CSCI, MATH 6860
FINITE ELEMENT ANALYSIS

Lecture Notes: Spring 2000

Joseph E. Flaherty
Amos Eaton Professor

Department of Computer Science
Department of Mathematical Sciences
Rensselaer Polytechnic Institute
Troy, New York 12180

CSCI, MATH 6860: Finite Element Analysis

Spring 2000

Outline

Bibliography

[1] A.K. Aziz, editor. *The Mathematical Foundations of the Finite Element Method with Applications to Partial Differential Equations*, New York, 1972. Academic Press.

[2] I. Babuška, J. Chandra, and J.E. Flaherty, editors. *Adaptive Computational Methods for Partial Differential Equations*, Philadelphia, 1983. SIAM.

[3] I. Babuška, O.C. Zienkiewicz, J. Gago, and E.R. de A. Oliveira, editors. *Accuracy Estimates and Adaptive Refinements in Finite Element Computations.* John Wiley and Sons, Chichester, 1986.

[4] K.-J. Bathe. *Finite Element Procedures.* Prentice Hall, Englewood Cliffs, 1995.

[5] E.B. Becker, G.F. Carey, and J.T. Oden. *Finite Elements: An Introduction*, volume I. Prentice Hall, Englewood Cliffs, 1981.

[6] M.W. Bern, J.E. Flaherty, and M. Luskin, editors. *Grid Generation and Adaptive Algorithms*, volume 113 of *The IMA Volumes in Mathematics and its Applications*, New York, 1999. Springer.

[7] C.A. Brebia. *The Boundary Element Method for Engineers.* Pentech Press, London, second edition, 1978.

[8] S.C. Brenner and L.R. Scott. *The Mathematical Theory of Finite Element Methods.* Springer-Verlag, New York, 1994.

[9] G.F. Carey. *Computational Grids: Generation, Adaptation, and Solution Strategies.* Series in Computational and Physical Processes in Mechanics and Thermal science. Taylor and Francis, New York, 1997.

[10] G.F. Carey and J.T. Oden. *Finite Elements: A Second Course*, volume II. Prentice Hall, Englewood Cliffs, 1983.

[11] G.F. Carey and J.T. Oden. *Finite Elements: Computational Aspects*, volume III. Prentice Hall, Englewood Cliffs, 1984.

[12] P.G. Ciarlet. *The Finite Element Method for Elliptic Problems.* North-Holland, Amsterdam, 1978.

[13] K. Clark, J.E. Flaherty, and M.S. Shephard, editors. *Applied Numerical Mathematics*, volume 14, 1994. Special Issue on Adaptive Methods for Partial Differential Equations.

[14] R.D. Cook, D.S. Malkus, and M.E. Plesha. *Concepts and Applications of Finite Element Analysis.* John Wiley and Sons, New York, third edition, 1989.

[15] K. Eriksson, D. Estep, P. Hansbo, and C. Johnson. *Computational Differential Equation.* Cambridge, Cambridge, 1996.

[16] G. Fairweather. *Finite Element Methods for Differential Equations.* Marcel Dekker, Basel, 1981.

[17] B. Finlayson. *The Method of Weighted Residuals and Variational Principles.* Academic Press, New York, 1972.

[18] J.E. Flaherty, P.J. Paslow, M.S. Shephard, and J.D. Vasilakis, editors. *Adaptive methods for Partial Differential Equations*, Philadelphia, 1989. SIAM.

[19] R.H. Gallagher, J.T. Oden, C. Taylor, and O.C. Zienkiewicz, editors. *Finite Elements in Fluids: Mathematical Foundations, Aerodynamics and Lubrication*, volume 2, London, 1975. John Wiley and Sons.

[20] R.H. Gallagher, J.T. Oden, C. Taylor, and O.C. Zienkiewicz, editors. *Finite Elements in Fluids: Viscous Flow and Hydrodynamics*, volume 1, London, 1975. John Wiley and Sons.

[21] R.H. Gallagher, O.C. Zienkiewicz, J.T. Oden, M. Morandi Cecchi, and C. Taylor, editors. *Finite Elements in Fluids*, volume 3, London, 1978. John Wiley and Sons.

[22] V. Girault and P.A. Raviart. *Finite Element Approximations of the Navier-Stokes Equations.* Number 749 in Lecture Notes in Mathematics. Springer-Verlag, Berlin, 1979.

[23] T.J.R. Hughes, editor. *Finite Element Methods for Convection Dominated Flows*, volume 34 of *AMD*, New York, 1979. ASME.

[24] T.J.R. Hughes. *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis.* Prentice Hall, Englewood Cliffs, 1987.

[25] B.M. Irons and S. Ahmed. *Techniques of Finite Elements.* Ellis Horwood, London, 1980.

[26] C. Johnson. *Numerical Solution of Partial Differential Equations by the Finite Element method.* Cambridge, Cambridge, 1987.

[27] N. Kikuchi. *Finite Element Methods in Mechanics.* Cambridge, Cambridge, 1986.

[28] Y.W. Kwon and H. Bang. *The Finite Element Method Using Matlab.* CRC Mechanical Engineering Series. CRC, Boca Raton, 1996.

[29] L. Lapidus and G.F. Pinder. *Numerical Solution of Partial Differential Equations in Science and Engineering.* Wiley-Interscience, New York, 1982.

[30] D.L. Logan. *A First Course in the Finite Element Method using ALGOR.* PWS, Boston, 1997.

[31] J.T. Oden. *Finite Elements of Nonlinear Continua.* Mc Graw-Hill, New York, 1971.

[32] J.T. Oden and G.F. Carey. *Finite Elements: Mathematical Aspects*, volume IV. Prentice Hall, Englewood Cliffs, 1983.

[33] D.R.J. Owen and E. Hinton. *Finite Elements in Plasticity-Theory and Practice.* Pineridge, Swansea, 1980.

[34] D.D. Reddy and B.D. Reddy. *Introductory Functional Analysis: With Applications to Boundary Value Problems and Finite Elements.* Number 27 in Texts in Applied Mathematics. Springer-Verlag, Berlin, 1997.

[35] J.N. Reddy. *The Finite Element Method in Heat Transfer and Fluid Dynamics.* CRC, Boca Raton, 1994.

[36] C. Schwab. *P- And Hp- Finite Element Methods: Theory and Applications in Solid and Fluid Mechanics.* Numerical Mathematics and Scientific Computation. Clarendon, London, 1999.

[37] G. Strang and G. Fix. *Analysis of the Finite Element Method.* Prentice-Hall, Englewood Cliffs, 1973.

[38] B. Szabó and I. Babuška. *Finite Element Analysis.* John Wiley and Sons, New York, 1991.

[39] F. Thomasset. *Implementation of Finite Element Methods for Navier-Stokes Equations.* Springer Series in Computational Physics. Springer-Verlag, New York, 1981.

[40] V. Thomée. *Galerkin Finite Element Methods for Parabolic Problems.* Number 1054 in Lecture Notes in Mathematics. Springer-Verlag, Berlin, 1984.

[41] R. Verfürth. *A Review of Posteriori Error Estimation and Adaptive Mesh-Refinement Techniques.* Teubner-Wiley, Stuttgart, 1996.

[42] R. Vichevetsky. *Computer Methods for Partial Differential Equations: Elliptic Equations and the Finite-Element Method*, volume 1. Prentice-Hall, Englewood Cliffs, 1981.

[43] R. Wait and A.R. Mitchell. *The Finite Element Analysis and Applications.* John Wiley and Sons, Chichester, 1985.

[44] R.E. White. *An Introduction to the Finite Element Method with Applications to Nonlinear Problems.* John Wiley and Sons, New York, 1985.

[45] J.R. Whiteman, editor. *The Mathematics of Finite Elements and Applications V, MAFELAP 1984*, London, 1985. Academic Press.

[46] J.R. Whiteman, editor. *The Mathematics of Finite Elements and Applications VI, MAFELAP 1987*, London, 1988. Academic Press.

[47] O.C. Zienkiewicz. *The Finite Element Method.* Mc Graw-Hill, New York, third edition, 1977.

[48] O.C. Zienkiewicz and R.L. Taylor. *Finite Element Method: Solid and Fluid Mechanics Dynamics and Non-Linearity.* Mc Graw-Hill, New York, 1991.

Chapter 1

Introduction

1.1 Historical Perspective

The finite element method is a computational technique for obtaining approximate solutions to the partial differential equations that arise in scientific and engineering applications. Rather than approximating the partial differential equation directly as with, *e.g.*, finite difference methods, the finite element method utilizes a *variational problem* that involves an integral of the differential equation over the problem domain. This domain is divided into a number of subdomains called *finite elements* and the solution of the partial differential equation is approximated by a simpler polynomial function on each element. These polynomials have to be pieced together so that the approximate solution has an appropriate degree of smoothness over the entire domain. Once this has been done, the variational integral is evaluated as a sum of contributions from each finite element. The result is an algebraic system for the approximate solution having a finite size rather than the original infinite-dimensional partial differential equation. Thus, like finite difference methods, the finite element process has *discretized* the partial differential equation but, unlike finite difference methods, the approximate solution is known throughout the domain as a pieceise polynomial function and not just at a set of points.

Logan [10] attributes the discovery of the finite element method to Hrennikof [8] and McHenry [11] who decomposed a two-dimensional problem domain into an assembly of one-dimensional bars and beams. In a paper that was not recognized for several years, Courant [6] used a *variational formulation* to describe a partial differential equation with a piecewise linear polynomial approximation of the solution relative to a decomposition of the problem domain into triangular elements to solve equilibrium and vibration problems. This is essentially the modern finite element method and represents the first application where the elements were pieces of a continuum rather than structural members.

Turner *et al.* [13] wrote a seminal paper on the subject that is widely regarded as the beginning of the finite element era. They showed how to solve one- and two-dimensional problems using actual structural elements and triangular- and rectangular-element decompositions of a continuum. Their timing was better than Courant's [6], since success of the finite element method is dependent on digital computation which was emerging in the late 1950s. The concept was extended to more complex problems such as plate and shell deformation (*cf.* the historical discussion in Logan [10], Chapter 1) and it has now become one of the most important numerical techniques for solving partial differential equations. It has a number of advantages relative to other methods, including

- the treatment of problems on complex irregular regions,
- the use of nonuniform meshes to reflect solution gradations,
- the treatment of boundary conditions involving fluxes, and
- the construction of high-order approximations.

Originally used for steady (elliptic) problems, the finite element method is now used to solve transient parabolic and hyperbolic problems. Estimates of discretization errors may be obtained for reasonable costs. These are being used to verify the accuracy of the computation, and also to control an adaptive process whereby meshes are automatically refined and coarsened and/or the degrees of polynomial approximations are varied so as to compute solutions to desired accuracies in an optimal fashion [1, 2, 3, 4, 5, 7, 14].

1.2 Weighted Residual Methods

Our goal, in this introductory chapter, is to introduce the basic principles and tools of the finite element method using a linear two-point boundary value problem of the form

$$\mathcal{L}[u] := -\frac{d}{dx}(p(x)\frac{du}{dx}) + q(x)u = f(x), \qquad 0 < x < 1, \tag{1.2.1a}$$

$$u(0) = u(1) = 0. \tag{1.2.1b}$$

The finite element method is primarily used to address partial differential equations and is hardly used for two-point boundary value problems. By focusing on this problem, we hope to introduce the fundamental concepts without the geometric complexities encountered in two and three dimensions.

Problems like (1.2.1) arise in many situations including the longitudinal deformation of an elastic rod, steady heat conduction, and the transverse deflection of a supported cable. In the latter case, for example, $u(x)$ represents the lateral deflection at position x of a cable having (scaled) unit length that is subjected to a tensile force p, loaded by a transverse force per unit length $f(x)$, and supported by a series of springs with elastic modulus q (Figure 1.2.1). The situation resembles the cable of a suspension bridge. The tensile force p is independent of x for the assumed small deformations of this model, but the applied loading and spring moduli could vary with position.

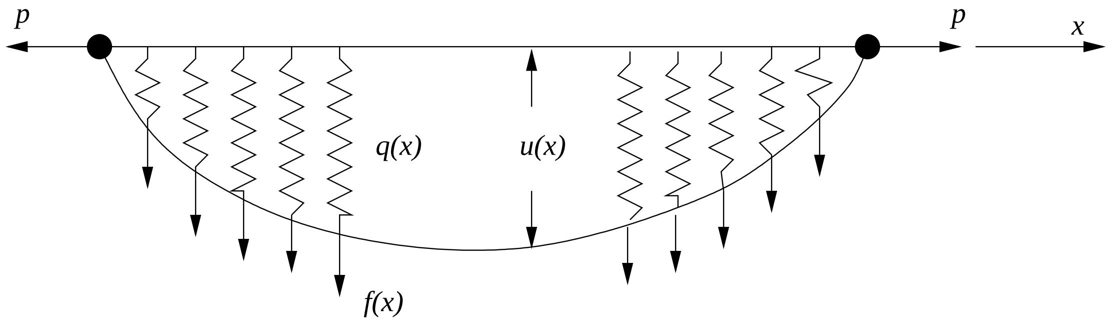


Figure 1.2.1: Deflection u of a cable under tension p, loaded by a force f per unit length, and supported by springs having elastic modulus q.

Mathematically, we will assume that $p(x)$ is positive and continuously differentiable for $x \in [0,1]$, $q(x)$ is non-negative and continuous on $[0,1]$, and $f(x)$ is continuous on $[0,1]$.

Even problems of this simplicity cannot generally be solved in terms of known functions; thus, the first topic on our agenda will be the development of a means of calculating approximate solutions of (1.2.1). With finite difference techniques, derivatives in (1.2.1a) are approximated by finite differences with respect to a mesh introduced on $[0,1]$ [12]. With the finite element method, the *method of weighted residuals (MWR)* is used to construct an integral formulation of (1.2.1) called a *variational problem.* To this end, let us multiply (1.2.1a) by a *test* or *weight function* v and integrate over $(0,1)$ to obtain

$$(v, \mathcal{L}[u] - f) = 0. \tag{1.2.2a}$$

We have introduced the L^2 *inner product*

$$(v, u) := \int_0^1 vu dx \tag{1.2.2b}$$

to represent the integral of a product of two functions.

The solution of (1.2.1) is also a solution of (1.2.2a) for all functions v for which the inner product exists. We'll express this requirement by writing $v \in L^2(0,1)$. All functions of class $L^2(0,1)$ are "square integrable" on $(0,1)$; thus, (v,v) exists. With this viewpoint and notation, we write (1.2.2a) more precisely as

$$(v, \mathcal{L}[u] - f) = 0, \qquad \forall v \in L^2(0,1). \tag{1.2.2c}$$

Equation (1.2.2c) is referred to as a *variational form* of problem (1.2.1). The reason for this terminology will become clearer as we develop the topic.

Using the method of weighted residuals, we construct approximate solutions by replacing u and v by simpler functions U and V and solving (1.2.2c) relative to these choices. Specifically, we'll consider approximations of the form

$$u(x) \approx U(x) = \sum_{j=1}^{N} c_j \phi_j(x), \tag{1.2.3a}$$

$$v(x) \approx V(x) = \sum_{j=1}^{N} d_j \psi_j(x). \tag{1.2.3b}$$

The functions $\phi_j(x)$ and $\psi_j(x)$, $j = 1, 2, \ldots, N$, are preselected and our goal is to determine the coefficients c_j, $j = 1, 2, \ldots, N$, so that U is a good approximation of u. For example, we might select

$$\phi_j(x) = \psi_j(x) = \sin j\pi x, \qquad j = 1, 2, \ldots, N,$$

to obtain approximations in the form of discrete Fourier series. In this case, every function satisfies the boundary conditions (1.2.1b), which seems like a good idea.

The approximation U is called a *trial function* and, as noted, V is called a *test function.* Since the differential operator $\mathcal{L}[u]$ is second order, we might expect $u \in C^2(0,1)$. (Actually, u can be slightly less smooth, but C^2 will suffice for the present discussion.) Thus, it's natural to expect U to also be an element of $C^2(0,1)$. Mathematically, we regard U as belonging to a finite-dimensional function space that is a subspace of $C^2(0,1)$. We express this condition by writing $U \in S^N(0,1) \subset C^2(0,1)$. (The restriction of these functions to the interval $0 < x < 1$ will, henceforth, be understood and we will no longer write the $(0,1)$.) With this interpretation, we'll call S^N the *trial space* and regard the preselected functions $\phi_j(x)$, $j = 1, 2, \ldots, N$, as forming a *basis* for S^N.

Likewise, since $v \in L^2$, we'll regard V as belonging to another finite-dimensional function space $\hat{S}^N$ called the *test space.* Thus, $V \in \hat{S}^N \subset L^2$ and $\psi_j(x)$, $j = 1, 2, \ldots, N$, provide a basis for $\hat{S}^N$.

Now, replacing v and u in (1.2.2c) by their approximations V and U, we have

$$(V, \mathcal{L}[U] - f) = 0, \qquad \forall V \in \hat{S}^N. \tag{1.2.4a}$$

The *residual*

$$r(x) := \mathcal{L}[U] - f(x) \tag{1.2.4b}$$

is apparent and clarifies the name "method of weighted residuals." The vanishing of the inner product (1.2.4a) implies that the residual is orthogonal in L^2 to all functions V in the test space $\hat{S}^N$.

Substituting (1.2.3) into (1.2.4a) and interchanging the sum and integral yields

$$\sum_{j=1}^{N} d_j(\psi_j, \mathcal{L}[U] - f) = 0, \qquad \forall d_j, \qquad j = 1, 2, \ldots, N. \tag{1.2.5}$$

Having selected the basis ψ_j, $j = 1, 2, \ldots, N$, the requirement that (1.2.4a) be satisfied for all $V \in \hat{S}^N$ implies that (1.2.5) be satisfied for all possible choices of d_k, $k = 1, 2, \ldots, N$. This, in turn, implies that

$$(\psi_j, \mathcal{L}[U] - f) = 0, \qquad j = 1, 2, \ldots, N. \tag{1.2.6}$$

Shortly, by example, we shall see that (1.2.6) represents a linear algebraic system for the unknown coefficients c_k, $k = 1, 2, \ldots, N$.

One obvious choice is to select the test space $\hat{S}^N$ to be the same as the trial space and use the same basis for each; thus, $\psi_k(x) = \phi_k(x)$, $k = 1, 2, \ldots, N$. This choice leads to *Galerkin's method*

$$(\phi_j, \mathcal{L}[u] - f) = 0, \qquad j = 1, 2, \ldots, N, \tag{1.2.7}$$

which, in a slightly different form, will be our "work horse." With $\phi_j \in C^2$, $j = 1, 2, \ldots, N$, the test space clearly has more continuity than necessary. Integrals like (1.2.4) or (1.2.6) exist for some pretty "wild" choices of V. Valid methods exist when V is a *Dirac delta* function (although such functions are not elements of L^2) and when V is a piecewise constant function (*cf.* Problems 1 and 2 at the end of this section).

There are many reasons to prefer a more symmetric variational form of (1.2.1) than (1.2.2), *e.g.*, problem (1.2.1) is symmetric (self-adjoint) and the variational form should reflect this. Additionally, we might want to choose the same trial and test spaces, as with Galerkin's method, but ask for less continuity on the trial space S^N. This is typically the case. As we shall see, it will be difficult to construct continuously differentiable approximations of finite element type in two and three dimensions. We can construct the symmetric variational form that we need by integrating the second derivative terms in (1.2.2a) by parts; thus, using (1.2.1a)

$$\int_0^1 v[-(pu')' + qu - f]dx = \int_0^1 (v'pu' + vqu - vf)dx - vpu'|_0^1 = 0 \tag{1.2.8}$$

where $(\ )' = d(\ )/dx$. The treatment of the last (boundary) term will need greater attention. For the moment, let v satisfy the same trivial boundary conditions (1.2.1b) as

u. In this case, the boundary term vanishes and (1.2.8) becomes

$$A(v,u) - (v,f) = 0 \tag{1.2.9a}$$

where

$$A(v,u) = \int_0^1 (v'pu' + vqu)dx. \tag{1.2.9b}$$

The integration by parts has eliminated second derivative terms from the formulation. Thus, solutions of (1.2.9) might have less continuity than those satisfying either (1.2.1) or (1.2.2). For this reason, they are called *weak solutions* in contrast to the *strong solutions* of (1.2.1) or (1.2.2). Weak solutions may lack the continuity to be strong solutions, but strong solutions are always weak solutions. In situations where weak and strong solutions differ, the weak solution is often the one of physical interest.

Since we've added a derivative to v by the integration by parts, v must be restricted to a space where functions have more continuity than those in L^2. Having symmetry in mind, we will select functions u and v that produce bounded values of

$$A(u,u) = \int_0^1 [p(u')^2 + qu^2]dx.$$

Actually, since p and q are smooth functions, it suffices for u and v to have bounded values of

$$\int_0^1 [(u')^2 + u^2]dx. \tag{1.2.10}$$

Functions where (1.2.10) exists are said to be elements of the Sobolev space H^1. We've also required that u and v satisfy the boundary conditions (1.2.1b). We identify those functions in H^1 that also satisfy (1.2.1b) as being elements of H_0^1. Thus, in summary, the variational problem consists of determining $u \in H_0^1$ such that

$$A(v,u) = (v,f), \qquad \forall v \in H_0^1. \tag{1.2.11}$$

The bilinear form $A(v,u)$ is called the *strain energy*. In mechanical systems it frequently corresponds to the stored or internal energy in the system.

We obtain approximate solutions of (1.2.11) in the manner described earlier for the more general method of weighted residuals. Thus, we replace u and v by their approximations U and V according to (1.2.3). Both U and V are regarded as belonging to the same finite-dimensional subspace S_0^N of H_0^1 and ϕ_j, $j = 1, 2, \ldots, N$, forms a basis for S_0^N. Thus, U is determined as the solution of

$$A(V,U) = (V,f), \qquad \forall V \in S_0^N. \tag{1.2.12a}$$

The substitution of (1.2.3b) with ψ_j replaced by ϕ_j in (1.2.12a) again reveals the more explicit form

$$A(\phi_j, U) = (\phi_j, f), \qquad j = 1, 2, \ldots, N. \tag{1.2.12b}$$

Finally, to make (1.2.12b) totally explicit, we eliminate U using (1.2.3a) and interchange a sum and integral to obtain

$$\sum_{k=1}^{N} c_k A(\phi_j, \phi_k) = (\phi_j, f), \qquad j = 1, 2, \ldots, N. \tag{1.2.12c}$$

Thus, the coefficients c_k, $k = 1, 2, \ldots, N$, of the approximate solution (1.2.3a) are determined as the solution of the linear algebraic equation (1.2.12c). Different choices of the basis ϕ_j, $j = 1, 2, \ldots, N$, will make the integrals involved in the strain energy (1.2.9b) and L^2 inner product (1.2.2b) easy or difficult to evaluate. They also affect the accuracy of the approximate solution. An example using a finite element basis is presented in the next section.

Problems

1. Consider the variational form (1.2.6) and select

$$\psi_j(x) = \delta(x - x_j), \qquad j = 1, 2, \ldots, N,$$

where $\delta(x)$ is the Dirac delta function satisfying

$$\delta(x) = 0, \qquad x \neq 0, \qquad \int_{-\infty}^{\infty} \delta(x)dx = 1,$$

and

$$0 < x_1 < x_2 < \ldots < x_N < 1.$$

Show that this choice of test function leads to the *collocation method*

$$\mathcal{L}[U] - f(x)|_{x=x_j} = 0, \qquad j = 1, 2, \ldots, N.$$

Thus, the differential equation (1.2.1) is satisfied exactly at N distinct points on $(0, 1)$.

2. The *subdomain method* uses piecewise continuous test functions having the basis

$$\psi_j(x) := \begin{cases} 1, & \text{if } x \in (x_{j-1/2}, x_{j+1/2}) \\ 0, & \text{otherwise} \end{cases} .$$

where $x_{j-1/2} = (x_j + x_{j-1})/2$. Using (1.2.6), show that the approximate solution $U(x)$ satisfies the differential equation (1.2.1a) on the average on each subinterval $(x_{j-1/2}, x_{j+1/2})$, $j = 1, 2, \ldots, N$.

3. Consider the two-point boundary value problem

$$-u'' + u = x, \qquad 0 < x < 1, \qquad u(0) = u(1) = 0,$$

which has the exact solution

$$u(x) = x - \frac{\sinh x}{\sinh 1}.$$

Solve this problem using Galerkin's method (1.2.12c) using the trial function

$$U(x) = c_1 \sin \pi x.$$

Thus, $N = 1$, $\phi_1(x) = \psi_1(x) = \sin \pi x$ in (1.2.3). Calculate the error in strain energy as $A(u,u) - A(U,U)$, where $A(u,v)$ is given by (1.2.9b).

1.3 A Simple Finite Element Problem

Finite element methods are weighted residuals methods that use bases of piecewise polynomials having small support. Thus, the functions $\phi(x)$ and $\psi(x)$ of (1.2.3, 1.2.4) are nonzero only on a small portion of problem domain. Since continuity may be difficult to impose, bases will typically use the minimum continuity necessary to ensure the existence of integrals and solution accuracy. The use of piecewise polynomial functions simplify the evaluation of integrals involved in the L^2 inner product and strain energy (1.2.2b, 1.2.9b) and help automate the solution process. Choosing bases with small support leads to a sparse, well-conditioned linear algebraic system (1.2.12c)) for the solution.

Let us illustrate the finite element method by solving the two-point boundary value problem (1.2.1) with constant coefficients, *i.e.*,

$$-pu'' + qu = f(x), \qquad 0 < x < 1, \qquad u(0) = u(1) = 0, \tag{1.3.1}$$

where $p > 0$ and $q \geq 0$. As described in Section 1.2, we construct a variational form of (1.2.1) using Galerkin's method (1.2.11). For this constant-coefficient problem, we seek to determine $u \in H_0^1$ satisfying

$$A(v,u) = (v,f), \qquad \forall v \in H_0^1, \tag{1.3.2a}$$

where

$$(v,u) = \int_0^1 vudx, \tag{1.3.2b}$$

$$A(v,u) = \int_0^1 (v'pu' + vqu)dx. \tag{1.3.2c}$$

With u and v belonging to H_0^1, we are sure that the integrals (1.3.2b,c) exist and that the trivial boundary conditions are satisfied.

We will subsequently show that functions (of one variable) belonging to H^1 must necessarily be continuous. Accepting this for the moment, let us establish the goal of finding the simplest continuous piecewise polynomial approximations of u and v. This would be a piecewise linear polynomial with respect to a mesh

$$0 = x_0 < x_1 < \ldots < x_N = 1 \tag{1.3.3}$$

introduced on $[0,1]$. Each subinterval (x_{j-1}, x_j), $j = 1, 2, \ldots, N$, is called a *finite element.* The basis is created from the "hat function"

$$\phi_j(x) = \begin{cases} \frac{x - x_{j-1}}{x_j - x_{j-1}}, & \text{if } x_{j-1} \le x < x_j \\ \frac{x_{j+1} - x}{x_{j+1} - x_j}, & \text{if} x_j \le x < x_{j+1} \\ 0, & \text{otherwise} \end{cases} . \tag{1.3.4a}$$

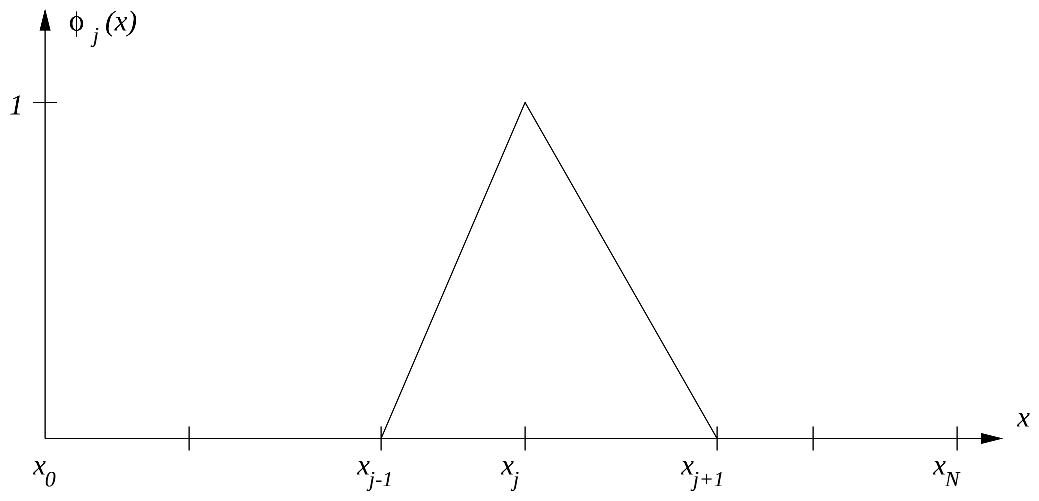


Figure 1.3.1: One-dimensional finite element mesh and piecewise linear hat function $\phi_j(x)$.

As shown in Figure 1.3.1, $\phi_j(x)$ is nonzero only on the two elements containing the *node* x_j. It rises and descends linearly on these two elements and has a maximal unit value at $x = x_j$. Indeed, it vanishes at all nodes but x_j, *i.e.*,

$$\phi_j(x_k) = \delta_{jk} := \begin{cases} 1, & \text{if } x_k = x_j \\ 0, & \text{otherwise} \end{cases} . \tag{1.3.4b}$$

Using this basis with (1.2.3), we consider approximations of the form

$$U(x) = \sum_{j=1}^{N-1} c_j \phi_j(x). \tag{1.3.5}$$

Let's examine this result more closely.

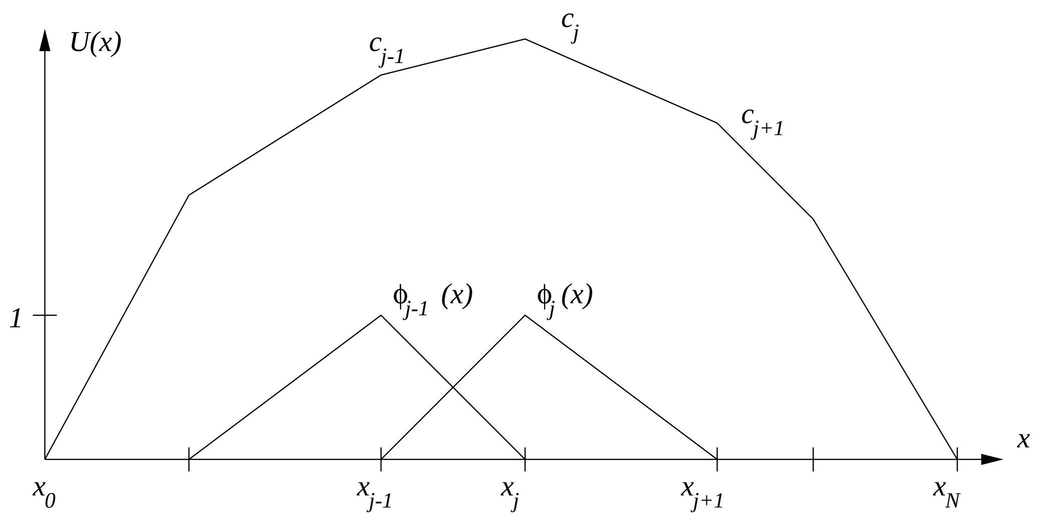


Figure 1.3.2: Piecewise linear finite element solution $U(x)$.

1. Since each $\phi_j(x)$ is a continuous piecewise linear function of x, their summation U is also continuous and piecewise linear. Evaluating U at a node x_k of the mesh using (1.3.4b) yields

$$U(x_k) = \sum_{j=1}^{N-1} c_j \phi_j(x_k) = c_k.$$

Thus, the coefficients c_k, $k = 1, 2, \ldots, N-1$, are the values of U at the interior nodes of the mesh (Figure 1.3.2).

2. By selecting the lower and upper summation indices as 1 and $N-1$ we have ensured that (1.3.5) satisfies the prescribed boundary conditions

$$U(0) = U(1) = 0.$$

As an alternative, we could have added basis elements $\phi_0(x)$ and $\phi_N(x)$ to the approximation and written the finite element solution as

$$U(x) = \sum_{j=0}^{N} c_j \phi_j(x). \tag{1.3.6}$$

Since, using (1.3.4b), $U(x_0) = c_0$ and $U(x_N) = c_N$, the boundary conditions are satisfied by requiring $c_0 = c_N = 0$. Thus, the representations (1.3.5) or (1.3.6) are identical; however, (1.3.6) would be useful with non-trivial boundary data.

3. The restriction of the finite element solution (1.3.5) or (1.3.6) to the element $[x_{j-1}, x_j]$ is the linear function

$$U(x) = c_{j-1}\phi_{j-1}(x) + c_j\phi_j(x), \qquad x \in [x_{j-1}, x_j], \tag{1.3.7}$$

since ϕ_{j-1} and ϕ_j are the only nonzero basis elements on $[x_{j-1}, x_j]$ (Figure 1.3.2).

Using Galerkin's method in the form (1.2.12c), we have to solve

$$\sum_{k=1}^{N-1} c_k A(\phi_j, \phi_k) = (\phi_j, f), \qquad j = 1, 2, \ldots, N-1. \tag{1.3.8}$$

Equation (1.3.8) can be evaluated in a straightforward manner by substituting replacing ϕ_k and ϕ_j using (1.3.4) and evaluating the strain energy and L^2 inner product according to (1.3.2b,c). This development is illustrated in several texts (*e.g.*, [9], Section 1.2). We'll take a slightly more complex path to the solution in order to focus on the computer implementation of the finite element method. Thus, write (1.2.12a) as the summation of contributions from each element

$$\sum_{j=1}^{N} [A_j(V, U) - (V, f)_j] = 0, \qquad \forall V \in S_0^N, \tag{1.3.9a}$$

where

$$A_j(V, U) = A_j^S(V, U) + A_j^M(V, U), \tag{1.3.9b}$$

$$A_j^S(V, U) = \int_{x_{j-1}}^{x_j} pV'U'dx, \tag{1.3.9c}$$

$$A_j^M(V, U) = \int_{x_{j-1}}^{x_j} qVUdx, \tag{1.3.9d}$$

$$(V, f)_j = \int_{x_{j-1}}^{x_j} Vfdx. \tag{1.3.9e}$$

It is customary to divide the strain energy into two parts with A_j^S arising from internal energies and A_j^M arising from inertial effects or sources of energy.

Matrices are simple data structures to manipulate on a computer, so let us write the restriction of $U(x)$ to $[x_{j-1}, x_j]$ according to (1.3.7) as

$$U(x) = [c_{j-1}, c_j] \begin{bmatrix} \phi_{j-1}(x) \\ \phi_j(x) \end{bmatrix} = [\phi_{j-1}(x), \phi_j(x)] \begin{bmatrix} c_{j-1} \\ c_j \end{bmatrix}, \qquad x \in [x_{j-1}, x_j]. \tag{1.3.10a}$$

We can, likewise, use (1.2.3b) to write the restriction of the test function $V(x)$ to $[x_{j-1}, x_j]$ in the same form

$$V(x) = [d_{j-1}, d_j] \begin{bmatrix} \phi_{j-1}(x) \\ \phi_j(x) \end{bmatrix} = [\phi_{j-1}(x), \phi_j(x)] \begin{bmatrix} d_{j-1} \\ d_j \end{bmatrix}, \qquad x \in [x_{j-1}, x_j]. \tag{1.3.10b}$$

Our task is to substitute (1.3.10) into (1.3.9c-e) and evaluate the integrals. Let us begin by differentiating (1.3.10a) while using (1.3.4a) to obtain

$$U'(x) = [c_{j-1}, c_j] \left[\begin{array}{c} -1/h_j \\ 1/h_j \end{array} \right] = [-1/h_j, 1/h_j] \left[\begin{array}{c} c_{j-1} \\ c_j \end{array} \right], \qquad x \in [x_{j-1}, x_j]. \qquad (1.3.11a)$$

where

$$h_j = x_j - x_{j-1}, \qquad j = 1, 2, \ldots, N. \qquad (1.3.11b)$$

Thus, $U'(x)$ is constant on $[x_{j-1}, x_j]$ and is given by the *first divided difference*

$$U'(x) = \frac{c_j - c_{j-1}}{h_j}, \qquad x \in [x_{j-1}, x_j].$$

Substituting (1.3.11) and a similar expression for $V'(x)$ into (1.3.9b) yields

$$A_j^S(V, U) = \int_{x_{j-1}}^{x_j} p[d_{j-1}, d_j] \left[\begin{array}{c} -1/h_j \\ 1/h_j \end{array} \right] [-1/h_j, 1/h_j] \left[\begin{array}{c} c_{j-1} \\ c_j \end{array} \right] dx$$

or

$$A_j^S(V, U) = [d_{j-1}, d_j] \left(\int_{x_{j-1}}^{x_j} p \left[\begin{array}{cc} 1/h_j^2 & -1/h_j^2 \\ -1/h_j^2 & 1/h_j^2 \end{array} \right] dx \right) \left[\begin{array}{c} c_{j-1} \\ c_j \end{array} \right].$$

The integrand is constant and can be evaluated to yield

$$A_j^S(V, U) = [d_{j-1}, d_j] \mathbf{K}_j \left[\begin{array}{c} c_{j-1} \\ c_j \end{array} \right], \qquad \mathbf{K}_j = \frac{p}{h_j} \left[\begin{array}{cc} 1 & -1 \\ -1 & 1 \end{array} \right]. \qquad (1.3.12)$$

The 2×2 matrix $\mathbf{K}_j$ is called the *element stiffness matrix*. It depends on j through h_j, but would also have such dependence if p varied with x. The key observation is that $\mathbf{K}_j$ can be evaluated without knowing c_{j-1}, c_j, d_{j-1}, or d_j and this greatly simplifies the automation of the finite element method.

The evaluation of A_j^M proceeds similarly by substituting (1.3.10) into (1.3.9d) to obtain

$$A_j^M(V, U) = \int_{x_{j-1}}^{x_j} q[d_{j-1}, d_j] \left[\begin{array}{c} \phi_{j-1} \\ \phi_j \end{array} \right] [\phi_{j-1}, \phi_j] \left[\begin{array}{c} c_{j-1} \\ c_j \end{array} \right] dx.$$

With q a constant, the integrand is a quadratic polynomial in x that may be integrated exactly (*cf.* Problem 1 at the end of this section) to yield

$$A_j^M(V, U) = [d_{j-1}, d_j] \mathbf{M}_j \left[\begin{array}{c} c_{j-1} c_j \end{array} \right], \qquad \mathbf{M}_j = \frac{q h_j}{6} \left[\begin{array}{cc} 2 & 1 \\ 1 & 2 \end{array} \right], \qquad (1.3.13)$$

where $\mathbf{M}_j$ is called the *element mass matrix* because, as noted, it often arises from inertial loading.

The final integral (1.3.9e) cannot be evaluated exactly for arbitrary functions $f(x)$. Without examining this matter carefully, let us approximate it by its linear interpolant

$$f(x) \approx f_{j-1}\phi_{j-1}(x) + f_j\phi_j(x), \qquad x \in [x_{j-1}, x_j], \tag{1.3.14}$$

where $f_j := f(x_j)$. Substituting (1.3.14) and (1.3.10b) into (1.3.9e) and evaluating the integral yields

$$(V, f)_j \approx \int_{x_{j-1}}^{x_j} [d_{j-1}, d_j] \begin{bmatrix} \phi_{j-1} \\ \phi_j \end{bmatrix} [\phi_{j-1}, \phi_j] \begin{bmatrix} f_{j-1} \\ f_j \end{bmatrix} dx = [d_{j-1}, d_j]\mathbf{l}_j \tag{1.3.15a}$$

where

$$\mathbf{l}_j = \frac{h_j}{6} \begin{bmatrix} 2f_{j-1} + f_j \\ f_{j-1} + 2f_j \end{bmatrix}. \tag{1.3.15b}$$

The vector $\mathbf{l}_j$ is called the *element load vector* and is due to the applied loading $f(x)$.

The next step in the process is the substitution of (1.3.12), (1.3.13), and (1.3.15) into (1.3.9a) and the summation over the elements. Since this our first example, we'll simplify matters by making the mesh uniform with $h_j = h = 1/N$, $j = 1, 2, \ldots, N$, and summing A_j^S, A_j^M, and $(V, f)_j$ separately. Thus, summing (1.3.12)

$$\sum_{j=1}^{N} A_j^S = \sum_{j=1}^{N} [d_{j-1}, d_j] \frac{p}{h} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} c_{j-1} \\ c_j \end{bmatrix}.$$

The first and last contributions have to be modified because of the boundary conditions which, as noted, prescribe $c_0 = c_N = d_0 = d_N = 0$. Thus,

$$\sum_{j=1}^{N} A_j^S = [d_1]\frac{p}{h}[1][c_1] + [d_1, d_2]\frac{p}{h} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} + \cdots$$

$$+[d_{N-2}, d_{N-1}]\frac{p}{h} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} c_{N-2} \\ c_{N-1} \end{bmatrix} + [d_{N-1}]\frac{p}{h}[1][c_{N-1}].$$

Although this form of the summation can be readily evaluated, it obscures the need for the matrices and complicates implementation issues. Thus, at the risk of further complexity, we'll expand each matrix and vector to dimension $N-1$ and write the summation as

$$\sum_{k=1}^{N} A_j^S = [d_1, d_2, \cdots, d_{N-1}]\frac{p}{h} \begin{bmatrix} 1 & & \\ & & \\ & & \\ & & \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_{N-1} \end{bmatrix}$$

$$+[d_1, d_2, \cdots, d_{N-1}]\frac{p}{h}\begin{bmatrix} 1 & -1 & & \\ -1 & 1 & & \\ & & & \\ & & & \\ & & & \end{bmatrix}\begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_{N-1} \end{bmatrix}$$

$$+\cdots+[d_1, d_2, \cdots, d_{N-1}]\frac{p}{h}\begin{bmatrix} & & & \\ & & & \\ & & & \\ & & 1 & -1 \\ & & -1 & 1 \end{bmatrix}\begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_{N-1} \end{bmatrix}$$

$$+[d_1, d_2, \cdots d_{N-1}]\frac{p}{h}\begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & 1 \end{bmatrix}\begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_{N-1} \end{bmatrix}$$

Zero elements of the matrices have not been shown for clarity. With all matrices and vectors having the same dimension, the summation is

$$\sum_{j=1}^{N} A_j^S = \mathbf{d}^T\mathbf{K}\mathbf{c}, \tag{1.3.16a}$$

where

$$\mathbf{K} = \frac{p}{h}\begin{bmatrix} 2 & -1 & & & & \\ -1 & 2 & -1 & & & \\ & -1 & 2 & -1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & -1 & 2 & -1 \\ & & & & -1 & 2 \end{bmatrix}, \tag{1.3.16b}$$

$$\mathbf{c} = [c_1, c_2, \cdots, c_{N-1}]^T, \tag{1.3.16c}$$

$$\mathbf{d} = [d_1, d_2, \cdots, d_{N-1}]^T. \tag{1.3.16d}$$

The matrix $\mathbf{K}$ is called the *global stiffness matrix.* It is symmetric, positive definite, and tridiagonal. In the form that we have developed the results, the summation over elements is regarded as an *assembly* process where the element stiffness matrices are added into their proper places in the global stiffness matrix. It is not necessary to actually extend the dimensions of the element matrices to those of the global stiffness matrix. As indicated in Figure 1.3.3, the elemental indices determine the proper location to add a local matrix into the global matrix. Thus, the 2×2 element stiffness matrix $\mathbf{K}_j$ is added to rows

$$A_1^S = d_1 \underbrace{\frac{p}{h}[1]}\, c_1 \qquad A_2^S = [d_1, d_2] \underbrace{\frac{p}{h}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}} \begin{bmatrix} c_1 \\ c_2 \end{bmatrix}$$

$$A_3^S = [d_2, d_3] \underbrace{\frac{p}{h}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}} \begin{bmatrix} c_2 \\ c_3 \end{bmatrix}$$

$$\mathbf{K} = \frac{p}{h}\begin{bmatrix} 2 & -1 & & & & \\ -1 & 2 & -1 & & & \\ & -1 & 1 & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \end{bmatrix}$$

Figure 1.3.3: Assembly of the first three element stiffness matrices into the global stiffness matrix.

$j-1$ and j and columns $j-1$ and j. Some modifications are needed for the first and last elements to account for the boundary conditions.

The summations of A_j^M and $(V, f)_j$ proceed in the same manner and, using (1.3.13) and (1.3.15), we obtain

$$\sum_{j=0}^{N} A_j^M = \mathbf{d}^T \mathbf{M} \mathbf{c}, \tag{1.3.17a}$$

$$\sum_{j=0}^{N} (V, f)_j = \mathbf{d}^T \mathbf{l} \tag{1.3.17b}$$

where

$$\mathbf{M} = \frac{qh}{6}\begin{bmatrix} 4 & 1 & & & \\ 1 & 4 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & 4 & 1 \\ & & & 1 & 4 \end{bmatrix}, \tag{1.3.17c}$$

$$\mathbf{l} = \frac{h}{6}\begin{bmatrix} f_0 + 4f_1 + f_2 \\ f_1 + 4f_2 + f_3 \\ \vdots \\ f_{N-2} + 4f_{N-1} + f_N \end{bmatrix}. \tag{1.3.17d}$$

The matrix $\mathbf{M}$ and the vector $\mathbf{l}$ are called the *global mass matrix* and *global load vector*, respectively.

Substituting (1.3.16a) and (1.3.17a,b) into (1.3.9a,b) gives

$$\mathbf{d}^T[(\mathbf{K}+\mathbf{M})\mathbf{c}-\mathbf{l}] = \mathbf{0}. \tag{1.3.18}$$

As noted in Section 1.2, the requirement that (1.3.9a) hold for *all* $V \in S_0^N$ is equivalent to satisfying (1.3.18) for all choices of $\mathbf{d}$. This is only possible when

$$(\mathbf{K}+\mathbf{M})\mathbf{c} = \mathbf{l}. \tag{1.3.19}$$

Thus, the nodal values c_k, $k = 1, 2, \dots, N-1$, of the finite element solution are determined by solving a linear algebraic system. With $\mathbf{c}$ known, the piecewise linear finite element U can be evaluated for any x using (1.2.3a). The matrix $\mathbf{K}+\mathbf{M}$ is symmetric, positive definite, and tridiagonal. Such systems may be solved by the tridiagonal algorithm (*cf.* Problem 2 at the end of this section) in $O(N)$ operations, where an operation is a scalar multiply followed by an addition.

The discrete system (1.3.19) is similar to the one that would be obtained from a centered finite difference approximation of (1.3.1), which is [12]

$$(\mathbf{K}+\mathbf{D})\hat{\mathbf{c}} = \hat{\mathbf{l}}, \tag{1.3.20a}$$

where

$$\mathbf{D} = qh\begin{bmatrix} 1 & & & \\ & 1 & & \\ & & \ddots & \\ & & & 1 \end{bmatrix}, \qquad \hat{\mathbf{l}} = h\begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_{N-1} \end{bmatrix}, \qquad \hat{\mathbf{c}} = \begin{bmatrix} \hat{c}_1 \\ \hat{c}_2 \\ \vdots \\ \hat{c}_{N-1} \end{bmatrix}. \tag{1.3.20b}$$

Thus, the qu and f terms in (1.3.1) are approximated by diagonal matrices with the finite difference method. In the finite element method, they are "smoothed" by coupling diagonal terms with their nearest neighbors using Simpson's rule weights. The diagonal matrix $\mathbf{D}$ is sometimes called a "lumped" approximation of the *consistent mass matrix* $\mathbf{M}$. Both finite difference and finite element solutions behave similarly for the present problem and have the same order of accuracy at the nodes of a uniform mesh.

Example 1.3.1. Consider the finite element solution of

$$-u'' + u = x, \qquad 0 < x < 1, \qquad u(0) = u(1) = 0,$$

which has the exact solution

$$u(x) = x - \frac{\sinh x}{\sinh 1}.$$

Relative to the more general problem (1.3.1), this example has $p = q = 1$ and $f(x) = x$. We solve it using the piecewise-linear finite element method developed in this section on uniform meshes with spacing $h = 1/N$ for $N = 4, 8, \ldots, 128$. Before presenting results, it is worthwhile mentioning that the load vector (1.3.15) is exact for this example. Even though we replaced $f(x)$ by its piecewise linear interpolant according to (1.3.14), this introduced no error since $f(x)$ is a linear function of x.

Letting

$$e(x) = u(x) - U(x) \tag{1.3.21}$$

denote the *discretization error,* in Table 1.3.1 we display the maximum error of the finite element solution and of its first derivative at the nodes of a mesh, *i.e.*,

$$|e|_\infty := \max_{0<j<N} |e(x_j)|, \qquad |e'|_\infty := \max_{1<j<N} |e'(x_j^-)|. \tag{1.3.22}$$

We have seen that $U'(x)$ is a piecewise constant function with jumps at nodes. Data in Table 1.3.1 were obtained by using derivatives from the left, *i.e.*, $x_j^- = \lim_{\epsilon\to 0} x_j - \epsilon$. With this interpretation, the results of second and fourth columns of Table 1.3.1 indicate that $|e|_\infty/h^2$ and $|e'|_\infty/h$ are (essentially) constants; hence, we may conclude that $|e|_\infty = O(h^2)$ and $|e'|_\infty = O(h)$.

N	$\|e\|_\infty$	$\|e\|_\infty/h^2$	$\|e'\|_\infty$	$\|e'\|_\infty/h$
4	0.269(-3)	0.430(-2)	0.111(0)	0.444
8	0.688(-4)	0.441(-2)	0.589(-1)	0.471
16	0.172(-4)	0.441(-2)	0.303(-1)	0.485
32	0.432(-5)	0.442(-2)	0.154(-1)	0.492
64	0.108(-5)	0.442(-2)	0.775(-2)	0.496
128	0.270(-6)	0.442(-2)	0.389(-2)	0.498

Table 1.3.1: Maximum nodal errors of the piecewise-linear finite element solution and its derivative for Example 1.3.1. (Numbers in parenthesis indicate a power of 10.)

The finite element and exact solutions of this problem are displayed in Figure 1.3.4 for a uniform mesh with eight elements. It appears that the pointwise discretization errors are much smaller at nodes than they are globally. We'll see that this phenomena, called *superconvergence*, applies more generally than this single example would imply.

Since finite element solutions are defined as continuous functions (of x), we can also appraise their behavior in some global norms in addition to the *discrete error norms* used in Table 1.3.1. Many norms could provide useful information. One that we will use quite

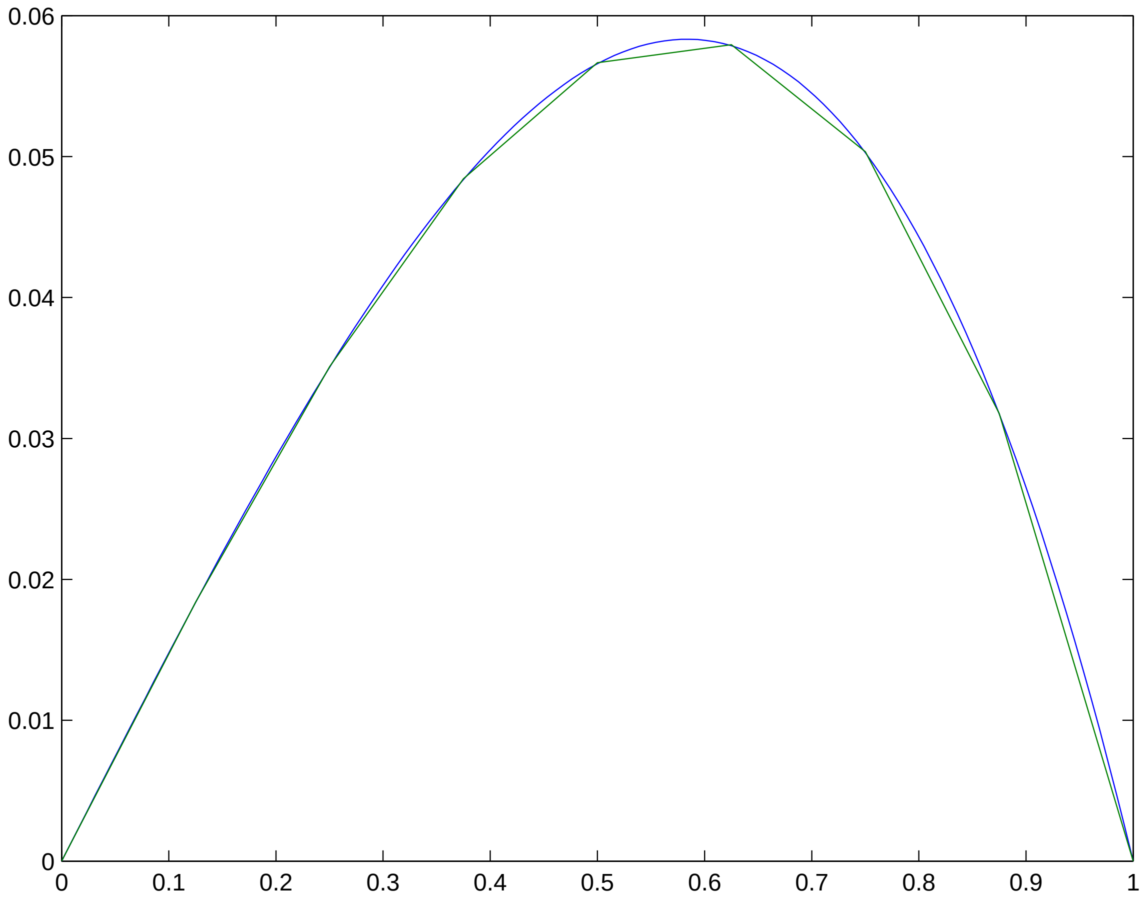


Figure 1.3.4: Exact and piecewise-linear finite element solutions of Example 1.3.1 on an 8-element mesh.

often is the square root of the strain energy of the error; thus, using (1.3.2c)

$$\|e\|_A := \sqrt{A(e,e)} = \left\{\int_0^1 [p(e')^2 + qe^2]dx\right\}^{1/2}. \tag{1.3.23a}$$

This expression may easily be evaluated as a summation over the elements in the spirit of (1.3.9a). With $p = q = 1$ for this example,

$$\|e\|_A^2 = \int_0^1 [(e')^2 + e^2]dx.$$

The integral is the square of the norm used on the Sobolev space H^1; thus,

$$\|e\|_1 := \left\{\int_0^1 [(e')^2 + e^2]dx\right\}^{1/2}. \tag{1.3.23b}$$

Other global error measures will be important to our analyses; however, the only one that we will introduce at the moment is the L^2 norm

$$\|e\|_0 := \left[\int_0^1 e^2(x)dx\right]^{1/2}. \tag{1.3.23c}$$

Results for the L^2 and strain energy errors, presented in Table 1.3.2 for this example, indicate that $\|e\|_0 = O(h^2)$ and $\|e\|_A = O(h)$. The error in the H^1 norm would be identical to that in strain energy. Later, we will prove that these a priori error estimates are correct for this and similar problems. Errors in strain energy converge slower than those in L^2 because solution derivatives are involved and their nodal convergence is $O(h)$ (Table 1.3.1).

N	$\|e\|_0$	$\|e\|_0/h^2$	$\|e\|_A$	$\|e\|_A/h$
4	0.265(-2)	0.425(-1)	0.390(-1)	0.156
8	0.656(-3)	0.426(-1)	0.195(-1)	0.157
16	0.167(-3)	0.427(-1)	0.979(-2)	0.157
32	0.417(-4)	0.427(-1)	0.490(-2)	0.157
64	0.104(-4)	0.427(-1)	0.245(-2)	0.157
128	0.260(-5)	0.427(-1)	0.122(-2)	0.157

Table 1.3.2: Errors in L^2 and strain energy for the piecewise-linear finite element solution of Example 1.3.1. (Numbers in parenthesis indicate a power of 10.)

Problems

1. The integral involved in obtaining the mass matrix according to (1.3.13) may, of course, be done symbolically. It may also be evaluated numerically by Simpson's rule which is exact in this case since the integrand is a quadratic polynomial. Recall, that Simpson's rule is

$$\int_0^h \mathbf{F}(x)dx \approx \frac{h}{6}[\mathbf{F}(0) + 4\mathbf{F}(h/2) + \mathbf{F}(h)].$$

The mass matrix is

$$\mathbf{M}_j = \int_{x_{j-1}}^{x_j} \begin{bmatrix} \phi_{j-1} \\ \phi_j \end{bmatrix} [\phi_{j-1}, \phi_j]dx.$$

Using (1.3.4), determine $\mathbf{M}_j$ by Simpson's rule to verify the result (1.3.13). The use of Simpson's rule may be simpler than symbolic integration for this example since the trial functions are zero or unity at the ends of an element and one half at its center.

2. Consider the solution of the linear system

$$\mathbf{AX} = \mathbf{F}, \tag{1.3.24a}$$

where $\mathbf{F}$ and $\mathbf{X}$ are N-dimensional vectors and $\mathbf{A}$ is an $N \times N$ tridiagonal matrix having the form

$$\mathbf{A} = \begin{bmatrix} a_1 & c_1 & & & \\ b_2 & a_2 & c_2 & & \\ & \ddots & \ddots & \ddots & \\ & & b_{N-1} & a_{N-1} & c_{N-1} \\ & & & b_N & a_N \end{bmatrix}. \tag{1.3.24b}$$

Assume that pivoting is not necessary and factor $\mathbf{A}$ as

$$\mathbf{A} = \mathbf{LU}, \tag{1.3.25a}$$

where $\mathbf{L}$ and $\mathbf{U}$ are lower and upper bidiagonal matrices having the form

$$\mathbf{L} = \begin{bmatrix} 1 & & & & \\ l_2 & 1 & & & \\ & l_3 & 1 & & \\ & & \ddots & \ddots & \\ & & & l_N & 1 \end{bmatrix}, \tag{1.3.25b}$$

$$\mathbf{U} = \begin{bmatrix} u_1 & v_1 & & & \\ & u_2 & v_2 & & \\ & & \ddots & \ddots & \\ & & & u_{N-1} & v_{N-1} \\ & & & & u_N \end{bmatrix}. \tag{1.3.25c}$$

Once the coefficients l_j, $j = 2, 3, \ldots, N$, u_j, $j = 1, 2, \ldots, N$, and v_j, $j = 1, 2, \ldots, N-1$, have been determined, the system (1.3.24a) may easily be solved by forward and backward substitution. Thus, using (1.3.25a) in (1.3.24a) gives

$$\mathbf{LUX} = \mathbf{F}. \tag{1.3.26a}$$

Let

$$\mathbf{UX} = \mathbf{Y}, \tag{1.3.26b}$$

then,

$$\mathbf{LY} = \mathbf{F}. \tag{1.3.26c}$$

2.1. Using (1.3.24) and (1.3.25), show

$$u_1 = a_1,$$

$$l_j = b_j / u_{j-1}, \qquad u_j = a_j - l_j c_{j-1}, \qquad j = 2, 3, \ldots, N,$$

$$v_j = c_j, \qquad j = 2, 3, \ldots, N.$$

2.2. Show that $\mathbf{Y}$ and $\mathbf{X}$ are computed as

$$Y_1 = F_1,$$

$$Y_j = F_j - l_j Y_{j-1}, \qquad j = 2, 3, \ldots, N,$$

$$X_N = y_N / u_N,$$

$$X_j = (Y_j - v_j X_{j+1})/u_j, \qquad j = N-1, N-2, \ldots, 1.$$

2.3. Develop a procedure to implement this scheme for solving tridiagonal systems. The input to the procedure should be N and vectors containing the coefficients a_j, b_j, c_j, f_j, $j = 1, 2, \ldots, N$. The procedure should output the solution $\mathbf{X}$. The coefficients a_j, b_j, *etc.*, $j = 1, 2, \ldots, N$, should be replaced by u_j, v_j, *etc.*, $j = 1, 2, \ldots, N$, in order to save storage. If you want, the solution $\mathbf{X}$ can be returned in $\mathbf{F}$.

2.4. Estimate the number of arithmetic operations necessary to factor $\mathbf{A}$ and for the forward and backward substitution process.

3. Consider the linear boundary value problem

$$-pu'' + qu = f(x), \qquad 0 < x < 1, \qquad u(0) = u'(1) = 0.$$

where p and q are positive constants and $f(x)$ is a smooth function.

3.1. Show that the Galerkin form of this boundary-value problem consists of finding $u \in H_0^1$ satisfying

$$A(v, u) - (v, f) = \int_0^1 (v'pu' + vqu)dx - \int_0^1 vf\,dx = 0, \qquad \forall v \in H_0^1.$$

For this problem, functions $u(x) \in H_0^1$ are required to be elements of H^1 and satisfy the Dirichlet boundary condition $u(0) = 0$. The Neumann boundary condition at $x = 1$ need not be satisfied by either u or v.

3.2. Introduce N equally spaced elements on $0 \leq x \leq 1$ with nodes $x_j = jh$, $j = 0, 1, \ldots, N$ ($h = 1/N$). Approximate u by U having the form

$$U(x) = \sum_{j=1}^{N} c_k \phi_k(x),$$

where $\phi_j(x)$, $j = 1, 2, \ldots, N$, is the piecewise linear basis (1.3.4), and use Galerkin's method to obtain the global stiffness and mass matrices and the load vector for this problem. (Again, the approximation $U(x)$ does not satisfy the *natural boundary condition* $u'(1) = 0$ nor does it have to. We will discuss this issue in Chapter 2.)

3.3. Write a program to solve this problem using the finite element method developed in Part 3.2b and the tridiagonal algorithm of Problem 2. Execute your program with $p = 1$, $q = 1$, and $f(x) = x$ and $f(x) = x^2$. In each case, use $N = 4$, 8, 16, and 32. Let $e(x) = u(x) - U(x)$ and, for each value of N, compute $|e|_\infty$, $|e'(x_N)|$, and $\|e\|_A$ according to (1.3.22) and (1.3.23a). You may (optionally) also compute $\|e\|_0$ as defined by (1.3.23c). In each case, estimate the rate of convergence of the finite element solution to the exact solution.

4. The Galerkin form of (1.3.1) consists of determining $u \in H_0^1$ such that (1.3.2) is satisfied. Similarly, the finite element solution $U \in S_0^N \subset H_0^1$ satisfies (1.2.12). Letting $e(x) = u(x) - U(x)$, show

$$A(e, e) = A(u, u) - A(U, U)$$

where the strain energy $A(v, u)$ is given by (1.3.2c). We have, thus, shown that the strain energy of the error is the error of the strain energy.

Bibliography

[1] I. Babuška, J. Chandra, and J.E. Flaherty, editors. *Adaptive Computational Methods for Partial Differential Equations*, Philadelphia, 1983. SIAM.

[2] I. Babuška, O.C. Zienkiewicz, J. Gago, and E.R. de A. Oliveira, editors. *Accuracy Estimates and Adaptive Refinements in Finite Element Computations.* John Wiley and Sons, Chichester, 1986.

[3] M.W. Bern, J.E. Flaherty, and M. Luskin, editors. *Grid Generation and Adaptive Algorithms*, volume 113 of *The IMA Volumes in Mathematics and its Applications*, New York, 1999. Springer.

[4] G.F. Carey. *Computational Grids: Generation, Adaptation, and Solution Strategies.* Series in Computational and Physical Processes in Mechanics and Thermal science. Taylor and Francis, New York, 1997.

[5] K. Clark, J.E. Flaherty, and M.S. Shephard, editors. *Applied Numerical Mathematics*, volume 14, 1994. Special Issue on Adaptive Methods for Partial Differential Equations.

[6] R. Courant. Variational methods for the solution of problems of equilibrium and vibrations. *Bulletin of the American Mathematics Society*, 49:1–23, 1943.

[7] J.E. Flaherty, P.J. Paslow, M.S. Shephard, and J.D. Vasilakis, editors. *Adaptive methods for Partial Differential Equations*, Philadelphia, 1989. SIAM.

[8] A. Hrennikoff. Solutions of problems in elasticity by the frame work method. *Journal of Applied Mechanics*, 8:169–175, 1941.

[9] C. Johnson. *Numerical Solution of Partial Differential Equations by the Finite Element method.* Cambridge, Cambridge, 1987.

[10] D.L. Logan. *A First Course in the Finite Element Method using ALGOR.* PWS, Boston, 1997.

[11] D. McHenry. A lattice analogy for the solution of plane stress problems. *Journal of the Institute of Civil Engineers*, 21:59–82, 1943.

[12] J.C. Strikwerda. *Finite Difference Schemes and Partial Differential Equations*. Chapman and Hall, Pacific Grove, 1989.

[13] M.J. Turner, R.W. Clough, H.C. Martin, and L.J. Topp. Stiffness and deflection analysis of complex structures. *Journal of the Aeronautical Sciences*, 23:805–824, 1956.

[14] R. Verfürth. *A Review of Posteriori Error Estimation and Adaptive Mesh-Refinement Techniques*. Teubner-Wiley, Stuttgart, 1996.

Chapter 2

One-Dimensional Finite Element Methods

2.1 Introduction

The piecewise-linear Galerkin finite element method of Chapter 1 can be extended in several directions. The most important of these is multi-dimensional problems; however, we'll postpone this until the next chapter. Here, we'll address and answer some other questions that may be inferred from our brief encounter with the method.

1. Is the Galerkin method the best way to construct a variational principal for a partial differential system?

2. How do we construct variational principals for more complex problems? Specifically, how do we treat boundary conditions other than Dirichlet?

3. The finite element method appeared to converge as $O(h)$ in strain energy and $O(h^2)$ in L^2 for the example of Section 1.3. Is this true more generally?

4. Can the finite element solution be improved by using higher-degree piecewise-polynomial approximations? What are the costs and benefits of doing this?

We'll tackle the Galerkin formulations in the next two sections, examine higher-degree piecewise polynomials in Sections 2.4 and 2.5, and conclude with a discussion of approximation errors in Section 2.6.

2.2 Galerkin's Method and Extremal Principles

"For since the fabric of the universe is most perfect and the work of a most wise creator, nothing at all takes place in the universe in which some rule of maximum or minimum does not appear."

- Leonhard Euler

Although the construction of variational principles from differential equations is an important aspect of the finite element method it will not be our main objective. We'll explore some properties of variational principles with a goal of developing a more thorough understanding of Galerkin's method and of answering the questions raised in Section 2.1. In particular, we'll focus on boundary conditions, approximating spaces, and extremal properties of Galerkin's method. Once again, we'll use the model two-point Dirichlet problem

$$\mathcal{L}[u] := -[p(x)u']' + q(x)u = f(x), \qquad 0 < x < 1, \tag{2.2.1a}$$

$$u(0) = u(1) = 0, \tag{2.2.1b}$$

with $p(x) > 0$, $q(x) \geq 0$, and $f(x)$ being smooth functions on $0 \leq x \leq 1$.

As described in Chapter 1, the Galerkin form of (2.2.1) is obtained by multiplying (2.2.1a) by a test function $v \in H_0^1$, integrating the result on $[0, 1]$, and integrating the second-order term by parts to obtain

$$A(v, u) = (v, f), \qquad \forall v \in H_0^1, \tag{2.2.2a}$$

where

$$(v, f) = \int_0^1 vf dx, \tag{2.2.2b}$$

and

$$A(v, u) = (v', pu') + (v, qu) = \int_0^1 (v'pu' + vqu)dx, \tag{2.2.2c}$$

and functions v belonging to the Sobolev space H^1 have bounded values of

$$\int_0^1 [(v')^2 + v^2]dx.$$

For (2.2.1), a function v is in H_0^1 if it also satisfies the trivial boundary conditions $v(0) = v(1) = 0$. As we shall discover in Section 2.3, the definition of H_0^1 will depend on the type of boundary conditions being applied to the differential equation.

There is a connection between self-adjoint differential problems such as (2.2.1) and the minimum problem: find $w \in H_0^1$ that minimizes

$$I[w] = A(w, w) - 2(w, f) = \int_0^1 [p(w')^2 + qw^2 - 2wf]dx. \tag{2.2.3}$$

Maximum and minimum variational principles occur throughout mathematics and physics and a discipline called the Calculus of Variations arose in order to study them. The initial goal of this field was to extend the elementary theory of the calculus of the maxima and minima of functions to problems of finding the extrema of functionals such as $I[w]$. (A *functional* is an operator that maps functions onto real numbers.)

The construction of the Galerkin form (2.2.2) of a problem from the differential form (2.2.1) is straight forward; however, the construction of the extremal problem (2.2.3) is not. We do not pursue this matter here. Instead, we refer readers to a text on the calculus of variations such as Courant and Hilbert [4]. Accepting (2.2.3), we establish that the solution u of Galerkin's method (2.2.2) is optimal in the sense of minimizing (2.2.3).

Theorem 2.2.1. *The function $u \in H_0^1$ that minimizes (2.2.3) is the one that satisfies (2.2.2a) and conversely.*

Proof. Suppose first that $u(x)$ is the solution of (2.2.2a). We choose a real parameter ϵ and any function $v(x) \in H_0^1$ and define the *comparison function*

$$w(x) = u(x) + \epsilon v(x). \tag{2.2.4}$$

For each function $v(x)$ we have a one parameter family of comparison functions $w(x) \in H_0^1$ with the solution $u(x)$ of (2.2.2a) obtained when $\epsilon = 0$. By a suitable choice of ϵ and $v(x)$ we can use (2.2.4) to represent any function in H_0^1. A comparison function $w(x)$ and its *variation* $\epsilon v(x)$ are shown in Figure 2.2.1.

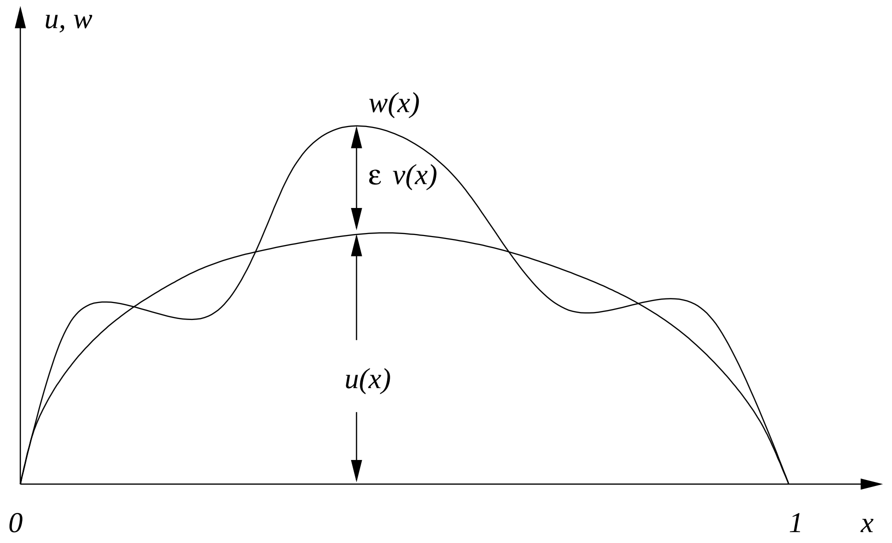


Figure 2.2.1: A comparison function $w(x)$ and its variation $\epsilon v(x)$ from $u(x)$.

Substituting (2.2.4) into (2.2.3)

$$I[w] = I[u + \epsilon v] = A(u + \epsilon v, u + \epsilon v) - 2(u + \epsilon v, f).$$

Expanding the strain energy and L^2 inner products using (2.2.2b,c)

$$I[w] = A(u,u) - 2(u,f) + 2\epsilon[A(v,u) - (v,f)] + \epsilon^2 A(v,v).$$

By hypothesis, u satisfies (2.2.2a), so the $O(\epsilon)$ term vanishes. Using (2.2.3), we have

$$I[w] = I[u] + \epsilon^2 A(v,v).$$

With $p > 0$ and $q \geq 0$, we have $A(v,v) \geq 0$; thus, u minimizes (2.2.3).

In order to prove the converse, assume that $u(x)$ minimizes (2.2.3) and use (2.2.4) to obtain

$$I[u] \leq I[u + \epsilon v].$$

For a particular choice of $v(x)$, let us regard $I[u+\epsilon v]$ as a function $\Phi(\epsilon)$, *i.e.*,

$$I[u+\epsilon v] := \Phi(\epsilon) = A(u+\epsilon v, u+\epsilon v) - 2(u+\epsilon v, f).$$

A necessary condition for a minimum to occur at $\epsilon = 0$ is $\Phi'(0) = 0$; thus, differentiating

$$\Phi'(\epsilon) = 2\epsilon A(v,v) + 2A(v,u) - 2(v,f)$$

and setting $\epsilon = 0$

$$\Phi'(0) = 2[A(v,u) - (v,f)] = 0.$$

Thus, u is a solution of (2.2.2a). □

The following corollary verifies that the minimizing function u is also unique.

Corollary 2.2.1. *The solution u of (2.2.2a) (or (2.2.3)) is unique.*

Proof. Suppose there are two functions $u_1, u_2 \in H_0^1$ satisfying (2.2.2a), *i.e.*,

$$A(v,u_1) = (v,f), \qquad A(v,u_2) = (v,f), \qquad \forall v \in H_0^1.$$

Subtracting

$$A(v, u_1 - u_2) = 0, \qquad \forall v \in H_0^1.$$

Since this relation is valid for *all* $v \in H_0^1$, choose $v = u_1 - u_2$ to obtain

$$A(u_1 - u_2, u_1 - u_2) = 0.$$

If $q(x) > 0$, $x \in (0,1)$, then $A(u_1 - u_2, u_1 - u_2)$ is positive unless $u_1 = u_2$. Thus, it suffices to consider cases when either *(i)* $q(x) \equiv 0$, $x \in [0,1]$, or *(ii)* $q(x)$ vanishes at isolated points or subintervals of $(0,1)$. For simplicity, let us consider the former case. The analysis of the latter case is similar.

When $q(x) \equiv 0$, $x \in [0,1]$, $A(u_1 - u_2, u_1 - u_2)$ can vanish when $u_1' - u_2' = 0$. Thus, $u_1 - u_2$ is a constant. However, both u_1 and u_2 satisfy the trivial boundary conditions (2.2.1b); thus, the constant is zero and $u_1 = u_2$. □

Corollary 2.2.2. *If u, w are smooth enough to permit integrating $A(u,v)$ by parts then the minimizer of (2.2.3), the solution of the Galerkin problem (2.2.2a), and the solution of the two-point boundary value problem (2.2.1) are all equivalent.*

Proof. Integrate the differentiated term in (2.2.3) by parts to obtain

$$I[w] = \int_0^1 [-w(pw')' + qw^2 - 2fw]dx + wpw'|_0^1.$$

The last term vanishes since $w \in H_0^1$; thus, using (2.2.1a) and (2.2.2b) we have

$$I[w] = (w, \mathcal{L}[w]) - 2(w, f). \tag{2.2.5}$$

Now, follow the steps used in Theorem 2.2.1 to show

$$A(v,u) - (v,f) = (v, \mathcal{L}[u] - f) = 0, \qquad \forall v \in H_0^1,$$

and, hence, establish the result. □

The minimization problems (2.2.3) and (2.2.5) are equivalent when w has sufficient smoothness. However, minimizers of (2.2.3) may lack the smoothness to satisfy (2.2.5). When this occurs, the solutions with less smoothness are often the ones of physical interest.

Problems

1. Consider the "stationary value" problem: find functions $w(x)$ that give stationary values (maxima, minima, or saddle points) of

$$I[w] = \int_0^1 F(x, w, w')dx \tag{2.2.6a}$$

when w satisfies the "essential" (Dirichlet) boundary conditions

$$w(0) = \alpha, \qquad w(1) = \beta. \tag{2.2.6b}$$

Let $w \in H_E^1$, where the subscript E denotes that w satisfies (2.2.6b), and consider comparison functions of the form (2.2.4) where $u \in H_E^1$ is the function that makes $I[w]$ stationary and $v \in H_0^1$ is arbitrary. (Functions in H_0^1 satisfy trivial versions of (2.2.6b), *i.e.*, $v(0) = v(1) = 0$.)

Using (2.2.1) as an example, we would have

$$F(x, w, w') = p(x)(w')^2 + q(x)w^2 - 2wf(x), \qquad \alpha = \beta = 0.$$

Smooth stationary values of (2.2.6) would be minima in this case and correspond to solutions of the differential equation (2.2.1a) and boundary conditions (2.2.1b).

Differential equations arising from minimum principles like (2.2.3) or from stationary value principles like (2.2.6) are called *Euler-Lagrange equations.*

Beginning with (2.2.6), follow the steps used in proving Theorem 2.2.1 to determine the Galerkin equations satisfied by u. Also determine the Euler-Lagrange equations for smooth stationary values of (2.2.6).

2.3 Essential and Natural Boundary Conditions

The analyses of Section 2.2 readily extend to problems having nontrivial Dirichlet boundary conditions of the form

$$u(0) = \alpha, \qquad u(1) = \beta. \tag{2.3.1a}$$

In this case, functions u satisfying (2.2.2a) or w satisfying (2.2.3) must be members of H^1 and satisfy (2.3.1a). We'll indicate this by writing $u, w \in H^1_E$, with the subscript E denoting that u and w satisfy the *essential* Dirichlet boundary conditions (2.3.1a). Since u and w satisfy (2.3.1a), we may use (2.2.4) or the interpretation of ϵv as a variation shown in Figure 2.2.1, to conclude that v should still vanish at $x = 0$ and 1 and, hence, belong to H^1_0.

When u is not prescribed at $x = 0$ and/or 1, the function v need not vanish there. Let us illustrate this when (2.2.1a) is subject to conditions

$$u(0) = \alpha, \qquad p(1)u'(1) = \beta. \tag{2.3.1b}$$

Thus, an essential or Dirichlet condition is specified at $x = 0$ and a *Neumann* condition is specified at $x = 1$. Let us construct a Galerkin form of the problem by again multiplying (2.2.1a) by a test function v, integrating on $[0, 1]$, and integrating the second derivative terms by parts to obtain

$$\int_0^1 v[-(pu')' + qu - f]dx = A(v, u) - (v, f) - vpu'|_0^1 = 0. \tag{2.3.2}$$

With an essential boundary condition at $x = 0$, we specify $u(0) = \alpha$ and $v(0) = 0$; however, $u(1)$ and $v(1)$ remain unspecified. We still classify $u \in H^1_E$ and $v \in H^1_0$ since they satisfy, respectively, the essential and trivial essential boundary conditions specified with the problem.

With $v(0) = 0$ and $p(1)u'(1) = \beta$, we use (2.3.2) to establish the Galerkin problem for (2.2.1a, 2.3.1b) as: determine $u \in H^1_E$ satisfying

$$A(v, u) = (v, f) + v(1)\beta, \qquad \forall v \in H^1_0. \tag{2.3.3}$$

Let us reiterate that the subscript E on H^1 restricts functions to satisfy Dirichlet (essential) boundary conditions, but not any Neumann conditions. The subscript 0 restricts functions to satisfy trivial versions of any Dirichlet conditions but, once again, Neumann conditions are not imposed.

As with problem (2.2.1), there is a minimization problem corresponding to (2.2.3): determine $w \in H_E^1$ that minimizes

$$I[w] = A(w, w) - 2(w, f) - 2w(1)\beta. \tag{2.3.4}$$

Furthermore, in analogy with Theorem 2.2.1, we have an equivalence between the Galerkin (2.3.3) and minimization (2.3.4) problems.

Theorem 2.3.1. *The function $u \in H_E^1$ that minimizes (2.3.4) is the one that satisfies (2.3.3) and conversely.*

Proof. The proof is so similar to that of Theorem 2.2.1 that we'll only prove that the function u that minimizes (2.3.4) also satisfies (2.3.3). (The remainder of the proof is stated as Problem 1 as the end of this section.)

Again, create the comparison function

$$w(x) = u(x) + \epsilon v(x); \tag{2.3.5}$$

however, as shown in Figure 2.3.1, $v(1)$ need not vanish. By hypothesis we have

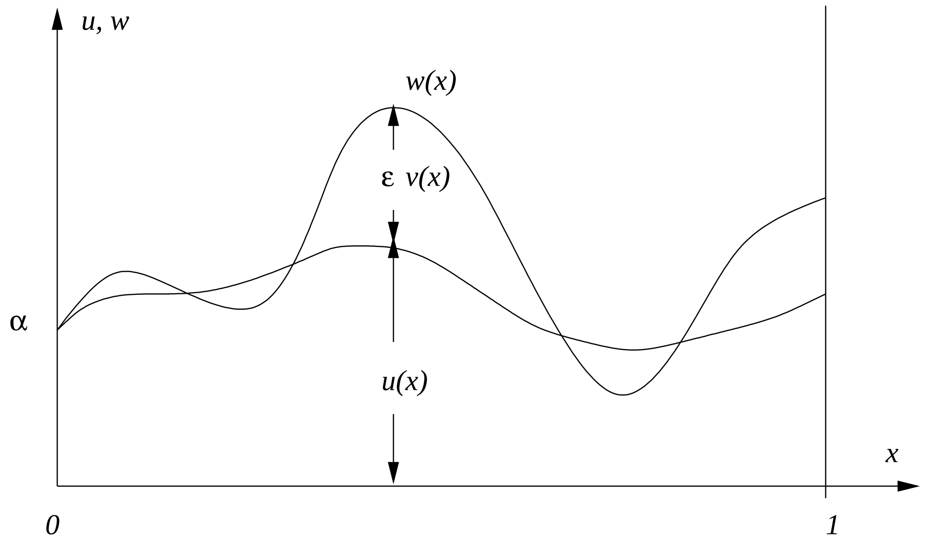


Figure 2.3.1: Comparison function $w(x)$ and variation $\epsilon v(x)$ when Dirichlet data is prescribed at $x = 0$ and Neumann data is prescribed at $x = 1$.

$$I[u] \le I[u + \epsilon v] = \Phi(\epsilon) = A(u + \epsilon v, u + \epsilon v) - 2(u + \epsilon v, f) - 2[u(1) + \epsilon v(1)]\beta.$$

Differentiating with respect to ϵ yields the necessary condition for a minimum as

$$\Phi'(0) = 2[A(v,u) - (v,f) - v(1)\beta] = 0;$$

thus, u satisfies (2.3.3). □

As expected, Theorem 2.3.1 can be extended when the minimizing function u is smooth.

Corollary 2.3.1. *Smooth functions $u \in H_E^1$ satisfying (2.3.3) or minimizing (2.3.4) also satisfy (2.2.1a, 2.3.1b).*

Proof. Using (2.2.2c), integrate the differentiated term in (2.3.3) by parts to obtain

$$\int_0^1 v[-(pu')' + qu - f]dx + v(1)[p(1)u'(1) - \beta] = 0, \qquad \forall v \in H_0^1. \tag{2.3.6}$$

Since (2.3.6) must be satisfied for all possible test functions, it must vanish for those functions satisfying $v(1) = 0$. Thus, we conclude that (2.2.1a) is satisfied. Similarly, by considering test functions v that are nonzero in just a small neighborhood of $x = 1$, we conclude that the boundary condition (2.3.1b) must be satisfied. Since (2.3.6) must be satisfied for *all* test functions v, the solution u must satisfy (2.2.1a) in the interior of the domain and (2.3.1b) at $x = 1$. □

Neumann boundary conditions, or other boundary conditions prescribing derivatives (*cf.* Problem 2 at the end of this section), are called *natural boundary conditions* because they follow directly from the variational principle and are not explicitly imposed. Essential boundary conditions constrain the space of functions that may be used as trial or comparison functions. Natural boundary conditions impose no constraints on the function spaces but, rather, alter the variational principle.

Problems

1. Prove the remainder of Theorem 2.3.1, *i.e.*, show that functions that satisfy (2.3.3) also minimize (2.3.4).

2. Show that the Galerkin form (2.2.1a) with the Robin boundary conditions

$$p(0)u'(0) + \gamma_0 u(0) = \alpha_0, \qquad p(1)u'(1) + \gamma_1 u(1) = \alpha_1$$

is: determine $u \in H^1$ satisfying

$$A(v,u) = (v,f) + v(1)(\alpha_1 - \gamma_1 u(1)) - v(0)(\alpha_0 - \gamma_0 u(0)), \qquad \forall v \in H^1.$$

Also show that the function $w \in H^1$ that minimizes

$$I[w] = A(w,w) - 2(w,f) - 2\alpha_1 w(1) + \gamma_1 w(1)^2 + 2\alpha_0 w(0) - \gamma_0 w(0)^2$$

is u, the solution of the Galerkin problem.

3. Construct the Galerkin form of (2.2.1) when

$$p(x) = \begin{cases} 1, & \text{if } 0 \le x < 1/2 \\ 2, & \text{if } 1/2 \le x \le 1 \end{cases} .$$

   Such a situation can arise in a steady heat-conduction problem when the medium is made of two different materials that are joined at $x = 1/2$. What conditions must u satisfy at $x = 1/2$?

2.4 Piecewise Lagrange Polynomials

The finite element method is not limited to piecewise-linear polynomial approximations and its extention to higher-degree polynomials is straight forward. There is, however, a question of the best basis. Many possibilities are available from design and approximation theory. Of these, splines and Hermite approximations [5] are generally not used because they offer more smoothness and/or a larger support than needed or desired. Lagrange interpolation [2] and a hierarchical approximation in the spirit of Newton's divided-difference polynomials will be our choices. The piecewise-linear "hat" function

$$\phi_j(x) = \begin{cases} \frac{x-x_{j-1}}{x_j-x_{j-1}}, & \text{if } x_{j-1} \le x < x_j \\ \frac{x_{j+1}-x}{x_{j+1}-x_j}, & \text{if } x_j \le x < x_{j+1} \\ 0, & \text{otherwise} \end{cases} \qquad (2.4.1a)$$

on the mesh

$$x_0 < x_1 < \ldots < x_N \qquad (2.4.1b)$$

is a member of both classes. It has two desirable properties: (i) $\phi_j(x)$ is unity at node j and vanishes at all other nodes and (ii) ϕ_j is only nonzero on those elements containing node j. The first property simplifies the determination of solutions at nodes while the second simplifies the solution of the algebraic system that results from the finite element discretization. The Lagrangian basis maintains these properties with increasing polynomial degree. Hierarchical approximations, on the other hand, maintain only the second property. They are constructed by adding high-degree corrections to lower-degree members of the series.

We will examine Lagrange bases in this section, beginning with the quadratic polynomial basis. These are constructed by adding an extra node $x_{j-1/2}$ at the midpoint of each element $[x_{j-1}, x_j]$, $j = 1, 2, \ldots, N$ (Figure 2.4.1). As with the piecewise-linear basis (2.4.1a), one basis function is associated with each node. Those associated with vertices are

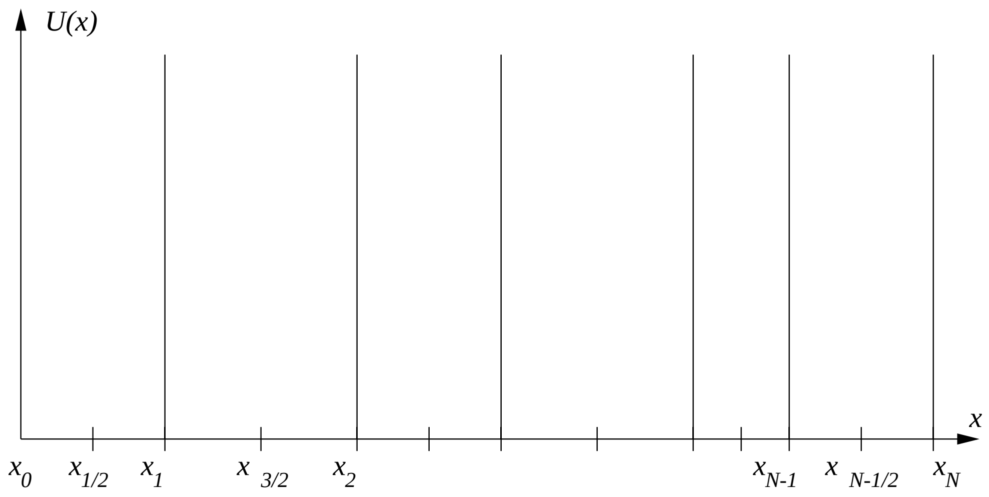


Figure 2.4.1: Finite element mesh for piecewise-quadratic Lagrange polynomial approximations.

$$\phi_j(x) = \begin{cases} 1+3(\frac{x-x_j}{h_j})+2(\frac{x-x_j}{h_j})^2, & \text{if } x_{j-1} \le x < x_j \\ 1-3(\frac{x-x_j}{h_{j+1}})+2(\frac{x-x_j}{h_{j+1}})^2, & \text{if } x_j \le x < x_{j+1} \\ 0, & \text{otherwise} \end{cases}, \qquad j = 0, 1, \ldots, N, \tag{2.4.2a}$$

and those associated with element midpoints are

$$\phi_{j-1/2}(x) = \begin{cases} 1-4(\frac{x-x_{j-1/2}}{h_j})^2, & \text{if } x_{j-1} \le x < x_j \\ 0, & \text{otherwise} \end{cases}, \qquad j = 1, 2, \ldots, N. \tag{2.4.2b}$$

Here

$$h_j = x_j - x_{j-1}, \qquad j = 1, 2, \ldots, N. \tag{2.4.2c}$$

These functions are shown in Figure 2.4.2. Their construction (to be described) invovles satsifying

$$\phi_j(x_k) = \begin{cases} 1, & \text{if } j = k \\ 0, & \text{otherwise} \end{cases}, \qquad j, k = 0, 1/2, 1, \ldots, N-1, N-1/2, N. \tag{2.4.3}$$

Basis functions associated with a vertex are nonzero on at most two elements and those associated with an element midpoint are nonzero on only one element. Thus, as noted, the Lagrange basis function ϕ_j is nonzero only on elements containing node j. The functions (2.4.2a,b) are quadratic polynomials on each element. Their construction and trivial extension to other finite elements guarantees that they are continuous over the entire mesh and, like (2.4.1), are members of H^1.

The finite element trial function $U(x)$ is a linear combination of (2.4.2a,b) over the vertices and element midpoints of the mesh that may be written as

$$U(x) = \sum_{j=0}^{N} c_j\phi_j(x) + \sum_{j=1}^{N} c_{j-1/2}\phi_{j-1/2}(x) = \sum_{j=0}^{2N} c_{j/2}\phi_{j/2}(x). \tag{2.4.4}$$

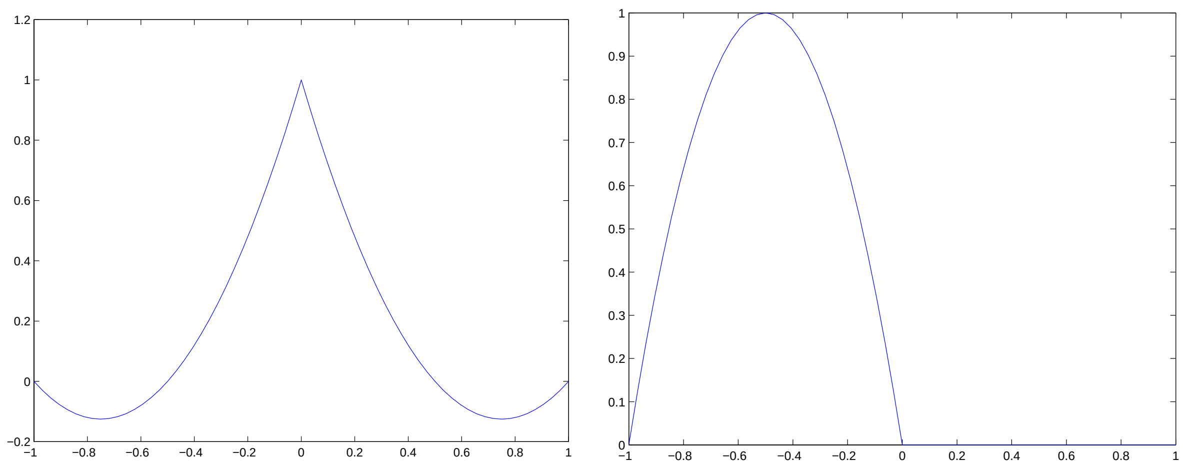


Figure 2.4.2: Piecewise-quadratic Lagrange basis functions for a vertex at $x = 0$ (left) and an element midpoint at $x = -0.5$ (right). When comparing with (2.4.2), set $x_{j-1} = -1$, $x_{j-1/2} = -0.5$, $x_j = 0$, $x_{j+1/2} = 0.5$, and $x_{j+1} = 1$.

Using (2.4.3), we see that $U(x_k) = c_k$, $k = 0, 1/2, 1, \ldots, N - 1/2, N$.

Cubic, quartic, *etc.* Lagrangian polynomials are generated by adding nodes to element interiors. However, prior to constructing them, let's introduce some terminology and simplify the node numbering to better suit our task. Finite element bases are constructed implicitly in an element-by-element manner in terms of shape functions. A *shape function* is the restriction of a basis function to an element. Thus, for the piecewise-quadratic Lagrange polynomial, there are three nontrivial shape functions on the element $\Omega_j := [x_{j-1}, x_j]$:

- the right portion of $\phi_{j-1}(x)$

$$N_{j-1,j}(x) = 1 - 3(\frac{x - x_{j-1}}{h_j}) + 2(\frac{x - x_{j-1}}{h_j})^2, \tag{2.4.5a}$$

- $\phi_{j-1/2}(x)$

$$N_{j-1/2,j}(x) = 1 - 4(\frac{x - x_{j-1/2}}{h_j})^2, \tag{2.4.5b}$$

- and the left portion of $\phi_j(x)$

$$N_{j,j}(x) = 1 + 3(\frac{x - x_j}{h_j}) + 2(\frac{x - x_j}{h_j})^2, \qquad x \in \Omega_j, \tag{2.4.5c}$$

(Figure 2.4.3). In these equations, $N_{k,j}$ is the shape function associated with node k, $k = j - 1, j - 1/2, j$, of element j (the subinterval Ω_j). We may use (2.4.4) and (2.4.5) to write the restriction of $U(x)$ to Ω_j as

$$U(x) = c_{j-1}N_{j-1,j} + c_{j-1/2}N_{j-1/2,j} + c_jN_{j,j}, \qquad x \in \Omega_j.$$

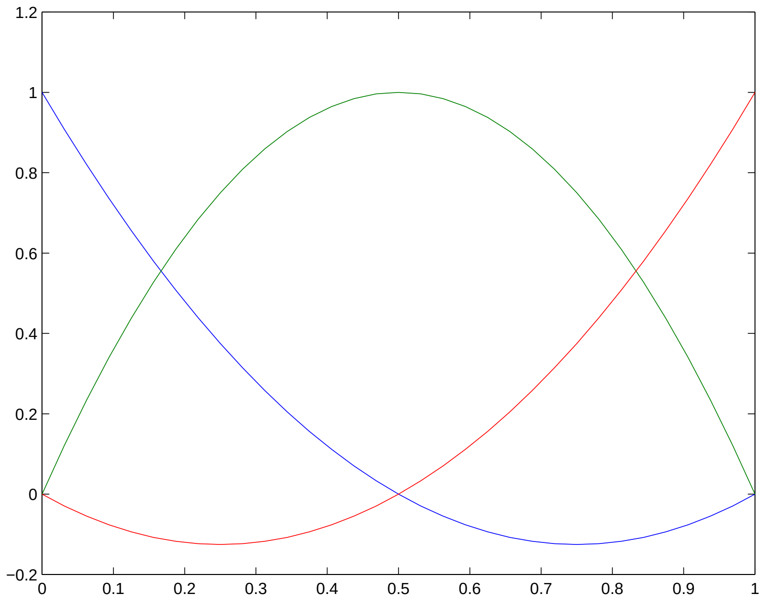


Figure 2.4.3: The three quadratic Lagrangian shape functions on the element $[x_{j-1}, x_j]$. When comparing with (2.4.5), set $x_{j-1} = 0$, $x_{j-1/2} = 0.5$, and $x_j = 1$.

More generally, we will associate the shape function $N_{k,e}(x)$ with *mesh entity* k of element e. At present, the only mesh entities that we know of are vertices and (nodes on) elements; however, edges and faces will be introduced in two and three dimensions. The key construction concept is that the shape function $N_{k,e}(x)$ is

1. nonzero only on element e and

2. nonzero only if mesh entity k belongs to element e.

A one-dimensional Lagrange polynomial shape function of degree p is constructed on an element e using two vertex nodes and $p-1$ nodes interior to the element. The generation of shape functions is straight forward, but it is customary and convenient to do this on a "canonical element." Thus, we map an arbitrary element $\Omega_e = [x_{j-1}, x_j]$ onto $-1 \le \xi \le 1$ by the linear transformation

$$x(\xi) = \frac{1-\xi}{2}x_{j-1} + \frac{1+\xi}{2}x_j, \qquad \xi \in [-1, 1]. \tag{2.4.6}$$

Nodes on the canonical element are numbered according to some simple scheme, *i.e.*, 0 to p with $\xi_0 = -1$, $\xi_p = 1$, and $0 < \xi_1 < \xi_2 < \ldots < \xi_{p-1} < 1$ (Figure 2.4.4). These are mapped to the actual physical nodes $x_{j-1}, x_{j-1+1/p}, \ldots, x_j$ on Ω_e using (2.4.6). Thus,

$$x_{j-1+i/p} = \frac{1-\xi_i}{2}x_{j-1} + \frac{1+\xi_i}{2}x_j, \qquad i = 0, 1, \ldots, p.$$

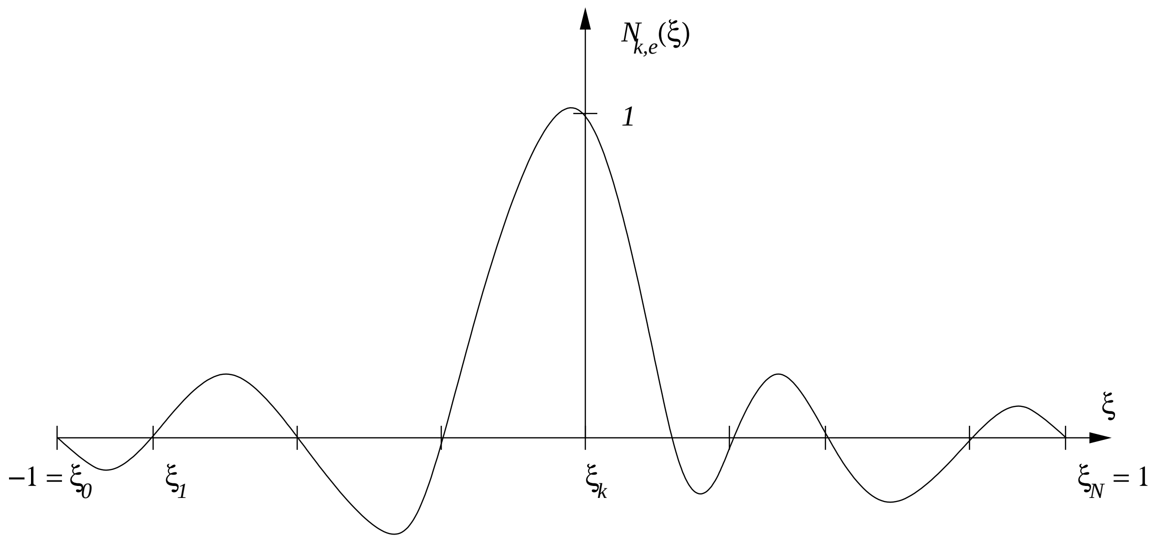


Figure 2.4.4: An element e used to construct a p th-degree Lagrangian shape function and the shape function $N_{k,e}(x)$ associated with node k.

The Lagrangian shape function $N_{k,e}(\xi)$ of degree p has a unit value at node k of element e and vanishes at all other nodes; thus,

$$N_{k,e}(\xi_l) = \delta_{kl} = \begin{cases} 1, & \text{if } k = l \\ 0, & \text{otherwise} \end{cases}, \qquad l = 0, 1, \ldots, p. \tag{2.4.7a}$$

It is extended trivially when $\xi \notin [-1, 1]$. The conditions expressed by (2.4.7a) imply that

$$N_{k,e}(\xi) = \prod_{l=0,\ l\neq k}^{p} \frac{\xi - \xi_l}{\xi_k - \xi_l} = \frac{(\xi - \xi_0)(\xi - \xi_1)\ldots(\xi - \xi_{k-1})(\xi - \xi_{k+1})\ldots(\xi - \xi_p)}{(\xi_k - \xi_0)(\xi_k - \xi_1)\ldots(\xi_k - \xi_{k-1})(\xi_k - \xi_{k+1})\ldots(\xi_k - \xi_p)}. \tag{2.4.7b}$$

We easily check that $N_{k,e}$ (i) is a polynomial of degree p in ξ and (ii) it satisfies conditions (2.4.7a). It is shown in Figure 2.4.4. Written in terms of shape function, the restriction of U to the canonical element is

$$U(\xi) = \sum_{k=0}^{p} c_k N_{k,e}(\xi). \tag{2.4.8}$$

Example 2.4.1. Let us construct the quadratic Lagrange shape functions on the canonical element by setting $p = 2$ in (2.4.7b) to obtain

$$N_{0,e}(\xi) = \frac{(\xi - \xi_1)(\xi - \xi_2)}{(\xi_0 - \xi_1)(\xi_0 - \xi_2)}, \qquad N_{1,e}(\xi) = \frac{(\xi - \xi_0)(\xi - \xi_2)}{(\xi_1 - \xi_0)(\xi_1 - \xi_2)},$$

$$N_{2,e}(\xi) = \frac{(\xi - \xi_0)(\xi - \xi_1)}{(\xi_2 - \xi_0)(\xi_2 - \xi_1)}.$$

Setting $\xi_0 = -1$, $\xi_1 = 0$, and $\xi_2 = 1$ yields

$$N_{0,e}(\xi) = \frac{\xi(\xi-1)}{2}, \qquad N_{1,e}(\xi) = (1-\xi^2), \qquad N_{2,e}(\xi) = \frac{(\xi+1)\xi}{2}. \tag{2.4.9}$$

These may easily be shown to be identical to (2.4.2) by using the transformation (2.4.6) (see Problem 1 at the end of this section).

Example 2.4.2. Setting $p = 1$ in (2.4.7b), we obtain the linear shape functions on the canonical element as

$$N_{0,e} = \frac{1-\xi}{2}, \qquad N_{1,e} = \frac{1+\xi}{2}. \tag{2.4.10}$$

The two nodes needed for these shape functions are at the vertices $\xi_0 = -1$ and $\xi_1 = 1$. Using the transformation (2.4.6), these yield the two pieces of the hat function (2.4.1a). We also note that these shape functions were used in the linear coordinate transformation (2.4.6). This will arise again in Chapter 5.

Problems

1. Show the the quadratic Lagrange shape functions (2.4.9) on the canonical $[-1, 1]$ element transform to those on the physical element (2.4.2) upon use of (2.4.6)

2. Construct the shape functions for a cubic Lagrange polynomial from the general formula (2.4.7) by using two vertex nodes and two interior nodes equally spaced on the canonical $[-1, 1]$ element. Sketch the shape functions. Write the basis functions for a vertex and an interior node.

2.5 Hierarchical Bases

With a hierarchical polynomial representation the basis of degree $p+1$ is obtained as a correction to that of degree p. Thus, the entire basis need not be reconstructed when increasing the polynomial degree. With finite element methods, they produce algebraic systems that are less susceptible to round-off error accumulation at high order than those produced by a Lagrange basis.

With the linear hierarchical basis being the usual hat functions (2.4.1), let us begin with the piecewise-quadratic hierarchical polynomial. The restriction of this function to element $\Omega_e = [x_{j-1}, x_j]$ has the form

$$U^2(x) = U^1(x) + c_{j-1/2} N^2_{j-1/2,e}(x), \qquad x \in \Omega_e, \tag{2.5.1a}$$

where $U^1(x)$ is the piecewise-linear finite element approximation on Ω_e

$$U^1(x) = c_{j-1} N^1_{j-1,e}(x) + c_j N^1_{j,e}(x). \tag{2.5.1b}$$

Superscripts have been added to U and $N_{j,e}$ to identify their polynomial degree. Thus,

$$N^1_{j-1,e}(x) = \begin{cases} \frac{x_j - x}{h_j}, & \text{if } x \in \Omega_e \\ 0, & \text{otherwise} \end{cases}, \tag{2.5.1c}$$

$$N^1_{j,e}(x) = \begin{cases} \frac{x - x_{j-1}}{h_j}, & \text{if } x \in \Omega_e \\ 0, & \text{otherwise} \end{cases} \tag{2.5.1d}$$

are the usual hat function (2.4.1) associated with a piecewise-linear approximation $U^1(x)$. The quadratic correction $N^2_{j-1/2,e}(x)$ is required to (i) be a quadratic polynomial, (ii) vanish when $x \notin \Omega_e$, and (iii) be continuous. These conditions imply that $N^2_{j-1/2,e}$ is proportional to the quadratic Lagrange shape function (2.4.5b) and we will take it to be identical; thus,

$$N^2_{j-1/2,e}(x) = \begin{cases} 1 - 4(\frac{x - x_{j-1/2}}{h_j})^2, & \text{if } x \in \Omega_e \\ 0, & \text{otherwise} \end{cases}. \tag{2.5.1e}$$

The normalization $N^2_{j-1/2,e}(x_{j-1/2}) = 1$ is not necessary, but seems convenient.

Like the quadratic Lagrange approximation, the quadratic hierarchical polynomial has three nontrivial shape functions per element; however, two of them are linear and only one is quadratic (Figure 2.5.1). The basis, however, still spans quadratic polynomials. Examining (2.5.1), we see that $c_{j-1} = U(x_{j-1})$ and $c_j = U(x_j)$; however,

$$U(x_{j-1/2}) = \frac{c_{j-1} + c_j}{2} + c_{j-1/2}.$$

Differentiating (2.5.1a) twice with respect to x gives an interpretation to $c_{j-1/2}$ as

$$c_{j-1/2} = -\frac{h^2}{8} U''(x_{j-1/2}).$$

This interpretation may be useful but is not necessary.

A basis may be constructed from the shape functions in the manner described for Lagrange polynomials. With a mesh having the structure used for the piecewise-quadratic Lagrange polynomials (Figure 2.4.1), the piecewise-quadratic hierarchical functions have the form

$$U(x) = \sum_{j=0}^{N} c_j \phi^1_j(x) + \sum_{j=1}^{N} c_{j-1/2} \phi^2_{j-1/2}(x) \tag{2.5.2}$$

where $\phi^1_j(x)$ is the hat function basis (2.4.1a) and $\phi^2_j(x) = N^2_{j,e}(x)$.

Higher-degree hierarchical polynomials are obtained by adding more correction terms to the lower-degree polynomials. It is convenient to construct and display these polynomials on the canonical $[-1, 1]$ element used in Section 2.4. The linear transformation

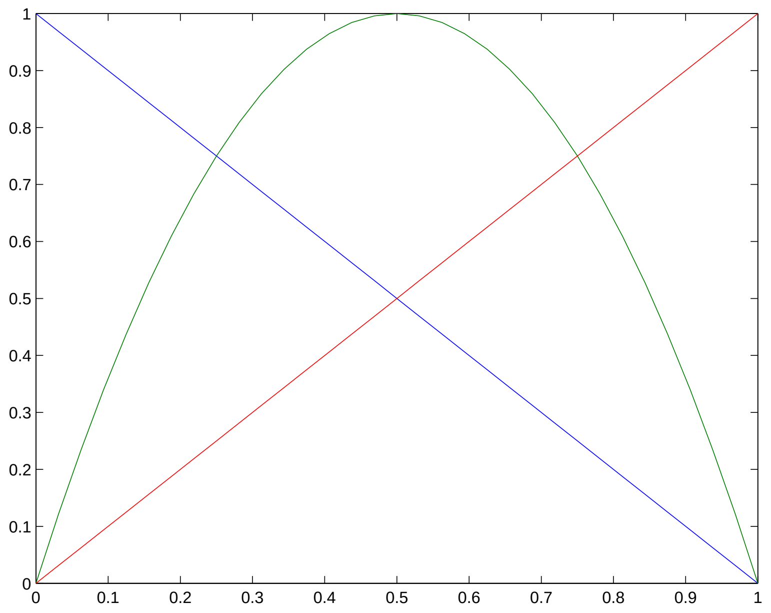


Figure 2.5.1: Quadratic hierarchical shape on $[x_{j-1}, x_j]$. When comparing with (2.5.1), set $x_{j-1} = 0$ and $x_j = 1$.

(2.4.6) is again used to map an arbitrary element $[x_{j-1}, x_j]$ onto $-1 \leq \xi \leq 1$. The vertex nodes at $\xi = -1$ and 1 are associated with the linear shape functions and, for simplicity, we will index them as -1 and 1. The remaining $p-1$ shape functions are on the element interior. They need not be associated with any nodes but, for convenience, we will associate all of them with a single node indexed by 0 at the center ($\xi = 0$) of the element. The restriction of the finite element solution $U(\xi)$ to the canonical element has the form

$$U(\xi) = c_{-1}N_{-1}^1(\xi) + c_1 N_1^1(\xi) + \sum_{i=2}^{p} c_i N_0^i(\xi), \qquad \xi \in [-1, 1]. \tag{2.5.3}$$

(We have dropped the elemental index e on $N_{j,e}^i$ since we are only concerned with approximations on the canonical element.) The vertex shape functions N_{-1}^1 and N_1^1 are the hat function segments (2.4.10) on the canonical element

$$N_{-1}^1(\xi) = \frac{1-\xi}{2}, \qquad N_1^1(\xi) = \frac{1+\xi}{2}, \qquad \xi \in [-1, 1]. \tag{2.5.4}$$

Once again, the higher-degree shape functions $N_0^i(\xi)$, $i = 2, 3, \ldots, p$, are required to have the proper degree and vanish at the element's ends $\xi = -1, 1$ to maintain continuity. Any normalization is arbitrary and may be chosen to satisfy a specified condition, *e.g.*, $N_0^2(0) = 1$. We use a normalization of Szabó and Babuška [7] which relies on Legendre polynomials. The Legendre polynomial $P_i(\xi)$, $i \geq 0$, is a polynomial of degree i in ξ satisfying [1]:

1. the differential equation

$$(1-\xi^2)P_i'' - 2\xi P_i' + i(i+1)P_i = 0, \qquad -1 < \xi < 1, \qquad i \geq 0; \tag{2.5.5a}$$

2. the normalization

$$P_i(1) = 1, \qquad i \geq 0; \tag{2.5.5b}$$

3. the orthogonality relation

$$\int_{-1}^{1} P_i(\xi)P_j(\xi)d\xi = \frac{2}{2i+1}\left\{\begin{array}{ll} 1, & \text{if } i=j \\ 0, & \text{otherwise} \end{array}\right. ; \tag{2.5.5c}$$

4. the symmetry condition

$$P_i(-\xi) = (-1)^i P_i(\xi), \qquad i \geq 0; \tag{2.5.5d}$$

5. the recurrence relation

$$(i+1)P_{i+1}(\xi) = (2i+1)\xi P_i(\xi) - iP_{i-1}(\xi), \qquad i \geq 1; \tag{2.5.5e}$$

and

6. the differentiation formula

$$P'_{i+1}(\xi) = (2i+1)P_i(\xi) + P'_{i-1}(\xi), \qquad i \geq 1. \tag{2.5.5f}$$

The first six Legendre polynomials are

$$\begin{aligned} P_0(\xi) = 1, \qquad & P_1(\xi) = \xi, \\ P_2(\xi) = \frac{3\xi^2-1}{2}, \qquad & P_3(\xi) = \frac{5\xi^3-3\xi}{2}, \\ P_4(\xi) = \frac{35\xi^4-30\xi^2+3}{2}, \qquad & P_5(\xi) = \frac{63\xi^5-70\xi^3+15\xi}{8}. \end{aligned} \tag{2.5.6}$$

With these preliminaries, we define the shape functions

$$N_0^i(\xi) = \sqrt{\frac{2i-1}{2}}\int_{-1}^{\xi} P_{i-1}(\sigma)d\sigma, \qquad i \geq 2. \tag{2.5.7a}$$

Using (2.5.5d,f), we readily show that

$$N_0^i(\xi) = \frac{P_i(\xi) - P_{i-2}(\xi)}{\sqrt{2(2i-1)}}, \qquad i \geq 2. \tag{2.5.7b}$$

Use of the normalization and symmetry properties (2.5.5b,d) further reveal that

$$N_0^i(-1) = N_0^i(1) = 0, \qquad i \geq 2, \tag{2.5.7c}$$

and use of the orthogonality property (2.5.5c) indicates that

$$\int_{-1}^{1} \frac{dN_0^i(\xi)}{d\xi}\frac{dN_0^j(\xi)}{d\xi}d\xi = \delta_{ij}, \qquad i, j \geq 2. \tag{2.5.7d}$$

Substituting (2.5.6) into (2.5.7b) gives

$$N_0^2(\xi) = \frac{3}{2\sqrt{6}}(\xi^2 - 1), \qquad N_0^3(\xi) = \frac{5}{2\sqrt{10}}\xi(\xi^2 - 1),$$
$$N_0^4(\xi) = \frac{7}{8\sqrt{14}}(5\xi^4 - 6\xi^2 + 1), \qquad N_0^5(\xi) = \frac{9}{8\sqrt{18}}(7\xi^5 - 10\xi^3 + 3\xi). \tag{2.5.8}$$

Shape functions $N_0^i(\xi)$, $i = 2, 3, \ldots, 6$, are shown in Figure 2.5.2.

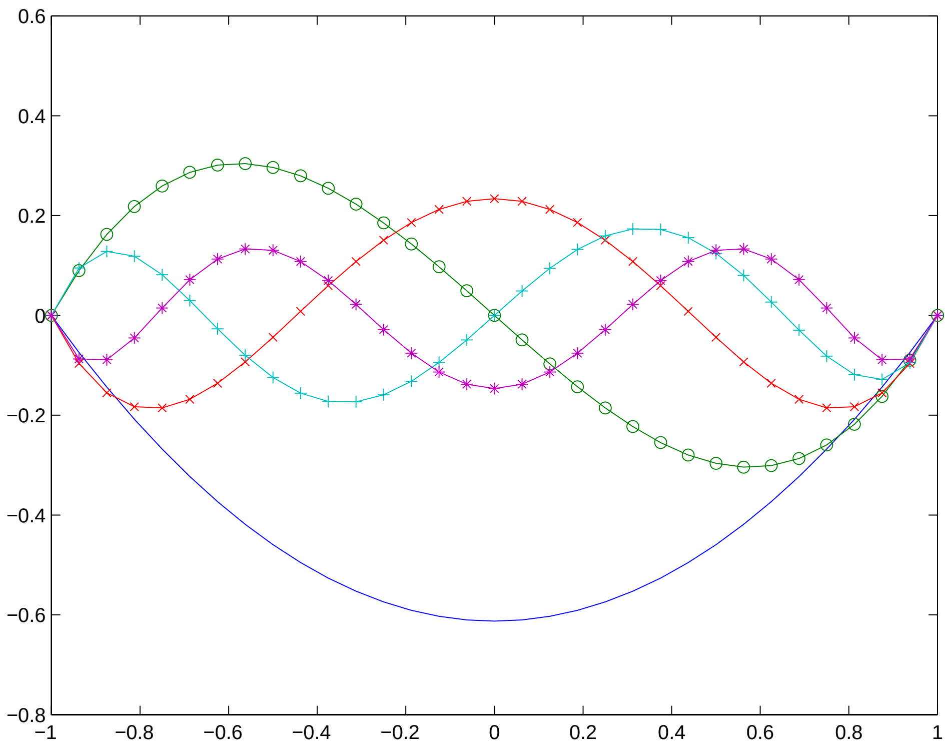


Figure 2.5.2: One-dimensional hierarchical shape functions of degrees 2 (solid), 3($\circ$), 4 ($\times$), 5 (+), and 6 (*) on the canonical element $-1 \leq \xi \leq 1$.

The representation (2.5.3) with use of (2.5.5b,d) reveals that the parameters c_{-1} and c_1 correspond to the values of $U(-1)$ and $U(1)$, respectively; however, the remaining parameters c_i, $i \geq 2$, do not correspond to solution values. In particular, using (2.5.3), (2.5.5d), and (2.5.7b) yields

$$U(0) = \frac{c_{-1} + c_1}{2} + \sum_{i=2,4}^{p} c_i N_0^i(0).$$

Hierarchical bases can be constructed so that c_i is proportional to $d^iU(0)/d\xi^i$, $i \geq 2$ (*cf.* [3], Section 2.8); however, the shape functions (2.5.8) based on Legendre polynomials reduce sensitivity of the basis to round-off error accumulation. This is very important when using high-order finite element approximations.

Example 2.5.1. Let us solve the two-point boundary value problem

$$-pu'' + qu = f(x), \qquad 0 < x < 1, \qquad u(0) = u(1) = 0, \tag{2.5.9}$$

using the finite element method with piecewise-quadratic hierarchical approximations. As in Chapter 1, we simplify matters by assuming that $p > 0$ and $q \geq 0$ are constants.

By now we are aware that the Galerkin form of this problem is given by (2.2.2). As in Chapter 1, introduce (*cf.* (1.3.9))

$$A_j^S(v, u) = \int_{x_{j-1}}^{x_j} pv'u'dx.$$

We use (2.4.6) to map $[x_{j-1}, x_j]$ to the canonical $[-1, 1]$ element as

$$A_j^S(v, u) = \frac{2}{h_j} \int_{-1}^{1} p \frac{dv}{d\xi} \frac{du}{d\xi} d\xi. \tag{2.5.10}$$

Using (2.5.3), we write the restriction of the piecewise-quadratic trial and text functions to $[x_{j-1}, x_j]$ as

$$U(\xi) = [c_{j-1}, c_j, c_{j-1/2}] \begin{bmatrix} N_{-1}^1 \\ N_1^1 \\ N_0^2 \end{bmatrix}, \qquad V(\xi) = [d_{j-1}, d_j, d_{j-1/2}] \begin{bmatrix} N_{-1}^1 \\ N_1^1 \\ N_0^2 \end{bmatrix}. \tag{2.5.11}$$

Substituting (2.5.11) into (2.5.10)

$$A_j^S(V, U) = [d_{j-1}, d_j, d_{j-1/2}] \mathbf{K}_j \begin{bmatrix} c_{j-1} \\ c_j \\ c_{j-1/2} \end{bmatrix} \tag{2.5.12a}$$

where $\mathbf{K}_j$ is the element stiffness matrix

$$\mathbf{K}_j = \frac{2p}{h_j} \int_{-1}^{1} \frac{d}{d\xi} \begin{bmatrix} N_{-1}^1 \\ N_1^1 \\ N_0^2 \end{bmatrix} \frac{d}{d\xi} [N_{-1}^1, N_1^1, N_0^2] d\xi.$$

Substituting for the basis definitions (2.5.4, 2.5.8)

$$\mathbf{K}_j = \frac{2p}{h_j}\int_{-1}^{1}\begin{bmatrix} -1/2 \\ 1/2 \\ \xi\sqrt{\frac{3}{2}} \end{bmatrix}[-1/2, 1/2, \xi\sqrt{\frac{3}{2}}]d\xi.$$

Integrating

$$\mathbf{K}_j = \frac{2p}{h_j}\int_{-1}^{1}\begin{bmatrix} 1/4 & -1/4 & -\xi\sqrt{3/8} \\ -1/4 & 1/4 & \xi\sqrt{3/8} \\ -\xi\sqrt{3/8} & \xi\sqrt{3/8} & 3\xi^2/2 \end{bmatrix}d\xi = \frac{p}{h_j}\begin{bmatrix} 1 & -1 & 0 \\ -1 & 1 & 0 \\ 0 & 0 & 2 \end{bmatrix}. \qquad (2.5.12b)$$

The orthogonality relation (2.5.7d) has simplified the stiffness matrix by uncoupling the linear and quadratic modes.

In a similar manner,

$$A_j^M(V,U) = \int_{x_{j-1}}^{x_j} qVU dx = \frac{qh_j}{2}\int_{-1}^{1} VU d\xi. \qquad (2.5.13a)$$

Using (2.5.11)

$$A_j^M(V,U) = [d_{j-1}, d_j, d_{j-1/2}]\mathbf{M}_j \begin{bmatrix} c_{j-1} \\ c_j \\ c_{j-1/2} \end{bmatrix} \qquad (2.5.13b)$$

where, upon use of (2.5.4, 2.5.8), the element mass matrix $\mathbf{M}_j$ satisfies

$$\mathbf{M}_j = \frac{qh_j}{2}\int_{-1}^{1}\begin{bmatrix} N_{-1}^1 \\ N_1^1 \\ N_0^2 \end{bmatrix}[N_{-1}^1, N_1^1, N_0^2]d\xi = \frac{qh_j}{6}\begin{bmatrix} 2 & 1 & -\sqrt{3/2} \\ 1 & 2 & -\sqrt{3/2} \\ -\sqrt{3/2} & -\sqrt{3/2} & 6/5 \end{bmatrix}. \qquad (2.5.13c)$$

The higher and lower order terms of the element mass matrix have not decoupled. Comparing (2.5.12b) and (2.5.13c) with the forms developed in Section 1.3 for piecewise-linear approximations, we see that the piecewise linear stiffness and mass matrices are contained as the upper 2×2 portions of these matrices. This will be the case for linear problems; thus, each higher-degree polynomial will add a "border" to the lower-degree stiffness and mass matrices.

Finally, consider

$$(V,f)_j = \int_{x_{j-1}}^{x_j} Vf dx = \frac{h_j}{2}\int_{-1}^{1} Vf d\xi. \qquad (2.5.14a)$$

Using (2.5.11)

$$(V,f)_j = [d_{j-1}, d_j, d_{j-1/2}]\mathbf{l}_j \qquad (2.5.14b)$$

where

$$\mathbf{l}_j = \frac{h_j}{2}\int_{-1}^{1} \begin{bmatrix} N_{-1}^1 \\ N_1^1 \\ N_0^2 \end{bmatrix} f(x(\xi))d\xi. \qquad (2.5.14c)$$

As in Section 1.3, we approximate $f(x)$ by piecewise-linear interpolation, which we write as

$$f(x) \approx N_{-1}^1(\xi)f_{j-1} + N_1^1(\xi)f_j$$

with $f_j := f(x_j)$. The manner of approximating $f(x)$ should clearly be related to the degree p and we will need a more careful analysis. Postponing this until Chapters 6 and 7, we have

$$\mathbf{l}_j = \frac{h_j}{2}\int_{-1}^{1} \begin{bmatrix} N_{-1}^1 \\ N_1^1 \\ N_0^2 \end{bmatrix} [N_{-1}^1, N_1^1]d\xi \begin{bmatrix} f_{j-1} \\ f_j \end{bmatrix} = \frac{h_j}{6}\begin{bmatrix} 2f_{j-1}+f_j \\ f_{j-1}+2f_j \\ -\sqrt{3/2}(f_{j-1}+f_j) \end{bmatrix} \qquad (2.5.14d)$$

Using (2.2.2a) with (2.5.12a), (2.5.13a), and (2.5.14a), we see that assembly requires evluating the sum

$$\sum_{j=1}^{N}[A_j^S(V,U) + A_j^M(V,U) - (V,f)_j] = 0.$$

Following the strategy used for the piecewise-linear solution of Section 1.3, the local stiffness and mass matrices and load vectors are added into their proper locations in their global counterparts. Imposing the condition that the system be satisfied for all choices of d_j, $j = 1/2, 1, 3/2, \ldots, N-1$, yields the linear algebraic system

$$(\mathbf{K}+\mathbf{M})\mathbf{c} = \mathbf{l}. \qquad (2.5.15)$$

The structure of the stiffness and mass matrices $\mathbf{K}$ and $\mathbf{M}$ and load vector $\mathbf{l}$ depend on the ordering of the unknowns $\mathbf{c}$ and virtual coordinates $\mathbf{d}$. One possibility is to order them by increasing index, *i.e.*,

$$\mathbf{c} = [c_{1/2}, c_1, c_{3/2}, c_2, \ldots, c_{N-1}, c_{N-1/2}]^T. \qquad (2.5.16)$$

As with the piecewise-linear basis, we have assumed that the homogeneous boundary conditions have explicitly eliminated $c_0 = c_N = 0$. Assembly for this ordering is similar to the one used in Section 1.3 (*cf.* Problem 2 at the end of this section). This is a natural ordering and the one most used for this approximation; however, for variety, let us order the unknowns by listing the vertices first followed by those at element midpoints, *i.e.*,

$$\mathbf{c} = \begin{bmatrix} \mathbf{c}_L \\ \mathbf{c}_Q \end{bmatrix}, \qquad \mathbf{c}_L = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_{N-1} \end{bmatrix}, \qquad \mathbf{c}_Q = \begin{bmatrix} c_{1/2} \\ c_{3/2} \\ \vdots \\ c_{N-1/2} \end{bmatrix}. \qquad (2.5.17)$$

In this case, $\mathbf{K}$, $\mathbf{M}$, and $\mathbf{l}$ have a block structure and may be partitioned as

$$\mathbf{K} = \begin{bmatrix} \mathbf{K}_L & \mathbf{0} \\ \mathbf{0} & \mathbf{K}_Q \end{bmatrix}, \qquad \mathbf{M} = \begin{bmatrix} \mathbf{M}_L & \mathbf{M}_{LQ} \\ \mathbf{M}_{LQ}^T & \mathbf{M}_Q \end{bmatrix}, \qquad \mathbf{l} = \begin{bmatrix} \mathbf{l}_L \\ \mathbf{l}_Q \end{bmatrix} \tag{2.5.18}$$

where, for uniform mesh spacing $h_j = h$, $j = 1, 2, \ldots, N$, these matrices are

$$\mathbf{K}_L = \frac{p}{h}\begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{bmatrix}, \qquad \mathbf{K}_Q = \frac{p}{h}\begin{bmatrix} 2 & & & & \\ & 2 & & & \\ & & \ddots & & \\ & & & 2 & \\ & & & & 2 \end{bmatrix}, \tag{2.5.19}$$

$$\mathbf{M}_L = \frac{qh}{6}\begin{bmatrix} 4 & 1 & & & \\ 1 & 4 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & 4 & 1 \\ & & & 1 & 4 \end{bmatrix}, \qquad \mathbf{M}_{LQ} = -\frac{qh}{6}\sqrt{\frac{3}{2}}\begin{bmatrix} 1 & 1 & & & & \\ & 1 & 1 & & & \\ & & \ddots & \ddots & & \\ & & & 1 & 1 & \\ & & & & 1 & 1 \end{bmatrix},$$

$$\mathbf{M}_Q = \frac{qh}{5}\begin{bmatrix} 1 & & & & \\ & 1 & & & \\ & & \ddots & & \\ & & & 1 & \\ & & & & 1 \end{bmatrix}, \tag{2.5.20}$$

$$\mathbf{l}_L = \frac{h}{6}\begin{bmatrix} f_0 + 4f_1 + f_2 \\ f_1 + 4f_2 + f_3 \\ \vdots \\ f_{N-2} + 4f_{N-1} + f_N \end{bmatrix}, \qquad \mathbf{l}_Q = -\frac{h}{\sqrt{24}}\begin{bmatrix} f_0 + f_1 \\ f_1 + f_2 \\ \vdots \\ f_{N-1} + f_N \end{bmatrix}. \tag{2.5.21}$$

With $N-1$ vertex unknowns $\mathbf{c}_L$ and N elemental unknowns $\mathbf{c}_Q$, the matrices $\mathbf{K}_L$ and $\mathbf{M}_L$ are $(N-1) \times (N-1)$, $\mathbf{K}_Q$ and $\mathbf{M}_Q$ are $N \times N$, and $\mathbf{M}_{LQ}$ is $(N-1) \times N$. Similarly, $\mathbf{l}_L$ and $\mathbf{l}_Q$ have dimension $N-1$ and N, respectively. The indicated ordering implies that the 3×3 element stiffness and mass matrices (2.5.12b) and (2.5.13c) for element j are added to rows and columns $j-1$, j, and $N-1+j$ of their global counterparts. The first row and column of the element stiffness and mass matrices are deleted when $j = 1$ to satisfy the left boundary condition. Likewise, the second row and column of these matrices are deleted when $j = N$ to satisfy the right boundary condition.

The structure of the system matrix $\mathbf{K} + \mathbf{M}$ is

$$\mathbf{K} + \mathbf{M} = \begin{bmatrix} \mathbf{K}_L + \mathbf{M}_L & \mathbf{M}_{LQ} \\ \mathbf{M}_{LQ}^T & \mathbf{K}_Q + \mathbf{M}_Q \end{bmatrix}. \tag{2.5.22}$$

The matrix $\mathbf{K}_L + \mathbf{M}_L$ is the same one used for the piecewise-linear solution of this problem in Section 1.3. Thus, an assembly and factorization of this matrix done during a prior piecewise-linear finite element analysis could be reused. A solution procedure using this factorization is presented as Problem 3 at the end of this section. Furthermore, if $q \equiv 0$ then $\mathbf{M}_{LQ} = 0$ (*cf.* (2.5.20b)) and the linear and quadratic portions of the system uncouple.

In Example 1.3.1, we solved (2.5.9) with $p = 1$, $q = 1$, and $f(x) = x$ using piecewise-linear finite elements. Let us solve this problem again using piecewise-quadratic hierarchical approximations and compare the results. Recall that the exact solution of this problem is

$$u(x) = x - \frac{\sinh x}{\sinh 1}.$$

Results for the error in the L^2 norm are shown in Table 2.5.1 for solutions obtained with piecewise-linear and quadratic approximations. The results indicate that solutions with piecewise-quadratic approximations are converging as $O(h^3)$ as opposed to $O(h^2)$ for piecewise-linear approximations. Subsequently, we shall show that smooth solutions generally converge as $O(h^{p+1})$ in the L^2 norm and as $O(h^p)$ in the strain energy (or H^1) norm.

N	Linear			Quadratic		
	DOF	$\|\|e\|\|_0$	$\|\|e\|\|_0/h^2$	DOF	$\|\|e\|\|_0$	$\|\|e\|\|_0/h^3$
4	3	0.265(-2)	0.425(-1)	7	0.126(-3)	0.807(-2)
8	7	0.656(-3)	0.426(-1)	15	0.158(-4)	0.809(-2)
16	15	0.167(-3)	0.427(-1)	31	0.198(-5)	0.809(-2)
32	31	0.417(-4)	0.427(-1)			

Table 2.5.1: Errors in L^2 and degrees of freedom (DOF) for piecewise-linear and piecewse-quadratic solutions of Example 2.5.1.

The number of elements N is not the only measure of computational complexity. With higher-order methods, the number of unknowns (degrees of freedom) provides a better index. Since the piecewise-quadratic solution has approximately twice the number of unknowns of the linear solution, we should compare the linear solution with spacing h and the quadratic solution with spacing $2h$. Even with this analysis, the superiority of the higher-order method in Table 2.5.1 is clear.

Problems

1. Consider the approximation in strain energy of a given function $u(\xi)$, $-1 < \xi < 1$, by a polynomial $U(\xi)$ in the hierarchical form (2.5.3). The problem consists of

determining $U(\xi)$ as the solution of the Galerkin problem

$$A(V, U) = A(V, u), \qquad \forall V \in S^p,$$

where S^p is a space of $p\,th$-degree polynomials on $[-1, 1]$. For simplicity, let us take the strain energy as

$$A(v, u) = \int_{-1}^{1} v_\xi u_\xi d\xi.$$

With $c_{-1} = u(-1)$ and $c_1 = u(1)$, find expressions for determining the remaining coefficients c_i, $i = 2, 3, \ldots, p$, so that the approximation satisfies the specified Galerkin projection.

2. Show how to generate the global stiffness and mass matrices and load vector for Example 2.5.1 when the equations and unknowns are written in order of increasing index (2.5.16).

3. Suppose $\mathbf{K}_L + \mathbf{M}_L$ have been assembled and factored by Gaussian elimination as part of a finite element analysis with piecewise-linear approximations. Devise an algorithm to solve (2.5.15) for $\mathbf{c}_L$ and $\mathbf{c}_Q$ that utilizes the given factorization.

2.6 Interpolation Errors

Errors of finite element solutions can be measured in several norms. We have already introduced pointwise and global metrics. In this introductory section on error analysis, we'll define some basic principles and study interpolation errors. As we shall see shortly, errors in interpolating a function u by a piecewise polynomial approximation U will provide bounds on the errors of finite element solutions.

Once again, consider a Galerkin problem for a second-order differential equation: find $u \in H_0^1$ such that

$$A(v, u) = (v, f), \qquad \forall v \in H_0^1. \tag{2.6.1}$$

Also consider its finite element counterpart: find $U \in S_0^N$ such that

$$A(V, U) = (V, f), \qquad \forall V \in S_0^N. \tag{2.6.2}$$

Let the approximating space $S_0^N \subset H_0^1$ consist of piecewise-polynomials of degree p on N-element meshes. We begin with two fundamental results regarding Galerkin's method and finite element approximations.

Theorem 2.6.1. *Let $u \in H_0^1$ and $U \in S_0^N \subset H_0^1$ satisfy (2.6.1) and (2.6.2), respectively, then*

$$A(V, u - U) = 0, \qquad \forall V \in S_0^N. \tag{2.6.3}$$

Proof. Since $V \in S_0^N$ it also belongs to H_0^1. Thus, it may be used to replace v in (2.6.1). Doing this and subtracting (2.6.2) yields the result. □

We shall subsequently show that the strain energy furnishes an inner product. With this interpretation, we may regard (2.6.3) as an orthogonality condition in a "strain energy space" where $A(v, u)$ is an inner product and $\sqrt{A(u, u)}$ is a norm. Thus, the finite element solution error

$$e(x) := u(x) - U(x) \tag{2.6.4}$$

is orthogonal in strain energy to all functions V in the subspace S_0^N. We use this orthogonality to show that solutions obtained by Galerkin's method are optimal in the sence of minimizing the error in strain energy.

Theorem 2.6.2. *Under the conditions of Theorem 2.6.1,*

$$A(u - U, u - U) = \min_{V \in S_0^N} A(u - V, u - V). \tag{2.6.5}$$

Proof. Consider

$$A(u - U, u - U) = A(u, u) - 2A(u, U) + A(U, U).$$

Use (2.6.3) with V replaced by U to write this as

$$A(u - U, u - U) = A(u, u) - 2A(u, U) + A(U, U) + 2A(u - U, U)$$

or

$$A(u - U, u - U) = A(u, u) - A(U, U).$$

Again, using (2.6.3) for any $V \in S_0^N$

$$A(u - U, u - U) = A(u, u) - A(U, U) + A(V, V) - A(V, V) - 2A(u - U, V)$$

or

$$A(u - U, u - U) = A(u - V, u - V) - A(U - V, U - V).$$

Since the last term on the right is non-negative, we can drop it to obtain

$$A(u - U, u - U) \leq A(u - V, u - V), \qquad \forall V \in S_0^N.$$

We see that equality is attained when $V = U$ and, thus, (2.6.5) is established. □

With optimality of Galerkin's method, we may obtain estimates of finite element discretization errors by bounding the right side of (2.6.5) for particular choices of V. Convenient bounds are obtained by selecting V to be an interpolant of the exact solution u. Bounds furnished in this manner generally provide the exact order of convergence in the mesh spacing h. Furthermore, results similar to (2.6.5) may be obtained in other norms. They are rarely as precise as those in strain energy and typically indicate that the finite element solution differs by no more than a constant from the optimal solution in the considered norm.

Thus, we will study the errors associated with interpolation problems. This can be done either on a physical or a canonical element, but we will proceed using a canonical element since we constructed shape functions in this manner. For our present purposes, we regard $u(\xi)$ as a known function that is interpolated by a p *th*-degree polynomial $U(\xi)$ on the canonical element $[-1, 1]$. Any form of the interpolating polynomial may be used. We use the Lagrange form (2.4.8), where

$$U(\xi) = \sum_{k=0}^{p} c_k N_k(\xi) \tag{2.6.6}$$

with $N_k(\xi)$ given by (2.4.7b). (We have omitted the elemental index e on N_k for clarity since we are concerned with one element.) An analysis of interpolation errors whith hierarchical shape functions may also be done (*cf.* Problem 1 at the end of this section). Although the Lagrangian and hierarchical shape functions differ, the resulting interpolation polynomials $U(\xi)$ and their errors are the same since the interpolation problem has a unique solution [2, 6].

Selecting $p+1$ distinct points $xi_i \in [-1, 1]$, $i = 0, 1, \ldots, p$, the interpolation conditions are

$$U(\xi_i) = u(\xi_i) := u_i = c_i, \qquad j = 0, 1, \ldots, p, \tag{2.6.7}$$

where the rightmost condition follows from (2.4.7a).

There are many estimates of pointwise interpolation errors. Here is a typical result.

Theorem 2.6.3. *Let $u(\xi) \in C^{p+1}[-1, 1]$ then, for each $\xi \in [-1, 1]$, there exists a point $\zeta(\xi) \in (-1, 1)$ such that the error in interpolating $u(\xi)$ by a p th-degree polynomial $U(\xi)$ is*

$$e(\xi) = \frac{u^{(p+1)}(\zeta)}{(p+1)!} \prod_{i=0}^{p} (\xi - \xi_i). \tag{2.6.8}$$

Proof. Although the proof is not difficult, we'll just sketch the essential details. A complete analysis is given in numerical analysis texts such as Burden and Faires [2], Chapter 3, and Isaacson and Keller [6], Chapter 5.

Since

$$e(\xi_0) = e(\xi_1) = \ldots = e(\xi_p) = 0$$

the error must have the form

$$e(\xi) = g(\xi) \prod_{i=0}^{p} (\xi - \xi_i).$$

The error in interpolating a polynomial of degree p or less is zero; thus, $g(\xi)$ must be proportional to $u^{(p+1)}$. We may use a Taylor's series argument to infer the existence of $\zeta(\xi) \in (-1, 1)$ and

$$e(\xi) = C u^{(p+1)}(\zeta) \prod_{i=0}^{p} (\xi - \xi_i).$$

Selecting u to be a polynomial of degree $p+1$ and differentiating this expression $p+1$ times yields C as $1/(p+1)!$ and (2.6.8). □

The pointwise error (2.6.8) can be used to obtain a variety of global error estimates. Let us estimate the error when interpolating a smooth function $u(\xi)$ by a linear polynomial $U(\xi)$ at the vertices $\xi_0 = -1$ and $\xi_1 = 1$ of an element. Using (2.6.8) with $p = 1$ reveals

$$e(\xi) = \frac{u''(\zeta)}{2}(\xi + 1)(\xi - 1), \qquad \xi \in (-1, 1). \tag{2.6.9}$$

Thus,

$$|e(\xi)| \le \frac{1}{2} \max_{-1 \le \xi \le 1} |u''(\xi)| \max_{-1 \le \xi \le 1} |\xi^2 - 1|.$$

Now,

$$\max_{-1 \le \xi \le 1} |\xi^2 - 1| = 1.$$

Thus,

$$|e(\xi)| \le \frac{1}{2} \max_{-1 \le \xi \le 1} |u''(\xi)|.$$

Derivatives in this expression are taken with respect to ξ. In most cases, we would like results expressed in physical terms. The linear transformation (2.4.6) provides the necessary conversion from the canonical element to element j: $[x_{j-1}, x_j]$. Thus,

$$\frac{d^2 u(\xi)}{d\xi^2} = \frac{h_j^2}{4} \frac{d^2 u(\xi)}{dx^2}$$

with $h_j = x_j - x_{j-1}$. Letting

$$\|f(\cdot)\|_{\infty,j} := \max_{x_{j-1} \le x \le x_j} |f(x)| \tag{2.6.10}$$

denote the local "maximum norm" of $f(x)$ on $[x_{j-1}, x_j]$, we have

$$\|e(\cdot)\|_{\infty,j} \leq \frac{h_j^2}{8}\|u''(\cdot)\|_{\infty,j}. \tag{2.6.11}$$

(Arguments have been replaced by a $\cdot$ to emphasize that the actual norm doesn't depend on x.)

If $u(x)$ were interpolated by a piecewise-linear function $U(x)$ on N elements $[x_{j-1}, x_j]$, $j = 1, 2, \ldots, N$, then (2.6.11) could be used on each element to obtain an estimate of the maximum error as

$$\|e(\cdot)\|_{\infty} \leq \frac{h^2}{8}\|u''(\cdot)\|_{\infty}, \tag{2.6.12a}$$

where

$$\|f(\cdot)\|_{\infty} := \max_{1\leq j\leq N} \|f(\cdot)\|_{\infty,j}, \tag{2.6.12b}$$

and

$$h := \max_{1\leq j\leq N}(x_j - x_{j-1}). \tag{2.6.12c}$$

As a next step, let us use (2.6.9) and (2.4.6) to compute an error estimate in the L^2 norm; thus,

$$\int_{x_{j-1}}^{x_j} e^2(x)dx = \frac{h_j}{2}\int_{-1}^{1}[\frac{u''(\zeta(\xi))}{2}(\xi^2-1)]^2 d\xi.$$

Since $|\xi^2 - 1| \leq 1$, we have

$$\int_{x_{j-1}}^{x_j} e^2(x)dx \leq \frac{h_j}{8}\int_{-1}^{1}[u''(\zeta(\xi))]^2 d\xi.$$

Introduce the "local L^2 norm" of a function $f(x)$ as

$$\|f(\cdot)\|_{0,j} := \left(\int_{x_{j-1}}^{x_j} f^2(x)dx\right)^{1/2}. \tag{2.6.13}$$

Then,

$$\|e(\cdot)\|_{0,j}^2 \leq \frac{h_j}{8}\int_{-1}^{1}[u''(\zeta(\xi))]^2 d\xi,$$

It is tempting to replace the integral on the right side of our error estimate by $\|u''\|_{0,j}^2$. This is almost correct; however, $\zeta = \zeta(\xi)$. We would have to verify that ζ varies smoothly with ξ. Here, we will assume this to be the case and expand u'' using Taylor's theorem to obtain

$$u''(\zeta) = u''(\xi) + u'''(\theta)(\zeta - \xi) = u''(\xi) + O(|\zeta - \xi|), \qquad \theta \in (\xi, \zeta),$$

or

$$|u''(\zeta)| \leq C|u''(\xi)|.$$

The constant C absorbs our careless treatment of the higher-order term in the Taylor's expansion. Thus, using (2.4.6), we have

$$\|e(\cdot)\|_{0,j}^2 \leq C^2 \frac{h_j}{8} \int_{-1}^{1} [u''(\xi)]^2 d\xi = C^2 \frac{h_j^4}{64} \int_{x_{j-1}}^{x_j} [u''(x)]^2 dx,$$

where derivatives in the rightmost expression are with respect to x. Using (2.6.13)

$$\|e(\cdot)\|_{0,j}^2 \leq C^2 \frac{h_j^4}{64} \|u''(\cdot)\|_{0,j}^2. \tag{2.6.14}$$

If we sum (2.6.14) over the N finite elements of the mesh and take a square root we obtain

$$\|e(\cdot)\|_0 \leq Ch^2 \|u''(\cdot)\|_0, \tag{2.6.15a}$$

where

$$\|f(\cdot)\|_0^2 = \sum_{j=1}^{N} \|f(\cdot)\|_{0,j}^2. \tag{2.6.15b}$$

(The constant C in (2.6.15a) replaces the constant $C/8$ of (2.6.14), but we won't be precise about identifying different constants.)

With a goal of estimating the error in H^1, let us examine the error $u'(\xi) - U'(\xi)$. Differentiating (2.6.9) with respect to ξ

$$e'(\xi) = u''(\zeta)\xi + \frac{u'''(\zeta)}{2} \frac{d\zeta}{d\xi}(\xi^2 - 1).$$

Assuming that $d\zeta/d\xi$ is bounded, we use (2.6.13) and (2.4.6) to obtain

$$\|e'\|_{0,j}^2 = \int_{x_{j-1}}^{x_j} [\frac{de(x)}{dx}]^2 dx = \frac{2}{h_j} \int_{-1}^{1} [u''(\zeta)\xi + \frac{u'''(\zeta)}{2} \frac{d\zeta}{d\xi}(\xi^2 - 1)]^2 d\xi.$$

Following the arguments that led to (2.6.14), we find

$$\|e'(\cdot)\|_{0,j}^2 \leq Ch_j^2 \|u''(\cdot)\|_{0,j}^2.$$

Summing over the N elements

$$\|e'(\cdot)\|_0^2 \leq Ch^2 \|u''(\cdot)\|_0. \tag{2.6.16}$$

To obtain an error estimate in the H^1 norm, we combine (2.6.15a) and (2.6.16) to get

$$\|e(\cdot)\|_1 \leq Ch\|u''(\cdot)\|_0 \tag{2.6.17a}$$

where

$$\|f(\cdot)\|_1^2 := \sum_{j=1}^{N}[\|f'(\cdot)\|_{0,j}^2 + \|f(\cdot)\|_{0,j}^2]. \tag{2.6.17b}$$

The methodology developed above may be applied to estimate interpolation errors of higher-degree polynomial approximations. A typical result follows.

Theorem 2.6.4. *Introduce a mesh $a \leq x_0 < x_1 < \ldots < x_N \leq b$ such that $U(x)$ is a polynomial of degree p or less on every subinterval (x_{j-1}, x_j) and $U \in H^1(a,b)$. Let $U(x)$ interpolate $u(x) \in H^{p+1}[a,b]$ such that no error results when $u(x)$ is any polynomial of degree p or less. Then, there exists a constant $C_p > 0$, depending on p, such that*

$$\|u - U\|_0 \leq C_p h^{p+1}\|u^{(p+1)}\|_0 \tag{2.6.18a}$$

and

$$\|u - U\|_1 \leq Ch_p^p\|u^{(p+1)}\|_0. \tag{2.6.18b}$$

where h satisfies (2.6.12c).

Proof. The analysis follows the one used for linear polynomials. □

Problems

1. Choose a hierarchical polynomial (2.5.3) on a canonical element $[-1,1]$ and show how to determine the coefficients c_j, $j = -1, 1, 2, \ldots, p$, to solve the interpolation problem (2.6.7).

Bibliography

[1] M. Abromowitz and I.A. Stegun. *Handbook of Mathematical Functions*, volume 55 of *Applied Mathematics Series.* National Bureau of Standards, Gathersburg, 1964.

[2] R.L. Burden and J.D. Faires. *Numerical Analysis.* PWS-Kent, Boston, fifth edition, 1993.

[3] G.F. Carey and J.T. Oden. *Finite Elements: A Second Course*, volume II. Prentice Hall, Englewood Cliffs, 1983.

[4] R. Courant and D. Hilbert. *Methods of Mathematical Physics*, volume 1. Wiley-Interscience, New York, 1953.

[5] C. de Boor. *A Practical Guide to Splines.* Springer-Verlag, New York, 1978.

[6] E. Isaacson and H.B. Keller. *Analysis of Numerical Methods.* John Wiley and Sons, New York, 1966.

[7] B. Szabó and I. Babuška. *Finite Element Analysis.* John Wiley and Sons, New York, 1991.

Chapter 3

Multi-Dimensional Variational Principles

3.1 Galerkin's Method and Extremal Principles

The construction of Galerkin formulations presented in Chapters 1 and 2 for one-dimensional problems readily extends to higher dimensions. Following our prior developments, we'll focus on the model two-dimensional self-adjoint diffusion problem

$$\mathcal{L}[u] = -(p(x,y)u_x)_x - (p(x,y)u_y)_y + q(x,y)u = f(x,y), \qquad (x,y) \in \Omega, \tag{3.1.1a}$$

where $\Omega \subset \Re^2$ with boundary $\partial\Omega$ (Figure 3.1.1) and $p(x,y) > 0$, $q(x,y) \geq 0$, $(x,y) \in \Omega$. Essential boundary conditions

$$u(x,y) = \alpha(x,y), \qquad (x,y) \in \partial\Omega_E, \tag{3.1.1b}$$

are prescribed on the portion $\partial\Omega_E$ of $\partial\Omega$ and natural boundary conditions

$$p(x,y)\frac{\partial u(x,y)}{\partial \mathbf{n}} = p\nabla u \cdot \mathbf{n} := p(u_x \cos\theta + u_y \sin\theta) = \beta(x,y), \qquad (x,y) \in \partial\Omega_N, \tag{3.1.1c}$$

are prescribed on the remaining portion $\partial\Omega_N$ of $\partial\Omega$. The angle θ is the angle between the x-axis and the outward normal $\mathbf{n}$ to $\partial\Omega$ (Figure 3.1.1).

The Galerkin form of (3.1.1) is obtained by multiplying (3.1.1a) by a test function v and integrating over Ω to obtain

$$\iint_{\Omega} v[-(pu_x)_x - (pu_y)_y + qu - f]dxdy = 0. \tag{3.1.2}$$

In order to integrate the second derivative terms by parts in two and three dimensions, we use Green's theorem or the divergence theorem

$$\iint_{\Omega} \nabla \cdot \mathbf{a} dxdy = \int_{\partial\Omega} \mathbf{a} \cdot \mathbf{n} ds \tag{3.1.3a}$$

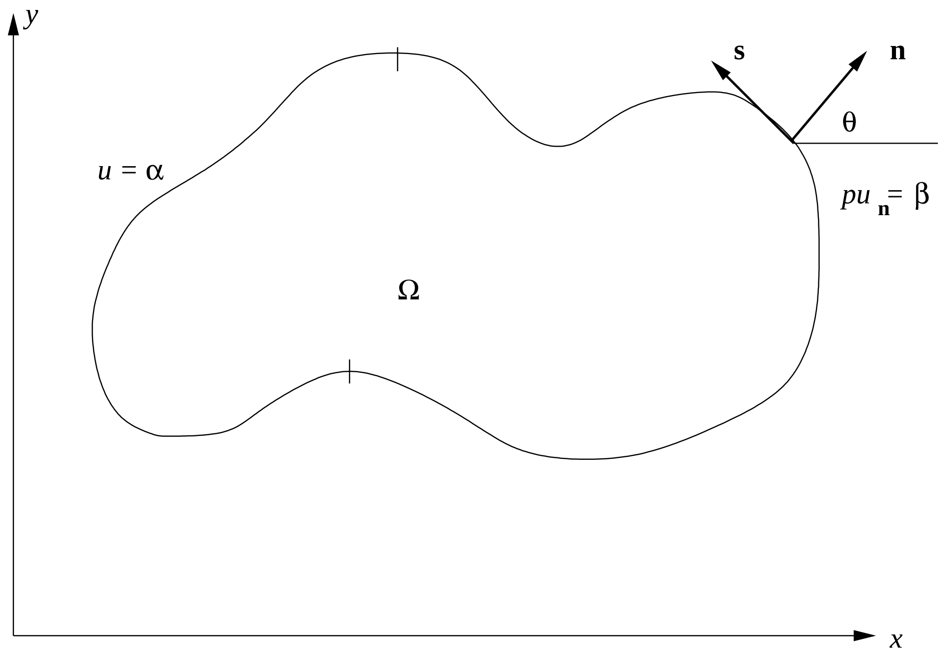


Figure 3.1.1: Two-dimensional region Ω with boundary $\partial\Omega$ and normal vector $\mathbf{n}$ to $\partial\Omega$.

where s is a coordinate on $\partial\Omega$, $\mathbf{a} = [a_1, a_2]^T$, and

$$\nabla \cdot \mathbf{a} = \frac{\partial a_1}{\partial x} + \frac{\partial a_2}{\partial y}. \tag{3.1.3b}$$

In order to use this result in the present circumstances, let us introduce vector notation

$$(pu_x)_x + (pu_y)_y := \nabla \cdot (p\nabla u)$$

and use the "product rule" for the divergence and gradient operators

$$\nabla \cdot (vp\nabla u) = (\nabla v) \cdot (p\nabla u) + v\nabla \cdot (p\nabla u). \tag{3.1.3c}$$

Thus,

$$\iint_{\Omega} -v\nabla \cdot (p\nabla u) dxdy = \iint_{\Omega} [(\nabla v) \cdot (p\nabla u) - \nabla \cdot (vp\nabla u)] dxdy.$$

Now apply the divergence theorem (3.1.3) to the second term to obtain

$$\iint_{\Omega} -v\nabla \cdot (p\nabla u) dxdy = \iint_{\Omega} \nabla v \cdot p\nabla u dxdy - \int_{\partial\Omega} vp\nabla u \cdot \mathbf{n} ds.$$

Thus, (3.1.2) becomes

$$\iint_{\Omega} [\nabla v \cdot p\nabla u + v(qu - f)] dxdy - \int_{\partial\Omega} vpu_{\mathbf{n}} ds = 0 \tag{3.1.4}$$

where (3.1.1c) was used to simplify the surface integral.

The integrals in (3.1.4) must exist and, with u and v of the same class and p and q smooth, this implies

$$\iint_\Omega (u_x^2 + u_y^2 + u^2)dxdy$$

exists. This is the two-dimensional Sobolev space H^1. Drawing upon our experiences in one dimension, we expect $u \in H_E^1$, where functions in H_E^1 are in H^1 and satisfy the Dirichlet boundary conditions (3.1.1b) on Ω_E. Likewise, we expect $v \in H_0^1$, which denotes that $v = 0$ on $\partial\Omega_E$. Thus, the variation v should vanish where the trial function u is prescribed.

Let us extend the one-dimensional notation as well. Thus, the L^2 inner product is

$$(v, f) := \iint_\Omega vf\,dxdy \tag{3.1.5a}$$

and the strain energy is

$$A(v, u) := (\nabla v, p\nabla u) + (v, qu) = \iint_\Omega [p(v_x u_x + v_y u_y) + qvu]dxdy. \tag{3.1.5b}$$

We also introduce a boundary L^2 inner product as

$$< v, w >= \int_{\partial\Omega_N} vwds. \tag{3.1.5c}$$

The boundary integral may be restricted to $\partial\Omega_N$ since $v = 0$ on $\partial\Omega_E$. With this nomenclature, the variational problem (3.1.4) may be stated as: find $u \in H_E^1$ satisfying

$$A(v, u) = (v, f)+ < v, \beta >, \qquad \forall v \in H_0^1. \tag{3.1.6}$$

The Neumann boundary condition (3.1.1c) was used to replace $pu_{\mathbf{n}}$ in the boundary inner product. The variational problem (3.1.6) has the same form as the one-dimensional problem (2.3.3). Indeed, the theory and extremal principles developed in Chapter 2 apply to multi-dimensional problems of this form.

Theorem 3.1.1. *The function $w \in H_E^1$ that minimizes*

$$I[w] = A(w, w) - 2(w, f) - 2 < w, \beta > . \tag{3.1.7}$$

is the one that satisfies (3.1.6), and conversely.

Proof. The proof is similar to that of Theorem 2.2.1 and appears as Problem 1 at the end of this section. □

Corollary 3.1.1. *Smooth functions $u \in H_E^1$ satisfying (3.1.6) or minimizing (3.1.7) also satisfy (3.1.1).*

Proof. Again, the proof is left as an exercise. □

Example 3.1.1. Suppose that the Neumann boundary conditions (3.1.1c) are changed to Robin boundary conditions

$$pu_{\mathbf{n}} + \gamma u = \beta, \qquad (x,y) \in \partial\Omega_N. \tag{3.1.8a}$$

Very little changes in the variational statement of the problem (3.1.1a,b), (3.1.8). Instead of replacing $pu_{\mathbf{n}}$ by β in the boundary inner product (3.1.5c), we replace it by $\beta - \gamma u$. Thus, the Galerkin form of the problem is: find $u \in H_E^1$ satisfying

$$A(v,u) = (v,f) + < v, \beta - \gamma u >, \qquad \forall v \in H_0^1. \tag{3.1.8b}$$

Example 3.1.2. Variational principles for nonlinear problems and vector systems of partial differential equations are constructed in the same manner as for the linear scalar problems (3.1.1). As an example, consider a thin elastic sheet occupying a two-dimensional region Ω. As shown in Figure 3.1.2, the Cartesian components (u_1, u_2) of the displacement vector vanish on the portion $\partial\Omega_E$ of of the boundary $\partial\Omega$ and the components of the traction are prescribed as (S_1, S_2) on the remaining portion $\partial\Omega_N$ of $\partial\Omega$.

The equations of equilibrium for such a problem are (*cf.*, *e.g.*, [6], Chapter 4)

$$\frac{\partial \sigma_{11}}{\partial x} + \frac{\partial \sigma_{12}}{\partial y} = 0, \tag{3.1.9a}$$

$$\frac{\partial \sigma_{12}}{\partial x} + \frac{\partial \sigma_{22}}{\partial y} = 0, \qquad (x,y) \in \Omega, \tag{3.1.9b}$$

where σ_{ij}, $i,j = 1,2$, are the components of the two-dimensional symmetric stress tensor (matrix). The stress components are related to the displacement components by Hooke's law

$$\sigma_{11} = \frac{E}{1-\nu^2}\left(\frac{\partial u_1}{\partial x} + \nu \frac{\partial u_2}{\partial y}\right), \tag{3.1.10a}$$

$$\sigma_{22} = \frac{E}{1-\nu^2}\left(\nu\frac{\partial u_1}{\partial x} + \frac{\partial u_2}{\partial y}\right), \tag{3.1.10b}$$

$$\sigma_{12} = \frac{E}{2(1+\nu)}\left(\frac{\partial u_1}{\partial y} + \frac{\partial u_2}{\partial x}\right), \tag{3.1.10c}$$

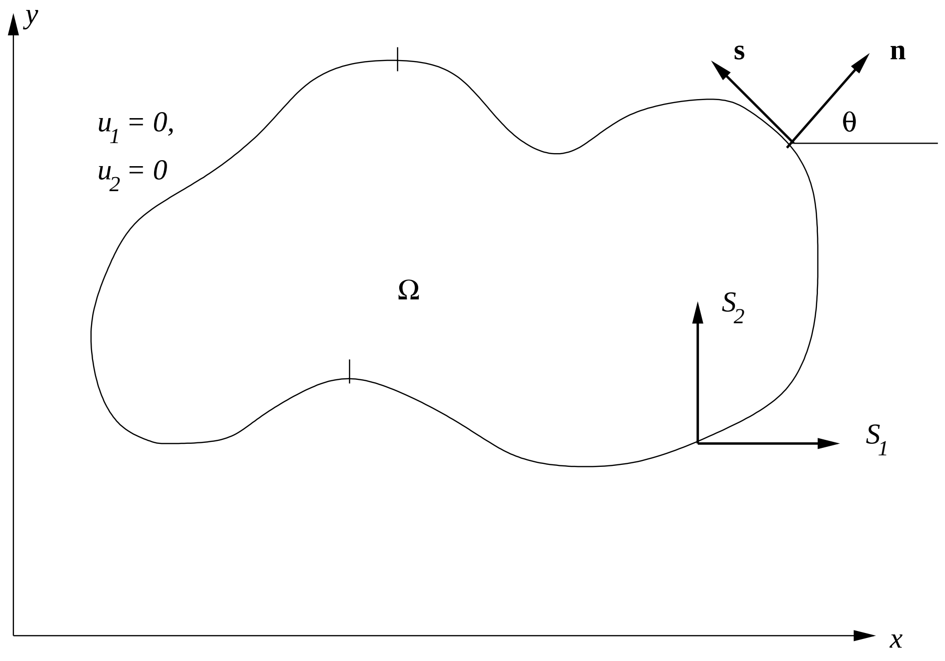


Figure 3.1.2: Two-dimensional elastic sheet occupying the region Ω. Displacement components (u_1, u_2) vanish on $\partial\Omega_E$ and traction components (S_1, S_2) are prescribed on $\partial\Omega_N$.

where E and ν are constants called Young's modulus and Poisson's ratio, respectively.

The displacement and traction boundary conditions are

$$u_1(x, y) = 0, \qquad u_2(x, y) = 0, \qquad (x, y) \in \partial\Omega_E, \tag{3.1.11a}$$

$$n_1\sigma_{11} + n_2\sigma_{12} = S_1, \qquad n_1\sigma_{12} + n_2\sigma_{22} = S_2, \qquad (x, y) \in \partial\Omega_N, \tag{3.1.11b}$$

where $\mathbf{n} = [n_1, n_2]^T = [\cos\theta, \sin\theta]^T$ is the unit outward normal vector to $\partial\Omega$ (Figure 3.1.2).

Following the one-dimensional formulations, the Galerkin form of this problem is obtained by multiplying (3.1.9a) and (3.1.9b) by test functions v_1 and v_2, respectively, integrated over Ω, and using the divergence theorem. With u_1 and u_2 being components of a displacement field, the functions v_1 and v_2 are referred to as components of the *virtual displacement field.*

We use (3.1.9a) to illustrate the process; thus, multiplying by v_1 and integrating over Ω, we find

$$\iint_{\Omega} v_1[\frac{\partial\sigma_{11}}{\partial x} + \frac{\partial\sigma_{12}}{\partial y}]dxdy = 0.$$

The three stress components are dependent on the two displacement components and are typically replaced by these using (3.1.10). Were this done, the variational principle would involve second derivatives of u_1 and u_2. Hence, we would want to use the divergence theorem to obtain a symmetric variational form and reduce the continuity requirements on u_1 and u_2. We'll do this, but omit the explicit substitution of (3.1.10) to simplify the presentation. Thus, we regard σ_{11} and σ_{12} as components of a two-vector, we use the divergence theorem (3.1.3) to obrain

$$\iint_{\Omega} [\frac{\partial v_1}{\partial x}\sigma_{11} + \frac{\partial v_1}{\partial y}\sigma_{12}]dxdy = \int_{\partial\Omega} v_1[n_1\sigma_{11} + n_2\sigma_{12}]ds.$$

Selecting $v_1 \in H_0^1$ implies that the boundary integral vanishes on $\partial\Omega_E$. This and the subsequent use of the natural boundary condition (3.1.11b) give

$$\iint_{\Omega} [\frac{\partial v_1}{\partial x}\sigma_{11} + \frac{\partial v_1}{\partial y}\sigma_{12}]dxdy = \int_{\partial\Omega_N} v_1 S_1 ds, \qquad \forall v_1 \in H_0^1. \tag{3.1.12a}$$

Similar treatment of (3.1.9b) gives

$$\iint_{\Omega} [\frac{\partial v_2}{\partial x}\sigma_{12} + \frac{\partial v_2}{\partial y}\sigma_{22}]dxdy = \int_{\partial\Omega_N} v_2 S_2 ds, \qquad \forall v_2 \in H_0^1. \tag{3.1.12b}$$

Equations (3.1.12a) and (3.1.12b) may be combined and written in a vector form. Letting $\mathbf{u} = [u_1, u_2]^T$, *etc.*, we add (3.1.12a) and (3.1.12b) to obtain the Galerkin problem: find $\mathbf{u} \in H_0^1$ such that

$$A(\mathbf{v}, \mathbf{u}) = < \mathbf{v}, \mathbf{S} >, \qquad \forall \mathbf{v} \in H_0^1, \tag{3.1.13a}$$

where

$$A(\mathbf{v}, \mathbf{u}) = \iint_{\Omega} [\frac{\partial v_1}{\partial x}\sigma_{11} + \frac{\partial v_2}{\partial y}\sigma_{22} + (\frac{\partial v_1}{\partial y} + \frac{\partial v_2}{\partial x})\sigma_{12}]dxdy, \tag{3.1.13b}$$

$$< \mathbf{v}, \mathbf{S} > = \int_{\partial\Omega_N} (v_1 S_1 + v_2 S_2)ds. \tag{3.1.13c}$$

When a vector function belongs to H^1, we mean that each of its components is in H^1. The spaces H_E^1 and H_0^1 are identical since the displacement is trivial on $\partial\Omega_E$.

The solution of (3.1.13) also satisfies the following minimum problem.

Theorem 3.1.2. *Among all functions* $\mathbf{w} = [w_1, w_2]^T \in H_E^1$ *the solution* $\mathbf{u} = [u_1, u_2]^T$ *of (3.1.13) is the one that minimizes*

$$I[\mathbf{w}] = \frac{E}{2(1-\nu^2)} \iint_{\Omega} \{(1-\nu)[(\frac{\partial w_1}{\partial x})^2 + (\frac{\partial w_2}{\partial y})^2] + \nu(\frac{\partial w_1}{\partial x} + \frac{\partial w_2}{\partial y})^2$$

$$+\frac{(1-\nu)}{2}(\frac{\partial w_1}{\partial y}+\frac{\partial w_2}{\partial x})^2\}dxdy-\int\limits_{\partial\Omega_N}(w_1S_1+w_2S_2)ds,$$

and conversely.

Proof. The proof is similar to that of Theorem 2.2.1. The stress components σ_{ij}, $i,j=1,2$, have been eliminated in favor of the displacements using (3.1.10). □

Let us conclude this section with a brief summary.

- A solution of the differential problem, *e.g.*, (3.1.1), is called a "classical" or "strong" solution. The function $u \in H^2_B$, where functions in H^2 have finite values of

  $$\iint\limits_{\Omega}[(u_{xx})^2+(u_{xy})^2+(u_{yy})^2+(u_x)^2+(u_y)^2+u^2]dxdy$$

  and functions in H^2_B also satisfy *all* prescribed boundary conditions, *e.g.*, (3.1.1b,c).

- Solutions of a Galerkin problem such as (3.1.6) are called "weak" solutions. They may be elements of a larger class of functions than strong solutions since the high-order derivatives are missing from the variational statement of the problem. For the second-order differential equations that we have been studying, the variational form (*e.g.*, (3.1.6)) only contains first derivatives and $u \in H^1_E$. Functions in H^1 have finite values of

  $$\iint\limits_{\Omega}[(u_x)^2+(u_y)^2+u^2]dxdy.$$

  and functions in H^1_E also satisfy the prescribed essential (Dirichlet) boundary condition (3.1.1b). Test functions v are not varied where essential data is prescribed and are elements of H^1_0. They satisfy trivial versions of the essential boundary conditions.

- While essential boundary conditions constrain the trial and test spaces, natural (Neumann or Robin) boundary conditions alter the variational statement of the problem. As with (3.1.6) and (3.1.13), inhomogeneous conditions add boundary inner product terms to the variational statement.

- Smooth solutions of the Galerkin problem satisfy the original partial differential equation(s) and natural boundary conditions, and conversely.

- Galerkin problems arising from self-adjoint differential equations also satisfy extremal problems. In this case, approximate solutions found by Galerkin's method are *best* in the sense of (2.6.5), *i.e.*, in the sense of minimizing the strain energy of the error.

Problems

1. Prove Theorem 3.1.1 and its Corollary.

2. Prove Theorem 3.1.2 and aslo show that smooth solutions of (3.1.13) satisfy the differential system (3.1.9) - (3.1.11).

3. Consider an infinite solid medium of material M containing an infinite number of periodically spaced circular cylindrical fibers made of material F. The fibers are arranged in a square array with centers two units apart in the x and y directions (Figure 3.1.3). The radius of each fiber is a (< 1). The aim of this problem is to find a Galerkin problem that can be used to determine the effective conductivity of the composite medium. Because of embedded symmetries, it suffices to solve a

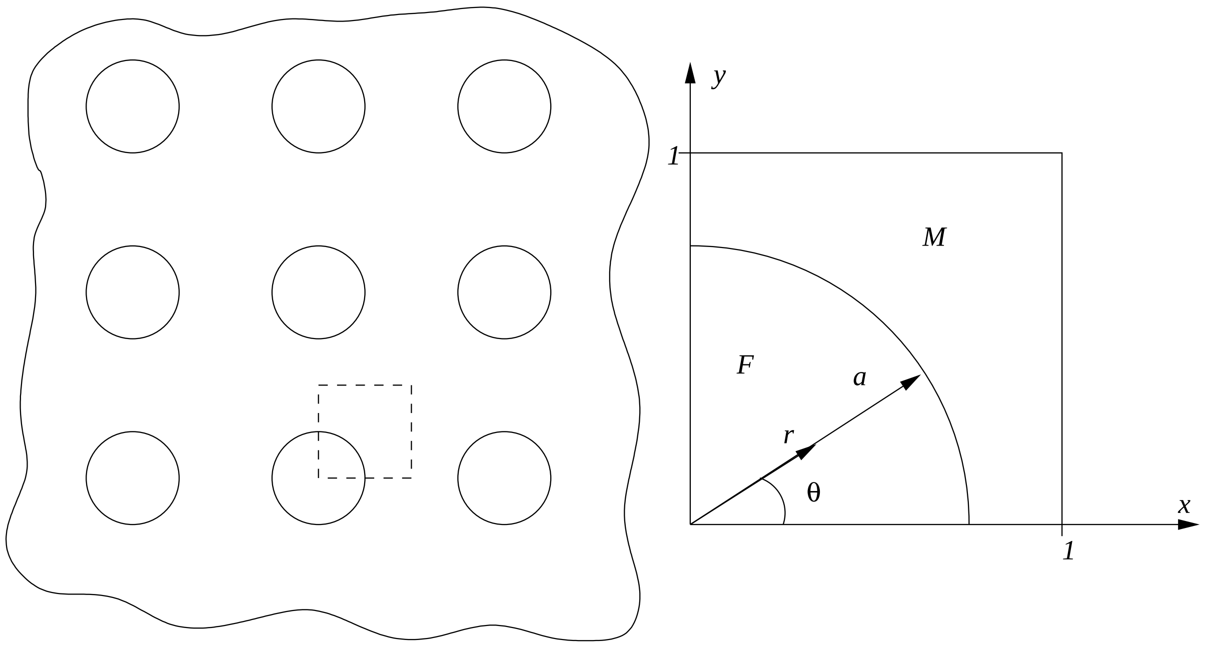


Figure 3.1.3: Composite medium consisting of a regular array of circular cylindrical fibers embedded in in a matrix (left). Quadrant of a Periodicity cell used to solve this problem (right).

problem on one quarter of a periodicity cell as shown on the right of Figure 3.1.3. The governing differential equations and boundary conditions for the temperature (or potential, *etc.*) $u(x,y)$ within this quadrant are

$$\begin{aligned} \nabla\cdot(p\nabla u) &= 0, && (x,y)\in\Omega_F\cup\Omega_M,\\ u_x(0,y) = u_x(1,y) &= 0, && 0\le y\le 1,\\ u(x,0)=0, \qquad u(x,1) &= 1, && 0\le x\le 1,\\ u\in C^0, \qquad pu_r &\in C^0, && (x,y)\in x^2+y^2=a^2. \end{aligned} \tag{3.1.14}$$

The subscripts F and M are used to indicate the regions and properties of the fiber and matrix, respectively. Thus, letting

$$\Omega := \{(x,y)|\ 0\le x\le 1,\ 0\le y\le 1\},$$

we have

$$\Omega_F := \{(r,\theta)|\ 0\le r\le a,\ 0\le\theta\le\pi/2\},$$

and

$$\Omega_M := \Omega - \Omega_F.$$

The conductivity p of the fiber and matrix will generally be different and, hence, p will jump at $r=a$. If necessary, we can write

$$p(x,y) = \begin{cases} p_F, & \text{if } x^2+y^2<a^2 \\ p_M, & \text{if } x^2+y^2>a^2 \end{cases}.$$

Although the conductivities are discontinuous, the last boundary condition confirms that the temperature u and flux pu_r are continuous at $r=a$.

3.1. Following the steps leading to (3.1.6), show that the Galerkin form of this problem consists of determining $u\in H_E^1$ as the solution of

$$\iint_{\Omega_F\cup\Omega_M} p(u_xv_x+u_yv_y)dxdy = 0, \qquad \forall v\in H_0^1.$$

Define the spaces H_E^1 and H_0^1 for this problem. The Galerkin problem appears to be the same as it would for a homogeneous medium. There is no indication of the continuity conditions at $r=a$.

3.2. Show that the function $w\in H_E^1$ that minimizes

$$I[w] = \iint_{\Omega_F\cup\Omega_M} p(w_x^2+w_y^2)dxdy$$

is the solution u of the Galerkin problem, and conversely. Again, there is little evidence that the problem involves an inhomogeneous medium.

3.2 Function Spaces and Approximation

Let us try to formalize some of the considerations that were raised about the properties of function spaces and their smoothness requirements. Consider a Galerkin problem in the form of (3.1.6). Using Galerkin's method, we find approximate solutions by solving (3.1.6) in a finite-dimensional subspace S^N of H^1. Selecting a basis $\{\phi_j\}_{j=1}^N$ for S^N, we consider approximations $U \in S_E^N$ of u in the form

$$U(x,y) = \sum_{j=1}^{N} c_j \phi_j(x,y). \tag{3.2.1}$$

With approximations $V \in S_0^N$ of v having a similar form, we determine U as the solution of

$$A(V,U) = (V,f) + < V, \beta >, \qquad \forall V \in S_0^N. \tag{3.2.2}$$

(Nontrivial essential boundary conditions introduce differences between S_E^N and S_0^N and we have not explicitly identified these differences in (3.2.2).)

We've mentioned the criticality of knowing the minimum smoothness requirements of an approximating space S^N. Smooth (*e.g.* C^1) approximations are difficult to construct on nonuniform two- and three-dimensional meshes. We have already seen that smoothness requirements of the solutions of partial differential equations are usually expressed in terms of Sobolev spaces, so let us define these spaces and examine some of their properties. First, let's review some preliminaries from linear algebra and functional analysis.

Definition 3.2.1. $\mathcal{V}$ is a *linear space* if

1. $u, v \in \mathcal{V}$ then $u + v \in \mathcal{V}$,

2. $u \in \mathcal{V}$ then $\alpha u \in \mathcal{V}$, for all constants α, and

3. $u, v \in \mathcal{V}$ then $\alpha u + \beta v \in \mathcal{V}$, for all constants α, β.

Definition 3.2.2. $A(u,v)$ is a *bilinear form* on $\mathcal{V} \times \mathcal{V}$ if, for $u, v, w \in \mathcal{V}$ and all constants α and β,

1. $A(u,v) \in \Re$, and

2. $A(u,v)$ is linear in each argument; thus,

$$A(u, \alpha v + \beta w) = \alpha A(u,v) + \beta A(u,w),$$

$$A(\alpha u + \beta v, w) = \alpha A(u,w) + \beta A(v,w).$$

Definition 3.2.3. An *inner product* $A(u,v)$ is a bilinear form on $\mathcal{V} \times \mathcal{V}$ that

1. is symmetric in the sense that $A(u,v) = A(v,u)$, $\forall u, v \in \mathcal{V}$, and
2. $A(u,u) > 0$, $u \neq 0$ and $A(0,0) = 0$, $\forall u \in \mathcal{V}$.

Definition 3.2.4. The *norm* $\|\cdot\|_A$ associated with the inner product $A(u,v)$ is

$$\|u\|_A = \sqrt{A(u,u)} \tag{3.2.3}$$

and it satisfies

1. $\|u\|_A > 0$, $u \neq 0$, $\|0\|_A = 0$,
2. $\|u+v\|_A \leq \|u\|_A + \|v\|_A$, and
3. $\|\alpha u\|_A = |\alpha| \|u\|_A$, for all constants α.

The integrals involved in the norms and inner products are Lebesgue integrals rather than the customary Riemann integrals. Functions that are Riemann integrable are also Lebesgue integrable but not conversely. We have neither time nor space to delve into Lebesgue integration nor will it be necessary for most of our discussions. It is, however, helpful when seeking understanding of the continuity requirements of the various function spaces. So, we'll make a few brief remarks and refer those seeking more information to texts on functional analysis [3, 4, 5].

With Lebesgue integration, the concept of the length of a subinterval is replaced by the *measure of an arbitrary point set.* Certain sets are so sparse as to have *measure zero.* An example is the set of rational numbers on $[0,1]$. Indeed, all countably infinite sets have measure zero. If a function $u \in \mathcal{V}$ possesses a given property except on a set of measure zero then it is said to have that property almost everywhere. A relevant property is the notion of an *equivalence class.* Two functions $u, v \in \mathcal{V}$ belong to the same *equivalence class* if

$$\|u - v\|_A = 0.$$

With Lebesgue integration, two functions in the same equivalence class are equal almost everywhere. Thus, if we are given a function $u \in \mathcal{V}$ and change its values on a set of measure zero to obtain a function v, then u and v belong to the same equivalence class.

We need one more concept, the notion of *completeness.* A *Cauchy sequence* $\{u_n\}_{n=1}^{\infty} \in \mathcal{V}$ is one where

$$\lim_{m,n\to\infty} \|u_m - u_n\|_A = 0.$$

If $\{u_n\}_{n=1}^{\infty}$ converges in $\|\cdot\|_A$ to a function $u \in \mathcal{V}$ then it is a Cauchy sequence. Thus, using the triangular inequality,

$$\lim_{m,n\to\infty} \|u_m - u_n\|_A \le \lim_{m,n\to\infty} \{\|u_m - u\|_A + \|u - u_n\|_A\} = 0.$$

A space $\mathcal{V}$ where the converse is true, *i.e.*, where all Cauchy sequences $\{u_n\}_{n=1}^{\infty}$ converge in $\|\cdot\|_A$ to functions $u \in \mathcal{V}$, is said to be *complete.*

Definition 3.2.5. A complete linear space $\mathcal{V}$ with inner product $A(u, v)$ and corresponding norm $\|u\|_A$, $u, v \in \mathcal{V}$ is called a *Hilbert space.*

Let's list some relevant Hilbert spaces for use with variational formulations of boundary value problems. We'll present their definitions in two space dimensions. Their extension to one and three dimensions is obvious.

Definition 3.2.6. The space $L^2(\Omega)$ consists of functions satisfying

$$L^2(\Omega) := \{u| \iint_{\Omega} u^2 dxdy < \infty\}. \tag{3.2.4a}$$

It has the inner product

$$(u, v) = \iint_{\Omega} uv dxdy \tag{3.2.4b}$$

and norm

$$\|u\|_0 = \sqrt{(u, u)}. \tag{3.2.4c}$$

Definition 3.2.7. The *Sobolev space* H^k consists of functions u which belong to L^2 with their first $k \ge 0$ derivatives. The space has the inner product and norm

$$(u, v)_k := \sum_{|\boldsymbol{\kappa}|\le k} (D^{\boldsymbol{\kappa}} u, D^{\boldsymbol{\kappa}} v), \tag{3.2.5a}$$

$$\|u\|_k = \sqrt{(u, u)_k}, \tag{3.2.5b}$$

where

$$\boldsymbol{\kappa} = [\kappa_1, \kappa_2]^T, \qquad |\boldsymbol{\kappa}| = \kappa_1 + \kappa_2, \tag{3.2.5c}$$

with κ_1 and κ_2 non-negative integers, and

$$D^{\boldsymbol{\kappa}} u := \frac{\partial^{\kappa_1+\kappa_2} u}{\partial x^{\kappa_1} \partial y^{\kappa_2}}. \tag{3.2.5d}$$

In particular, the space H^1 has the inner product and norm

$$(u,v)_1 = (u,v) + (u_x, v_x) + (u_y, v_y) = \iint_\Omega (uv + u_x v_x + u_y v_y)dxdy \tag{3.2.6a}$$

$$\|u\|_1 = \left[\iint_\Omega (u^2 + u_x^2 + u_y^2)dxdy\right]^{1/2}. \tag{3.2.6b}$$

Likewise, functions $u \in H^2$ have finite values of

$$\|u\|_2^2 = \iint_\Omega [u_{xx}^2 + u_{xy}^2 + u_{yy}^2 + u_x^2 + u_y^2 + u^2]dxdy.$$

Example 3.2.1. We have been studying second-order differential equations of the form (3.1.1) and seeking weak solutions $u \in H^1$ and $U \in S^N \subset H^1$ of (3.1.6) and (3.2.2), respectively. Let us verify that H^1 is the correct space, at least in one dimension. Thus, consider a basis of the familiar piecewise-linear hat functions on a uniform mesh with spacing $h = 1/N$

$$\phi_j(x) = \begin{cases} (x - x_{j-1})/h, & \text{if } x_{j-1} \le x < x_j \\ (x_{j+1} - x)/h, & \text{if } x_j \le x < x_{j+1} \\ 0, & \text{otherwise} \end{cases}. \tag{3.2.7}$$

Since $S^N \subset H^1$, ϕ_j and ϕ_j' must be in L^2, $j = 1, 2, ..., N$. Consider C^∞ approximations of $\phi_j(x)$ and $\phi_j'(x)$ obtained by "rounding corners" in $O(h/n)$-neighborhoods of the nodes x_{j-1}, x_j, x_{j+1} as shown in Figure3.2.1. A possible smooth approximation of $\phi_j'(x)$ is

$$\phi_j'(x) \approx \phi_{j,n}'(x) = \frac{1}{2h}[\tanh\frac{n(x - x_{j+1})}{h} + \tanh\frac{n(x - x_{j-1})}{h} - 2\tanh\frac{n(x - x_j)}{h}].$$

A smooth approximation $\phi_{j,n}$ of ϕ_j is obtained by integration as

$$\phi_{j,n}(x) = \frac{h}{2n}\ln\left[\frac{\cosh n((x - x_{j+1})/h)\cosh n((x - x_{j-1})/h)}{\cosh^2 n((x - x_j)/h)}\right].$$

Clearly, $\phi_{j,n}$ and $\phi_{j,n}'$ are elements of L^2. The "rounding" disappears as $n \to \infty$ and

$$\lim_{n\to\infty}\int_0^1 [\phi_{j,n}'(x)]^2 dx \approx 2h(1/h)^2 = 2/h.$$

The explicit calculations are somewhat involved and will not be shown. However, it seems clear that the limiting function $\phi_j' \in L^2$ and, hence, $\phi_j \in S^N$ for fixed h.

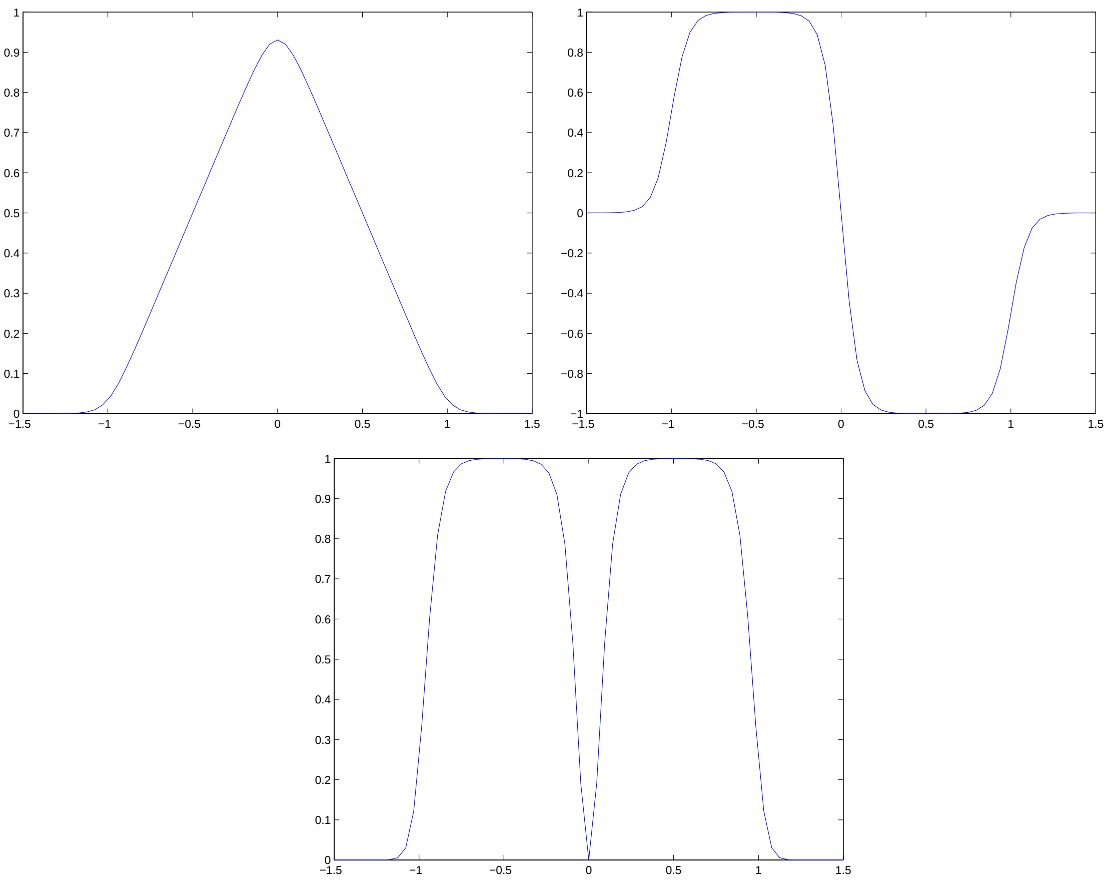


Figure 3.2.1: Smooth version of a piecewise linear hat function (3.2.7) (top), its first derivative (center), and the square of its first derivative (bottom). Results are shown with $x_{j-1} = -1$, $x_j = 0$, $x_{j+1} = 1$ $(h = 1)$, and $n = 10$.

Example 3.2.2. Consider the piecewise-constant basis function on a uniform mesh

$$\phi_j(x) = \begin{cases} 1, & \text{if } x_{j-1} \le x < x_j \\ 0, & \text{otherwise} \end{cases} . \tag{3.2.8}$$

A smooth version of this function and its first derivative are shown in Figure 3.2.2 and may be written as

$$\phi_{j,n}(x) = \frac{1}{2}[\tanh \frac{n(x-x_{j-1})}{h} - \tanh \frac{n(x-x_j)}{h}]$$

$$\phi'_{j,n}(x) = \frac{n}{2h}[\text{sech}^2 \frac{n(x-x_{j-1})}{h} - \text{sech}^2 \frac{n(x-x_j)}{h}].$$

As $n \to \infty$, $\phi_{j,n}$ approaches a square pulse; however, $\phi'_{j,n}$ is proportional to the combination of delta functions

$$\phi'_{j,n}(x) \propto \delta(x-x_{j-1}) - \delta(x-x_j).$$

Thus, we anticipate problems since delta functions are not elements of L^2. Squaring $\phi'_{j,n}(x)$

$$[\phi'_{j,n}(x)]^2 = (\frac{n}{2h})^2[\text{sech}^4\frac{n(x-x_{j-1})}{h} - 2\text{sech}^2\frac{n(x-x_{j-1})}{h}\text{sech}^2\frac{n(x-x_j)}{h} + \text{sech}^4\frac{n(x-x_j)}{h}].$$

As shown in Figure 3.2.2, the function $\text{sech}n(x-x_j)/h$ is largest at x_j and decays exponentially fast from x_j; thus, the center term in the above expression is exponentially small relative to the first and third terms. Neglecting it yields

$$[\phi'_{j,n}(x)]^2 \approx (\frac{n}{2h})^2[\text{sech}^4\frac{n(x-x_{j-1})}{h} + \text{sech}^4\frac{n(x-x_j)}{h}].$$

Thus,

$$\int_0^1 [\phi'_{j,n}(x)]^2dx \approx \frac{n}{12h}[\tanh\frac{n(x-x_{j-1})}{h}(2+\text{sech}^2\frac{n(x-x_{j-1})}{h}) + \tanh\frac{n(x-x_j)}{h}(2+\text{sech}^2\frac{n(x-x_j)}{h})]_0^1.$$

This is unbounded as $n \to \infty$; hence, $\phi'_j(x) \notin L^2$ and $\phi_j(x) \notin H^1$.

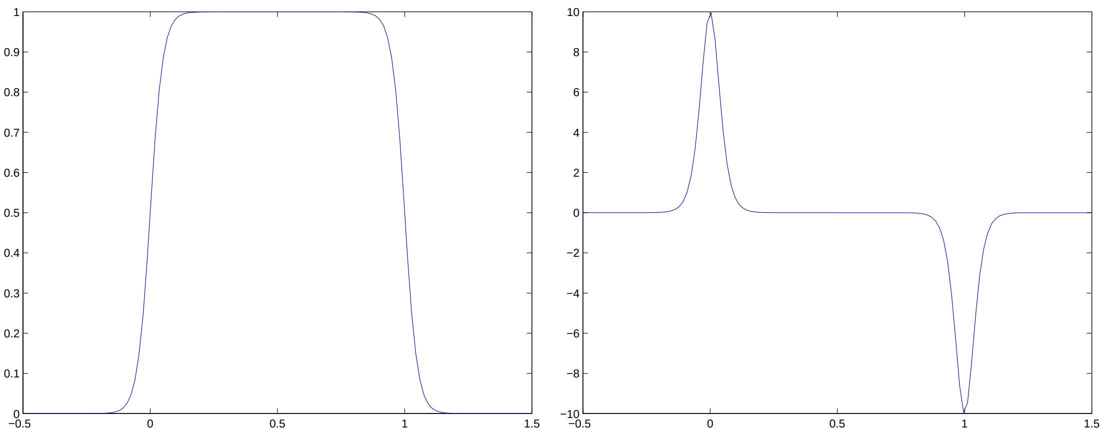


Figure 3.2.2: Smooth version of a piecewise constant function (3.2.8) (left) and its first derivative (right). Results are shown with $x_{j-1}=0$, $x_j=1$ ($h=1$), and $n=20$.

Although the previous examples lack rigor, we may conclude that a basis of continuous functions will belong to H^1 in one dimension. More generally, $u \in H^k$ implies that $u \in C^{k-1}$ in one dimension. The situation is not as simple in two and three dimensions. The Sobolev space H^k is the completion with respect to the norm (3.2.5) of C^k functions whose first k partial derivatives are elements of L^2. Thus, for example, $u \in H^1$ implies that u, u_x, and u_y are all elements of L^2. This is not sufficient to ensure that u is continuous in two and three dimensions. Typically, if $\partial\Omega$ is smooth then $u \in H^k$ implies that $u \in C^s(\Omega \cup \partial\Omega)$ where s is the largest integer *less than* $(k-d/2)$ in d dimensions [1, 2]. In two and three dimensions, this condition implies that $u \in C^{k-2}$.

Problems

1. Assuming that $p(x, y) > 0$ and $q(x, y) \geq 0$, $(x, y) \in \Omega$, find any other conditions that must be satisfied for the strain energy

$$A(v, u) = \iint_{\Omega} [p(v_x u_x + v_y u_y) + qvu] dxdy$$

to be an inner product and norm, *i.e.*, to satisfy Definitions 3.2.3 and 3.2.4.

2. Construct a variational problem for the fourth-order biharmonic equation

$$\Delta(p\Delta u) = f(x, y), \qquad (x, y) \in \Omega,$$

where

$$\Delta u = u_{xx} + u_{yy}$$

and $p(x, y) > 0$ is smooth. Assume that u satisfies the essential boundary conditions

$$u(x, y) = 0, \qquad u_{\mathbf{n}}(x, y) = 0, \qquad (x, y) \in \partial\Omega,$$

where $\mathbf{n}$ is a unit outward normal vector to $\partial\Omega$. To what function space should the weak solution of the variational problem belong?

3.3 Overview of the Finite Element Method

Let us conclude this chapter with a brief summary of the key steps in constructing a finite-element solution in two or three dimensions. Although not necessary, we will continue to focus on (3.1.1) as a model.

1. Construct a variational form of the problem. Generally, we will use Galerkin's method to construct a variational problem. As described, this involves multiplying the differential equation be a suitable test function and using the divergence theorem to get a symmetric formulation. The trial function $u \in H_E^1$ and, hence, satisfies any prescribed essential boundary conditions. The test function $v \in H_0^1$ and, hence, vanishes where essential boundary conditions are prescribed. Any prescribed Neumann or Robin boundary conditions are used to alter the variational problem as, *e.g.*, with (3.1.6) or (3.1.8b), respectively.

Nontrivial essential boundary conditions introduce differences in the spaces H_E^1 and H_0^1. Furthermore, the finite element subspace S_E^N cannot satisfy non-polynomial boundary conditions. One way of overcoming this is to transform the differential equation to one having trivial essential boundary conditions (*cf.* Problem 1 at the end of this section). This approach is difficult to use when the boundary data is discontinuous or when the problem is nonlinear. It is more important for theoretical than for practical reasons.

The usual approach for handling nontrivial Dirichlet data is to interpolate it by the finite element trial function. Thus, consider approximations in the usual form

$$U(x,y) = \sum_{j=1}^{N} c_j \phi_j(x,y); \tag{3.3.1}$$

however, we include basis functions ϕ_k for mesh entities (vertices, edges) k that are on $\partial\Omega_E$. The coefficients c_k associated with these nodes are not varied during the solution process but, rather, are selected to interpolate the boundary data. Thus, with a Lagrangian basis where $\phi_k(x_j, y_j) = \delta_{k,j}$, we have

$$U(x_k, y_k) = \alpha(x_k, y_k) = c_k, \qquad (x_k, y_k) \in \partial\Omega_E.$$

The interpolation is more difficult with hierarchical functions, but it is manageable (*cf.* Section 4.4). We will have to appraise the effect of this interpolation on solution accuracy. Although the spaces S_E^N and S_0^N differ, the stiffness and mass matrices can be made symmetric for self-adjoint linear problems (*cf.* Section 5.5).

A third method of satisfying essential boundary conditions is given as Problem 2 at the end of this section.

2. Discretize the domain. Divide Ω into *finite elements* having simple shapes, such as triangles or quadrilaterals in two dimensions and tetrahedra and hexahedra in three dimensions. This nontrivial task generally introduces errors near $\partial\Omega$. Thus, the problem is typically solved on a polygonal region $\tilde{\Omega}$ defined by the finite element mesh (Figure 3.3.1) rather than on Ω. Such errors may be reduced by using finite elements with curved sides and/or faces near $\partial\Omega$ (*cf.* Chapter 4). The relative advantages of using fewer curved elements or a larger number of smaller straight-sided or planar-faced elements will have to be determined.

3. Generate the element stiffness and mass matrices and element load vector. Piecewise polynomial approximations $U \in S_E^N$ of u and $V \in S_0^N$ of v are chosen. The approximating spaces S_E^N and S_0^N are supposed to be subspaces of H_E^1 and H_0^1, respectively; however, this may not be the case because of errors introduced in approximating the essential boundary conditions and/or the domain Ω. These effects will also have to be appraised (*cf.* Section 7.3). Choosing a basis for S^N, we write U and V in the form of (3.3.1).

The variational problem is written as a sum of contributions over the elements and the element stiffness and mass matrices and load vectors are generated. For the model problem (3.1.1) this would involve solving

$$\sum_{e=1}^{N_\Delta} [A_e(V,U) - (V,f)_e - < V, \beta >_e] = 0, \qquad \forall V \in S_0^N, \tag{3.3.2a}$$

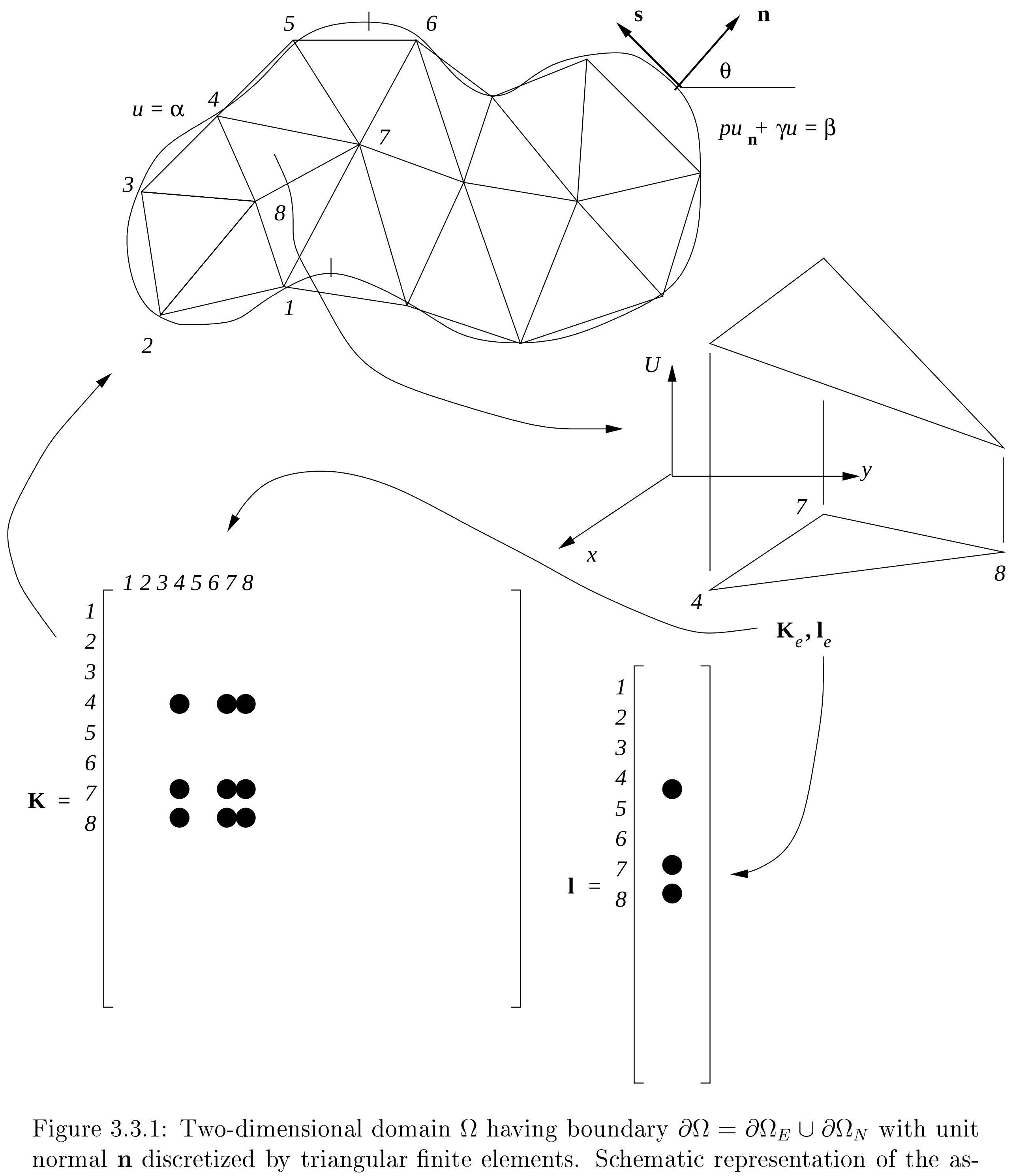

5
6
s
n
θ
u = α
4
7
$pu_{\mathbf{n}} + \gamma u = \beta$
3
8
1
2
U
y
7
x
4
8
1 2 3 4 5 6 7 8
1
2
3
4
5
6
7
8
K =
$\mathbf{K}_e, \mathbf{l}_e$
1
2
3
4
5
6
7
8
l =

Figure 3.3.1: Two-dimensional domain Ω having boundary $\partial\Omega = \partial\Omega_E \cup \partial\Omega_N$ with unit normal $\mathbf{n}$ discretized by triangular finite elements. Schematic representation of the assembly of the element stiffness matrix $\mathbf{K}_e$ and element load vector $\mathbf{l}_e$ into the global stiffness matrix $\mathbf{K}$ and load vector $\mathbf{l}$.

where

$$A_e(V, U) = \iint_{\Omega_e} (V_x p U_x + V_y p U_y + VqU) dx dy, \tag{3.3.2b}$$

$$(V, f)_e = \iint_{\Omega_e} V f dx dy, \tag{3.3.2c}$$

$$< V, \beta >_e = \int_{\partial\Omega_e \cap \partial\tilde{\Omega}_N} V \beta ds, \tag{3.3.2d}$$

Ω_e is the domain occupied by element e, and N_Δ is the number of elements in the mesh. The boundary integral (3.3.2d) is zero unless a portion of $\partial\Omega_e$ coincides with the boundary of the finite element domain $\partial\tilde{\Omega}$.

Galerkin formulations for self-adjoint problems such as (3.1.6) lead to minimum problems in the sense of Theorem 3.1.1. Thus, the finite element solution is the best solution in S^N in the sense of minimizing the strain energy of the error $A(u - U, u - U)$. The strain energy of the error is orthogonal to all functions V in S^N_E as illustrated in Figure 3.3.2 for three-vectors.

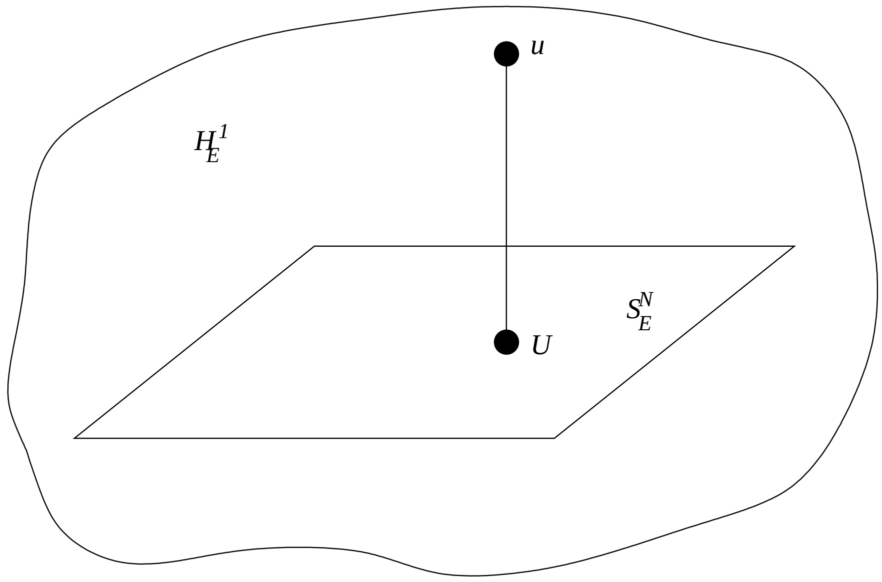


Figure 3.3.2: Subspace S^N_E of H^1_E illustrating the "best" approximation property of the solution of Galerkin's method.

4. Assemble the global stiffness and mass matrices and load vector. The element stiffness and mass matrices and load vectors that result from evaluating (3.3.2b-d) are added directly into global stiffness and mass matrices and a load vector. As depicted in Figure 3.3.1, the indices assigned to unknowns associated with mesh entities (vertices as shown) determine the correct positions of the elemental matrices and vectors in the global stiffness and mass matrices and load vector.

5. Solve the algebraic system. For linear problems, the assembly of (3.3.2) gives rise to a system of the form

$$\mathbf{d}^T[(\mathbf{K}+\mathbf{M})\mathbf{c}-\mathbf{l}] = \mathbf{0}, \tag{3.3.3a}$$

where $\mathbf{K}$ and $\mathbf{M}$ are the global stiffness and mass matrices, $\mathbf{l}$ is the global load vector,

$$\mathbf{c}^T = [c_1, c_2, ..., c_N]^T, \tag{3.3.3b}$$

and

$$\mathbf{d}^T = [d_1, d_2, ..., d_N]^T. \tag{3.3.3c}$$

Since (3.3.3a) must be satisfied for *all* choices of $\mathbf{d}$, we must have

$$(\mathbf{K}+\mathbf{M})\mathbf{c} = \mathbf{l}. \tag{3.3.4}$$

For the model problem (3.1.1), $\mathbf{K}+\mathbf{M}$ will be sparse and positive definite. With proper treatment of the boundary conditions, it will also be symmetric (*cf.* Chapter 5).

Each step in the finite element solution will be examined in greater detail. Basis construction is described in Chapter 4, mesh generation and assembly appear in Chapter 5, error analysis is discussed in Chapter 7, and linear algebraic solution strategies are presented in Chapter 11.

Problems

1. By introducing the transformation

$$\hat{u} = u - \alpha$$

show that (3.1.1) can be changed to a problem with homogeneous essential boundary conditions. Thus, we can seek $\hat{u} \in H_0^1$.

2. Another method of treating essential boundary conditions is to remove them by using a "penalty function." Penalty methods are rarely used for this purpose, but they are important for other reasons. This problem will introduce the concept and reinforce the material of Section 3.1. Consider the variational statement (3.1.6) as an example, and modify it by including the essential boundary conditions

$$A(v,u) = (v,f) + < v, \beta >_{\partial\Omega_N} + \lambda < v, \alpha - u >_{\partial\Omega_E}, \qquad \forall v \in H^1.$$

Here λ is a penalty parameter and subscripts on the boundary integral indicate their domain. No boundary conditions are applied and the problem is solved for u and v ranging over the whole of H^1.

Show that smooth solutions of this variational problem satisfy the differential equation (3.1.1a) as well as the natural boundary conditions (3.1.1c) and

$$u + \frac{p}{\lambda}\frac{\partial u}{\partial \mathbf{n}} = \alpha, \qquad (x, y) \in \Omega_E.$$

The penalty parameter λ must be selected large enough for this natural boundary condition to approximate the prescribed essential condition (3.1.1b). This can be tricky. If selected too large, it will introduce ill-conditioning into the resulting algebraic system.

Bibliography

[1] R.A. Adams. *Sobolev Spaces.* Academic Press, New York, 1975.

[2] O. Axelsson and V.A. Barker. *Finite Element Solution of Boundary Value Problems.* Academic Press, Orlando, 1984.

[3] C. Geoffman and G. Pedrick. *First Course in Functional Analysis.* Prentice-Hall, Englewood Cliffs, 1965.

[4] P.R. Halmos. *Measure Theory.* Springer-Verlag, New York, 1991.

[5] J.T. Oden and L.F. Demkowicz. *Applied Functional Analysis.* CRC Press, Boca Raton, 1996.

[6] R. Wait and A.R. Mitchell. *The Finite Element Analysis and Applications.* John Wiley and Sons, Chichester, 1985.

Chapter 4

Finite Element Approximation

4.1 Introduction

Our goal in this chapter is the development of piecewise-polynomial approximations U of a two- or three-dimensional function u. For this purpose, it suffices to regard u as being known and to determine U as its interpolant on a domain Ω. Concentrating on two dimensions for the moment, let us partition Ω into a collection of finite elements and write U in the customary form

$$U(x,y) = \sum_{j=1}^{N} c_j \phi_j(x,y). \tag{4.1.1}$$

As we discussed, it is convenient to associate each basis function ϕ_j with a mesh entity, *e.g.*, a vertex, edge, or element in two dimensions and a vertex, edge, face, or element in three dimensions. We will discuss these entities and their hierarchical relationship further in Chapter 5. For now, if ϕ_j is associated with the entity indexed by j, then, as described in Chapters 1 and 2, finite element bases are constructed so that ϕ_j is nonzero only on elements containing entity j. The support of two-dimensional basis functions associated with a vertex, an edge, and an element interior is shown in Figure 4.1.1.

As in one dimension, finite element bases are constructed implicitly in an element-by-element manner in terms of "shape functions" (*cf.* Section 2.4). Once again, a shape function on an element e is the restriction of a basis function $\phi_j(x,y)$ to element e. We proceed by constructing shape functions on triangular elements (Section 4.2, 4.4), quadrilaterals (Sections 4.3, 4.4), tetrahedra (Section 4.5.1), and hexahedra (Section 4.5.2).

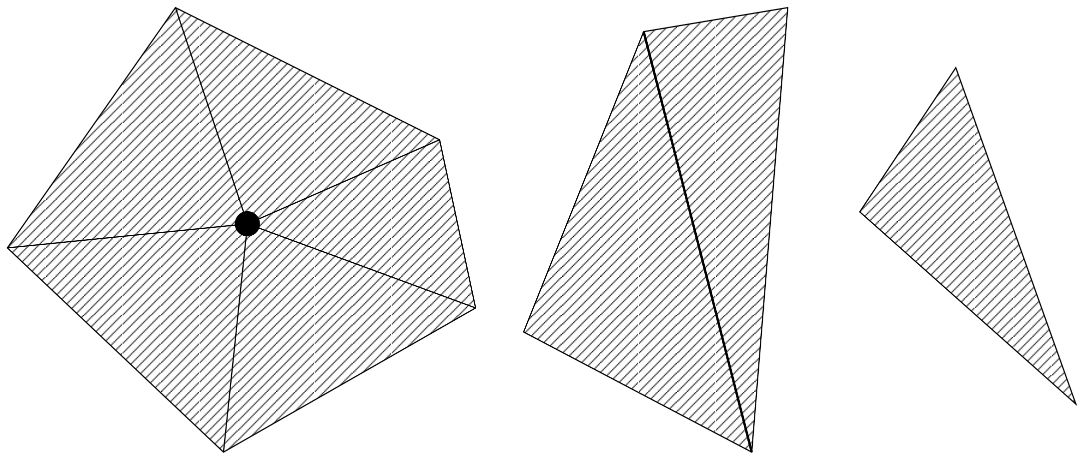


Figure 4.1.1: Support of basis functions associated with a vertex, edge, and element interior (left to right).

4.2 Lagrange Shape Functions on Triangles

Perhaps the simplest two-dimensional Lagrangian finite element basis is a piecewise-linear polynomial on a grid of triangular elements. It is the two-dimensional analog of the hat functions introduced in Section 1.3. Consider an arbitrary triangle e with its vertices indexed as 1, 2, and 3 and vertex j having coordinates (x_j, y_j), $j = 1, 2, 3$ (Figure 4.2.1). The linear shape function $N_j(x, y)$ associated with vertex j satisfies

$$N_j(x_k, y_k) = \delta_{j,k}, \qquad j, k = 1, 2, 3. \tag{4.2.1}$$

(Again, we omit the subscript e from $N_{j,e}$ whenever it is clear that we are discussing a single element.) Let N_j have the form

$$N_j(x, y) = a + bx + cy, \qquad (x, y) \in \Omega_e,$$

where Ω_e is the domain occupied by element e. Imposing conditions (4.2.1) produces

$$\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & x_j & y_j \\ 1 & x_k & y_k \\ 1 & x_l & y_l \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix}, \qquad k \neq l \neq j, \qquad j, k, l = 1, 2, 3.$$

Solving this system by Crammer's rule yields

$$N_j(x, y) = \frac{D_{k,l}(x, y)}{C_{j,k,l}}, \qquad k \neq l \neq j, \qquad j, k, l = 1, 2, 3, \tag{4.2.2a}$$

where

$$D_{k,l} = \det \begin{bmatrix} 1 & x & y \\ 1 & x_k & y_k \\ 1 & x_l & y_l \end{bmatrix}, \tag{4.2.2b}$$

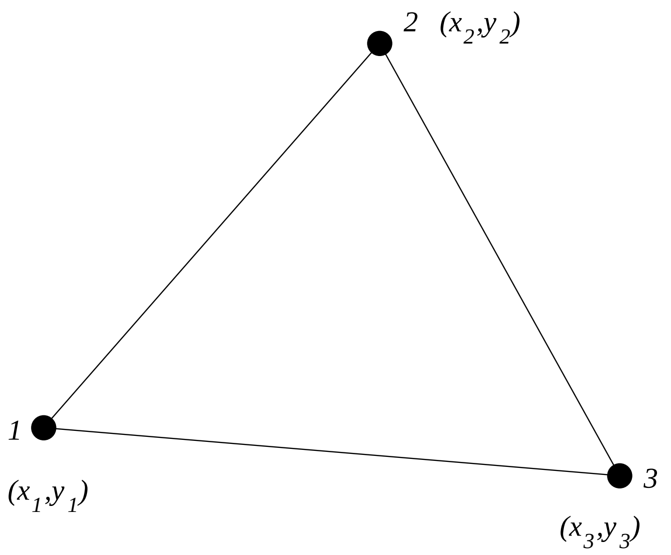


Figure 4.2.1: Triangular element with vertices $1, 2, 3$ having coordinates (x_1, y_1), (x_2, y_2), and (x_3, y_3).

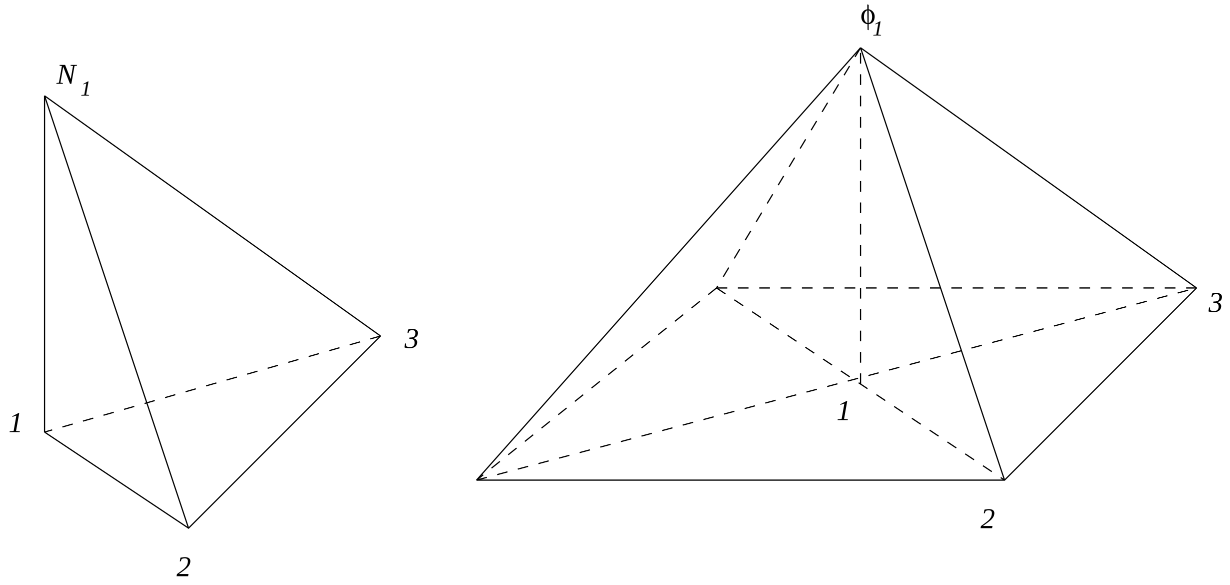


Figure 4.2.2: Shape function N_1 for Node 1 of element e (left) and basis function ϕ_1 for a cluster of four finite elements at Node 1.

$$C_{j,k,l} = \det \begin{bmatrix} 1 & x_j & y_j \\ 1 & x_k & y_k \\ 1 & x_l & y_l \end{bmatrix}. \tag{4.2.2c}$$

Basis functions are constructed by combining shape functions on neighboring elements as described in Section 2.4. A sample basis function for a four-element cluster is shown in Figure 4.2.2. The implicit construction of the basis in terms of shape function eliminates the need to know detailed geometric information such as the number of elements sharing a node. Placing the three nodes at element vertices guarantees a continuous basis. While interpolation at three non-colinear points is (necessary and) sufficient to determine a unique linear polynomial, it will not determine a continuous approximation. With vertex placement, the shape function (*e.g.*, N_j) along any element edge is a linear function of a variable along that edge. This linear function is determined by the nodal values at the two vertex nodes on that edge (*e.g.*, j and k). As shown in Figure 4.2.2, the shape function on a neighboring edge is determined by the same two nodal values; thus, the basis (*e.g.*, ϕ_j) is continuous.

The restriction of $U(x,y)$ to element e has the form

$$U(x,y) = c_1 N_1(x,y) + c_2 N_2(x,y) + c_3 N_3(x,y), \qquad (x,y) \in \Omega_e. \tag{4.2.3}$$

Using (4.2.1), we have $c_j = U(x_j, y_j)$, $j = 1,2,3$.

The construction of higher-order Lagrangian shape functions proceeds in the same manner. In order to construct a p *th*-degree polynomial approximation on element e, we introduce $N_j(x,y)$, $j = 1,2,\ldots,n_p$, shape functions at n_p nodes, where

$$n_p = \frac{(p+1)(p+2)}{2} \tag{4.2.4}$$

is the number of monomial terms in a complete polynomial of degree p in two dimensions. We may write a shape function in the form

$$N_j(x,y) = \sum_{i=1}^{n_p} a_i q_i(x,y) = \mathbf{a}^T \mathbf{q}(x,y) \tag{4.2.5a}$$

where

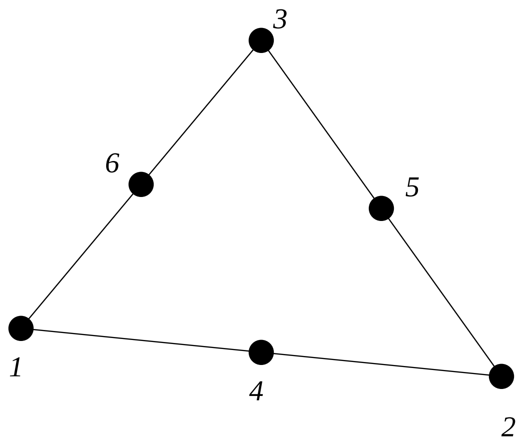
$$\mathbf{q}^T(x,y) = [1, x, y, x^2, xy, y^2, \ldots, y^p]. \tag{4.2.5b}$$

Thus, for example, a second degree ($p = 2$) polynomial would have $n_2 = 6$ coefficients and

$$\mathbf{q}^T(x,y) = [1, x, y, x^2, xy, y^2].$$

Including all n_p monomial terms in the polynomial approximation ensures isotropy in the sense that the degree of the trial function is conserved under coordinate translation and rotation.

With six parameters, we consider constructing a quadratic Lagrange polynomial by placing nodes at the vertices and midsides of a triangular element. The introduction of nodes is unnecessary, but it is a convenience. Indexing of nodes and other entities will be discussed in Chapter 5. Here, since we're dealing with a single element, we number the



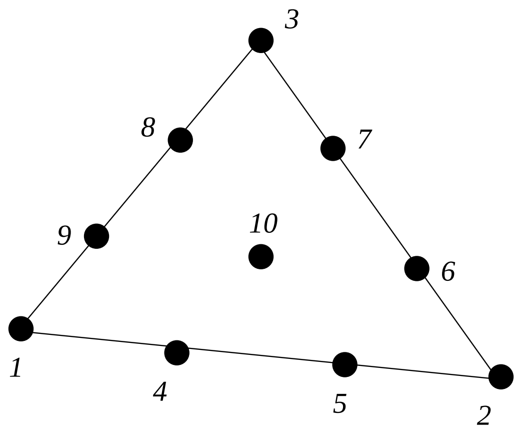


Figure 4.2.3: Arrangement of nodes for quadratic (left) and cubic (right) Lagrange finite element approximations.

nodes from 1 to 6 as shown in Figure 4.2.3. The shape functions have the form (4.2.5) with $n_2 = 6$

$$N_j = a_1 + a_2x + a_3y + a_4x^2 + a_5xy + a_6y^2,$$

and the six coefficients a_j, $j = 1, 2, \ldots, 6$, are determined by requiring

$$N_j(x_k, y_k) = \delta_{j,k}, \qquad j, k = 1, 2, \ldots, 6.$$

The basis

$$\phi_j = \cup_{e=1}^{N_\Delta} N_{j,e}(x, y)$$

is continuous by virtue of the placement of the nodes. The shape function $N_{j,e}$ is a quadratic function of a local coordinate on each edge of the triangle. This quadratic function of a single variable is uniquely determined by the values of the shape functions at the three nodes on the given edge. Shape functions on shared edges of neighboring triangles are determined by the same nodal values; hence, ensuring that the basis is globally of class C^0.

The construction of cubic approximations would proceed in the same manner. A complete cubic in two dimensions has 10 parameters. These parameters can be determined by selecting 10 nodes on each element. Following the reasoning described above, we should place four nodes on each edge since a cubic function of one variable is uniquely determined by prescribing four quantities. This accounts for nine of the ten nodes. The last node can be placed at the centroid as shown in Figure 4.2.3.

The construction of Lagrangian approximations is straight forward but algebraically complicated. Complexity can be significantly reduced by using one of the following two coordinate transformations.

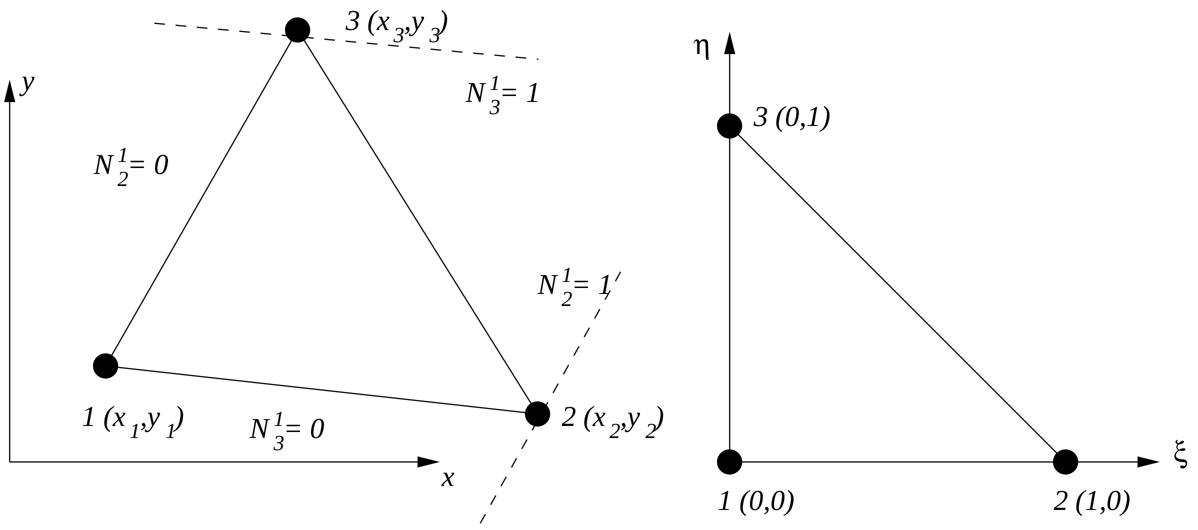


Figure 4.2.4: Mapping an arbitrary triangular element in the (x, y)-plane (left) to a canonical 45° right triangle in the (ξ, η)-plane (right).

1. Transformation to a canonical element. The idea is to transform an arbitrary element in the physical (x, y)-plane to one having a simpler geometry in a computational (ξ, η)-plane. For purposes of illustration, consider an arbitrary triangle having vertex nodes numbered 1, 2, and 3 which is mapped by a linear transformation to a unit 45° right triangle, as shown in Figure 4.2.4.

Consider N_2^1 and N_3^1 as defined by (4.2.2). (A superscript 1 has been added to emphasize that the shape functions are linear polynomials.) The equation of the line connecting Nodes 1 and 3 of the triangular element shown on the left of Figure 4.2.4 is $N_2^1 = 0$. Likewise, the equation of a line passing through Node 2 and parallel to the line passing through Nodes 1 and 3 is $N_2^1 = 1$. Thus, to map the line $N_2^1 = 0$ onto the line $\xi = 0$ in the canonical plane, we should set $\xi = N_2^1(x, y)$. Similarly, the line joining Nodes 1 and 2 satisfies the equation $N_3^1 = 0$. We would like this line to become the line $\eta = 0$ in the transformed plane, so our mapping must be $\eta = N_3^1(x, y)$. Therefore, using (4.2.2)

$$\xi = N_2^1(x,y) = \frac{\det\begin{bmatrix} 1 & x & y \\ 1 & x_1 & y_1 \\ 1 & x_3 & y_3 \end{bmatrix}}{\det\begin{bmatrix} 1 & x_2 & y_2 \\ 1 & x_1 & y_1 \\ 1 & x_3 & y_3 \end{bmatrix}}, \qquad \eta = N_3^1(x,y) = \frac{\det\begin{bmatrix} 1 & x & y \\ 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \end{bmatrix}}{\det\begin{bmatrix} 1 & x_3 & y_3 \\ 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \end{bmatrix}}. \tag{4.2.6}$$

As a check, evaluate the determinants and verify that $(x_1, y_1) \to (0, 0)$, $(x_2, y_2) \to (1, 0)$, and $(x_3, y_3) \to (0, 1)$.

Polynomials may now be developed on the canonical triangle to simplify the algebraic

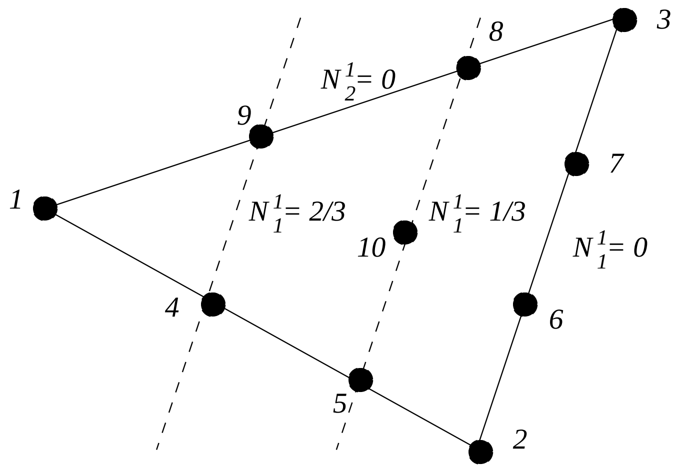


Figure 4.2.5: Geometry of a triangular finite element for a cubic polynomial Lagrange approximation.

complexity and subsequently transformed back to the physical element.

2. Transformation using triangular coordinates. A simple procedure for constructing Lagrangian approximations involves the use of a redundant coordinate system. The construction may be described in general terms, but an example suffices to illustrate the procedure. Thus, consider the construction of a cubic approximation on the triangular element shown in Figure 4.2.5. The vertex nodes are numbered 1, 2, and 3; edge nodes are numbered 4 to 9; and the centroid is numbered as Node 10.

Observe that

- the line $N_1^1 = 0$ passes through Nodes 2, 6, 7, and 3;
- the line $N_1^1 = 1/3$ passes through Nodes 5, 10, and 8; and
- the line $N_1^1 = 2/3$ passes through Nodes 4 and 9.

Since N_1^3 must vanish at Nodes 2 - 10 and be a cubic polynomial, it must have the form

$$N_1^3(x,y) = \alpha N_1^1(N_1^1 - 1/3)(N_1^1 - 2/3)$$

where the constant α is determined by normalizing $N_1^3(x_1,y_1) = 1$. Since $N_1^1(x_1,y_1) = 1$, we find $\alpha = 9/2$ and

$$N_1^3(x,y) = \frac{9}{2}N_1^1(N_1^1 - 1/3)(N_1^1 - 2/3).$$

The shape function for an edge node is constructed in a similar manner. For example, in order to obtain N_4^3 we observe that

- the line $N_2^1 = 0$ passes through Nodes 1, 9, 8, and 3;

- the line $N_1^1 = 0$ passes through Nodes 2, 6, 7, and 3; and
- the line $N_1^1 = 1/3$ passes through Nodes 5, 10, and 8.

Thus, N_4^3 must have the form

$$N_4^3(x, y) = \alpha N_1^1 N_2^1 (N_1^1 - 1/3).$$

Normalizing $N_4^3(x_4, y_4) = 1$ gives

$$N_4^3(x_4, y_4) = \alpha \frac{2}{3}\frac{1}{3}(\frac{2}{3} - \frac{1}{3}).$$

Hence, $\alpha = 27/2$ and

$$N_4^3(x, y) = \frac{27}{2} N_1^1 N_2^1 (N_1^1 - 1/3).$$

Finally, the shape function N_{10}^3 must vanish on the boundary of the triangle and is, thus, determined as

$$N_{10}^3(x, y) = 27 N_1^1 N_2^1 N_3^1.$$

The cubic shape functions N_1^3, N_4^3, and N_{10}^3 are shown in Figure 4.2.6.

The three linear shape functions N_j^1, $j = 1, 2, 3$, can be regarded as a redundant coordinate system known as "triangular" or "barycentric" coordinates. To be more specific, consider an arbitrary triangle with vertices numbered 1, 2, and 3 as shown in Figure 4.2.7. Let

$$\zeta_1 = N_1^1, \qquad \zeta_2 = N_2^1, \qquad \zeta_3 = N_3^1, \tag{4.2.7}$$

and define the transformation from triangular to physical coordinates as

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} \zeta_1 \\ \zeta_2 \\ \zeta_3 \end{bmatrix}. \tag{4.2.8}$$

Observe that $(\zeta_1, \zeta_2, \zeta_3)$ has value (1,0,0) at vertex 1, (0,1,0) at vertex 2 and (0,0,1) at vertex 3.

An alternate, and more common, definition of the triangular coordinate system involves ratios of areas of subtriangles to the whole triangle. Thus, let P be an arbitrary point in the interior of the triangle, then the triangular coordinates of P are

$$\zeta_1 = \frac{A_{P23}}{A_{123}}, \qquad \zeta_2 = \frac{A_{P31}}{A_{123}}, \qquad \zeta_3 = \frac{A_{P12}}{A_{123}}, \tag{4.2.9}$$

where A_{123} is the area of the triangle, A_{P23} is the area of the subtriangle having vertices P, 2, 3, *etc.*

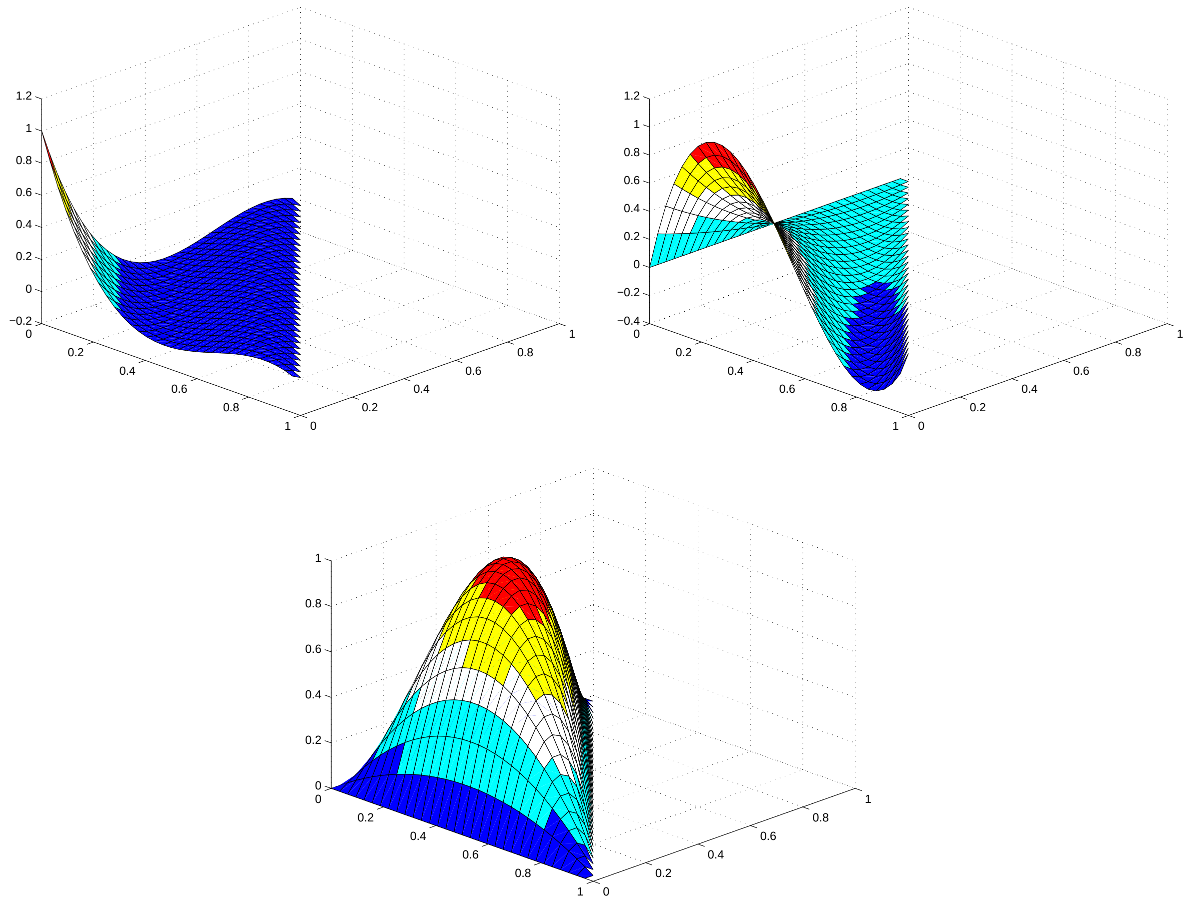


Figure 4.2.6: Cubic Lagrange shape functions associated with a vertex (left), an edge(right), and the centroid (bottom) of a right 45° triangular element.

The triangular coordinate system is redundant since two quantities suffice to locate a point in a plane. This redundancy is expressed by the third of equations (4.2.8), which states that

$$\zeta_1 + \zeta_2 + \zeta_3 = 1.$$

This relation also follows by adding equations (4.2.9).

Although seemingly distinct, triangular coordinates and the canonical coordinates are closely related. The triangular coordinate ζ_2 is equivalent to the canonical coordinate ξ and ζ_3 is equivalent to η, as seen from (4.2.6) and (4.2.7).

Problems

1. With reference to the nodal placement and numbering shown on the left of Figure 4.2.3, construct the shape functions for Nodes 1 and 4 of the quadratic Lagrange polynomial. Derive your answer using triangular coordinates. Having done this, also express your answer in terms of the canonical (ξ, η) coordinates. Plot or sketch

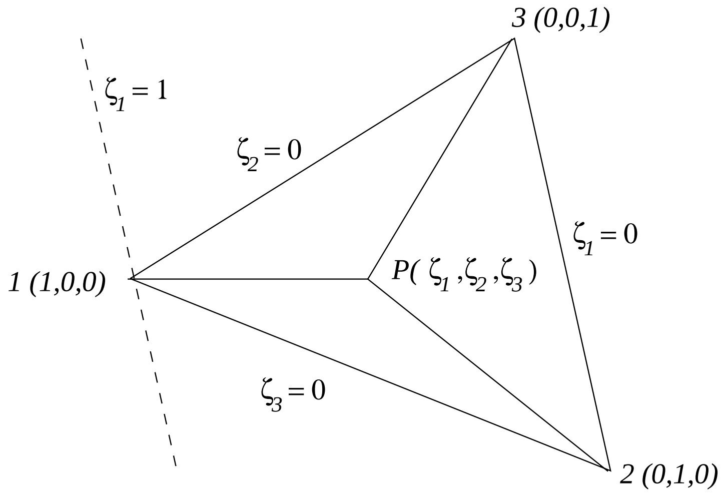


Figure 4.2.7: Triangular coordinate system.

the two shape functions on the canonical element.

2. A Lagrangian approximation of degree p on a triangle has three nodes at the vertices and $p-1$ nodes along each edge that are not at vertices. As we've discussed, the latter placement ensures continuity on a mesh of triangular elements. If no additional nodes are placed on edges, how many nodes are interior to the element if the approximation is to be complete?

4.3 Lagrange Shape Functions on Rectangles

The triangle in two dimensions and the tetrahedron in three dimensions are the polyhedral shapes having the minimum number of edges and faces. They are optimal for defining complete C^0 Lagrangian polynomials. Even so, Lagrangian interpolants are simple to construct on rectangles and hexahedra by taking products of one-dimensional Lagrange polynomials. Multi-dimensional polynomials formed in this manner are called "tensor-product" approximations. we'll proceed by constructing polynomial shape functions on canonical 2×2 square elements and mapping these elements to an arbitrary quadrilateral elements. We describe a simple bilinear mapping here and postpone more complex mappings to Chapter 5.

We consider the canonical 2×2 square $\{(\xi, \eta) | -1 \leq \xi, \eta \leq 1\}$ shown in Figure 4.3.1. For simplicity, the vertices of the element have been indexed with a double subscript as $(1,1)$, $(2,1)$, $(1,2)$, and $(2,2)$. At times it will be convenient to index the vertex coordinats as $\xi_1 = -1$, $\xi_2 = 1$, $\eta_1 = -1$, and $\eta_2 = 1$. With nodes at each vertex, we construct a bilinear Lagrangian polynomial $U(\xi, \eta)$ whose restriction to the canonical

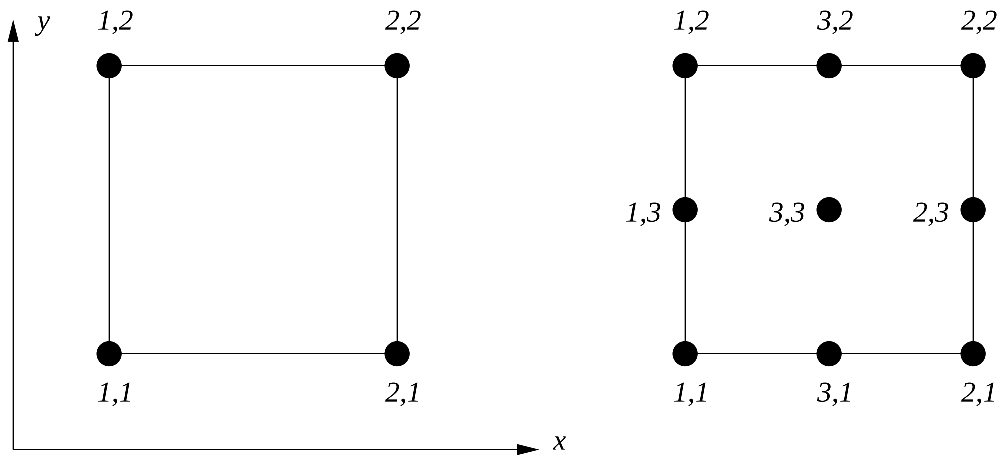


Figure 4.3.1: Node indexing for canonical square elements with bilinear (left) and bi-quadratic (right) polynomial shape functions.

element has the form

$$U(\xi,\eta) = c_{1,1}N_{1,1}(\xi,\eta) + c_{2,1}N_{2,1}(\xi,\eta) + c_{2,2}N_{2,2}(\xi,\eta) + c_{1,2}N_{1,2}(\xi,\eta). \tag{4.3.1a}$$

As with Lagrangian polynomials on triangles, the shape function $N_{i,j}(\xi,\eta)$ satisfies

$$N_{i,j}(\xi_k,\eta_l) = \delta_{i,k}\delta_{j,l}, \qquad k,l = 1,2. \tag{4.3.1b}$$

Once again, $U(\xi_k,\eta_l) = c_{k,l}$; however, now $N_{i,j}$ is the product of one-dimensional hat functions

$$N_{i,j}(\xi,\eta) = \bar{N}_i(\xi)\bar{N}_j(\eta) \tag{4.3.1c}$$

with

$$\bar{N}_1(\xi) = \frac{1-\xi}{2}, \tag{4.3.1d}$$

$$\bar{N}_2(\xi) = \frac{1+\xi}{2}, \qquad -1 \le \xi \le 1. \tag{4.3.1e}$$

Similar formulas apply to $\bar{N}_j(\eta)$, $j = 1,2$, with ξ replaced by η and i replaced by j. The shape function $N_{1,1}$ is shown in Figure 4.3.2. By examination of either this figure or (4.3.1c-e), we see that $N_{i,j}(\xi,\eta)$ is a bilinear function of the form

$$N_{i,j}(\xi,\eta) = a_1 + a_2\xi + a_3\eta + a_4\xi\eta, \qquad -1 \le \xi,\eta \le 1. \tag{4.3.2}$$

The shape function is linear along the two edges containing node (i,j) and it vanishes along the two opposite edges.

A basis may be constructed by uniting shape functions on elements sharing a node. The piecewise bilinear basis functions $\phi_{i,j}$ when Node (i,j) is at the intersection of four

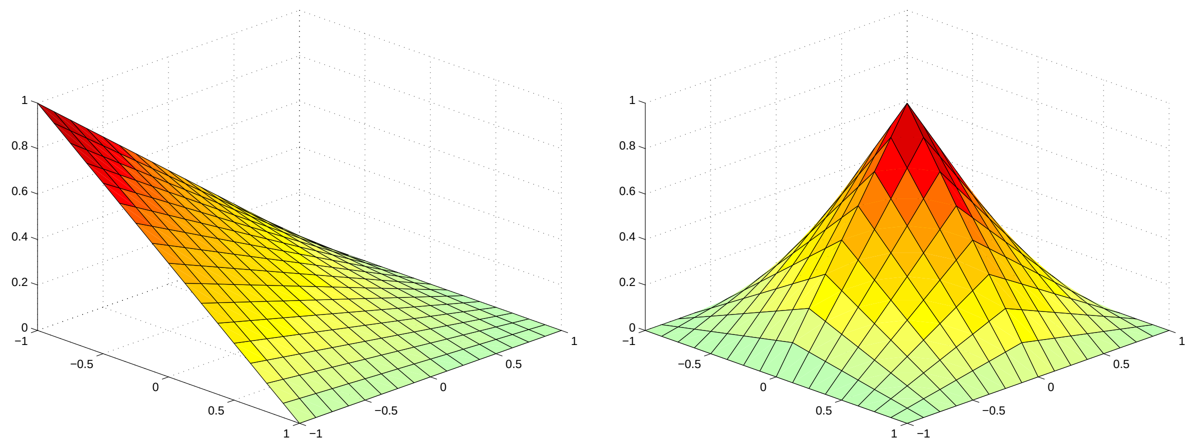


Figure 4.3.2: Bilinear shape function $N_{1,1}$ on the $[-1,1]\times[-1,1]$ canonical square element (left) and bilinear basis function at the intersection of four square elements (right).

square elements is shown in Figure 4.3.2. Since each shape function is a linear polynomial along element edges, the basis will be continuous on a grid of square (or rectangular) elements. The restriction to a square (or rectangular) grid is critical and the approximation would *not* be continuous on an arbitrary mesh of quadrilateral elements.

To construct biquadratic shape functions on the canonical square, we introduce 9 nodes: (1,1), (2,1), (2,2), and (1,2) at the vertices; (3,1), (2,3), (3,2), and (1,3) at mid-sides; and (3,3) at the center (Figure 4.3.1). The restriction of the interpolant U to this element has the form

$$U(\xi,\eta) = \sum_{i=1}^{3}\sum_{j=1}^{3} c_{i,j} N_{i,j}(\xi,\eta) \qquad (4.3.3a)$$

where the shape functions $N_{i,j}$, $i,j = 1,2,3$, are products of the one-dimensional quadratic polynomial Lagrange shape functions

$$N_{i,j}(\xi,\eta) = \bar{N}_i(\xi)\bar{N}_j(\eta), \qquad i,j = 1,2,3, \qquad (4.3.3b)$$

with (*cf.* Section 2.4)

$$\begin{aligned} \bar{N}_1(\xi) &= -\xi(1-\xi)/2, && (4.3.3c)\\ \bar{N}_2(\xi) &= \xi(1+\xi)/2, && (4.3.3d)\\ \bar{N}_3(\xi) &= (1-\xi^2), \qquad -1 \le \xi \le 1. && (4.3.3e) \end{aligned}$$

Shape functions for a vertex, an edge, and the centroid are shown in Figure 4.3.3. Using (4.3.3b-e), we see that shape functions are biquadratic polynomials of the form

$$N_{i,j}(\xi,\eta) = a_1 + a_2\xi + a_3\eta + a_4\xi^2 + a_5\xi\eta + a_6\eta^2 + a_7\xi\eta^2 + a_8\xi^2\eta + a_9\xi^2\eta^2. \qquad (4.3.4)$$

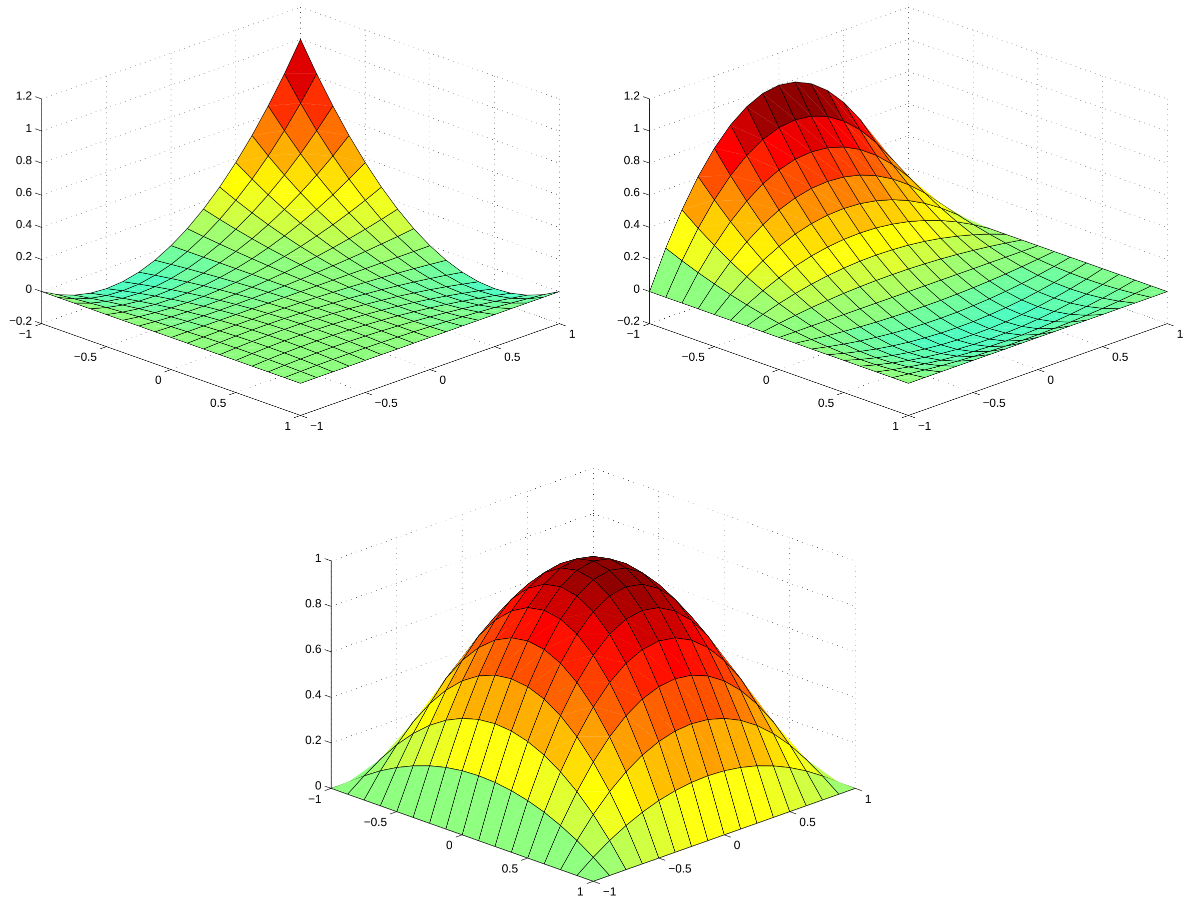


Figure 4.3.3: Biquadratic shape functions associated with a vertex (left), an edge (right), and the centroid (bottom).

Although (4.3.4) contains some cubic and quartic monomial terms, interpolation accuracy is determined by the highest-degree *complete* polynomial that can be represented exactly, which, in this case, is a quadratic polynomial.

Higher-order shape functions are constructed in similar fashion.

4.3.1 Bilinear Coordinate Transformations

Shape functions on the canonical square elements may be mapped to arbitrary quadrilaterals by a variety of transformations (*cf.* Chapter 5). The simplest of these is a picewise-bilinear function that uses the same shape functions (4.3.1d,e) as the finite element solution (4.3.1a). Thus, consider a mapping of the canonical 2×2 square S to a quadrilateral Q having vertices at $(x_{i,j}, y_{i,j})$, $i, j = 1, 2$, in the physical (x, y)-plane (Figure 4.3.4) using a bilinear transformation written in terms of (4.3.1d,e) as

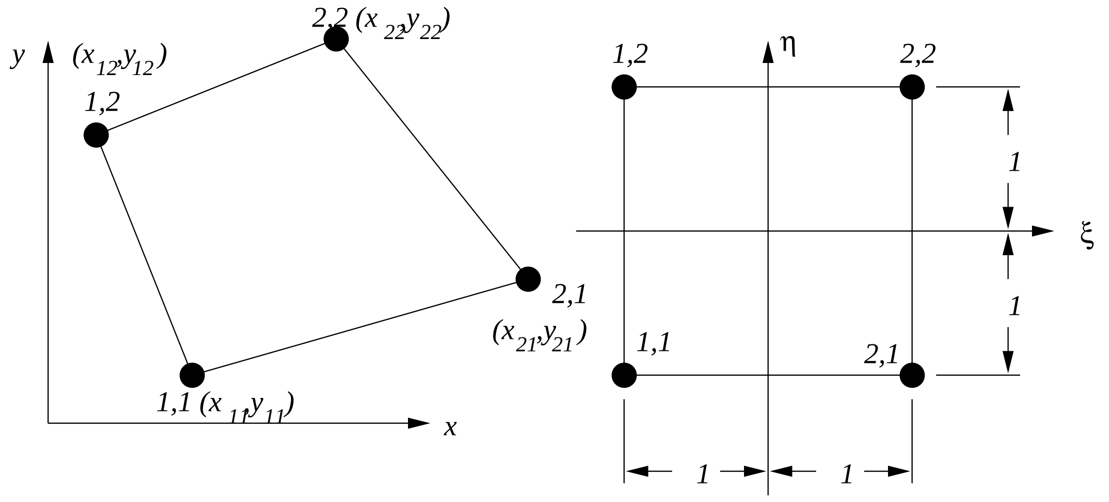


Figure 4.3.4: Bilinear mapping of the canonical square to a quadrilateral.

$$\begin{bmatrix} x(\xi,\eta) \\ y(\xi,\eta) \end{bmatrix} = \sum_{i=1}^{2}\sum_{j=1}^{2} \begin{bmatrix} x_{ij} \\ y_{ij} \end{bmatrix} N_{i,j}(\xi,\eta), \tag{4.3.5}$$

where $N_{i,j}(\xi,\eta)$ is given by (4.3.1b).

The transformation is linear on each edge of the element. In particular, transforming the edge $\eta = -1$ to the physical edge $(x_{11}, y_{11}$ - $(x_{21}, y21)$ yields

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x_{11} \\ y_{11} \end{bmatrix} \frac{1-\xi}{2} + \begin{bmatrix} x_{21} \\ y_{21} \end{bmatrix} \frac{1+\xi}{2}, \qquad -1 \le \xi \le 1.$$

As ξ varies from -1 to 1, x and y vary linearly from (x_{11}, y_{11}) to (x_{21}, y_{21}). The locations of the vertices (1,2) and (2,2) have no effect on the transformation. This ensures that a continuous approximation in the (ξ,η)-plane will remain continuous when mapped to the (x,y)-plane. We have to ensure that the mapping is invertible and we'll show in Chapter 5 that this is the case when Q is convex.

Problems

1. As noted, interpolation errors of the biquadratic approximation (4.3.3) are the same order as for a quadratic approximation on a triangle. Thus, for example, the L^2 error in interpolating a smooth function $u(x,y)$ by a piecewise biquadratic function $U(x,y)$ is $O(h^3)$, where h is the length of the longest edge of an element. The extra degrees of freedom associated with the cubic and quartic terms do not generally improve the order of accuracy. Hence, we might try to eliminate some shape functions and reduce the complexity of the approximation. Unknowns associated with interior shape functions are only coupled to unknowns on the element and can easily be eliminated by a variety of techniques. Considering the biquadratic polynomial in the form (4.3.3a), we might determine $c_{3,3}$ so that the coefficient of the

quartic term x^2y^2 vanishes. Show how this may be done for a 2×2 square canonical element. Polynomials of this type have been called *serendipity* by Zienkiewicz [8]. In the next section, we shall see that they are also a part of the hierarchical family of approximations. The parameter $c_{3,3}$ is said to be "constrained" since it is prescribed in advance and not determined as part of the Galerkin procedure. Plot or sketch shape functions associated with a vertex and a midside.

4.4 Hierarchical Shape Functions

We have discussed the advantages of hierarchical bases relative to Lagrangian bases for one-dimensional problems in Section 2.5. Similar advantages apply in two and three dimensions. We'll again use the basis of Szabó and Babuška [7], but follow the construction procedure of Shephard *et al.* [6] and Dey *et al.* [5]. Hierarchical bases of degree p may be constructed for triangles and squares. Squares are the simpler of the two, so let us handle them first.

4.4.1 Hierarchical Shape Functions on Squares

We'll construct the basis on the canonical element $\{(\xi,\eta)| -1 \leq \xi,\eta \leq 1\}$, indexing the vertices, edges, and interiors as described for the biquadratic approximation shown in Figure 4.3.1. The hierarchical polynomial of order p has a basis consisting of the following shape functions.

Vertex shape functions. The four vertex shape functions are the bilinear functions (4.3.1c-e)

$$N^1_{i,j} = \bar{N}_i(\xi)\bar{N}_j(\eta), \qquad i,j = 1,2, \tag{4.4.1a}$$

where

$$\bar{N}_1(\xi) = \frac{1-\xi}{2}, \qquad \bar{N}_2(\xi) = \frac{1+\xi}{2}. \tag{4.4.1b}$$

The shape function $N^1_{1,1}$ is shown in the upper left portion of Figure 4.4.1.

Edge shape functions. For $p \geq 2$, there are $4(p-1)$ shape functions associated with the midside nodes $(3,1)$, $(2,3)$, $(3,2)$, and $(1,3)$:

$$N^k_{3,1}(\xi,\eta) = \bar{N}_1(\eta)\bar{N}^k(\xi), \tag{4.4.2a}$$

$$N^k_{3,2}(\xi,\eta) = \bar{N}_2(\eta)\bar{N}^k(\xi), \tag{4.4.2b}$$

$$N^k_{1,3}(\xi,\eta) = \bar{N}_1(\xi)\bar{N}^k(\eta), \tag{4.4.2c}$$

$$N^k_{2,3}(\xi,\eta) = \bar{N}_2(\xi)\bar{N}^k(\eta), \qquad k = 2,3,\ldots,p, \tag{4.4.2d}$$

where $\bar{N}^k(\xi)$, $k = 2, 3, \ldots, p$, are the one-dimensional hierarchical shape functions given by (2.5.8a) as

$$\bar{N}^k(\xi) = \sqrt{\frac{2k-1}{2}} \int_{-1}^{\xi} P_{k-1}(\sigma) d\sigma. \tag{4.4.2e}$$

Edge shape functions $N_{3,1}^k$ are shown for $k = 2, 3, 4$, in Figure 4.4.1. The edge shape functions are the product of a linear function of the variable normal to the edge to which they are associated and a hierarchical polynomial of degree k in a variable on this edge. The linear function ($\bar{N}_j(\xi)$, $\bar{N}_j(\eta)$, $j = 1, 2$) "blends" the edge function ($\bar{N}^k(\xi)$, $\bar{N}^k(\eta)$) onto the element so as to ensure continuity of the basis.

Interior shape functions. For $p \geq 4$, there are $(p-2)(p-3)/2$ internal shape functions associated with the centroid, Node $(3, 3)$. The first internal shape function is the "bubble function"

$$N_{3,3}^{4,0,0} = (1 - \xi^2)(1 - \eta^2). \tag{4.4.3a}$$

The remaining shape functions are products of $N_{3,3}^{4,0,0}$ and the Legendre polynomials as

$$N_{3,3}^{5,1,0} = N_{3,3}^{4,0,0} P_1(\xi), \tag{4.4.3b}$$
$$N_{3,3}^{5,0,1} = N_{3,3}^{4,0,0} P_1(\eta), \tag{4.4.3c}$$
$$N_{3,3}^{6,2,0} = N_{3,3}^{4,0,0} P_2(\xi), \tag{4.4.3d}$$
$$N_{3,3}^{6,1,1} = N_{3,3}^{4,0,0} P_1(\xi) P_1(\eta), \tag{4.4.3e}$$
$$N_{3,3}^{6,0,2} = N_{3,3}^{4,0,0} P_2(\eta), \qquad \ldots. \tag{4.4.3f}$$

The superscripts k, λ, and μ, resectively, give the polynomial degree, the degree of $P_\lambda(\xi)$, and the degree of $P_\mu(\eta)$. The first six interior bubble shape functions $N_{3,3}^{k,\lambda,\mu}$, $\lambda+\mu = k-4$, $k = 4, 5, 6$, are shown in Figure 4.4.2. These functions vanish on the element boundary to maintain continuity.

On the canonical element, the interpolant $U(\xi, \eta)$ is written as the usual linear combination of shape functions

$$U(\xi, \eta) = \sum_{i=1}^{2} \sum_{j=1}^{2} c_{i,j}^1 N_{i,j}^1 + \sum_{k=2}^{p} [\sum_{j=1}^{2} c_{3,j}^k N_{3,j}^k + \sum_{i=1}^{2} c_{i,3}^k N_{i,3}^k] + \sum_{k=4}^{p} \sum_{\lambda+\mu=k-4} c_{3,3}^{k,\lambda,\mu} N_{3,3}^{k,\lambda,\mu}. \tag{4.4.4}$$

The notation is somewhat cumbersome but it is explicit. The first summation identifies unknowns and shape functions associated with vertices. The two center summations identify edge unknowns and shape functions for polynomial orders 2 to p. And, the third summation identifies the interior unknowns and shape functions of orders 4 to p.

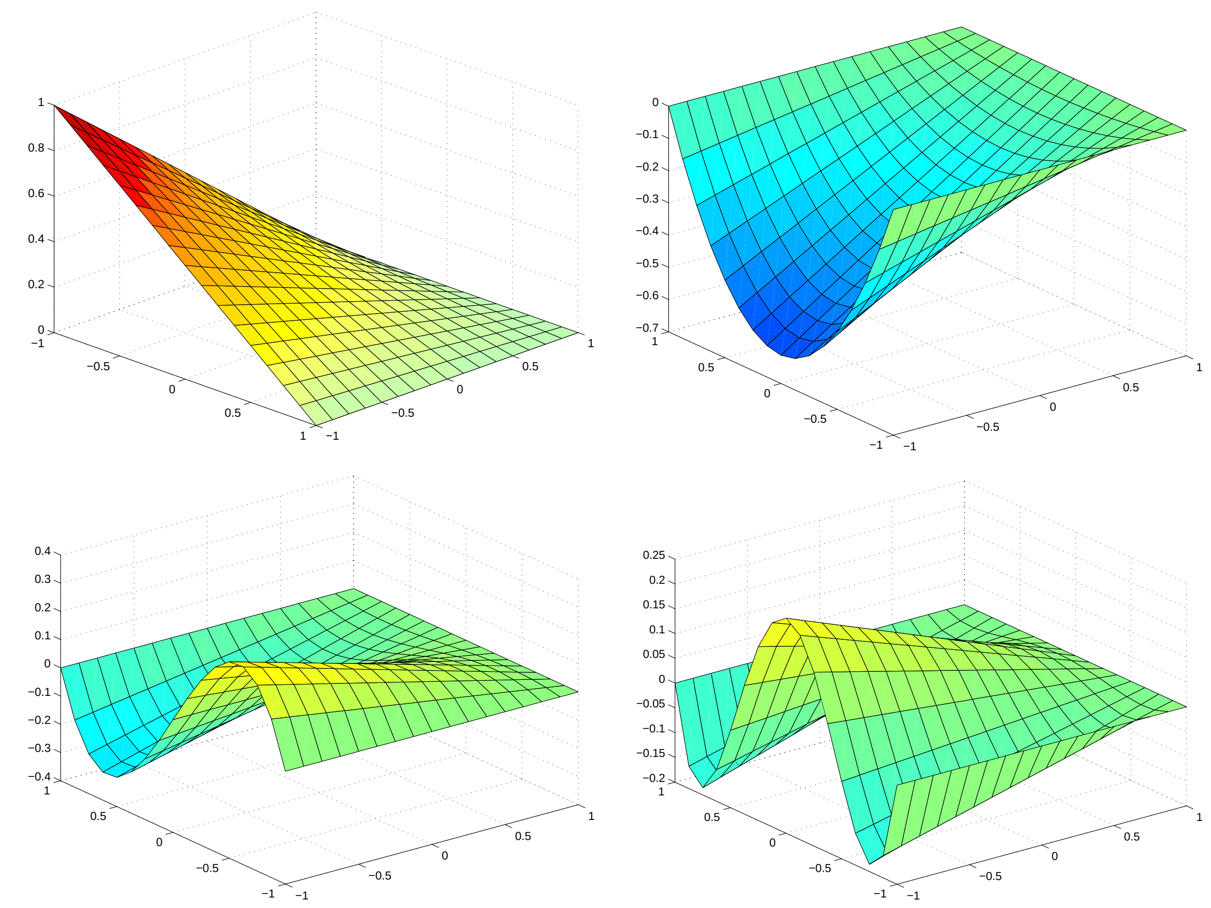


Figure 4.4.1: Hierarchical vertex and edge shape functions for $k = 1$ (upper left), $k = 2$ (upper right), $k = 3$ (lower left), and $k = 4$ (lower right).

Summations are understood to be zero when their initial index exceeds the final index. A degree p approximation has $4 + 4(p-1)_+ + (p-2)_+(p-3)_+/2$ unknowns and shape functions, where $q_+ = \max(q, 0)$. This function is listed in Table 4.4.1 for p ranging from 1 to 8. For large values of p there are $O(p^2)$ internal shape functions and $O(p)$ edge functions.

4.4.2 Hierarchical Shape Functions on Triangles

We'll express the hierarchical shape functions for triangular elements in terms of triangular coordinates, indexing the vertices as 1, 2, and 3; the edges as 4, 5, and 6; and the centroid as 7 (Figure 4.4.3). The basis consists of the following shape functions.

Vertex Shape functions. The three vertex shape functions are the linear barycentric coordinates (4.2.7)

$$N_i^1(\zeta_1, \zeta_2, \zeta_3) = \zeta_i, \qquad i = 1, 2, 3. \tag{4.4.5}$$

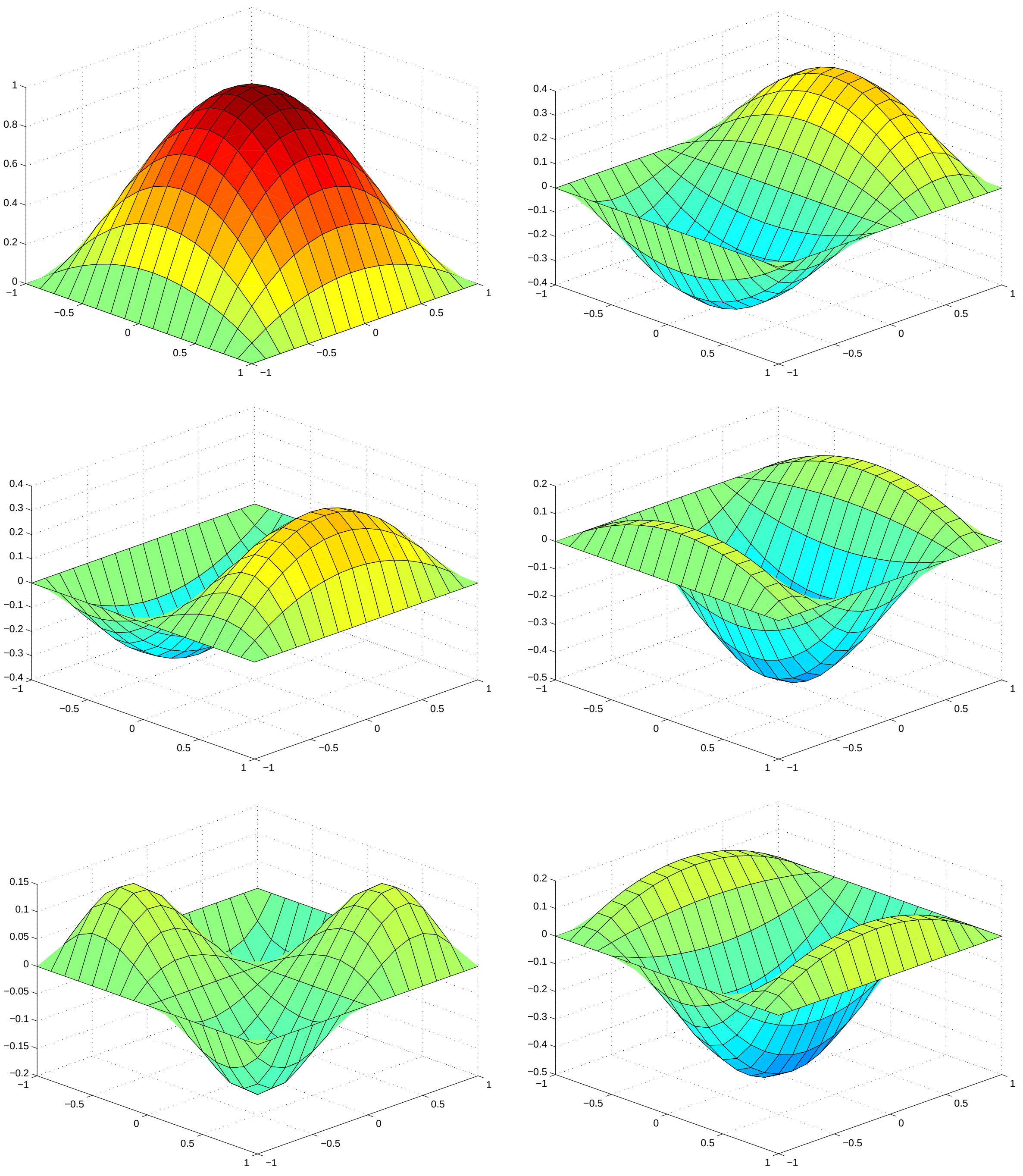


Figure 4.4.2: Hierarchical interior shape functions $N_{3,3}^{4,0,0}$, $N_{3,3}^{5,1,0}$ (top), $N_{3,3}^{5,0,1}$, $N_{3,3}^{6,2,0}$ (middle), and $N_{3,3}^{6,1,1}$, $N_{3,3}^{6,0,2}$ (bottom).

p	Square Dimension	Triangle Dimension
1	4	3
2	8	6
3	12	10
4	17	15
5	23	21
6	30	28
7	38	36
8	47	45

Table 4.4.1: Dimension of the hierarchical basis of order p on square and triangular elements.

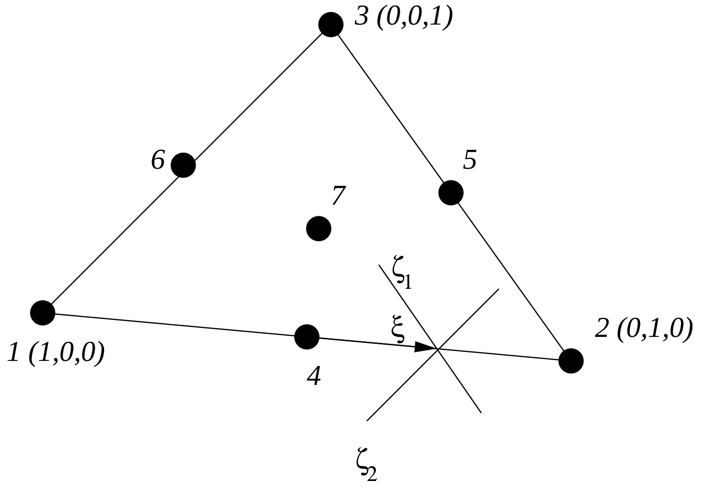


Figure 4.4.3: Node placement and coordinates for hierarchical approximations on a triangle.

Edge shape functions. For $p \geq 2$ there are $3(p-1)$ edge shape functions which are each nonzero on one edge (to which they are associated) and vanish on the other two. Each shape function is selected to match the corresponding edge shape function on a square element so that a continuous approximation may be obtained on meshes with both triangular and quadrilateral elements. Let us construct of the shape functions N_4^k, $k = 2, 3, \ldots, p$, associated with Edge 4. They are required to vanish on Edges 5 and 6 and must have the form

$$N_4^k(\zeta_1, \zeta_2, \zeta_3) = \zeta_1\zeta_2\bar{\chi}^k(\xi), \qquad k = 2, 3, \ldots, p, \tag{4.4.6a}$$

where $\bar{\chi}^k(\xi)$ is a shape function to be determined and ξ is a coordinate on Edge 4 that has value -1 at Node 1, 0 at Node 4, and 1 at Node 2. Since Edge 4 is $\zeta_3 = 0$, we have

$$N_4^k(\zeta_1, \zeta_2, 0) = \zeta_1\zeta_2\bar{\chi}^k(\xi), \qquad \zeta_1 + \zeta_2 = 1.$$

The latter condition follows from (4.2.8) with $\zeta_3 = 0$. Along Edge 4, ζ_1 ranges from 1 to 0 and ζ_2 ranges from 0 to 1 as ξ ranges from -1 to 1; thus, we may select

$$\zeta_1 = (1-\xi)/2, \qquad \zeta_2 = (1+\xi)/2, \qquad \zeta_3 = 0. \tag{4.4.6b}$$

While ξ may be defined in other ways, this linear mapping ensures that $\zeta_1 + \zeta_2 = 1$ on Edge 4. Compatibility with the edge shape function (4.4.2) requires

$$N_4^k(\zeta_1, \zeta_2, 0) = \bar{N}^k(\xi) = \frac{(1-\xi)(1+\xi)}{4}\bar{\chi}^k(\xi)$$

where $\bar{N}^k(\xi)$ is the one-dimensional hierarchical shape function (4.4.2e). Thus,

$$\bar{\chi}^k(\xi) = \frac{4\bar{N}^k(\xi)}{1-\xi^2}. \tag{4.4.6c}$$

The result can be written in terms of triangular coordinates by using (4.4.6b) to obtain $\xi = \zeta_2 - \zeta_1$; hence,

$$N_4^k(\zeta_1, \zeta_2, \zeta_3) = \zeta_1\zeta_2\bar{\chi}^k(\zeta_2 - \zeta_1), \qquad k = 2, 3, \ldots, p. \tag{4.4.7a}$$

Shape functions along other edges follow by permuting indices, *i.e.*,

$$N_5^k(\zeta_1, \zeta_2, \zeta_3) = \zeta_2\zeta_3\bar{\chi}^k(\zeta_3 - \zeta_2), \tag{4.4.7b}$$

$$N_6^k(\zeta_1, \zeta_2, \zeta_3) = \zeta_3\zeta_1\bar{\chi}^k(\zeta_1 - \zeta_3), \qquad k = 2, 3, \ldots, p. \tag{4.4.7c}$$

It might appear that the shape functions $\bar{\chi}^k(\xi)$ has singularities at $\xi = \pm 1$; however, the one-dimensional hierarchical shape functions have $(1-\xi^2)$ as a factor. Thus, $\bar{\chi}^k(\xi)$ is a polynomial of degree $k-2$. Using (2.5.8), the first four of them are

$$\bar{\chi}^2(\xi) = -\sqrt{6}, \qquad \bar{\chi}^3(\xi) = -\sqrt{10}\xi,$$

$$\bar{\chi}^4(\xi) = -\sqrt{\frac{7}{8}}(5\xi^2 - 1), \qquad \bar{\chi}^5(\xi) = -\sqrt{\frac{9}{8}}(7\xi^3 - 3\xi). \tag{4.4.8}$$

Interior shape functions. The $(p-1)(p-2)/2$ internal shape functions for $p \geq 3$ are products of the bubble function

$$N_7^{3,0,0} = \zeta_1\zeta_2\zeta_3 \tag{4.4.9a}$$

and Legendre polynomials. The Legendre polynomials are functions of two of the three triangular coordinates. Following Szabó and Babuška [7], we present them in terms of $\zeta_2 - \zeta_1$ and ζ_3. Thus,

$$N_7^{4,1,0} = N_7^{3,0,0}P_1(\zeta_2 - \zeta_1), \tag{4.4.9b}$$

$$N_7^{4,0,1} = N_7^{3,0,0}P_1(2\zeta_3 - 1), \tag{4.4.9c}$$

$$N_7^{5,2,0} = N_7^{3,0,0}P_2(\zeta_2 - \zeta_1), \tag{4.4.9d}$$

$$N_7^{5,1,1} = N_7^{3,0,0}P_1(\zeta_2 - \zeta_1)P_1(2\zeta_3 - 1), \tag{4.4.9e}$$

$$N_7^{5,0,2} = N_7^{3,0,0}P_2(2\zeta_3 - 1), \qquad \ldots. \tag{4.4.9f}$$

The shift in ζ_3 ensures that the range of the Legendre polynomials is $[-1, 1]$.

Like the edge shape functions for a square (4.4.2), the edge shape functions for a triangle (4.4.7) are products of a function on the edge ($\bar{\chi}^k(\zeta_i - \zeta_j)$) and a function ($\zeta_i\zeta_j$, $i \neq j$) that blends the edge function onto the element. However, the edge functions for the triangle are not the same as those for the square. The two are related by (4.4.6c). Having the same edge functions for all element shapes simplifies construction of the element stiffness matrices [6]. We can, of course, make the edge functions the same by redefining the blending functions. Thus, using (4.4.6a,c), the edge function for Edge 4 can be $\bar{N}^k(\xi)$ if the blending function is

$$\frac{4\zeta_1\zeta_2}{1-\xi^2}.$$

In a similar manner, using (4.4.2a) and (4.4.6c), the edge function for the shape function $N^k_{3,1}$ can be $\bar{\chi}^k(\xi)$ if the blending function is

$$\frac{\bar{N}_1(\eta)(1-\xi^2)}{4}.$$

Shepard *et al.* [6] show that representations in terms of $\bar{\chi}^k$ involve fewer algebraic operations and, hence, are preferred.

The first three edge and interior shape functions are shown in Figure 4.4.4. A degree p hierarchical approximation on a triangle has $3+3(p-1)_+ + (p-1)_+(p-2)_+/2$ unknowns and shape functions. This function is listed in Table 4.4.1. We see that for $p > 1$, there are two fewer shape functions with triangular elements than with squares. The triangular element is optimal in the sense of using the minimal number of shape functions for a complete polynomial of a given degree. This, however, does not mean that the complexity of solving a given problem is less with triangular elements than with quadrilaterals. This issue depends on the partial differential equations, the geometry, the mesh structure, and other factors.

Carnevali *et al.* [4] introduced shape functions that produce better conditioned element stiffness matrices at higher values of p than the bases presented here [7]. Adjerid *et al.* [1] construct an alternate basis that appears to further reduce ill conditioning at high p.

4.5 Three-Dimensional Shape Functions

Three-dimensional finite element shape functions are constructed in the same manner as in two dimensions. Common element shapes are tetrahedra and hexahedra and we will examine some Lagrange and hierarchical approximations on these elements.

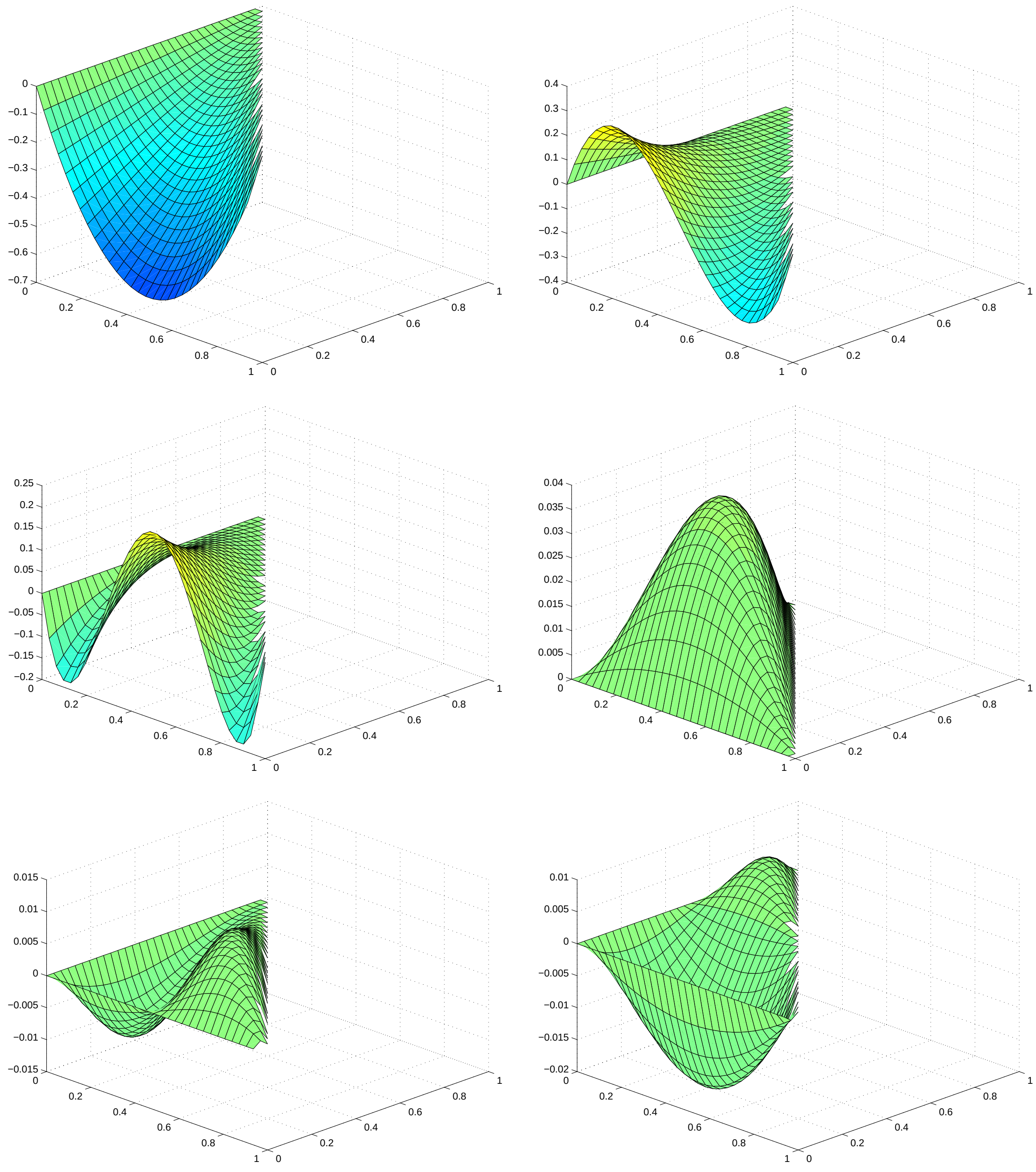


Figure 4.4.4: Hierarchical edge and interior shape functions N_4^2 (top left), N_4^3 (top right), N_4^4 (middle left), $N_7^{3,0,0}$ (middle right), $N_7^{4,1,0}$ (bottom left), $N_7^{4,0,1}$ (bottom right).

4.5.1 Lagrangian Shape Functions on Tetrahedra

Let us begin with a linear shape function on a tetrahedron. We introduce four nodes numbered (for convenience) as 1 to 4 at the vertices of the element (Figure 4.5.1). Imposing the usual Lagrangian conditions that $N_j(x_k, y_k, z_k) = \delta_{jk}$, $j, k = 1, 2, 3, 4$, gives the shape functions as

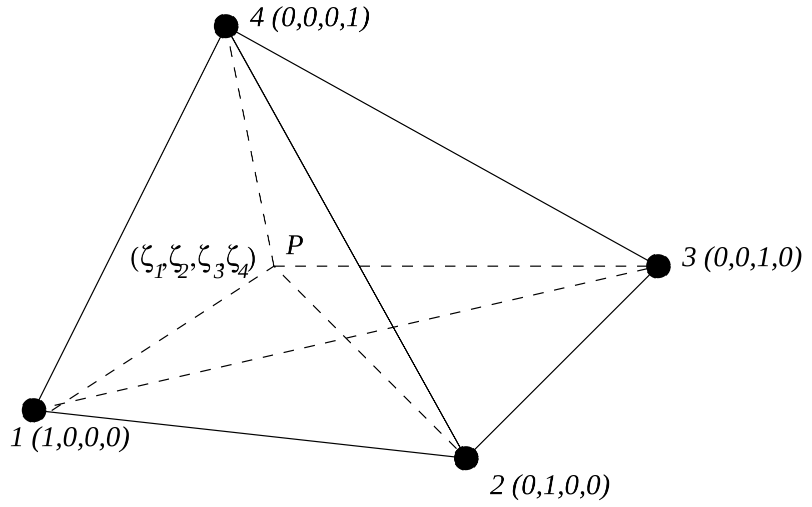


Figure 4.5.1: Node placement for linear shape functions on a tetrahedron and definition of tetrahedral coordinates.

$$N_j(x,y,z) = \frac{D_{k,l,m}(x,y,z)}{C_{j,k,l,m}}, \qquad (j,k,l,m) \text{ a permutation of } 1,2,3,4, \tag{4.5.1a}$$

where

$$D_{k,l,m}(x,y,z) = \det \begin{bmatrix} 1 & x & y & z \\ 1 & x_k & y_k & z_k \\ 1 & x_l & y_l & z_l \\ 1 & x_m & y_m & z_m \end{bmatrix}, \tag{4.5.1b}$$

$$C_{j,k,l,m} = \det \begin{bmatrix} 1 & x_j & y_j & z_j \\ 1 & x_k & y_k & z_k \\ 1 & x_l & y_l & z_l \\ 1 & x_m & y_m & z_m \end{bmatrix}. \tag{4.5.1c}$$

Placing nodes at the vertices produces a linear shape function on each face that is uniquely determined by its values at the three vertices on the face. This guarantees continuity of bases constructed from the shape functions. The restriction of U to element e is

$$U(x,y,z) = \sum_{j=1}^{4} c_j N_j(x,y,z). \tag{4.5.2}$$

As in two dimensions, we may construct higher-order polynomial interpolants by either mapping to a canonical element or by introducing "tetrahedral coordinates." Focusing on the latter approach, let

$$\zeta_j = N_j(x,y,z), \qquad j = 1,2,3,4, \tag{4.5.3a}$$

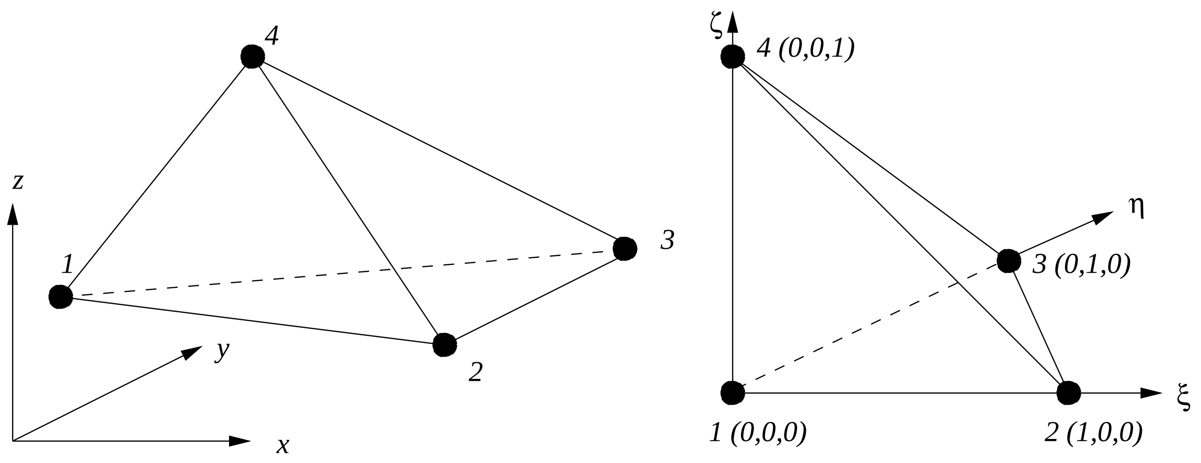


Figure 4.5.2: Transformation of an arbitrary tetrahedron to a right, unit canonical tetrahedron.

and regard ζ_j, $j = 1,2,3,4$, as forming a redundant coordinate system on a tetrahedron. The coordinates of a point P located at $(\zeta_1, \zeta_2, \zeta_3, \zeta_4)$ are (Figure 4.5.1)

$$\zeta_1 = \frac{V_{P234}}{V_{1234}}, \qquad \zeta_2 = \frac{V_{P134}}{V_{1234}}, \qquad \zeta_3 = \frac{V_{P124}}{V_{1234}}, \qquad \zeta_4 = \frac{V_{P123}}{V_{1234}}, \tag{4.5.3b}$$

where V_{ijkl} is the volume of the tetrahedron with vertices at i, j, k, and l. Hence, the coordinates of Vertex 1 are $(1,0,0,0)$, those of Vertex 2 are $(0,1,0,0)$, *etc.* The plane $\zeta = 0$ is the plane A_{234} opposite to vertex 1, *etc.* The transformation from physical to tetrahedral coordinates is

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \\ z_1 & z_2 & z_3 & z_4 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} \zeta_1 \\ \zeta_2 \\ \zeta_3 \\ \zeta_4 \end{bmatrix}. \tag{4.5.4}$$

The coordinate system is redundant as expressed by the last equation.

The transformation of an arbitrary tetrahedron to a right, unit canonical tetrahedron (Figure 4.5.2) follows the same lines, and we may define it as

$$\xi = N_2(x,y,z), \qquad \eta = N_3(x,y,z), \qquad \zeta = N_4(x,y,z). \tag{4.5.5}$$

The face A_{134} (Figure 4.5.2) is mapped to the plane $\xi = 0$, the face A_{124} is mapped to $\eta = 0$, and A_{123} is mapped to $\zeta = 0$. In analogy with the two-dimensional situation, this transformation is really the same as the mapping (4.5.3) to tetrahedral coordinates.

A complete polynomial of degree p in three dimensions has

$$n_p = \frac{(p+1)(p+2)(p+3)}{6} \tag{4.5.6}$$

monomial terms (*cf.*, *e.g.*, Brenner and Scott [3], Section 3.6). With $p = 2$, we have $n_2 = 10$ monomial terms and we can determine Lagrangian shape functions by placing nodes at the four vertices and at the midpoints of the six edges (Figure 4.5.3). With $p = 3$, we have $n_3 = 20$ and we can specify shape functions by placing a node at each of the four vertices, two nodes on each of the six edges, and one node on each of the four faces (Figure 4.5.3). Higher degree polynomials also have nodes in the element's interior. In general there is 1 node at each vertex, $p-1$ nodes on each edge, $(p-1)(p-2)/2$ nodes on each face, and $(p-1)(p-2)(p-3)/6$ nodes in the interior.

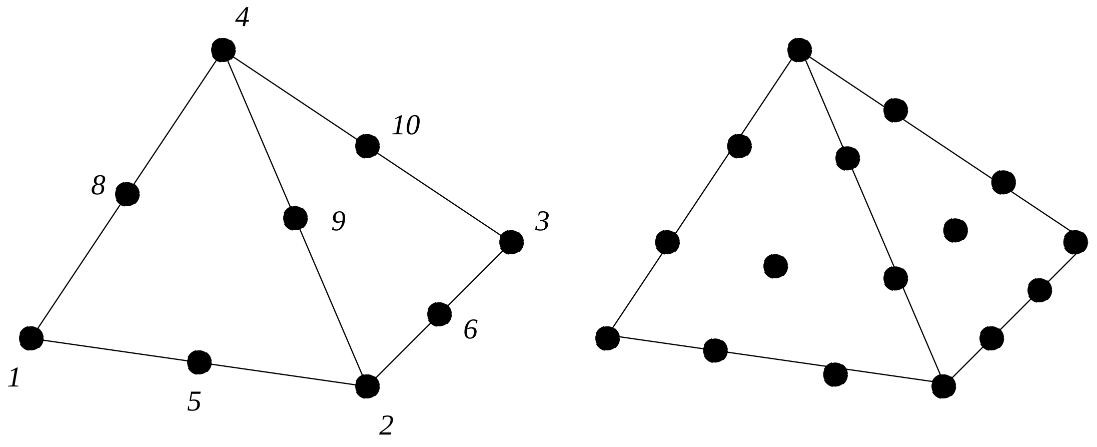


Figure 4.5.3: Node placement for quadratic (left) and cubic (right) interpolants on tetrahedra.

Example 4.5.1. The quadratic shape function N_1^2 associated with vertex Node 1 of a tetrahedron (Figure 4.5.3, left) is required to vanish at all nodes but Node 1. The plane $\zeta_1 = 0$ passes through face A_{234} and, hence, Nodes 2, 3, 4, 6, 9, 10. Likewise, the plane $\zeta_1 = 1/2$ passes through Nodes 5, 7 (not shown), and 8. Thus, N_1^2 must have the form

$$N_1^2(\zeta_1, \zeta_2, \zeta_3, \zeta_4) = \alpha\zeta_1(\zeta_1 - 1/2).$$

Since $N_1^2 = 1$ at Node 1 ($\zeta_1 = 1$), we find $\alpha = 2$ and

$$N_1^2(\zeta_1, \zeta_2, \zeta_3, \zeta_4) = 2\zeta_1(\zeta_1 - 1/2).$$

Similarly, the shape function N_5^2 associated with edge Node 5 (Figure 4.5.3, left) is required to vanish on the planes $\zeta_1 = 0$ (Nodes 2, 3, 4, 6, 9, 10) and $\zeta_2 = 0$ (Nodes 1, 3, 4, 7, 8, 10) and have unit value at Node 5 ($\zeta_1 = \zeta_2 = 1/2$). Thus, it must be

$$N_5^2(\zeta_1, \zeta_2, \zeta_3, \zeta_4) = 4\zeta_1\zeta_2.$$

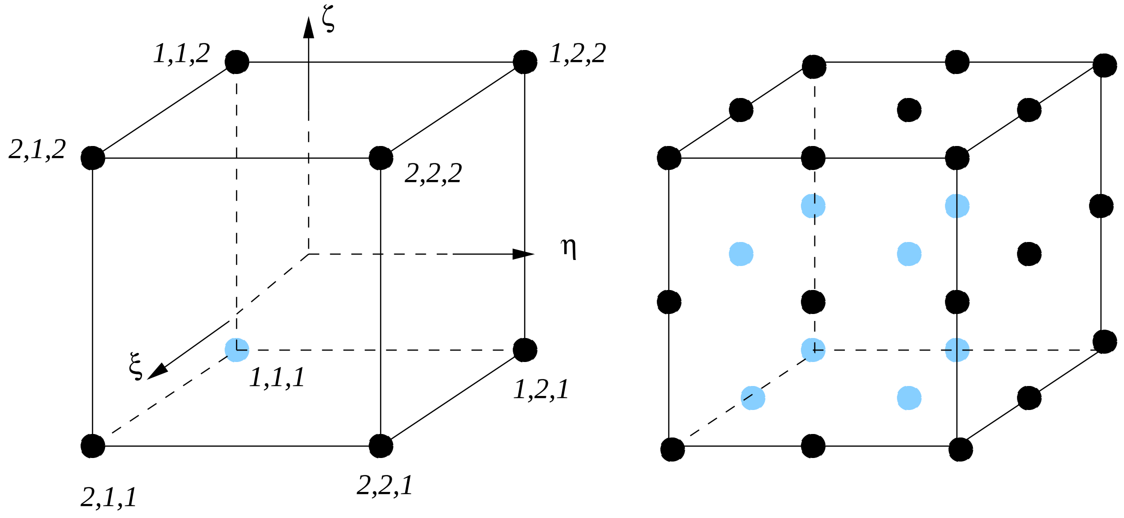


Figure 4.5.4: Node placement for a trilinear (left) and tri-quadratic (right) polynomial interpolants on a cube.

4.5.2 Lagrangian Shape Functions on Cubes

In order to construct a trilinear approximation on the canonical cube $\{\xi, \eta, \zeta | -1 \leq \xi, \eta, \zeta \leq 1\}$, we place eight nodes numbered (i, j, k), $i, j, k = 1, 2$, at its vertices (Figure 4.5.4). The shape function associated with Node (i, j, k) is taken as

$$N_{i,j,k}(\xi, \eta, \zeta) = \bar{N}_i(\xi)\bar{N}_j(\eta)\bar{N}_k(\zeta) \tag{4.5.7a}$$

where $\bar{N}_i(\xi)$, $i = 1, 2$, are the hat function (4.3.1d,e). The restriction of U to this element has the form

$$U(\xi, \eta, \zeta) = \sum_{i=1}^{2}\sum_{j=1}^{2}\sum_{k=1}^{2} c_{i,j,k}N_{i,j,k}(\xi, \eta, \zeta), \tag{4.5.7b}$$

Once again, $c_{i,j,k} = U_{i,j,k} = U(\xi_i, \eta_j, \zeta_k)$.

The placement of nodes at the vertices produces bilinear shape functions on each face of the cube that are uniquely determined by values at their four vertices on that face. Once again, this ensures that shape functions and U are C^0 functions on a uniform grid of cubes or rectangular parallelepipeds. Since each shape function is the product of one-dimensional linear polynomials, the interpolant is a trilinear function of the form

$$U(\xi, \eta, \zeta) = a_1 + a_2\xi + a_3\eta + a_4\zeta + a_5\xi\eta + a_6\eta\zeta + a_7\zeta\xi + a_8\xi\eta\zeta.$$

Other approximations and transformations follow their two-dimensional counterparts. For example, tri-quadratic shape functions on the canonical cube are constructed by placing 27 nodes at the vertices, midsides, midfaces, and centroid of the element (Figure 4.5.4). The shape function associated with Node (i, j, k) is given by (4.5.7a) with $\bar{N}_i(\xi)$ given by (4.3.3b-d).

4.5.3 Hierarchical Approximations

As with the two-dimensional hierarchical approximations described in Section 4.4, we use Szabó and Babuška's [7] shape function with the representation of Shephard *et al.* [6]. The basis for a tetrahedral or a canonical cube begins with the vertex functions (4.5.1) or (4.5.7), respectively. As noted in Section 4.4, higher-order shape functions are written as products

$$N_i^k(x,y,z) = \bar{\chi}^k(\xi,\eta,\zeta)\beta_i(\xi,\eta,\zeta) \tag{4.5.8}$$

of an entity function $\bar{\chi}^k$ and a blending function β_i.

- The *entity function* is defined on a mesh entity (vertex, edge, face, or element) and varies with the degree k of the approximation. It does not depend on the shapes of higher-dimensional entities.

- The *blending function* distributes the entity function over higher-dimensional entities. It depends on the shapes of the higher-dimensional entities but not on k.

The entity functions that are used to construct shape functions for cubic and tetrahedral elements follow.

Edge functions for both cubes and tetrahedra are given by (4.4.6c) and (4.4.2e) as

$$\bar{\chi}^k(\xi) = \frac{\sqrt{2(2k-1)}}{1-\xi^2}\int_{-1}^{\xi} P_{k-1}(\sigma)d\sigma, \qquad k \geq 2, \tag{4.5.9a}$$

where $\xi \in [-1,1]$ is a coordinate on the edge. The first four edge functions are presented in (4.4.8).

Face functions for squares are given by (4.4.3) divided by the square face blending function (4.4.3a)

$$\bar{\chi}^{k,\lambda,\mu}(\xi,\eta) = P_\lambda(\xi)P_\mu(\eta), \qquad \lambda+\mu = k-4, \qquad k \geq 4. \tag{4.5.9b}$$

Here, (ξ,η) are canonical coordinates on the face. The first six square face functions are

$$\begin{aligned} \bar{\chi}^{4,0,0} &= 1, & \bar{\chi}^{5,1,0} &= \xi, \\ \bar{\chi}^{5,0,1} &= \eta, & \bar{\chi}^{6,2,0} &= \frac{3\xi^2-1}{2}, \\ \bar{\chi}^{6,1,1} &= \xi\eta, & \bar{\chi}^{6,0,2} &= \frac{3\eta^2-1}{2}. \end{aligned}$$

Face functions for triangles are given by (4.4.9) divided the triangular face blending function (4.4.9a)

$$\bar{\chi}^{k,\lambda,\mu}(\zeta_1,\zeta_2,\zeta_3) = P_\lambda(\zeta_2-\zeta_1)P_\mu(2\zeta_3-1), \qquad \lambda+\mu = k-3, \qquad k \geq 3. \tag{4.5.9c}$$

As with square faces, $(\zeta_1, \zeta_2, \zeta_3)$ form a canonical coordinate system on the face. The first six triangular face functions are

$$\bar{\chi}^{3,0,0} = 1, \qquad \bar{\chi}^{4,1,0} = \zeta_2 - \zeta_1,$$
$$\bar{\chi}^{4,0,1} = 2\zeta_3 - 1, \qquad \bar{\chi}^{5,2,0} = \frac{3(\zeta_2 - \zeta_1)^2 - 1}{2},$$
$$\bar{\chi}^{5,1,1} = (\zeta_2 - \zeta_1)(2\zeta_3 - 1), \qquad \bar{\chi}^{5,0,2} = \frac{3(2\zeta_3 - 1)^2 - 1}{2}.$$

Now, let's turn to the blending functions.

The *tetrahedral element blending function* for an edge is

$$\beta_{ij}(\zeta_1, \zeta_2, \zeta_3, \zeta_4) = \zeta_i \zeta_j \tag{4.5.10a}$$

when the edge is directed from Vertex i to Vertex j. Using either Figure 4.5.2 or Figure 4.5.3 as references, we see that the blending function ensures that the shape function vanishes on the two faces not containing the edge to maintain continuity. Thus, if $i = 1$ and $j = 2$, the blending function for Edge $(1, 2)$ (which is marked with a 5 on the left of Figure 4.5.3) vanishes on the faces $\zeta_1 = 0$ (Face A_{234}) and $\zeta_2 = 0$ (Face A_{134}).

The blending function for a face is

$$\beta_{ijk}(\zeta_1, \zeta_2, \zeta_3, \zeta_4) = \zeta_i \zeta_j \zeta_k \tag{4.5.10b}$$

when the vertices on the face are i, j, and k. Again, the blending function ensures that the shape function vanishes on all faces but A_{ijk}. Again referring to Figures 4.5.2 or 4.5.3, the blending function β_{123} vanishes when $\zeta_1 = 0$ (Face A_{234}), $\zeta_2 = 0$ (Face A_{134}), and $\zeta_3 = 0$ (Face A_{124}).

The *cubic element blending function* for an edge is more difficult to write with our notation. Instead of writing the general result, let's consider an edge parallel to the ξ axis. Then

$$\beta_{1-2,j,k}(\xi, \eta, \zeta) = \frac{1 - \xi^2}{4} \bar{N}_j(\eta) \bar{N}_k(\zeta). \tag{4.5.11a}$$

The factor $(1 - \xi^2)/4$ adjusts the edge function to (4.5.9) as described in the paragraph following (4.4.9). The one-dimensional shape functions $\bar{N}_j(\eta)$ and $\bar{N}_k(\zeta)$ ensure that the shape function vanishes on all faces not containing the edge. Blending functions for other edges are obtained by cyclic permutation of ξ, η, and ζ and the index. Thus, referring to Figure 4.5.4, the edge function for the edge connecting vertices $2, 1, 1$ and $2, 2, 1$ is

$$\beta_{2,1-2,1}(\xi, \eta, \zeta) = \frac{1 - \eta^2}{4} \bar{N}_2(\xi) \bar{N}_1(\zeta).$$

Since $\bar{N}_2(-1) = 0$ (*cf.* (4.5.7b)), the shape function vanishes on the rear face of the cube shown in Figure 4.5.4. Since $\bar{N}_1(1) = 0$, the shape function vanishes on the top face of the cube of Figure 4.5.4. Finally, the shape function vanishes at $\eta = \pm 1$ and, hence, on the left and right faces of the cube of Figure 4.5.4. Thus, the blending function (4.5.11a) has ensured that the shape function vanishes on all but the bottom and front faces of the cube of Figure 4.5.4.

The cubic face blending function for a face perpendicular to the ξ axis is

$$\beta_{i,j,k}(\xi,\eta,\zeta) = \bar{N}_i(\xi)(1-\eta^2)(1-\zeta^2). \tag{4.5.11b}$$

Referring to Figure 4.5.4, the quadratic terms in η and ζ ensure that the shape function vanishes on the right, left ($\eta = \pm 1$), top, and bottom ($\zeta = \pm 1$) faces. The one-dimensional shape function $\bar{N}_i(\xi)$ vanishes on the rear ($\xi = -1$) face when $i = 1$ and on the front ($\xi = 1$) face when $i = 2$; thus, the shape function vanishes on all faces but the one to which it is associated.

Finally, there are *elemental* shape functions. For tetrahedra, there are $(p-1)(p-2)(p-3)/6$ elemental functions for $p \geq 4$ that are given by

$$\begin{aligned} N_0^{k,\lambda,\mu,\nu}(\zeta_1,\zeta_2,\zeta_3,\zeta_4) &= \zeta_1\zeta_2\zeta_3\zeta_4 P_\lambda(\zeta_2-\zeta_1)P_\mu(2\zeta_3-1)P_\nu(2\zeta_4-1), \\ &\forall\ \lambda+\mu+\nu = k-4, \qquad k = 4,5,\ldots,p. \end{aligned} \tag{4.5.12a}$$

The subscript 0 is used to identify the element's centroid. The shape functions vanish on all element faces as indicated by the presence of the multiplier $\zeta_1\zeta_2\zeta_3\zeta_4$. We could also split this function into the product of an elemental function involving the Legendre polynomials and the blend involving the product of the tetrahedral coordinates. However, this is not necessary.

For $p \geq 6$ there are the following elemental shape functions for a cube

$$N_0^{k,\lambda,\mu,\nu}(\xi,\eta,\zeta) = (1-\xi^2)(1-\eta^2)(1-\zeta^2)P_\lambda(\xi)P_\mu(\eta)P_\nu(\zeta), \qquad \forall\ \lambda+\mu+\nu = k-6. \tag{4.5.12b}$$

Again, the shape function vanishes on all faces of the element to maintain continuity. Adding, we see that there are $(p-5)_+(p-4)_+(p-3)_+/6$ element modes for a polynomial of order p.

Shephard *et al.* [6] also construct blending functions for pyramids, wedges, and prisms. They display several shape functions and also present entity functions using the basis of Carnevali *et al.* [4].

Problems

1. Construct the shape functions associated with a vertex, an edge, and a face node for a cubic Lagrangian interpolant on the tetrahedron shown on the right of Figure 4.5.3. Express your answer in the tetrahedral coordinates (4.5.3).

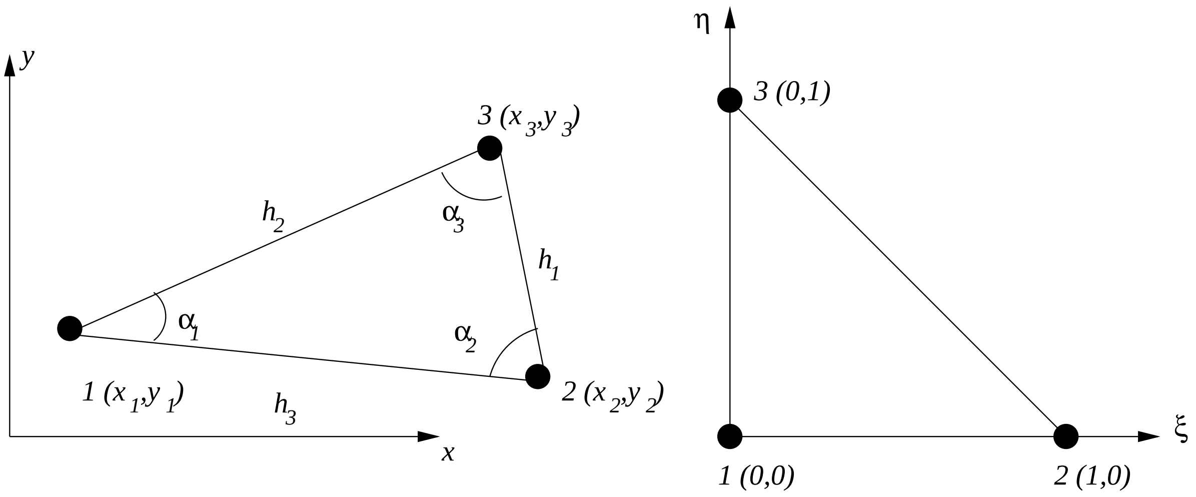


Figure 4.6.1: Nomenclature for a finite element in the physical (x, y)-plane and for its mapping to a canonical element in the computational (ξ, η)-plane.

4.6 Interpolation Error Analysis

We conclude this chapter with a brief discussion of the errors in interpolating a function u by a piecewise polynomial function U. This work extends our earlier study in Section 2.6 to multi-dimensional situations. Two- and three-dimensional interpolation is, naturally, more complex. In one dimension, it was sufficient to study limiting processes where mesh spacings tend to zero. In two and three dimensions, we must also ensure that element shapes cannot be too distorted. This usually means that elements cannot become too thin as the mesh is refined. We have been using coordinate mappings to construct bases. Concentrating on two-dimensional problems, the coordinate transformation from a canonical element in, say, the (ξ, η)-plane to an actual element in the (x, y)-plane must be such that no distorted elements are produced.

Let's focus on triangular elements and consider a linear mapping of a canonical unit, right, 45° triangle in the (ξ, η)-plane to an element e in the (x, y)-plane (Figure 4.6.1). More complex mappings will be discussed in Chapter 5. Using the transformation (4.2.8) to triangular coordinates in combination with the definitions (4.2.6) and (4.2.7) of the canonical variables, we have

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} \zeta_1 \\ \zeta_2 \\ \zeta_3 \end{bmatrix} = \begin{bmatrix} x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1-\xi-\eta \\ \xi \\ \eta \end{bmatrix}. \tag{4.6.1}$$

The Jacobian of this transformation is

$$\mathbf{J}_e := \begin{bmatrix} x_\xi & x_\eta \\ y_\xi & y_\eta \end{bmatrix}. \tag{4.6.2a}$$

Differentiating (4.6.1), we find the determinant of this Jacobian as

$$\det(\mathbf{J}_e) = (x_2 - x_1)(y_3 - y_1) - (x_3 - x_1)(y_2 - y_1). \tag{4.6.2b}$$

Lemma 4.6.1. *Let h_e be the longest edge and α_e be the smallest angle of Element e, then*

$$\frac{h_e^2}{2}\sin\alpha_e \leq \det(\mathbf{J}_e) \leq h_e^2 \sin\alpha_e. \tag{4.6.3}$$

Proof. Label the vertices of Element e as 1, 2, and 3; their angles as $\alpha_1 \leq \alpha_2 \leq \alpha_3$; and the lengths of the edges opposite these angles as h_1, h_2, and h_3 (Figure 4.6.1). With $\alpha_1 = \alpha_e$ being the smallest angle of Element e, write the determinant of the Jacobian as

$$\det(\mathbf{J}_e) = h_2 h_3 \sin\alpha_e.$$

Using the law of sines we have $h_1 \leq h_2 \leq h_3 = h_e$. Replacing h_2 by h_3 in the above expression yields the right-hand inequality of (4.6.3). The triangular inequality gives $h_3 < h_1 + h_2$. Thus, at least one edge, say, $h_2 > h_3/2$. This yields the left-hand inequality of (4.6.3). □

Theorem 4.6.1. *Let $\theta(x, y) \in H^s(\Omega_e)$ and $\tilde{\theta}(\xi, \eta) \in H^s(\Omega_0)$ be such that $\theta(x, y) = \tilde{\theta}(\xi, \eta)$ where Ω_e is the domain of element e and Ω_0 is the domain of the canonical element. Under the linear transformation (4.6.1), there exist constants c_s and C_s, independent of θ, $\tilde{\theta}$, h_e, and α_e such that*

$$c_s \sin^{s-1/2}\alpha_e h_e^{s-1}|\theta|_{s,\Omega_e} \leq |\tilde{\theta}|_{s,\Omega_0} \leq C_s \sin^{-1/2}\alpha_e h_e^{s-1}|\theta|_{s,\Omega_e} \tag{4.6.4a}$$

where the Sobolev seminorm is

$$|\theta|_{s,\Omega_e}^2 = \sum_{|\boldsymbol{\kappa}|=s} \iint_{\Omega_e} (D^{\boldsymbol{\kappa}}\theta)^2 dxdy \tag{4.6.4b}$$

with $D^{\boldsymbol{\kappa}}u$ being a partial derivative of order $|\boldsymbol{\kappa}| = s$ (cf. Section 3.2).

Proof. Let us begin with $s = 0$, where

$$\iint_{\Omega_e} \theta^2 dxdy = \det(\mathbf{J}_e) \iint_{\Omega_0} \tilde{\theta}^2 d\xi d\eta$$

or

$$|\theta|_{0,\Omega_e}^2 = \det(\mathbf{J}_e)|\tilde{\theta}|_{0,\Omega_0}^2.$$

Dividing by $\det(\mathbf{J}_e)$ and using (4.6.3)

$$\frac{|\theta|_{0,\Omega_e}^2}{\sin\alpha_e h_e^2} \leq |\tilde{\theta}|_{0,\Omega_0}^2 \leq \frac{2|\theta|_{0,\Omega_e}^2}{\sin\alpha_e h_e^2}.$$

Taking a square root, we see that (4.6.4a) is satisfied with $c_0 = 1$ and $C_0 = \sqrt{2}$.

With $s = 1$, we use the chain rule to get

$$\theta_x = \tilde{\theta}_\xi \xi_x + \tilde{\theta}_\eta \eta_x, \qquad \theta_y = \tilde{\theta}_\xi \xi_y + \tilde{\theta}_\eta \eta_y.$$

Then,

$$|\theta|^2_{1,\Omega_e} = \iint\limits_{\Omega_e} (\theta_x^2 + \theta_y^2) dxdy = \det(\mathbf{J}_e) \iint\limits_{\Omega_0} (g_{1,e}\tilde{\theta}_\xi^2 + 2g_{2,e}\tilde{\theta}_\xi\tilde{\theta}_\eta + g_{3,e}\tilde{\theta}_\eta^2) d\xi d\eta$$

where

$$g_{1,e} = \xi_x^2 + \xi_y^2, \qquad g_{2,e} = \xi_x\eta_x + \xi_y\eta_y, \qquad g_{3,e} = \eta_x^2 + \eta_y^2.$$

Applying the inequality $ab \leq (a^2 + b^2)/2$ to the center term on the right yields

$$|\theta|^2_{1,e} \leq \det(\mathbf{J}_e) \iint\limits_{\Omega_0} [g_{1,e}\tilde{\theta}_\xi^2 + g_{2,e}(\tilde{\theta}_\xi^2 + \tilde{\theta}_\eta^2) + g_{3,e}\tilde{\theta}_\eta^2] d\xi d\eta.$$

Letting

$$\delta = \max(|g_{1,e} + g_{2,e}|, |g_{3,e} + g_{2,e}|)$$

and using (4.6.4b), we have

$$|\theta|^2_{1,\Omega_e} \leq \det(\mathbf{J}_e)\delta|\tilde{\theta}|^2_{1,\Omega_0}. \tag{4.6.5a}$$

Either by using the chain rule above with $\theta = x$ and y or by inverting the mapping (4.6.1), we may show that

$$\xi_x = \frac{y_\eta}{\det(\mathbf{J}_e)}, \qquad \xi_y = -\frac{x_\eta}{\det(\mathbf{J}_e)}, \qquad \eta_x = -\frac{y_\xi}{\det(\mathbf{J}_e)}, \qquad \eta_y = -\frac{x_\xi}{\det(\mathbf{J}_e)}.$$

From (4.6.2), $|x_\xi|$, $|x_\eta|$, $|y_\xi|$, $|y_\eta| \leq h_e$; thus, using (4.6.3), we have $|\xi_x|$, $|\xi_y|$, $|\eta_x|$, $|\eta_y| \leq 2/(h_e \sin\alpha_e)$. Hence,

$$\delta \leq \frac{16}{(h_e \sin\alpha_e)^2}.$$

Using this result and (4.6.3) with (4.6.5a), we find

$$|\theta|^2_{1,\Omega_e} \leq \frac{16}{\sin\alpha_e}|\tilde{\theta}|^2_{1,\Omega_0}. \tag{4.6.5b}$$

Hence, the left-hand inequality of (4.6.4a) is established with $c_1 = 1/4$.

To establish the right inequality, we invert the transformation and proceed from Ω_0 to Ω_e to obtain

$$|\tilde{\theta}|^2_{1,\Omega_0} \leq \frac{\tilde{\delta}|\theta|^2_{1,\Omega_e}}{\det(\mathbf{J}_e)} \tag{4.6.6a}$$

with

$$\tilde{\delta} = \max(|\tilde{g}_{1,e} + \tilde{g}_{2,e}|, |\tilde{g}_{3,e} + \tilde{g}_{2,e}|),$$

$$\tilde{g}_{1,e} = x_\xi^2 + x_\eta^2, \qquad \tilde{g}_{2,e} = x_\xi y_\xi + x_\eta y_\eta, \qquad \tilde{g}_{3,e} = y_\xi^2 + y_\eta^2.$$

We've indicated that $|x_\xi|$, $|x_\eta|$, $|y_\xi|$, $|y_\eta| \leq h_e$. Thus, $\tilde{\delta} \leq 4h_e^2$ and, using (4.6.3), we find

$$|\tilde{\theta}|_{1,\Omega_0}^2 \leq \frac{8}{\sin \alpha_e} |\theta|_{1,\Omega_e}^2. \tag{4.6.6b}$$

Thus, the right inequality of (4.6.4b) is established with $C_1 = 2\sqrt{2}$.

The remainder of the proof follows the same lines and is described in Axelsson and Barker [2]. □

With Theorem 4.6.1 established, we can concentrate on estimating interpolation errors on the canonical triangle. For simplicity, we'll use the Lagrange interpolating polynomial

$$\tilde{U}(\xi, \eta) = \sum_{j=1}^{n} \tilde{u}(\xi_j, \eta_j) N_j(\xi, \eta), \tag{4.6.7}$$

with n being the number of nodes on the standard triangle. However, with minor alterations, the results apply to other bases and, indeed, other element shapes. We proceed with one preliminary theorem and then present the main result.

Theorem 4.6.2. *Let p be the largest integer for which the interpolant (4.6.7) is exact when $\tilde{u}(\xi, \eta)$ is a polynomial of degree p. Then, there exists a constant $C > 0$ such that*

$$|\tilde{u} - \tilde{U}|_{s,\Omega_0} \leq C |\tilde{u}|_{p+1,\Omega_0}, \qquad \forall u \in H^{p+1}(\Omega_0), \qquad s = 0, 1, \ldots, p+1. \tag{4.6.8}$$

Proof. The proof utilizes the Bramble-Hilbert Lemma and is presented in Axelsson and Barker [2]. □

Theorem 4.6.3. *Let Ω be a polygonal domain that has been discretized into a net of triangular elements Ω_e, $e = 1, 2, \ldots, N_\Delta$. Let h and α denote the largest element edge and smallest angle in the mesh, respectively. Let p be the largest integer for which (4.6.7) is exact when $\tilde{u}(\xi, \eta)$ is a complete polynomial of degree p. Then, there exists a constant $C > 0$, independent of $u \in H^{p+1}$ and the mesh, such that*

$$|u - U|_s \leq \frac{Ch^{p+1-s}}{[\sin \alpha]^s} |u|_{p+1}, \qquad \forall u \in H^{p+1}(\Omega), \qquad s = 0, 1. \tag{4.6.9}$$

Remark 1. The results are restricted $s = 0, 1$ because, typically, $U \in H^1 \cap H^{p+1}$.

Proof. Consider an element e and use the left inequality of (4.6.4a) with θ replaced by $u-U$ to obtain

$$|u-U|^2_{s,\Omega_e} \leq c_s^{-2} \sin^{-2s+1} \alpha_e h_e^{-2s+2} |\tilde{u} - \tilde{U}|^2_{s,\Omega_0}.$$

Next, use (4.6.8)

$$|u-U|^2_{s,\Omega_e} \leq c_s^{-2} \sin^{-2s+1} \alpha_e h_e^{-2s+2} C |\tilde{u}|^2_{p+1,\Omega_0}.$$

Finally, use the right inequality of (4.6.4a) to obtain

$$|u-U|^2_{s,\Omega_e} \leq c_s^{-2} \sin^{-2s+1} \alpha_e h_e^{-2s+2} C C^2_{p+1} \sin^{-1} \alpha_e h_e^{2p} |u|^2_{p+1,\Omega_e}.$$

Combining the constants

$$|u-U|^2_{s,\Omega_e} \leq C \sin^{-2s} \alpha_e h_e^{2(p+1-s)} |u|^2_{p+1,\Omega_e}.$$

Summing over the elements and taking a square root gives (4.6.9). □

A similar result for rectangles follows.

Theorem 4.6.4. *Let the rectangular domain Ω be discretized into a mesh of rectangular elements Ω_e, $e = 1, 2, \ldots, N_\Delta$. Let h and β denote the largest element edge and smallest edge ratio in the mesh, respectively. Let p be the largest integer for which (4.6.7) is exact when $\tilde{u}(\xi,\eta)$ is a complete polynomial of degree p. Then, there exists a constant $C > 0$, independent of $u \in H^{p+1}$ and the mesh, such that*

$$|u-U|_s \leq \frac{Ch^{p+1-s}}{\beta^s} |u|_{p+1}, \qquad \forall u \in H^{p+1}(\Omega), \qquad s = 0, 1. \tag{4.6.10}$$

Proof. The proof follows the lines of Theorem 4.6.3 [2]. □

Thus, small and large (near π) angles in triangular meshes and small aspect ratios (the minimum to maximum edge ratio of an element) β in a rectangular mesh must be avoided. If these quantities remain bounded then the mesh is uniform as expressed by the following definition.

Definition 4.6.1. A family of finite element meshes Δ_h is *uniform* if all angles of all elements are bounded away from 0 and π and all aspect ratios are bounded away from zero as the element size $h \to 0$.

With such uniform meshes, we can combine Theorems 4.6.2, 4.6.3, and 4.6.4 to obtain a result that appears more widely in the literature.

Theorem 4.6.5. *Let a family of meshes Δ_h be uniform and let the polynomial interpolant U of $u \in H^{p+1}$ be exact whenever u is a complete polynomial of degree p. Then there exists a constant $C > 0$ such that*

$$|u-U|_s \leq Ch^{p+1-s} |u|_{p+1}, \qquad s = 0, 1. \tag{4.6.11}$$

Proof. Use the bounds on α and β with (4.6.9) and (4.6.10) to redefine the constant C and obtain (4.6.11). $\square$

Theorems 4.6.2 - 4.6.5 only apply when $u \in H^{p+1}$. If u has a singularity and belongs to H^{q+1}, $q < p$, then the convergence rate is reduced to

$$|u - U|_s \leq Ch^{q+1-s}|u|_{q+1}, \qquad s = 0, 1. \tag{4.6.12}$$

Thus, there appears to be little benefit to using p *th*-degree piecewise-polynomial interpolants in this case. However, in some cases, highly graded nonuniform meshes can be created to restore a higher convergence rate.

Bibliography

[1] S. Adjerid, M. Aiffa, and J.E. Flaherty. Hierarchical finite element bases for triangular and tetrahedral elements. *Computer Methods in Applied Mechanics and Engineering*, 2000. to appear.

[2] O. Axelsson and V.A. Barker. *Finite Element Solution of Boundary Value Problems.* Academic Press, Orlando, 1984.

[3] S.C. Brenner and L.R. Scott. *The Mathematical Theory of Finite Element Methods.* Springer-Verlag, New York, 1994.

[4] P. Carnevali, R.V. Morric, Y.Tsuji, and B. Taylor. New basis functions and computational procedures for p-version finite element analysis. *International Journal of Numerical Methods in Enginneering*, 36:3759–3779, 1993.

[5] S. Dey, M.S. Shephard, and J.E. Flaherty. Geometry-based issues associated with p-version finite element computations. *Computer Methods in Applied Mechanics and Engineering*, 150:39 – 50, 1997.

[6] M.S. Shephard, S. Dey, and J.E. Flaherty. A straightforward structure to construct shape functions for variable p-order meshes. *Computer Methods in Applied Mechanics and Engineering*, 147:209–233, 1997.

[7] B. Szabó and I. Babuška. *Finite Element Analysis.* John Wiley and Sons, New York, 1991.

[8] O.C. Zienkiewicz. *The Finite Element Method.* McGraw-Hill, New York, third edition, 1977.

Chapter 5

Mesh Generation and Assembly

5.1 Introduction

There are several reasons for the popularity of finite element methods. Large code segments can be implemented for a wide class of problems. The software can handle complex geometry. Little or no software changes are needed when boundary conditions change, domain shapes change, or coefficients vary. A typical finite element software framework contains a *preprocessing module* to define the problem geometry and data; a *processing module* to assemble and solve the finite element system; and a *postprocessing module* to output the solution and calculate additional quantities of interest. The preprocessing module

- creates a computer model of the problem domain Ω, perhaps, using a computer aided design (CAD) system;
- discretizes Ω into a finite element mesh;
- creates geometric and mesh databases describing the mesh entities (vertices, edges, faces and elements) and their relationships to each other and to the problem geometry; and
- defines problem-dependent data such as coefficient functions, loading, initial data, and boundary data.

The processing module

- generates element stiffness and mass matrices and load vectors;
- assembles the global stiffness and mass matrices and load vector;
- enforces any essential boundary conditions; and

- solves the linear (or nonlinear) algebraic system for the finite element solution.

The postprocessing modules

- calculates additional quantities of interest, such as stresses, total energy, and *a posteriori* error estimates, and
- stores and displaying solution information.

In this chapter, we study the preprocessing and processing steps with the exception of the geometrical description and solution procedures. The former topic is not addressed while the latter subject will be covered in Chapter 11.

5.2 Mesh Generation

Discretizing two-dimensional domains into triangular or quadrilateral finite element meshes can either be a simple or difficult task depending on geometric or solution complexities. Discretizing three-dimensional domains is currently not simple. Uniform meshes may be appropriate for some problems having simple geometric shapes, but, even there, nonuniform meshes might provide better performance when solutions vary rapidly, *e.g.*, in boundary layers. Finite element techniques and software have always been associated with unstructured and nonuniform meshes. Early software left it to users to generate meshes manually. This required the entry of the coordinates of all element vertices. Node and element indexing, typically, was also done manually. This is a tedious and error prone process that has largely been automated, at least in two dimensions. Adaptive solution-based mesh refinement procedures concentrate meshes in regions of rapid solution variation and attempt to automate the task of modifying (refining/coarsening) an existing mesh [1, 5, 6, 9, 11]. While we will not attempt a thorough treatment of all approaches, we will discuss the essential ideas of mesh generation by (*i*) mapping techniques where a complex domain is transformed into a simpler one where a mesh may be easily generated and (*ii*) direct techniques where a mesh is generated on the original domain.

5.2.1 Mesh Generation by Coordinate Mapping

Scientists and engineers have used coordinate mappings for some time to simplify geometric difficulties. The mappings can either employ analytical functions or piecewise polynomials as presented in Chapter 4. The procedure begins with mappings

$$x = f_1(\xi, \eta), \qquad y = f_2(\xi, \eta)$$

that relate the problem domain in physical (x, y) space to its image in the simpler (ξ, η) space. A simply connected region and its computational counterpart appear in Figure 5.2.1. It will be convenient to introduce the vectors

$$\mathbf{x}^T = [x, y], \qquad \mathbf{f}(\xi, \eta)^T = [f_1(\xi, \eta), f_2(\xi, \eta)] \tag{5.2.1a}$$

and write the coordinate transformation as

$$\mathbf{x} = \mathbf{f}(\xi, \eta) \tag{5.2.1b}$$

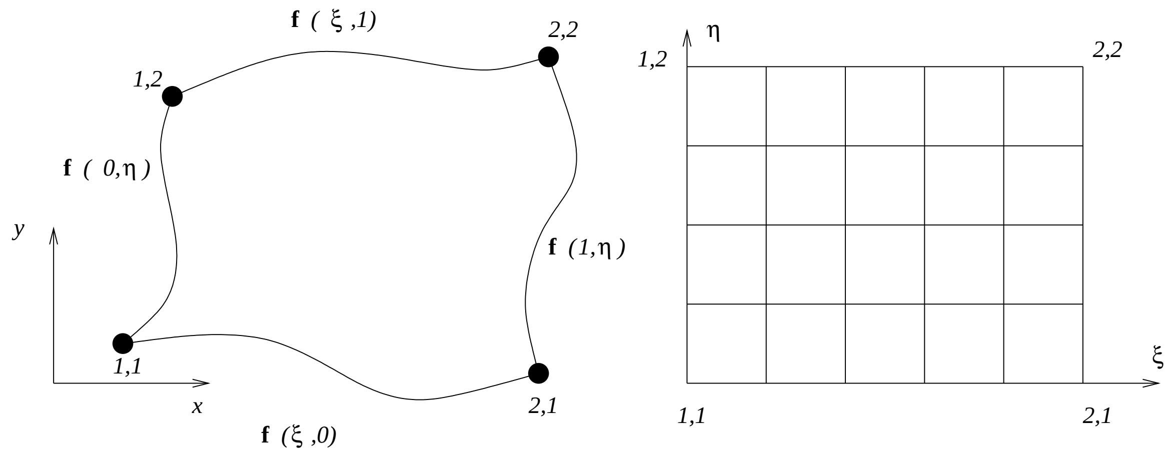


Figure 5.2.1: Mapping of a simply connected region (left) onto a rectangular computational domain (right).

In Figure 5.2.1, we show a region with four segments $\mathbf{f}(\xi, 0)$, $\mathbf{f}(\xi, 1)$, $\mathbf{f}(0, \eta)$, and $\mathbf{f}(1, \eta)$ that are related to the computational lines $\xi = 0$, $\xi = 1$, $\eta = 0$, and $\eta = 1$, respectively. (The four curved segments may involve different functions, but we have written them all as $\mathbf{f}$ for simplicity.)

Also consider the projection operators

$$\mathbf{x} = \mathcal{P}_\xi(\mathbf{f}) = \bar{N}_1(\xi)\mathbf{f}(0, \eta) + \bar{N}_2(\xi)\mathbf{f}(1, \eta), \tag{5.2.2a}$$

$$\mathbf{x} = \mathcal{P}_\eta(\mathbf{f}) = \bar{N}_1(\eta)\mathbf{f}(\xi, 0) + \bar{N}_2(\eta)\mathbf{f}(\xi, 1), \tag{5.2.2b}$$

where

$$\bar{N}_1(\xi) = 1 - \xi, \tag{5.2.2c}$$

and

$$\bar{N}_2(\xi) = \xi \tag{5.2.2d}$$

are the familiar hat functions scaled to the interval $0 \leq \xi \leq 1$.

As shown in Figure 5.2.2, the mapping $\mathbf{x} = \mathcal{P}_\xi(\mathbf{f})$ transforms the left and right edges of the domain correctly, but ignores the top and bottom while the mapping $\mathbf{x} = \mathcal{P}_\eta(\mathbf{f})$ transforms the top and bottom boundaries correctly but not the sides. Coordinate lines of constant ξ and η are mapped as either curves or straight lines on the physical domain.

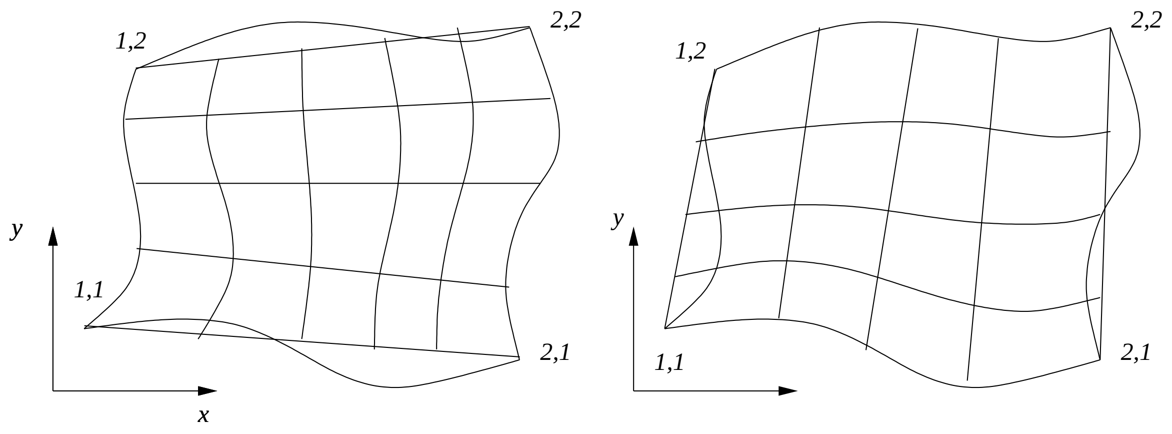


Figure 5.2.2: The transformations $\mathbf{x} = \mathcal{P}_\xi(\mathbf{f})$ (left) and $\mathbf{x} = \mathcal{P}_\eta(\mathbf{f})$ (right) as applied to the simply-connected domain shown in Figure 5.2.1.

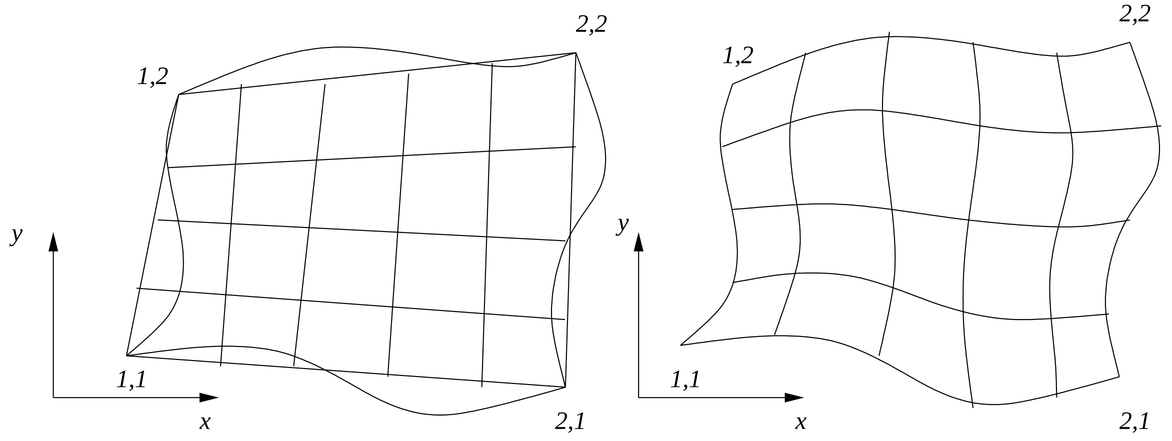


Figure 5.2.3: Illustrations of the transformations $\mathbf{x} = \mathcal{P}_\xi\mathcal{P}_\eta(\mathbf{f})$ (left) and $\mathbf{x} = \mathcal{P}_\xi \oplus \mathcal{P}_\eta(\mathbf{f})$ (right) as applied to the simply-connected domain shown in Figure 5.2.1.

With a goal of constructing an effective mapping, let us introduce the tensor product and Boolean sums of the projections (5.2.2) as

$$\mathbf{x} = \mathcal{P}_\xi\mathcal{P}_\eta(\mathbf{f}) = \sum_{i=1}^{2}\sum_{j=1}^{2} \bar{N}_i(\xi)\bar{N}_j(\eta)\mathbf{f}(i-1, j-1) \tag{5.2.3a}$$

$$\mathbf{x} = \mathcal{P}_\xi \oplus \mathcal{P}_\eta(\mathbf{f}) = \mathcal{P}_\xi(\mathbf{f}) + \mathcal{P}_\eta(\mathbf{f}) - \mathcal{P}_\xi\mathcal{P}_\eta(\mathbf{f}). \tag{5.2.3b}$$

An application of these transformations to a simply-connected domain is shown in Figure 5.2.3. The transformation (5.2.3a) is a bilinear function of ξ and η while (5.2.3b) is clearly the one needed to map the simply connected domain onto the computational plane. Lines of constant ξ and η become curves in the physical domain (Figure 5.2.3).

Although these transformations are simple, they have been used to map relatively complex two- and three-dimensional regions. Two examples involving the flow about an airfoil are shown in Figure 5.2.4. With the transformation shown at the top of the figure, the entire surface of the airfoil is mapped to $\eta = 0$ (2-3). A cut is made from the trailing edge of the airfoil and the curve so defined is mapped to the left ($\xi = 0$, 2-1) and right ($\xi = 0$, 3-4) edges of the computational domain. The entire far field is mapped to the top ($\eta = 1$, 1-4) of the computational domain. Lines of constant ξ are rays from the airfoil surface to the far field boundary in the physical plane. Lines of constant η are closed curves encircling the airfoil. Meshes constructed in this manner are called "O-grids." In the bottom of Figure 5.2.4, the surface of the airfoil is mapped to a portion (2-3) of the ξ axis. The cut from the trailing edge is mapped to the rest (1-2 and 3-4) of the axis. The (right) outflow boundary is mapped to the left (1-5) and right (4-6) edges of the computational domain, and the top, left, and bottom far field boundaries are mapped to the top ($\eta = 1$, 5-6) of the computational domain. Lines of constant ξ become curves beginning and ending at the outflow boundary and surrounding the airfoil. Lines of constant η are rays from the airfoil surface or the cut to the outer boundary. This mesh is called a "C-grid."

5.2.2 Unstructured Mesh Generation

There are several approaches to unstructured mesh generation. Early attempts used manual techniques where point-coordinates were explicitly defined. Semi-automatic mesh generation required manual input of a coarse mesh which could be uniformly refined by dividing each element edge into K segments and connecting segments on opposite sides of an element to create K^2 (triangular) elements. More automatic procedures use advancing fronts, point insertion, and recursive bisection. We'll discuss the latter procedure and briefly mention the former.

With recursive bisection [3], a two-dimensional region Ω is embedded in a square "universe" that is recursively quartered to create a set of disjoint squares called *quadrants.* Quadrants are related through a hierarchical *quadtree* structure. The original square universe is regarded as the root of the tree and smaller quadrants created by subdivision are regarded as offspring of larger ones. Quadrants intersecting $\partial\Omega$ are recursively

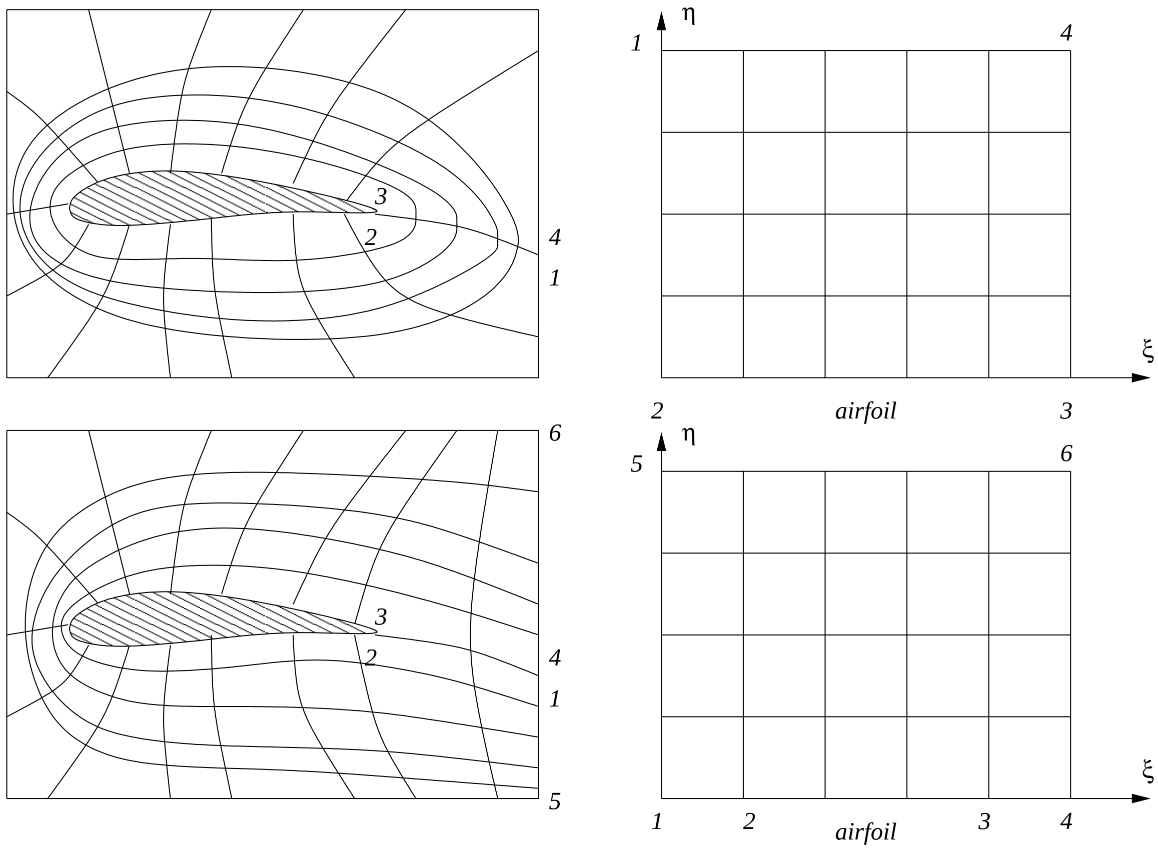


Figure 5.2.4: "O-grid" (top) and "C-grid" (bottom) mappings of the flow about an airfoil.

quartered until a prescribed spatial resolution of Ω is obtained. At this stage, quadrants that are leaf nodes of the tree and intersect $\Omega \cup \partial\Omega$ are further divided into small sets of triangular or quadrilateral elements. Severe mesh gradation is avoided by imposing a maximal one-level difference between quadrants sharing a common edge. This implies a maximal two-level difference between quadrants sharing a common vertex.

A simple example involving a domain consisting of a rectangle and a region within a curved arc, as shown in Figure 5.2.5, will illustrate the quadtree process. In the upper portion of the figure, the square universe containing the problem domain is quartered creating the one-level tree structure shown at the upper right. The quadrant containing the curved arc is quartered and the resulting quadrant that intersects the arc is quartered again to create the three-level tree shown in the lower right portion of the figure. A triangular mesh generated for this tree structure is also shown. The triangular elements are associated with quadrants of the tree structure. Quadrants and a mixed triangular- and quadrilateral-element mesh for a more complex example are shown in Figure 5.2.6.

Elements produced by the quadtree and octree techniques may have poor geometric shapes near boundaries. A final "smoothing" of the mesh improves element shapes and

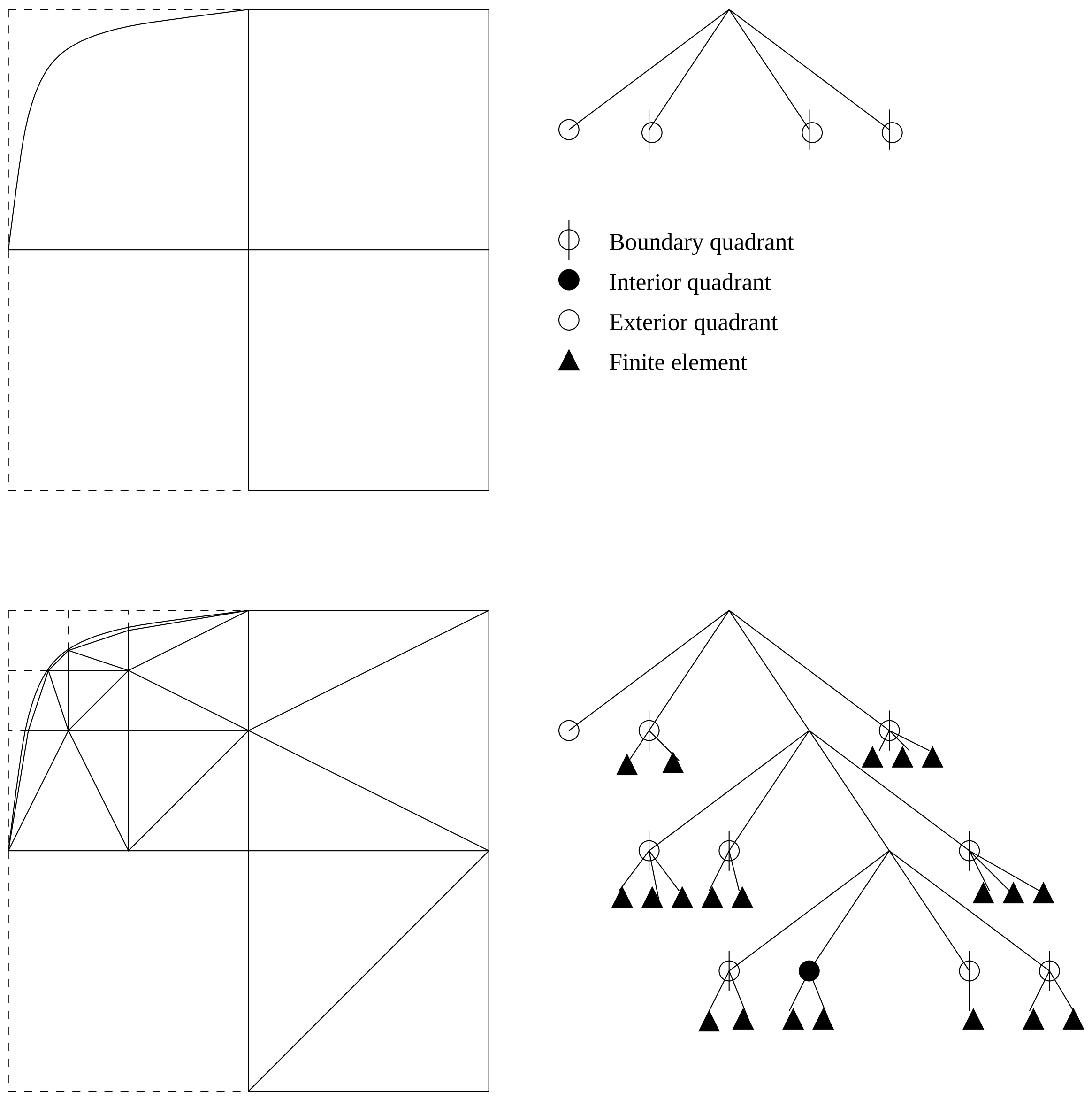


Figure 5.2.5: Finite quadtree mesh generation for a domain consisting of a rectangle and a region within a curved arc. One-level (top) and three-level (bottom) tree structures are shown. The mesh of triangular elements associated with the three-level quadtree is shown superimposed.

further reduces mesh gradation near $\partial\Omega$. Element vertices on $\partial\Omega$ are moved along the boundary to provide a better approximation to it. Pairs of boundary vertices that are too close to each other may be *collapsed* to a single vertex. Interior vertices are smoothed by a *Laplacian* operation that places each vertex at the "centroid" of its neighboring vertices. To be specific, let i be the index of a node to be re-positioned; $\mathbf{x}_i$ be its coordinates; P_i be the set of indices of all vertices that are connected to Node i by an element edge; and Q_i contain the indices of vertices that are in the same quadrant as Node i but are not

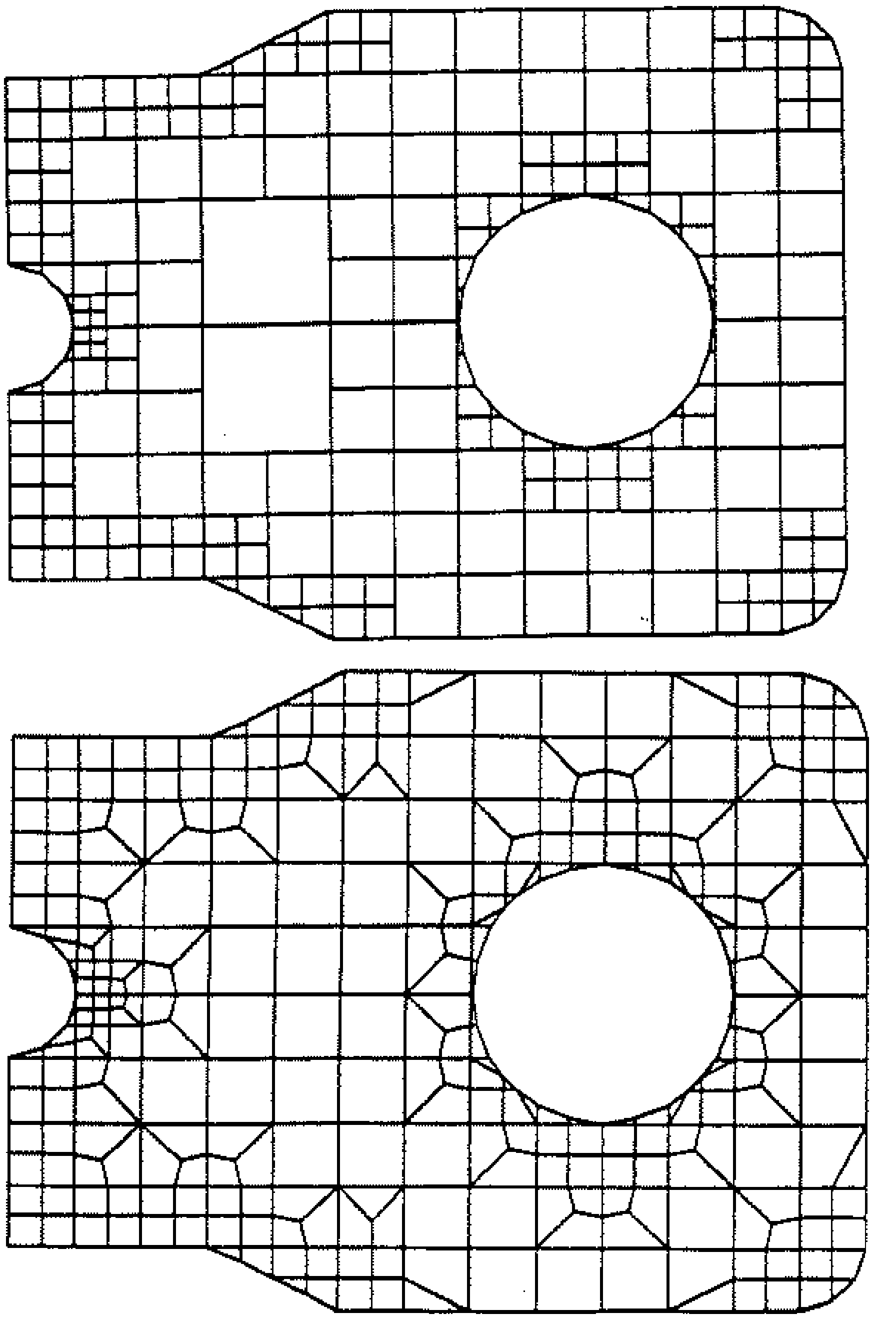


Figure 5.2.6: Quadtree structure and mixed triangular- and quadrilateral-element mesh generated from it.

connected to it by an edge. Then

$$\mathbf{x}_i = \frac{2\sum_{j\in P_i}\mathbf{x}_j + \sum_{j\in Q_i}\mathbf{x}_j}{2\dim(P_i) + \dim(Q_i)} \tag{5.2.4}$$

where $\dim(S)$ is the number of element vertices in set S. Additional details appear in

Baehmann *et al.* [2].

Arbitrarily complex two- and three-dimensional domains may be discretized by quadtree and octree decomposition to produce unstructured grids. Further solution-based mesh refinement may be done by subdividing appropriate terminal quadrants or octants and generating a new mesh locally. This unites mesh generation and adaptive mesh refinement by a common tree data structure [2]. The underlying tree structure is also suitable for load balancing on a parallel computer [8, 7].

The advancing front technique constructs a mesh by "notching" elements from $\partial\Omega$ and propagating this process into the interior of the domain. An example is shown in Figure 5.2.7. This procedure provides better shape control than quadtree or octree but problems arise as the advancing fronts intersect. Löhner [10] has a description of this and other mesh generation techniques. Carey [6] presents a more recent treatment of mesh generation.

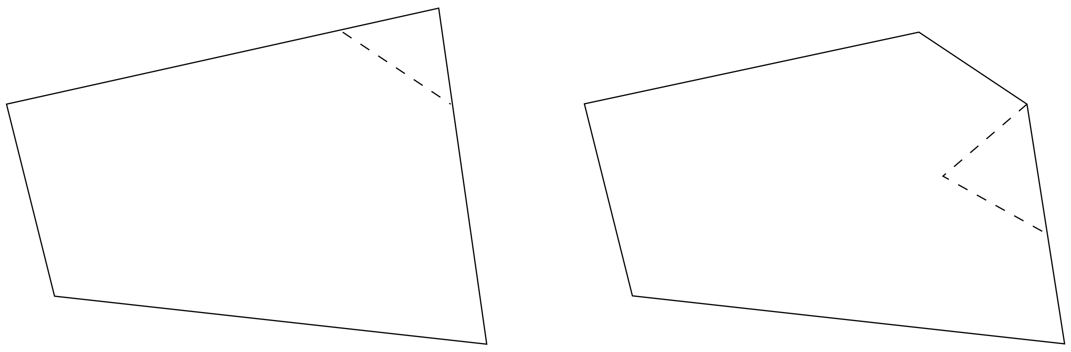


Figure 5.2.7: Mesh generation by the advancing front technique.

5.3 Data Structures

Unstructured mesh computation requires a data structure to store the geometric information. There is some ambiguity concerning the information that should be computed at the preprocessing stage, but, at the very least, the processing module would have to know

- the vertices belonging to each element,
- the spatial coordinates of each vertex, and
- the element edges, faces, or vertices that are on $\partial\Omega$.

The processing module would need more information when adaptivity is performed. It, for example, would need a link to the geometric information in order to refine elements along a curved boundary. Even without adaptivity, the processing software may want access to geometric information when using elements with curved edges or faces (*cf.* Section 5.4). If the finite element basis were known at the preprocessing stage, space could be reserved for edge and interior nodes or for a symbolic factorization of the resulting algebraic system (*cf.* Chapter 11).

Beall and Shephard [4] introduced a database and data structure that have great flexibility. It is suitable for use with high-order and hierarchical bases, adaptive mesh refinement and/or order variation, and arbitrarily complex domains. It has a hierarchical structure with three-dimensional elements (regions) having pointers to their bounding faces, faces having pointers to their bounding edges, and edges having pointers to their bounding vertices. Those mesh entities (elements, faces, edges, and vertices) on domain boundaries have pointers to relevant geometric structures defining the problem domain. This structure, called the *SCOREC mesh database*, is shown in Figure 5.3.1. Nodes may be introduced as fixed points in space to be associated with shape functions. When done, these may be located by pointers from any mesh entity.

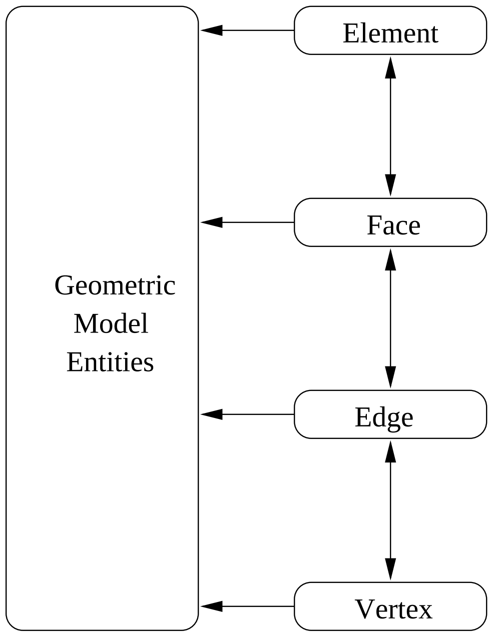


Figure 5.3.1: SCOREC hierarchical mesh database.

Let us illustrate the data structure for the two-dimensional domain shown in Figure 5.2.5. As shown in Figure 5.3.2, this mesh has 20 faces (two-dimensional elements), 36 edges, and 17 vertices. The face and edge-pointer information is shown in Table 5.3.1. Each edge has two pointers back to the faces that contain it. These are shown within brackets in the table. The use of tables and integer indices for pointers is done for convenience and does not imply an array implementation of pointer data. The edge and vertex-pointer information and the vertex-point coordinate data are shown in Table 5.3.2. Backward pointers from vertices to edges and pointers from vertices and edges on the boundary to the geometric database have not been shown to simplify the presentation. We have shown a small portion of the pointer structure near Edge 18 in Figure 5.3.3. Links between common entities allow the mesh to be traversed by faces, edges, or vertices in two dimensions. Problem and solution data is stored with the appropriate entities.

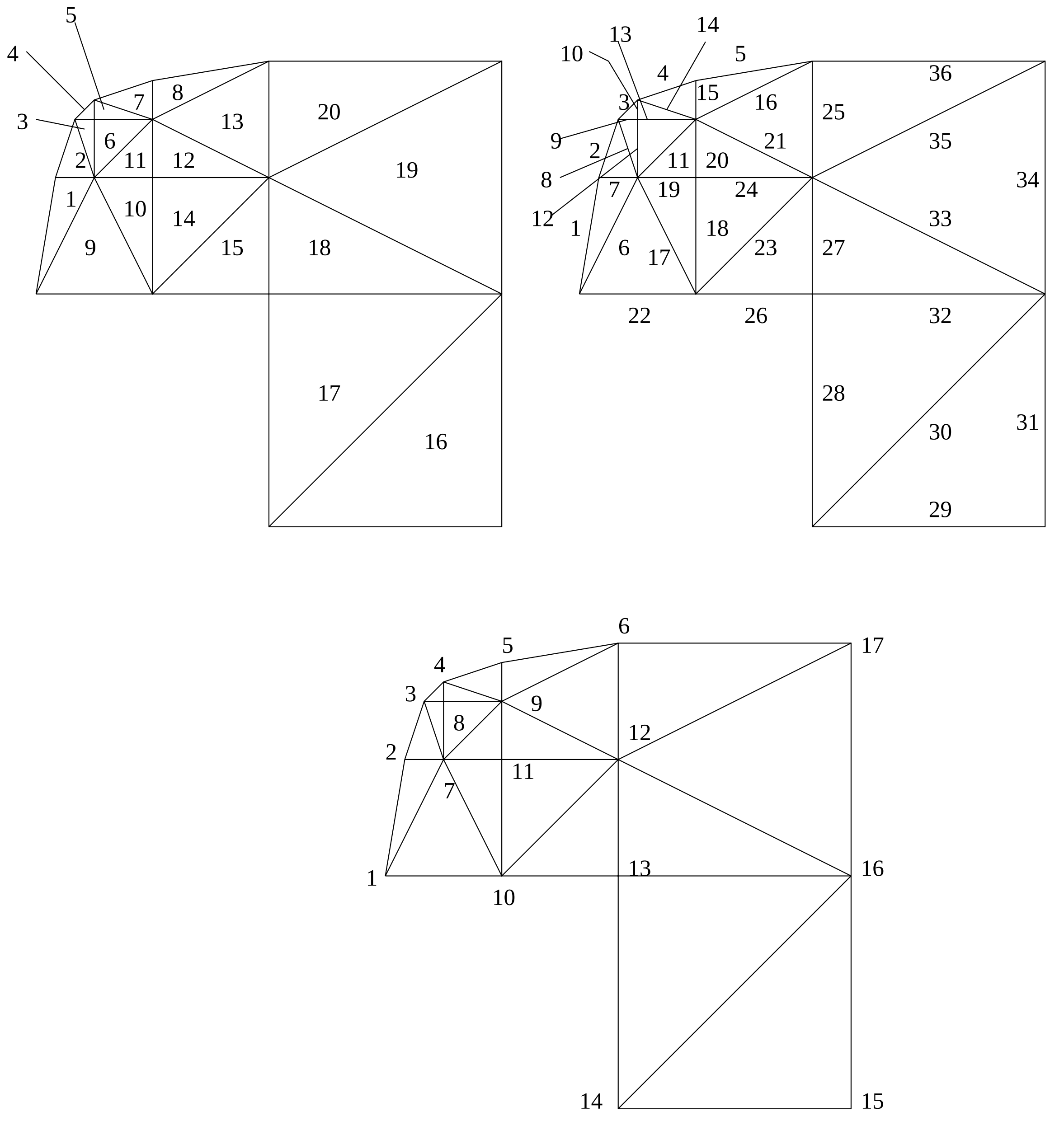


Figure 5.3.2: Example illustrating the SCOREC mesh database. Faces are indexed as shown at the upper left, edge numbering is shown at the upper right, and vertex numbering is shown at the bottom.

Face	Edge	Edge	Edge
1	1 [1]	7 [1 2]	6 [1 9]
2	2 [2]	8 [2 3]	7 [2 1]
3	8 [3 2]	9 [3 4]	12 [3 6]
4	3 [4]	10 [4 5]	9 [4 3]
5	10 [5 4]	14 [5 7]	13 [5 6]
6	12 [6 3]	13 [6 5]	11 [6 11]
7	4 [7]	15 [7 8]	14 [7 5]
8	15 [8 7]	5 [8]	16 [8 13]
9	6 [9 1]	17 [9 10]	22 [9]
10	17 [10 9]	19 [10 11]	18 [10 14]
11	11 [11 6]	20 [11 12]	19 [11 10]
12	20 [12 11]	21 [12 13]	24 [12 14]
13	16 [13 8]	25 [13 20]	21 [13 12]
14	18 [14 10]	24 [14 12]	23 [14 15]
15	23 [15 14]	27 [15 18]	26 [15]
16	29 [16]	30 [16 17]	31 [16]
17	28 [17]	32 [17 18]	30 [17 16]
18	27 [18 15]	33 [18 19]	32 [18 17]
19	33 [19 18]	35 [19 20]	34 [19]
20	25 [20 13]	36 [20]	35 [20 19]

Table 5.3.1: Face and edge-pointer data for the mesh shown in Figure 5.2.5. Backward pointers from edges to their bounding faces are shown in brackets.

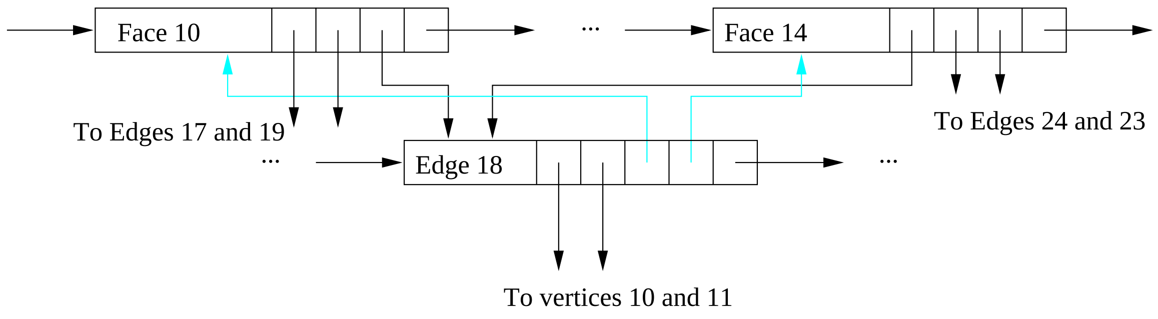


Figure 5.3.3: Pointer structure in the vicinity of Edge 18.

The SCOREC mesh database contains more information than necessary for a typical finite element solution. For example, the edge information may be eliminated and faces may point directly to vertices. This would be a more traditional finite element data structure. Although it saves storage and simplifies the data structure, it may be wise to keep the edge information. Adaptive mesh refinement procedures often work by edge splitting and these are simplified when edge data is available. Edge information also simplifies the application of boundary conditions, especially when the boundary is

Edge	Vertices		Edge	Vertices	
1	1	2	19	7	11
2	2	3	20	9	11
3	3	4	21	9	12
4	4	5	22	1	10
5	5	6	23	10	12
6	1	7	24	11	12
7	2	7	25	12	6
8	3	7	26	10	13
9	3	8	27	13	12
10	4	8	28	13	14
11	7	9	29	14	15
12	7	8	30	14	16
13	8	9	31	15	16
14	4	9	32	13	16
15	5	9	33	16	12
16	9	6	34	16	17
17	7	10	35	12	17
18	10	11	36	6	17

Vertex	Coordinates	
1	-1.00	0.00
2	-0.90	0.50
3	-0.80	0.75
4	0.75	0.80
5	-0.50	0.90
6	0.00	1.00
7	-0.75	0.50
8	-0.75	0.75
9	-0.50	0.75
10	-0.50	0.00
11	-0.50	0.50
12	0.00	0.50
13	0.00	0.00
14	0.00	-1.00
15	1.00	-1.00
16	1.00	0.00
17	1.00	1.00

Table 5.3.2: Edge and vertex-pointer data (left) and vertex and coordinate data (right) for the mesh shown in Figure 5.2.5.

curved. Only pointers are required for the edge information and, in many implementations, pointers require less storage than integers. Nevertheless, let us illustrate face and vertex information for the simple mesh shown in Figure 5.3.4, which contains a mixture of triangular and quadrilateral elements. The face-vertex information is shown in Table 5.3.3 and the vertex-coordinate data is shown in Table 5.3.4. Assuming quadratic shape functions on the triangles and biquadratic shape functions on the rectangles, a traditional data structure would typically add nodes at the centers of all edges and the centers of the rectangular faces. In this example, the midside and face nodes are associated with faces; however, they could also have been associated with vertices.

Without edge data, the database generally requires additional *a priori* assumptions. For example, we could agree to list vertices in counterclockwise order. Edge nodes could follow in counterclockwise order beginning with the node that is closest in the counterclockwise direction to the first vertex. Finally, interior nodes may be listed in any order. The choice of the first vertex is arbitrary. This strategy is generally a compromise between storing a great deal of data with fast access and having low storage costs but having to recompute information. We could further reduce storage, for example, by not saving the coordinates of the edge nodes.

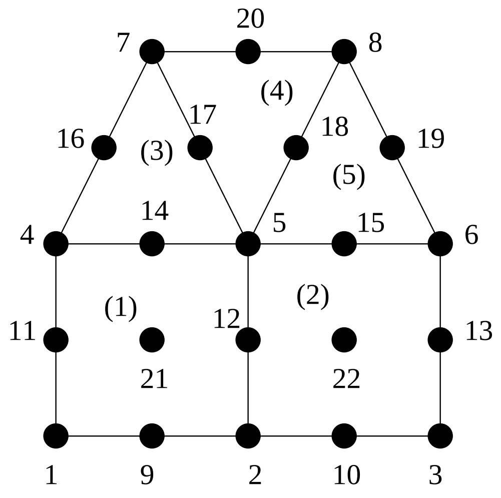


Figure 5.3.4: Sample finite element mesh involving a mixture of quadratic approximations on triangles and biquadratic approximations on rectangles. Face indices are shown in parentheses.

Face	Vertices				Nodes				
1	1	2	5	4	9	12	14	11	21
2	2	3	6	5	10	13	15	12	22
3	4	5	7		14	17	16		
4	5	8	7		18	20	17		
5	5	6	8		15	19	18		

Table 5.3.3: Simplified face-vertex data for the mesh of Figure 5.3.4.

The type of finite element basis must also be stored. In the present example, we could attach it to the face-vertex table. With the larger database described earlier, we could attach it to the appropriate entity. In the spirit of the shape function decomposition described in Sections 4.4 and 4.5, we could store information about a face shape function with the face and information about an edge shape function with the edge. This would allow us to use variable-order approximations (p-refinement).

Without edge data, we need a way of determining those edges that are on $\partial\Omega$. This can be done by adopting a convention that the edge between the first and second vertices of each face is Edge 1. Remaining edges are numbered in counterclockwise order. A sample boundary data table for the mesh of Figure 5.3.4 is shown on the right of Table 5.3.4. The first row of the table identifies Edge 1 of Face 1 as being on a boundary of the domain. Similarly, the second row of the table identifies Edge 4 of Face 1 as being a boundary edge, *etc.* Regions with curved edges would need pointers back to the geometric database.

Vertex	Coordinates	
1	0.00	0.00
2	1.00	0.00
3	2.00	0.00
4	0.00	1.00
5	1.00	1.00
6	2.00	1.00
7	0.50	2.00
8	1.50	2.00
9	0.50	0.00
10	1.50	0.00
11	0.00	0.50
12	1.00	0.50
13	2.00	0.50
14	0.50	1.00
15	1.50	1.00
16	0.25	1.50
17	0.75	1.50
18	1.25	1.50
19	1.75	1.50
21	0.50	0.50
22	1.50	0.50

Face	Edge
1	1
1	4
2	1
2	2
3	3
4	2
5	2

Table 5.3.4: Vertex and coordinate data (left) and boundary data (right) for the finite element mesh shown in Figure 5.3.4.

5.4 Coordinate Transformations

Coordinate transformations enable us to develop element stiffness and mass matrices and load vectors on canonical triangular, square, tetrahedral, and cubic elements in a computational domain and map these to actual elements in the physical domain. Useful transformations must (*i*) be simple to evaluate, (*ii*) preserve continuity of the finite element solution and geometry, and (*iii*) be invertible. The latter requirement ensures that each point within the actual element corresponds to one and only one point in the canonical element. Focusing on two dimensions, this requires the Jacobian

$$\mathbf{J}_e := \begin{bmatrix} x_\xi & x_\eta \\ y_\xi & y_\eta \end{bmatrix} \tag{5.4.1}$$

of the transformation of Element e in the physical (x, y)-plane to the canonical element in the computational (ξ, η)-plane to be nonsingular.

The most popular coordinate transformations are, naturally, piecewise-polynomial functions. These mappings are called *subparametric*, *isoparametric*, and *superparametric* when their polynomial degree is, respectively, lower than, equal to, and greater than that used for the trial function. As we have seen in Chapter 4, the transformations use the same shape functions as the finite element solutions. We illustrated linear (Section 4.2) and bilinear (Section 4.3) transformations for, respectively, mapping triangles and quadrilaterals to canonical elements. We have two tasks in front of us: (*i*) determining whether higher-degree piecewise polynomial mappings can be used to advantage and (*ii*) ensuring that these transformations will be nonsingular.

Example 5.4.1. Recall the bilinear transformation of a 2×2 canonical square to a quadrilateral that was introduced in Section 4.3 (Figure 5.4.1)

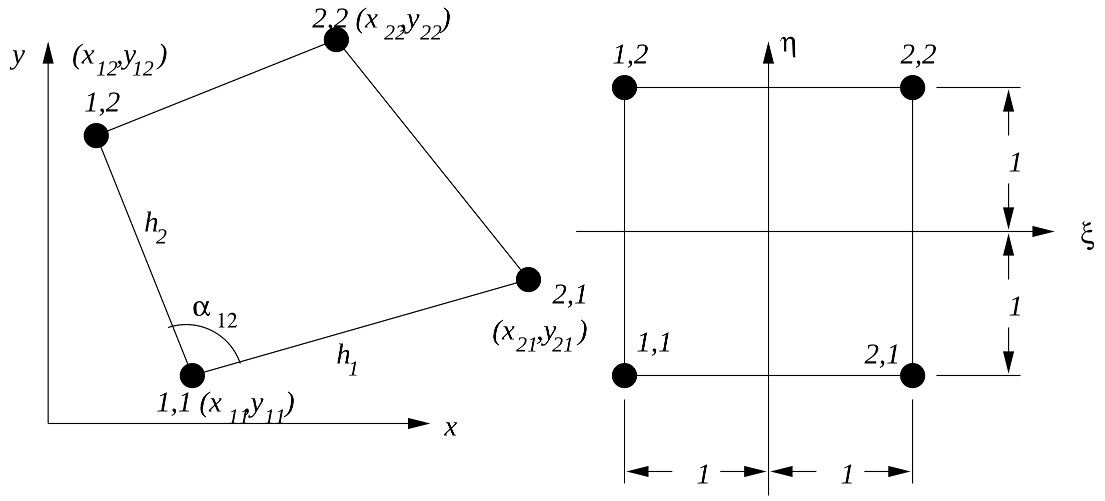


Figure 5.4.1: Bilinear mapping of a quadrilateral to a 2×2 square.

$$\begin{bmatrix} x(\xi,\eta) \\ y(\xi,\eta) \end{bmatrix} = \sum_{i=1}^{2}\sum_{j=1}^{2} \begin{bmatrix} x_{ij} \\ y_{ij} \end{bmatrix} N_{i,j}(\xi,\eta), \tag{5.4.2a}$$

where

$$N_{i,j}(\xi,\eta) = \bar{N}_i(\xi)\bar{N}_j(\eta), \qquad i,j = 1,2, \tag{5.4.2b}$$

and

$$\bar{N}_i(\xi) = \begin{cases} (1-\xi)/2, & \text{if } i = 1 \\ (1+\xi)/2, & \text{if } i = 2 \end{cases}. \tag{5.4.2c}$$

The vertices of the square $(-1,-1)$, $(1,-1)$, $(-1,1)$, $(1,1)$ are mapped to the vertices of the quadrilateral $(x_{1,1}, y_{1,1})$, $(x_{2,1}, y_{2,1})$, $(x_{1,2}, y_{1,2})$, $(x_{2,2}, y_{2,2})$. The bilinear transformation is linear along each edge, so the quadrilateral element has straight sides.

Differentiating (5.4.2a) while using (5.4.2b,c)

$$x_\xi = \frac{x_{21}-x_{11}}{2}\bar{N}_1(\eta) + \frac{x_{22}-x_{12}}{2}\bar{N}_2(\eta),$$

$$y_\xi = \frac{y_{21}-y_{11}}{2}\bar{N}_1(\eta) + \frac{y_{22}-y_{12}}{2}\bar{N}_2(\eta),$$

$$x_\eta = \frac{x_{12}-x_{11}}{2}\bar{N}_1(\xi) + \frac{x_{22}-x_{21}}{2}\bar{N}_2(\xi),$$

$$y_\eta = \frac{y_{12}-y_{11}}{2}\bar{N}_1(\xi) + \frac{y_{22}-y_{21}}{2}\bar{N}_2(\xi).$$

Substituting these formulas into (5.4.1) and evaluating the determinant reveals that the quadratic terms cancel; hence, the determinant of $\mathbf{J}_e$ is a linear function of ξ and η rather than a bilinear function. Therefore, it suffices to check that $\det(\mathbf{J}_e)$ has the same sign at each of the four vertices. For example,

$$\det(\mathbf{J}_e(-1,-1)) = x_\xi(-1,-1)y_\eta(-1,-1) - x_\eta(-1,-1)y_\xi(-1,-1)$$

or

$$\det(\mathbf{J}_e(-1,-1)) = (x_{21}-x_{11})(y_{12}-y_{11}) - (x_{12}-x_{11})(y_{21}-y_{11}).$$

The cross product formula for two-component vectors indicates that

$$\det(\mathbf{J}_e(-1,-1)) = h_1 h_2 \sin\alpha_{12},$$

where h_1, h_2, and α_{12} are the lengths of two adjacent sides and the angle between them (Figure 5.4.1). Similar formulas apply at the other vertices. Therefore, $\det(\mathbf{J}_e)$ will not vanish if and only if $\alpha_{ij} < \pi$ at each vertex, *i.e.*, if and only if the quadrilateral is convex.

Polynomial shape functions and bases are constructed on the canonical element as described in Chapter 4. For example, the restriction of a bilinear (isoparametric) trial function to the canonical element would have the form

$$U(\xi,\eta) = \sum_{i=1}^{2}\sum_{j=1}^{2} c_{i,j}N_{i,j}(\xi,\eta).$$

A subparametric approximation might, for example, use a piecewise-bilinear coordinate transformation (5.4.2) with a piecewise-biquadratic trial function. Let us illustrate this using the element node numbering of Section 4.3 as shown in Figure 5.4.2. Using (4.3.3), the restriction of the piecewise-biquadratic polynomial trial function to the canonical element is

$$U(\xi,\eta) = \sum_{i=1}^{3}\sum_{j=1}^{3} c_{i,j}N_{i,j}^2(\xi,\eta) \qquad (5.4.3a)$$

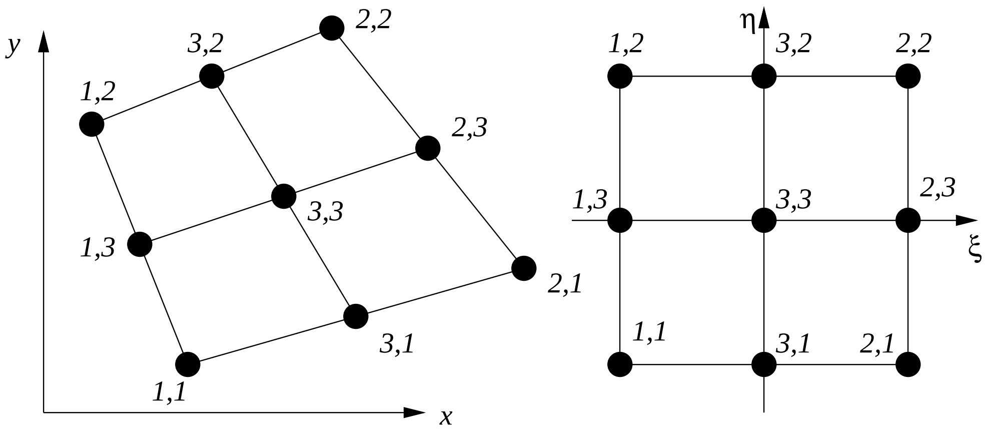


Figure 5.4.2: Bilinear mapping to a unit square with a biquadratic trial function.

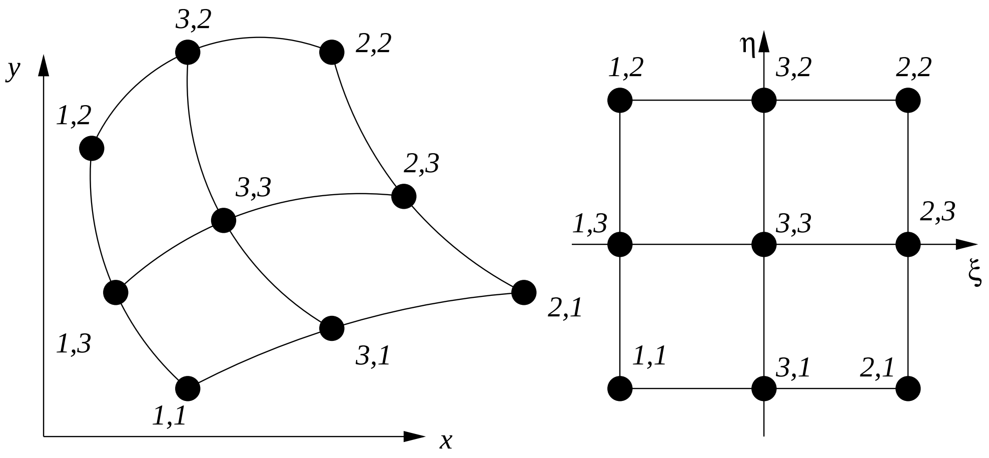


Figure 5.4.3: Biquadratic mapping of the unit square to a curvilinear element.

where the superscript 2 is used to identify biquadratic shape functions

$$N^2_{i,j}(\xi,\eta) = \bar{N}^2_i(\xi)\bar{N}^2_j(\eta), \qquad i,j = 1,2,3, \tag{5.4.3b}$$

with

$$\bar{N}^2_i(\xi) = \begin{cases} -\xi(1-\xi)/2, & \text{if } i = 1 \\ \xi(1+\xi)/2, & \text{if } i = 2 \\ 1-\xi^2, & \text{if } i = 3 \end{cases}. \tag{5.4.3c}$$

Example 5.4.2. A biquadratic transformation of the canonical square has the form

$$\begin{bmatrix} x(\xi,\eta) \\ y(\xi,\eta) \end{bmatrix} = \sum_{i=1}^{3}\sum_{j=1}^{3} \begin{bmatrix} x_{ij} \\ y_{ij} \end{bmatrix} N^2_{i,j}(\xi,\eta), \tag{5.4.4}$$

where $N^2_{i,j}(\xi,\eta)$, $i,j = 1,2,3$, is given by (5.4.3).

This transformation produces an element in the (x,y)-plane having curved (quadratic) edges as shown in Figure 5.4.3. An isoparametric approximation would be biquadratic

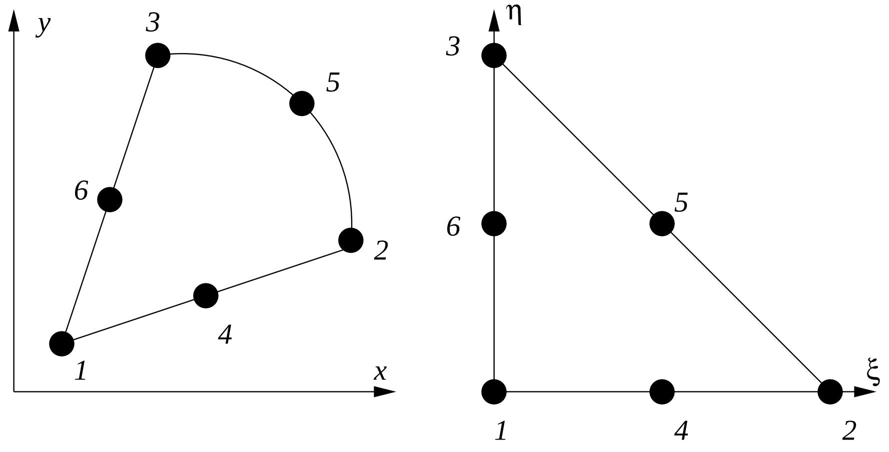


Figure 5.4.4: Quadratic mapping of a triangle having one curved side.

and have the form of (5.4.3). The interior node (3,3) is awkward and can be eliminated by using a second-order serendipity (*cf.* Problems 4.3.1) or hierarchical transformation (*cf.* Section 4.4).

Example 5.4.3. The biquadratic transformation described in Example 5.4.2 is useful for discretizing domains having curved boundaries. With a similar goal, we describe a transformation for creating triangular elements having one curved and two straight sides (Figure 5.4.4). Let us approximate the curved boundary by a quadratic polynomial and map the element onto a canonical right triangle by the quadratic transformation

$$\begin{bmatrix} x(\xi,\eta) \\ y(\xi,\eta) \end{bmatrix} = \sum_{i=1}^{6} \begin{bmatrix} x_i \\ y_i \end{bmatrix} N_i^2(\xi,\eta), \tag{5.4.5a}$$

where the quadratic Lagrange shape functions are (*cf.* Problem 4.2.1)

$$N_j^2 = 2\zeta_j(\zeta_j - 1/2), \qquad j = 1,2,3, \tag{5.4.5b}$$

$$N_4^2 = 4\zeta_1\zeta_2, \qquad N_5^2 = 4\zeta_2\zeta_3, \qquad N_6^2 = 4\zeta_3\zeta_1, \tag{5.4.5c}$$

and

$$\zeta_1 = 1 - \xi - \eta, \qquad \zeta_2 = \xi, \qquad \zeta_3 = \eta. \tag{5.4.6}$$

Equations (5.4.5) and (5.4.6) describe a general quadratic transformation. We have a more restricted situation with

$$x_4 = (x_1 + x_2)/2, \qquad y_4 = (y_1 + y_2)/2,$$

$$x_6 = (x_1 + x_3)/2, \qquad y_6 = (y_1 + y_3)/2.$$

This simplifies the transformation (5.4.5a) to

$$\begin{bmatrix} x(\xi,\eta) \\ y(\xi,\eta) \end{bmatrix} = \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} \hat{N}_1^2 + \begin{bmatrix} x_2 \\ y_2 \end{bmatrix} \hat{N}_2^2 + \begin{bmatrix} x_3 \\ y_3 \end{bmatrix} \hat{N}_3^2 + \begin{bmatrix} x_5 \\ y_5 \end{bmatrix} \hat{N}_5^2, \tag{5.4.7a}$$

where, upon use of (5.4.5) and (5.4.6),

$$\hat{N}_1^2 = N_1^2 + (N_4^2 + N_6^2)/2 = \zeta_1 = 1 - \xi - \eta, \tag{5.4.7b}$$

$$\hat{N}_2^2 = N_2^2 + N_4^2/2 = \xi(1 - 2\eta), \tag{5.4.7c}$$

$$\hat{N}_3^2 = N_3^2 + N_6^2/2 = \eta(1 - 2\xi), \tag{5.4.7d}$$

$$\hat{N}_5^2 = N_5^2 = 4\xi\eta. \tag{5.4.7e}$$

From these results, we see that the mappings on edges 1-2 ($\eta = 0$) and 1-3 ($\xi = 0$) are linear and are, respectively, given by

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} (1-\xi) + \begin{bmatrix} x_2 \\ y_2 \end{bmatrix} \xi, \qquad \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} (1-\eta) + \begin{bmatrix} x_3 \\ y_3 \end{bmatrix} \eta.$$

The Jacobian determinant of the transformation can vanish depending on the location of Node 5. The analysis may be simplified by constructing the transformation in two steps. In the first step, we use a linear transformation to map an arbitrary element onto a canonical element having vertices at $(0,0)$, $(1,0)$, and $(0,1)$ but with one curved side. In the second step, we remove the curved side using the quadratic transformation (5.4.7). The linear mapping of the first step has a constant Jacobian determinant and, therefore, cannot affect the invertibility of the system. Thus, it suffices to consider the second step of the transformation as shown in Figure 5.4.5. Setting $(x_1, y_1) = (0,0)$, $(x_2, y_2) = (1,0)$, and $(x_3, y_3) = (0,1)$ in (5.4.7a) yields

$$\begin{bmatrix} x(\xi,\eta) \\ y(\xi,\eta) \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \hat{N}_2^2 + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \hat{N}_3^2 + \begin{bmatrix} x_5 \\ y_5 \end{bmatrix} \hat{N}_5^2.$$

Using (5.4.7c-e)

$$\begin{bmatrix} x(\xi,\eta) \\ y(\xi,\eta) \end{bmatrix} = \begin{bmatrix} \xi(1-2\eta) \\ \eta(1-2\xi) \end{bmatrix} + 4\xi\eta \begin{bmatrix} x_5 \\ y_5 \end{bmatrix}.$$

Calculating the Jacobian

$$\mathbf{J}_e(\xi,\eta) = \begin{bmatrix} x_\xi & x_\eta \\ y_\xi & y_\eta \end{bmatrix} = \begin{bmatrix} 1 - 2\eta + 4x_5\eta & -2\xi + 4x_5\xi \\ -2\eta + 4y_5\eta & 1 - 2\xi + 4y_5\xi \end{bmatrix},$$

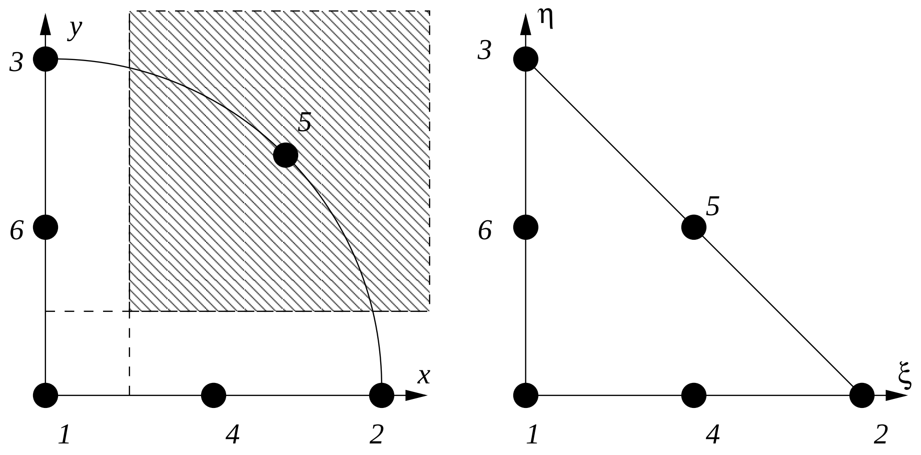


Figure 5.4.5: Quadratic mapping of a right triangle having one curved side. The shaded region indicates where Node 5 can be placed without introducing a singularity in the mapping.

we find the determinant as

$$\det(\mathbf{J}_e(\xi, \eta) = 1 + (4x_5 - 2)\eta + (4y_5 - 2)\xi.$$

The Jacobian determinant is a linear function of ξ and η; thus, as with Example 5.4.1, we need only ensure that it has the same sign at each of its three vertices. We have

$$\det(\mathbf{J}_e(0,0)) = 1, \qquad \det(\mathbf{J}_e(0,1)) = 4x_5 - 1, \qquad \det(\mathbf{J}_e(1,0)) = 4y_5 - 1.$$

Hence, the Jacobian determinant will not vanish and the mapping will be invertible when $x_5 > 1/4$ and $y_5 > 1/4$ (*cf.* Problem 2 at the end of this section). This region is shown shaded on the triangle of Figure 5.4.5.

Problems

1. Consider the second-order serendipity shape functions of Problem 4.3.1 or the second-order hierarchical shape functions of Section 4.4. Let the four vertex nodes be numbered $(1,1)$, $(2,1)$, $(1,1)$, and $(2,1)$ and the four midside nodes be numbered $(3,1)$, $(1,3)$, $(2,3)$, and $(3,2)$. Use the serendipity shape functions of Problem 4.3.1 to map the canonical 2×2 square element onto an eight-noded quadrilateral element with curved sides in the (x,y)-plane. Assume that the vertex and midside nodes of the physical element have the same numbering as the canonical element but have coordinates at (x_{ij}, y_{ij}), $i,j = 1,2,3$, $i = j \neq 3$. Can the Jacobian of the transformation vanish for some particular choices of (x,y)? (This is not a simple question. It suffices to give some qualitative reasoning as to how and why the Jacobian may or may not vanish.)

2. Consider the transformation (5.4.7) of Example 5.4.3 with $x_5 = y_5 = 1/4$ and sketch the element in the (x, y)-plane. Sketch the element for some choice of $x_5 = y_5 < 1/4$.

5.5 Generation of Element Matrices and Vectors and Their Assembly

Having discretized the domain, the next step is to select a finite element basis and generate and assemble the element stiffness and mass matrices and load vectors. As a review, we summarize some of the two-dimensional shape functions that were developed in Chapter 4 in Tables 5.5.1 and 5.5.2. Nodes are shown on the mesh entities for the Lagrangian and hierarchical shape functions. As noted in Section 5.3, however, the shape functions may be associated with the entities without introducing modal points. The number of parameters n_p for an element having order p shape functions is presented for $p = 1, 2, 3, 4$. We also list an estimate of the number of unknowns (degrees of freedom) N for scalar problems solved on unit square domains using uniform meshes of $2n^2$ triangular or n^2 square elements.

Both the Lagrange and hierarchical bases of order p have the same number of parameters and degrees of freedom on the uniform triangular meshes. Without constraints for Dirichlet data, the number of degrees of freedom is $N = (pn + 1)^2$ (*cf.* Problem 1 at the end of this section). Dirichlet data on the entire boundary would reduce N by $O(pn)$ and, hence, be a higher-order effect when n is large. The asymptotic approximation $N \approx (pn)^2$ is recorded in Table 5.5.1. Similarly, bi-polynomial approximations of order p on squares with n^2 uniform elements have $N = (pn + 1)^2$ degrees of freedom (again, *cf.* Problem 1). The asymptotic approximation $(pn)^2$ is reported in Table 5.5.2. Under the same conditions, hierarchical bases on squares have

$$N = \begin{cases} (2p-1)n^2 + 2pn + 1, & \text{if } p < 4 \\ (p^2 - p + 4)n^2/2 + 2pn + 1, & \text{if } p \geq 4 \end{cases}.$$

degrees of freedom. The asymptotic values $N \approx (2p - 1)N^2$, $p < 4$, and $N \approx (p^2 - p + 4)n^2/2$, $p \geq 4$, are reported in Table 5.5.2.

The Lagrange and hierarchical bases on triangles and the Lagrange bi-polynomial bases on squares have approximately the same number of degrees of freedom for a given order p. The hierarchical bases on squares have about half the degrees of freedom of the others. The bi-polynomial Lagrange shape functions on a square have the largest number of parameters per element for a given p. The number of parameters per element affects the element matrix and vector computations while the number of degrees of freedom affects the solution time. We cannot, however, draw firm conclusions about the superiority of one basis relative to another. The selection of an optimal basis for an intended level

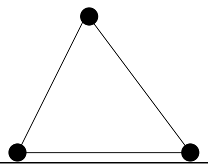
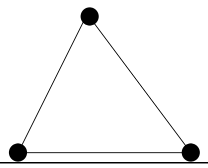
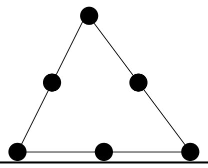
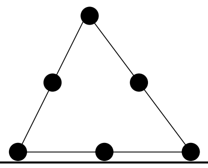
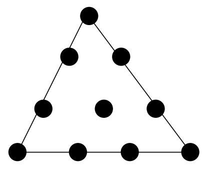
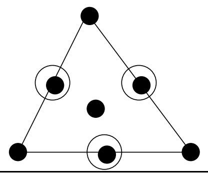
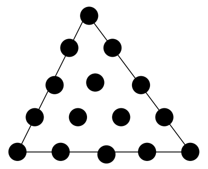
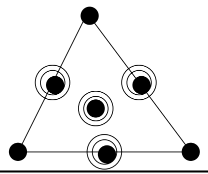
p	Lagrange Stencil	Hierarchical Stencil	n_p	$N \approx p^2n^2$
1			3	n^2
2			6	$4n^2$
3			10	$9n^2$
4			15	$16n^2$

Table 5.5.1: Shape function placement for Lagrange and hierarchical finite element approximations of degrees $p = 1, 2, 3, 4$ on triangular elements with their number of parameters per element n_p and degrees of freedom N on a square with $2n^2$ elements. Circles indicate additional shape functions located on a mesh entity.

of accuracy is a complex issue that depends on solution smoothness, geometry, and the partial differential system. We'll examine this topic in a later chapter. At least it seems clear that bi-polynomial bases are not competitive with hierarchical ones on square elements.

5.5.1 Generation of Element Matrices and Vectors

The generation of the element stiffness and mass matrices and load vectors is largely independent of the partial differential system being solved; however, let us focus on the model problem of Section 3.1 in order to illustrate the procedures less abstractly. Thus, consider the two-dimensional Galerkin problem: determine $u \in H^1_E$ satisfying

$$A(v, u) = (v, f), \qquad \forall v \in H^1_0, \tag{5.5.1a}$$

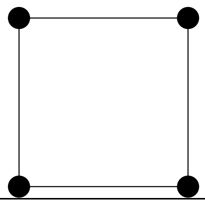
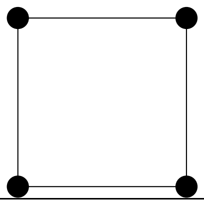
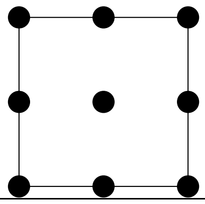
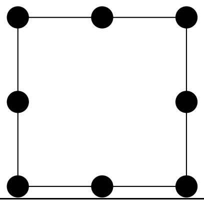
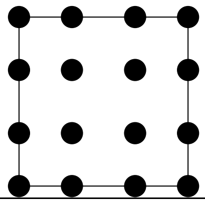
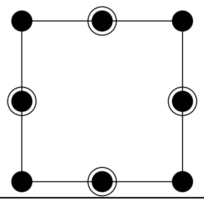
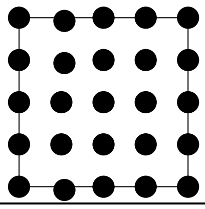
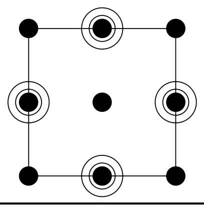
p	Lagrange			Hierarchical		
	Stencil	n_p	$N \approx Mn^2$	Stencil	n_p	$N \approx Mn^2$
1		4	n^2		4	n^2
2		9	$4n^2$		8	$3n^2$
3		16	$9n^2$		12	$5n^2$
4		25	$16n^2$		17	$8n^2$

Table 5.5.2: Shape function placement for bi-polynomial Lagrange and hierarchical approximations of degrees $p = 1, 2, 3, 4$ on square elements with their number of parameters per element n_p and degrees of freedom N on a square with n^2 elements. Circles indicate additional shape functions located on a mesh entity.

where

$$(v, f) = \iint_{\Omega} vf\,dxdy \tag{5.5.1b}$$

$$A(v, u) = \iint_{\Omega} [p(v_x u_x + v_y u_y) + qvu]dxdy \tag{5.5.1c}$$

As usual, Ω is a two-dimensional domain with boundary $\partial\Omega = \partial\Omega_E \cup \partial\Omega_N$. Recall that smooth solutions of (5.5.1) satisfy

$$-(pu_x)_x - (pu_y)_y + qu = f, \qquad (x, y) \in \Omega, \tag{5.5.2a}$$

$$u = \alpha, \qquad (x,y) \in \partial\Omega_E, \tag{5.5.2b}$$

$$u_{\mathbf{n}} = 0, \qquad (x,y) \in \partial\Omega_N, \tag{5.5.2c}$$

where $\mathbf{n}$ is the unit outward normal vector to $\partial\Omega$. Trivial natural boundary conditions are considered for simplicity. More complicated situations will be examined later in this section.

Following the one-dimensional examples of Chapters 1 and 2, we select finite-dimensional subspaces S_E^N and S_0^N of H_E^1 and H_0^1 and write (5.5.1b,c) as the sum of contributions over elements

$$(V,f) = \sum_{e=1}^{N_\Delta} (V,f)_e, \tag{5.5.3a}$$

$$A(V,U) = \sum_{e=1}^{N_\Delta} A_e(V,U). \tag{5.5.3b}$$

Here, N_Δ is the number of elements in the mesh,

$$(V,f)_e = \iint_{\Omega_e} Vf\,dxdy \tag{5.5.3c}$$

is the local L^2 inner product,

$$A_e(V,U) = \iint_{\Omega_e} [p(V_xU_x + V_yU_y) + qVU]dxdy \tag{5.5.3d}$$

is the local strain energy, and Ω_e is the portion of Ω occupied by element e.

The evaluation of (5.5.3c,d) can be simple or complex depending on the functions p, q, and f and the mesh used to discretize Ω. If p and q were constant, for example, the local strain energy (5.5.3d) could be integrated exactly as illustrated in Chapters 1 and 2 for one-dimensional problems. Let's pursue a more general approach and discuss procedures based on transforming integrals (5.5.3c,d) on element e to a canonical element 0 and evaluating them numerically. Thus, let $U_0(\xi,\eta) = U(x(\xi,\eta), y(\xi,\eta))$ and $V_0(\xi,\eta) = V(x(\xi,\eta), y(\xi,\eta))$ and transform the integrals (5.5.3c,d)) to element 0 to get

$$(V,f)_e = \iint_{\Omega_0} V_0(\xi,\eta) f(x(\xi,\eta), y(\xi,\eta)) \det(\mathbf{J}_e) d\xi d\eta. \tag{5.5.4a}$$

$$A_e(V,U) = \iint_{\Omega_0} [p(V_{0_\xi}\xi_x + V_{0_\eta}\eta_x)(U_{0_\xi}\xi_x + U_{0_\eta}\eta_x) +$$

$$p(V_{0_\xi}\xi_y + V_{0_\eta}\eta_y)(U_{0_\xi}\xi_y + U_{0_\eta}\eta_y) + qV_0U_0] \det(\mathbf{J}_e)d\xi d\eta$$

where $\mathbf{J}_e$ is the Jacobian of the transformation (*cf.* (5.4.1)).

Expanding the terms in the strain energy

$$A_e(V,U) = \iint_{\Omega_0} [g_{1e}V_{0_\xi}U_{0_\xi} + g_{2e}(V_{0_\xi}U_{0_\eta} + V_{0_\eta}U_{0_\xi}) + g_{3e}V_{0_\eta}U_{0_\eta} + qV_0U_0] \det(\mathbf{J}_e)d\xi d\eta \tag{5.5.4b}$$

where

$$g_{1e} = p(x(\xi,\eta), y(\xi,\eta))[\xi_x^2 + \xi_y^2], \tag{5.5.4c}$$

$$g_{2e} = p(x(\xi,\eta), y(\xi,\eta))[\xi_x\eta_x + \xi_y\eta_y], \tag{5.5.4d}$$

$$g_{3e} = p(x(\xi,\eta), y(\xi,\eta))[\eta_x^2 + \eta_y^2]. \tag{5.5.4e}$$

The integrand of (5.5.4b) might appear to be polynomial for constant p and a polynomial mapping; however, this is not the case. In Section 4.6, we showed that the inverse coordinate mapping satisfies

$$\xi_x = \frac{y_\eta}{\det(\mathbf{J}_e)}, \qquad \xi_y = -\frac{x_\eta}{\det(\mathbf{J}_e)}, \qquad \eta_x = -\frac{y_\xi}{\det(\mathbf{J}_e)}, \qquad \eta_y = \frac{x_\xi}{\det(\mathbf{J}_e)}. \tag{5.5.5}$$

The functions g_{ie}, $i = 1,2,3$, are proportional to $[1/\det(\mathbf{J}_e)]^2$; thus, the integrand of (5.5.4b) is a rational function unless, of course, $\det(\mathbf{J}_e)$ is a constant.

Let us write U_0 and V_0 in the form

$$U_0(\xi,\eta) = \mathbf{c}_e^T\mathbf{N}(\xi,\eta) = \mathbf{N}(\xi,\eta)^T\mathbf{c}_e, \qquad V_0(\xi,\eta) = \mathbf{d}_e^T\mathbf{N}(\xi,\eta) = \mathbf{N}(\xi,\eta)^T\mathbf{d}_e \tag{5.5.6}$$

where the vectors $\mathbf{c}_e$ and $\mathbf{d}_e$ contain the elemental parameters and $\mathbf{N}(\xi,\eta)$ is a vector containing the elemental shape functions.

Example 5.5.1. For a linear polynomial on the canonical right 45° triangular element having vertices numbered 1 to 3 as shown in Figure 5.5.1,

$$\mathbf{c}_e = \begin{bmatrix} c_{e,1} \\ c_{e,2} \\ c_{e,3} \end{bmatrix}, \qquad \mathbf{N}(\xi,\eta) = \begin{bmatrix} 1-\xi-\eta \\ \xi \\ \eta \end{bmatrix}.$$

The actual vertex indices, shown as i, j, abd k, are mapped to the canonical indices 1, 2, and 3.

Example 5.5.2. The treatment of hierarchical polynomials is more involved because there can be more than one parameter per node. Consider the case of a cubic hierarchical

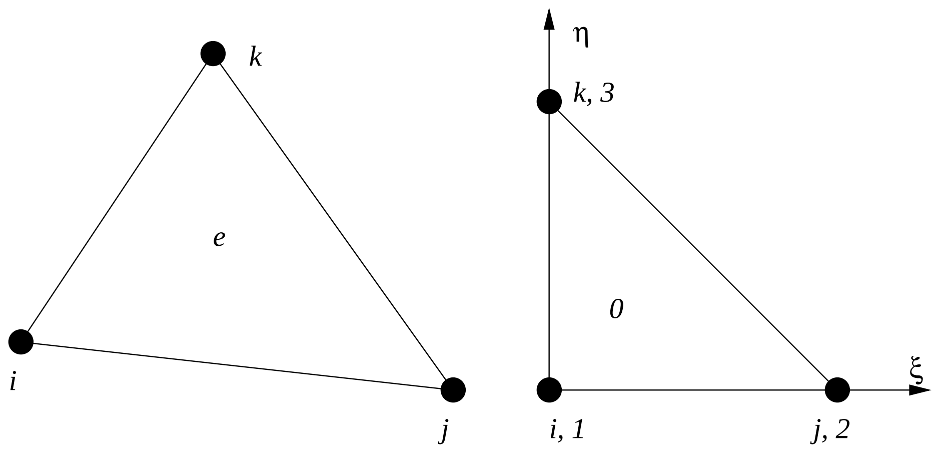


Figure 5.5.1: Linear transformation of a triangular element e to a canonical right 45° triangle.

function on a triangle. Translating the basis construction of Section 4.4 to the canonical element, we obtain an approximation of the form (5.5.6) with

$$\mathbf{c}_e^T = [c_{e,1}, c_{e,2}, ..., c_{e,10}]$$

$$\mathbf{N}^T = [N_1(\xi,\eta), N_2(\xi,\eta), ..., N_{10}(\xi,\eta)].$$

The basis has ten shape functions per element (*cf.* (4.4.5-9)), which are ordered as

$$N_1(\xi,\eta) = \zeta_1 = 1-\xi-\eta, \qquad N_2(\xi,\eta) = \zeta_2 = \xi, \qquad N_3(\xi,\eta) = \zeta_3 = \eta,$$

$$N_4(\xi,\eta) = -\sqrt{6}\zeta_1\zeta_2, \qquad N_5(\xi,\eta) = -\sqrt{6}\zeta_2\zeta_3, \qquad N_6(\xi,\eta) = -\sqrt{6}\zeta_3\zeta_1,$$

$$N_7(\xi,\eta) = -\sqrt{10}\zeta_1\zeta_2(2\xi-1), \qquad N_8(\xi,\eta) = -\sqrt{10}\zeta_2\zeta_3(2\xi-1),$$

$$N_9(\xi,\eta) = -\sqrt{10}\zeta_1\zeta_3(1-2\eta), \qquad N_{10}(\xi,\eta) = \zeta_1\zeta_2\zeta_3.$$

With this ordering, the first three shape functions are associated with the vertices, the next three are quadratic corrections at the midsides, the next three are cubic corrections at the midsides, and the last is a cubic "bubble function" associated with the centroid (Figure 5.5.2).

An array implementation, as described by (5.5.6) and Examples 5.5. 1 and 5.5.2, may be the simplest data structure; however, implementations with structures linked to geometric entities (Section 5.3) are also possible.

Substituting the polynomial representation (5.5.6) into the transformed strain energy expression (5.5.4b) and external load (5.5.4a) yields

$$A_e(V,U) = \mathbf{d}_e^T(\mathbf{K}_e + \mathbf{M}_e)\mathbf{c}_e, \tag{5.5.7a}$$

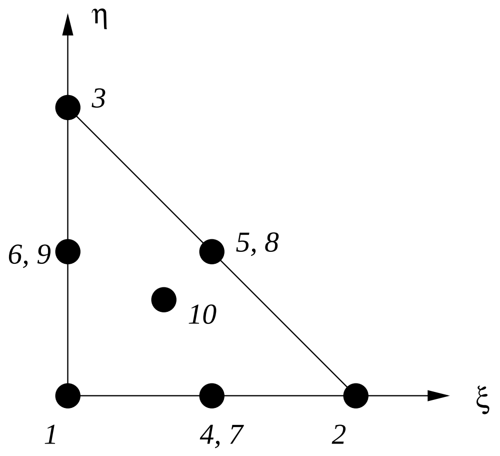


Figure 5.5.2: Shape function placement and numbering for a hierarchical cubic approximation on a canonical right 45° triangle.

$$(V, f)_e = \mathbf{d}_e^T \mathbf{f}_e, \tag{5.5.7b}$$

where

$$\mathbf{K}_e = \iint_{\Omega_0} [g_{1e}\mathbf{N}_\xi \mathbf{N}_\xi^T + g_{2e}(\mathbf{N}_\xi \mathbf{N}_\eta^T + \mathbf{N}_\eta \mathbf{N}_\xi^T) + g_{3e}\mathbf{N}_\eta \mathbf{N}_\eta^T] \det(\mathbf{J}_e) d\xi d\eta, \tag{5.5.8a}$$

$$\mathbf{M}_e = \iint_{\Omega_0} q\mathbf{N}\mathbf{N}^T \det(\mathbf{J}_e) d\xi d\eta, \tag{5.5.8b}$$

$$\mathbf{f}_e = \iint_{\Omega_0} \mathbf{N} f \det(\mathbf{J}_e) d\xi d\eta. \tag{5.5.8c}$$

Here, $\mathbf{K}_e$ and $\mathbf{M}_e$ are the element stiffness and mass matrices and $\mathbf{f}_e$ is the element load vector. Numerical integration will generally be necessary to evaluate these arrays when the coordinate transformation is not linear and we will study procedures to do this in Chapter 6.

Element mass and stiffness matrices and load vectors are generated for all elements in the mesh and *assembled* into their proper locations in the global stiffness and mass matrix and load vector. The positions of the elemental matrices and vectors in their global counterparts are determined by their indexing. In order to illustrate this point, consider a linear shape function on an element with Vertices 4, 7, and 8 as shown in Figure 5.5.3. These vertex indices are mapped onto local indices, *e.g.*, 1, 2, 3, of the canonical element and the correspondence is recorded as shown in Figure 5.5.3. After generating the element matrices and vectors, the global indexing determines where to add these entries into the global stiffnes and mass matrix and load vector. In the example shown in Figure 5.5.3, the entry k^e_{11} is added to Row 4 and Column 4 of the global stiffness matrix $\mathbf{K}$. The entry k^e_{12} is added to Row 4 and Column 7 of $\mathbf{K}$, *etc.*

The assembly process avoids the explicit summations implied by (5.5.3) and yields

$$A(V,U) = \mathbf{d}^T(\mathbf{K}+\mathbf{M})\mathbf{c}, \tag{5.5.9a}$$

$$(V,f) = \mathbf{d}^T\mathbf{f}, \tag{5.5.9b}$$

where

$$\mathbf{c}^T = [c_1, c_2, ..., c_N], \tag{5.5.9c}$$

$$\mathbf{d}^T = [d_1, d_2, ..., d_N], \tag{5.5.9d}$$

where $\mathbf{K}$ is the global stiffness matrix, $\mathbf{M}$ is the global mass matrix, $\mathbf{f}$ is the global load vector, and N is the dimension of the trial space (or the number of degrees of freedom). Imposing the Galerkin condition (5.5.1a)

$$A(V,U) - (V,f) = \mathbf{d}^T[(\mathbf{K}+\mathbf{M})\mathbf{c} - \mathbf{f}] = \mathbf{0}, \qquad \forall \mathbf{d} \in \Re^N, \tag{5.5.10a}$$

yields

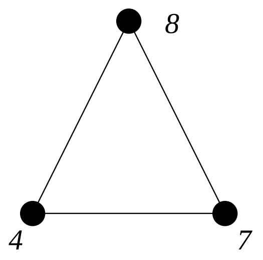
$$(\mathbf{K}+\mathbf{M})\mathbf{c} = \mathbf{f}. \tag{5.5.10b}$$

5.5.2 Essential and Neumann Boundary Conditions

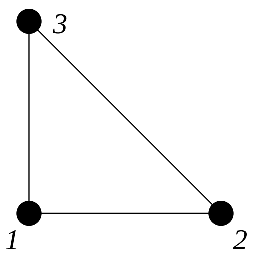
It's customary to ignore any essential boundary conditions during the assembly phase. Were boundary conditions not imposed, the matrix $\mathbf{K}+\mathbf{M}$ would be singular. Essential boundary conditions constrain some of the c_i, $i = 1, 2, ..., N$, and they must be imposed before the algebraic system (5.5.10b) can be solved. In order to simplify the discussion, let us suppose that either $\mathbf{M} = 0$ or that $\mathbf{M}$ has been added to $\mathbf{K}$ so that (5.5.10) may be written as

$$\mathbf{d}^T[\mathbf{K}c - \mathbf{f}] = \mathbf{0}, \qquad \forall \mathbf{d} \in \Re^N, \tag{5.5.11a}$$

$$\mathbf{K}\mathbf{c} = \mathbf{f}. \tag{5.5.11b}$$



Global	Local
4	1
7	2
8	3



$$\mathbf{K}_e = \begin{bmatrix} k^e_{11} & k^e_{12} & k^e_{13} \\ k^e_{21} & k^e_{22} & k^e_{23} \\ k^e_{31} & k^e_{32} & k^e_{33} \end{bmatrix} \qquad \mathbf{f}_e = \begin{bmatrix} f^e_1 \\ f^e_2 \\ f^e_3 \end{bmatrix}$$

$$\mathbf{K} \;=\; \left.\begin{array}{ccccccccc|c} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & \\ \hline & & & & & & & & & 1 \\ & & & & & & & & & 2 \\ & & & & & & & & & 3 \\ & & & +k^e_{11} & & & +k^e_{12} & +k^e_{13} & & 4 \\ & & & & & & & & & 5 \\ & & & & & & & & & 6 \\ & & & +k^e_{21} & & & +k^e_{22} & +k^e_{23} & & 7 \\ & & & +k^e_{31} & & & +k^e_{32} & +k^e_{33} & & 8 \\ & & & & & & & & & 9 \end{array}\right. \qquad \mathbf{f} = \begin{bmatrix} \\ \\ \\ +f^e_1 \\ \\ \\ +f^e_2 \\ +f^e_3 \\ \\ \end{bmatrix}$$

Figure 5.5.3: Assembly of an element stiffness matrix and load vector into their global counterparts for a piecewise-linear polynomial approximation. The actual vertex indices are recorded and stored (top), the element stiffness matrix and load vector are calculated (center), and the indices are used to determine where to add the entries of the elemental matrix and vector into the global stiffness and mass matrix.

Essential boundary conditions may either constrain a single c_i or impose constraints between several nodal variables. In the former case, we partition (5.5.11a) as

$$[\mathbf{d}_1 \mathbf{d}_2] \left\{ \begin{bmatrix} \mathbf{K}_{11} & \mathbf{K}_{12} \\ \mathbf{K}_{21} & \mathbf{K}_{22} \end{bmatrix} \begin{bmatrix} \mathbf{c}_1 \\ \mathbf{c}_2 \end{bmatrix} - \begin{bmatrix} \mathbf{f}_1 \\ \mathbf{f}_2 \end{bmatrix} \right\} = \mathbf{0}, \tag{5.5.12a}$$

where the essential boundary conditions are

$$\mathbf{c}_2 = \boldsymbol{\alpha}_2. \tag{5.5.12b}$$

Recall (Chapters 2 and 3), that the test function V should vanish on $\partial\Omega_E$; thus, corresponding to (5.5.12b)

$$\mathbf{d}_2 = \mathbf{0}. \tag{5.5.12c}$$

The second "block" of equations in (5.5.12a) should never have been generated and, actually, we should have been solving

$$\mathbf{d}_1^T[\mathbf{K}_{11}\mathbf{c}_1 + \mathbf{K}_{12}\mathbf{c}_2 - \mathbf{f}_1] = \mathbf{d}_1^T[\mathbf{K}_{11}\mathbf{c}_1 + \mathbf{K}_{12}\boldsymbol{\alpha}_2 - \mathbf{f}_1] = \mathbf{0}. \tag{5.5.13a}$$

Imposing the Galerkin condition that (5.5.13a) vanish for *all* $\mathbf{d}_1$,

$$\mathbf{K}_{11}\mathbf{c}_1 = \mathbf{f}_1 - \mathbf{K}_{12}\boldsymbol{\alpha}_2. \tag{5.5.13b}$$

Partitioning (5.5.11) need not be done explicitly as in (5.5.11). It can be done implicitly without rearranging equations. Consider the original system (5.5.11b)

$$\begin{bmatrix} k_{11} & & k_{1j} & & k_{1N} \\ \vdots & & \vdots & & \vdots \\ k_{j1} & \cdots & k_{jj} & \cdots & k_{jN} \\ \vdots & & \vdots & & \vdots \\ k_{N1} & & k_{Nj} & & k_{NN} \end{bmatrix} \begin{bmatrix} c_1 \\ \vdots \\ c_j \\ \vdots \\ c_N \end{bmatrix} = \begin{bmatrix} f_1 \\ \vdots \\ f_j \\ \vdots \\ f_N \end{bmatrix}. \tag{5.5.14}$$

Suppose that one boundary condition specifies $c_j = \alpha_j$, then the j *th* equation (row) of the system is deleted, c_j is replaced by the boundary condition, and the coefficients of c_j are moved to the right-hand side to obtain

$$\begin{bmatrix} k_{11} & & k_{1,j-1} & k_{1,j+1} & & k_{1N} \\ \vdots & & \vdots & \vdots & & \vdots \\ k_{j-1,1} & \cdots & k_{j-1,j-1} & k_{j-1,j+1} & \cdots & k_{j-1,N} \\ k_{j+1,1} & \cdots & k_{j+1,j-1} & k_{j+1,j+1} & \cdots & k_{j+1,N} \\ \vdots & & \vdots & \vdots & & \vdots \\ k_{N1} & & k_{N,j-1} & k_{N,j+1} & & k_{NN} \end{bmatrix} \begin{bmatrix} c_1 \\ \vdots \\ c_{j-1} \\ c_{j+1} \\ \vdots \\ c_N \end{bmatrix} = \begin{bmatrix} f_1 - k_{1,j}\alpha_j \\ \vdots \\ f_{j-1} - k_{j-1,j}\alpha_j \\ f_{j+1} - k_{j+1,j}\alpha_j \\ \vdots \\ f_N - k_{N,j}\alpha_j \end{bmatrix}.$$

When the algebraic system is large, the cost of moving data when rows and columns are removed from the system may outweigh the cost of solving a larger algebraic system. In this case, the boundary condition $c_j = \alpha_j$ can be inserted as the j th equation of (5.5.14). Although not necessary, the j th column is usually moved to the right-hand side to preserve symmetry. The resulting larger problem is

$$\begin{bmatrix} k_{11} & & 0 & & k_{1N} \\ \vdots & & \vdots & & \vdots \\ 0 & \cdots & 1 & \cdots & 0 \\ \vdots & & \vdots & & \vdots \\ k_{N1} & & 0 & & k_{NN} \end{bmatrix} \begin{bmatrix} c_1 \\ \vdots \\ c_j \\ \vdots \\ c_N \end{bmatrix} = \begin{bmatrix} f_1 - k_{1,j}\alpha_j \\ \vdots \\ \alpha_j \\ \vdots \\ f_N - k_{N,j}\alpha_j \end{bmatrix}.$$

The treatment of essential boundary conditions that impose constraints among several nodal variables is much more difficult. Suppose, for example, there are l boundary conditions of the form

$$\mathbf{Tc} = \boldsymbol{\alpha}, \tag{5.5.15}$$

where $\mathbf{T}$ is an $l \times N$ matrix and $\boldsymbol{\alpha}$ is an l-vector. In vector systems of partial differential equations, such boundary conditions arise when constraints are specified between different components of the solution vector. In scalar problems, conditions having the form (5.5.15) arise when a "global" boundary condition like

$$\int_{\partial\Omega} u ds = \alpha$$

is specified. They could also arise with periodic boundary conditions which might, for example, specify $u(0, y) = u(1, y)$ if u were periodic in x on a rectangle of unit length.

One could possibly solve (5.5.15) for l values of c_i, $i = 1, 2, ..., N$, in terms of the others. Sometimes there is an obvious choice; however, often there is no clear way to choose the unknowns to eliminate. A poor choice can lead to ill-conditioning of the algebraic system. An alternate way of treating problems with boundary conditions such as (5.5.15) is to embed Problem (5.5.11) in a constrained minimization problem which may be solved using Lagrange multipliers. Assuming $\mathbf{K}$ to be symmetric and positive semi-definite, (5.5.11) can be regarded as the minimum of

$$I[\mathbf{c}] = \mathbf{c}^T\mathbf{Kc} - 2\mathbf{c}^T\mathbf{f}.$$

Using Lagrange multipliers, we minimize the modified functional

$$\tilde{I}[\mathbf{c}, \boldsymbol{\lambda}] = \mathbf{c}^T\mathbf{Kc} - 2\mathbf{c}^T\mathbf{f} + 2\boldsymbol{\lambda}^T(\mathbf{Tc} - \boldsymbol{\alpha}),$$

where $\boldsymbol{\lambda}$ is an l-vector of Lagrange multipliers. Minimizing $\tilde{I}$ with respect to $\mathbf{c}$ and $\boldsymbol{\lambda}$ yields

$$\begin{bmatrix} \mathbf{K} & \mathbf{T}^T \\ \mathbf{T} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{c} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ \boldsymbol{\alpha} \end{bmatrix}. \tag{5.5.16}$$

The system (5.5.16) may or may not be simple to solve. If $\mathbf{K}$ is non-singular then the algorithm described in Problem 2 at the end of this section is effective. However, since boundary conditions are prescribed by (5.5.15), $\mathbf{K}$ may not be invertible.

Nontrivial Neumann boundary conditions on $\partial\Omega_N$ require the evaluation of an extra line integral for those elements having edges on $\partial\Omega_N$. Suppose, for example, that the variational principle (5.5.1) is replaced by: determine $u \in H_E^1$ satisfying

$$A(v,u) = (v,f) + < v, \beta >, \qquad \forall v \in H_0^1, \tag{5.5.17a}$$

where

$$< v, \beta > = \int_{\partial\Omega_N} v\beta(x,y)ds, \tag{5.5.17b}$$

s being a coordinate on $\partial\Omega_N$. As discussed in Chapter 3, smooth solutions of (5.5.17) satisfy (5.5.2a), the essential boundary conditions (5.5.2b), and the natural boundary condition

$$pu_{\mathbf{n}} = \beta, \qquad (x,y) \in \partial\Omega_N. \tag{5.5.18}$$

The line integral (5.5.17b) is evaluated in the same manner as the area integrals and it will alter the load vector $\mathbf{f}$ (*cf.* Problem 3 at the end of this section).

Problems

1. Determine the number of degrees of freedom when a scalar finite element Galerkin problem is solved using either Lagrange or hierarchical bases on a square region having a uniform mesh of either $2n^2$ triangular or n^2 square elements. Express your answer in terms of p and n and compare it with the results of Tables 5.5.1 and 5.5.2.

2. Assume that $\mathbf{K}$ is invertible and show that the following algorithm provides a solution of (5.5.16).

```
Solve $\mathbf{KW} = \mathbf{T}^T$ for $\mathbf{W}$
Let $\mathbf{Y} = \mathbf{TW}$
Solve $\mathbf{Ky} = \mathbf{f}$ for $\mathbf{y}$
Solve $\mathbf{Y}\boldsymbol{\lambda} = \mathbf{Ty} - \boldsymbol{\alpha}$ for $\boldsymbol{\lambda}$
Solve $\mathbf{Kc} = \mathbf{f} - \mathbf{T}^T\boldsymbol{\lambda}$ for $\mathbf{c}$
```

3. Calculate the effect on the element load vector $\mathbf{f}_e$ of a nontrivial Neumann condition having the form (5.5.18).

4. Consider the solution of Laplace's equation

$$u_{xx} + u_{yy} = 0, \qquad (x, y) \in \Omega,$$

on the unit square $\Omega := \{(x, y) | 0 < x, y < 1\}$ with Dirichlet boundary conditions

$$u = \alpha, \qquad (x, y) \in \partial\Omega.$$

As described in the beginning of this section, create a mesh by dividing the unit square into n^2 uniform square elements and then into $2n^2$ triangles by cutting each square element in half along its positive sloping diagonal.

   4.1. Using a Galerkin formulation with a piecewise-linear basis, develop the element stiffness matrices for each of the two types of elements in the mesh.

   4.2. Assemble the element stiffness matrices to form the global stiffness matrix.

   4.3. Apply the Dirichlet boundary conditions and exhibit the final linear algebraic system for the nodal unknowns.

5. The task is is to solve a Dirichlet problem on a square using available finite element software. The problem is

$$-u_{xx} - u_{yy} + f(x, y) = 0, \qquad (x, y) \in \Omega,$$

with $u = 0$ on the boundary of the unit square $\Omega = \{(x, y) | 0 \le x, y \le 1\}$. Select $f(x, y)$ so that the exact solution of the problem is

$$u(x, y) = e^x y \sin \pi x \sin 2\pi y.$$

The Galerkin form of this problem is to find $u \in H_0^1$ satisfying

$$\iint_\Omega [v_x u_x + v_y u_y + vf] dx dy = 0, \qquad \forall v \in H_0^1.$$

Solve this problem on a sequence of finer and finer grids using piecewise linear, quadratic, and cubic finite element bases. Select a basic grid with either two or four elements in it and obtain finer grids by uniform refinement of each element into four elements. Present plots of the energy error as functions of the number of degrees of freedom (DOF), the mesh spacing h, and the CPU time for the three polynomial bases. Define h as the square root of the area of an average element.

You may combine the convergence histories for the three polynomial solutions on one graph. Thus, you'll have three graphs, error vs. h, error vs. DOF, and error vs. CPU time, each having results for the three polynomial solutions. Estimate the convergence rates of the solutions. Comment on the results. Are they converging at the theoretical rates? Are there any unexpected anomalies? If so, try to explain them. You may include plots of solutions and/or meshes to help answer these questions.

6. Consider the Dirichlet problem for Laplace's equation

$$\Delta u = u_{xx} + u_{yy} = 0, \qquad (x,y) \in \Omega,$$

$$u(x,y) = \alpha(x,y), \qquad (x,y) \in \partial\Omega,$$

where Ω is the L-shaped region with lines connecting the Cartesian vertices (0,0), (1,0), (1,1), (-1,1), (-1,-1), (0,-1), (0,0). Select $\alpha(x,y)$ so that the exact solution expressed in polar coordinates is

$$u(r,\theta) = r^{2/3} \sin \frac{2\theta}{3}.$$

with

$$x = r\cos\theta, \qquad y = r\sin\theta.$$

This solution has a singularity in its first derivative at $r = 0$. The singularity is typical of those associated with the solution of elliptic problems at re-entrant corners such as the one found at the origin.

Because of symmetries, the problem need only be solved on half of the L-shaped domain, *i.e.*, the trapezoidal region $\tilde{\Omega}$ with lines connecting the Cartesian vertices (0,0), (1,0), (1,1), (-1,1), (0,0).

The Galerkin form of this problem consists of determining $u \in H^1_E$

$$\iint_{\tilde{\Omega}} [v_x u_x + v_y u_y] dxdy = 0, \qquad \forall v \in H^1_0.$$

Functions $u \in H^1_E$ satisfy the essential boundary conditions

$$u(x,y) = 0, \qquad y = 0, \qquad 0 < x < 1,$$

$$u(r,\theta) = r^{2/3} \sin \frac{2\theta}{3}, \qquad x = 1, \qquad 0 \le y < 1, \qquad y = 1, \qquad -1 < x \le 1.$$

These boundary conditions may be expressed in Cartesian coordinates by using

$$r^2 = x^2 + y^2, \qquad \tan\theta = \frac{y}{x}.$$

The solution of the Galerkin problem will also satisfy the natural boundary condition $u_{\mathbf{n}} = u_\theta = 0$ along the diagonal $y = -x$.

Solve this problem using available finite element software. To begin, create a three-element initial mesh by placing lines between the vertices (0,0) and (1,1) and between (0,0) and (0,1). Generate finer meshes by uniform refinement and use piecewise-polynomial bases of degrees one through three.

As in Problem 5, present plots of the energy error as functions of the number of degrees of freedom, the mesh spacing h, and the CPU time for the three polynomial bases. You may combine the convergence histories for the three polynomial solutions on one graph. Define h as the square root of the area of an average element. Estimate the convergence rates of the solutions. Is accuracy higher with a high-order method on a coarse mesh or with a low-order method on a fine mesh?

If adaptivity is available, use a piecewise-linear basis to calculate a solution using adaptive h-refinement. Plot the energy error of this solution with those of the uniform-mesh solutions. Is the adaptive solution more efficient? Efficiency may be defined as less CPU time or fewer degrees of freedom for the same accuracy. Contrast the uniform and adaptive meshes.

5.6 Assembly of Vector Systems

Vector systems of partial differential equations may be treated in the same manner as the scalar problems described in the previous section. As an example, consider the vector version of the model problem (5.5.1): determine $\mathbf{u} \in H_E^1$ satisfying

$$A(\mathbf{v}, \mathbf{u}) = (\mathbf{v}, \mathbf{f}), \qquad \forall v \in H_0^1, \tag{5.6.1a}$$

where

$$(\mathbf{v}, \mathbf{f}) = \iint_\Omega \mathbf{v}^T \mathbf{f} dxdy, \tag{5.6.1b}$$

$$A(\mathbf{v}, \mathbf{u}) = \iint_\Omega [\mathbf{v}_x^T \mathbf{P} \mathbf{u}_x + \mathbf{v}_y^T \mathbf{P} \mathbf{u}_y + \mathbf{v}^T \mathbf{Q} \mathbf{u}] dxdy. \tag{5.6.1c}$$

The functions $\mathbf{u}(x,y)$, $\mathbf{v}(x,y)$, and $\mathbf{f}(x,y)$ are m-vectors and $\mathbf{P}$ and $\mathbf{Q}$ are $m \times m$ matrices. Smooth solutions of (5.6.1) satisfy

$$-(\mathbf{P}\mathbf{u}_x)_x - (\mathbf{P}\mathbf{u}_y)_y + \mathbf{Q}\mathbf{u} = \mathbf{f}, \qquad (x,y) \in \Omega, \tag{5.6.2a}$$

$$\mathbf{u} = \boldsymbol{\alpha}, \qquad (x, y) \in \partial\Omega_D, \qquad \mathbf{u_n} = \mathbf{0}, \qquad (x, y) \in \partial\Omega_N. \tag{5.6.2b}$$

Example 5.6.1. Consider the biharmonic equation

$$\Delta^2 w = f(x, y), \qquad (x, y) \in \Omega,$$

where

$$\Delta(\ ) := (\ )_{xx} + (\ )_{yy}$$

is the Laplacian and Ω is a bounded two-dimensional region. Problems involving biharmonic operators arise in elastic plate deformation, slow viscous flow, combustion, *etc.* Depending on the boundary conditions, this problem may be transformed to a system of two second-order equations having the form (5.6.2). For example, it seems natural to let

$$u_1 = -\Delta w$$

then

$$-\Delta u_1 = f.$$

Let $w = -u_2$ to obtain the vector system

$$-\Delta u_1 = f, \qquad -\Delta u_2 + u_1 = 0, \qquad (x, y) \in \Omega.$$

This system has the form (5.6.2) with

$$\mathbf{u} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}, \qquad \mathbf{P} = \mathbf{I}, \qquad \mathbf{Q} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}, \qquad \mathbf{f} = \begin{bmatrix} f \\ 0 \end{bmatrix}.$$

The simplest boundary conditions to prescribe are

$$w = -u_2 = \alpha_2, \qquad \Delta w = -u_1 = \alpha_1, \qquad (x, y) \in \partial\Omega.$$

With these (Dirichlet) boundary conditions, the variational form of this problem is (5.6.1a) with

$$(\mathbf{v}, \mathbf{f}) = \iint_\Omega v_1 f \, dxdy$$

and

$$A(\mathbf{v}, \mathbf{u}) = \iint_\Omega [(v_1)_x(u_1)_x + (v_1)_y(u_1)_y + (v_2)_x(u_2)_x + (v_2)_y(u_2)_y + v_2 u_1] dxdy.$$

The requirement that (5.6.1a) be satisfied for all vector functions $\mathbf{v} \in H_0^1$ gives the two scalar variational problems

$$\iint_\Omega [(v_1)_x(u_1)_x + (v_1)_y(u_1)_y - v_1 f] dxdy = 0, \qquad \forall v_1 \in H_0^1,$$

$$\iint_{\Omega} [(v_2)_x(u_2)_x + (v_2)_y(u_2)_y + v_2 u_1]dxdy = 0, \qquad \forall v_2 \in H_0^1.$$

We may check that smooth solutions of these variational problems satisfy the pair of second-order differential equations listed above.

We note in passing, that the boundary conditions presented with this example are not the only ones of interest. Other boundary conditions do not separate the system as neatly.

Following the procedures described in Section 5.5, we evaluate (5.6.1) in an element-by-element manner and transform the elemental strain energy and load vector to the canonical element to obtain

$$A_e(\mathbf{V}, \mathbf{U}) = \iint_{\Omega_0} [\mathbf{V}_{0_\xi}^T \mathbf{G}_{1e} \mathbf{U}_{0_\xi} + \mathbf{V}_{0_\xi}^T \mathbf{G}_{2e} \mathbf{U}_{0_\eta} + \mathbf{V}_{0_\eta}^T \mathbf{G}_{2e} \mathbf{U}_{0_\xi} +$$

$$\mathbf{V}_{0_\eta}^T \mathbf{G}_{3e} \mathbf{U}_{0_\eta} + \mathbf{V}_0^T \mathbf{Q} \mathbf{U}_0] \det(\mathbf{J}_e) d\xi d\eta, \tag{5.6.3a}$$

where

$$\mathbf{G}_{1e} = \mathbf{P}[\xi_x^2 + \xi_y^2], \qquad \mathbf{G}_{2e} = \mathbf{P}[\xi_x \eta_x + \xi_y \eta_y], \qquad \mathbf{G}_{3e} = \mathbf{P}[\eta_x^2 + \eta_y^2], \tag{5.6.3b}$$

and

$$(\mathbf{V}, \mathbf{f})_e = \iint_{\Omega_0} \mathbf{V}_0^T \mathbf{f} \det(\mathbf{J}_e) d\xi d\eta. \tag{5.6.3c}$$

The restriction of the piecewise-polynomial approximation $\mathbf{U}_0$ to element e is written in terms of shape functions as

$$\mathbf{U}_0(\xi, \eta) = \sum_{j=1}^{n_p} \mathbf{c}_{e,j} N_j(\xi, \eta) \tag{5.6.4a}$$

where n_p is the number of shape functions on element e. We have divided the vector $\mathbf{c}_e$ of parameters into its contributions $\mathbf{c}_{e,j}$, $j = 1, 2, ..., n$, from the shape functions of element e. Thus, we may write

$$\mathbf{c}_e^T = [\mathbf{c}_{e,1}^T, \mathbf{c}_{e,2}^T, ..., \mathbf{c}_{e,n_p}^T]. \tag{5.6.4b}$$

In this form, we may write $\mathbf{U}_0$ as

$$\mathbf{U}_0 = \mathbf{N}^T \mathbf{c}_e = \mathbf{c}_e^T \mathbf{N} \tag{5.6.4c}$$

where $\mathbf{N}$ is the $n_pm \times m$ matrix

$$\mathbf{N}^T = [N_1\mathbf{I}, N_2\mathbf{I}, ..., N_{n_p}\mathbf{I}]. \tag{5.6.4d}$$

and the identity matrices have the dimension m of the partial differential system. The simple linear shape functions will illustrate the formulation.

Example 5.6.2. Consider the solution of a system of $m = 2$ equations using a piecewise-linear finite element basis on triangles. Suppose, for convenience, that the node numbers of element e are 1, 2, and 3. In order to simplify the notation, we suppress the subscript 0 on $\mathbf{U}_0$ and $\mathbf{V}_0$ and the subscript e on $\mathbf{c}_e$. The linear approximation on element e then takes the form

$$\begin{bmatrix} U_1 \\ U_2 \end{bmatrix} = \begin{bmatrix} c_{11} \\ c_{21} \end{bmatrix} N_1(\xi,\eta) + \begin{bmatrix} c_{12} \\ c_{22} \end{bmatrix} N_2(\xi,\eta) + \begin{bmatrix} c_{13} \\ c_{23} \end{bmatrix} N_3(\xi,\eta),$$

where

$$N_1(\xi,\eta) = 1 - \xi - \eta, \qquad N_2(\xi,\eta) = \xi, \qquad N_3(\xi,\eta) = \eta.$$

The first subscript on c_{ij} denotes its index in $\mathbf{c}$ and the second subscript identifies the vertex of element e. The expression (5.6.4c) takes the form

$$\begin{bmatrix} U_1 \\ U_2 \end{bmatrix} = \begin{bmatrix} N_1 & 0 & N_2 & 0 & N_3 & 0 \\ 0 & N_1 & 0 & N_2 & 0 & N_3 \end{bmatrix} \begin{bmatrix} c_{11} \\ c_{21} \\ c_{12} \\ c_{22} \\ c_{13} \\ c_{23} \end{bmatrix}.$$

Substituting (5.6.4c) and a similar expression for $\mathbf{V}_0$ into (5.6.3a,e) yields

$$A_e(\mathbf{V},\mathbf{U}) = \mathbf{d}_e^T(\mathbf{K}_e + \mathbf{M}_e)\mathbf{c}_e, \qquad (\mathbf{V},\mathbf{f})_e = \mathbf{d}_e^T\mathbf{f}_e, \tag{5.6.5}$$

where

$$\mathbf{K}_e = \iint_{\Omega_0} [\mathbf{N}_\xi\mathbf{G}_{1e}\mathbf{N}_\xi^T + \mathbf{N}_\xi\mathbf{G}_{2e}\mathbf{N}_\eta^T + \mathbf{N}_\eta\mathbf{G}_{2e}\mathbf{N}_\xi^T + \mathbf{N}_\eta\mathbf{G}_3 e\mathbf{N}_\eta^T]\det(\mathbf{J}_e)^d\xi d\eta, \tag{5.6.6a}$$

$$\mathbf{M}_e = \iint_{\Omega_0} \mathbf{N}\mathbf{Q}\mathbf{N}^T \ \det(\mathbf{J}_e)d\xi d\eta, \mathbf{f}_e = \iint_{\Omega_0} \mathbf{N}\mathbf{f}\det(\mathbf{J}_e)d\xi d\eta. \tag{5.6.6b}$$

Problems

1. It is, of course, possible to use different shape functions for different solution components. This is often done with incompressible flows where the pressure is approximated by a basis having one degree less than that used for velocity. Variational formulations with different fields are called *mixed variational principles.* The resulting finite element formulations are called *mixed methods.* As an example, consider a vector problem having two components. Suppose that a piecewise-linear basis is used for the first variable and piecewise quadratics are used for the second. Using hierarchical bases, select an ordering of unknowns and write the form of the finite element solution on a canonical two-dimensional element. What are the components of the matrix $\mathbf{N}$? For this approximation, develop a formula for the element stiffness matrix (5.6.6a). Express your answer in terms of the matrices $\mathbf{G}_{ie}$, $i = 1, 2, 3$, and integrals of the shape functions.

Bibliography

[1] I. Babuška, J.E. Flaherty, W.D. Henshaw, J.E. Hopcroft, J.E. Oliger, and T. Tezduyar, editors. *Modeling, Mesh Generation, and Adaptive Numerical Methods for Partial Differential Equations*, volume 75 of *The IMA Volumes in Mathematics and its Applications*, New York, 1995. Springer-Verlag.

[2] P.L. Baehmann, M.S. Shephard, and J.E. Flaherty. Adaptive analysis for automated finite element modeling. In J.R. Whiteman, editor, *The Mathematics of Finite Elements and Applications VI, MAFELAP 1987*, pages 521–532, London, 1988. Academic Press.

[3] P.L. Baehmann, S.L. Witchen, M.S. Shephard, K.R. Grice, and M.A. Yerry. Robust, geometrically-based, automatic two-dimensional mesh generation. *International Journal of Numerical Methods in Engineering*, 24:1043–1078, 1987.

[4] M.W. Beall and M.S. Shephard. A general topology-based mesh data structure. *International Journal of Numerical Methods in Engineering*, 40:1573–1596, 1997.

[5] M.W. Bern, J.E. Flaherty, and M. Luskin, editors. *Grid Generation and Adaptive Algorithms*, volume 113 of *The IMA Volumes in Mathematics and its Applications*, New York, 1999. Springer.

[6] G.F. Carey. *Computational Grids: Generation, Adaptation, and Solution Strategies.* Series in Computational and Physical Processes in Mechanics and Thermal science. Taylor and Francis, New York, 1997.

[7] J.E. Flaherty, R. Loy, M.S. Shephard, B.K. Szymanski, J. Teresco, and L. Ziantz. Adaptive local refinement with octree load-balancing for the parallel solution of three-dimensional conservation laws. *Parallel and Distributed Computing*, 1998. to appear.

[8] J.E. Flaherty, R.M. Loy, C. Özturan, M.S. Shephard, B.K. Szymanski, J.D. Teresco, and L.H. Ziantz. Parallel structures and dynamic load balancing for adaptive finite element computation. *Applied Numerical Mathematics*, 26:241–265, 1998.

[9] J.E. Flaherty, P.J. Paslow, M.S. Shephard, and J.D. Vasilakis, editors. *Adaptive methods for Partial Differential Equations*, Philadelphia, 1989. SIAM.

[10] R. Löhner. Finite element methods in CFD: Grid generation, adaptivity and parallelization. In H. Deconinck and T. Barth, editors, *Unstructured Grid Methods for Advection Dominated Flows*, number AGARD Report AGARD-R-787, Neuilly sur Seine, 1992. Chapter 8.

[11] R. Verfürth. *A Review of Posteriori Error Estimation and Adaptive Mesh-Refinement Techniques*. Teubner-Wiley, Stuttgart, 1996.

Chapter 6

Numerical Integration

6.1 Introduction

After transformation to a canonical element Ω_0, typical integrals in the element stiffness or mass matrices (*cf.* (5.5.8)) have the forms

$$\mathbf{Q} = \iint\limits_{\Omega_0} \alpha(\xi,\eta)\mathbf{N}_s\mathbf{N}_t^T \det(\mathbf{J}_e) d\xi d\eta, \tag{6.1.1a}$$

where $\alpha(\xi,\eta)$ depends on the coefficients of the partial differential equation and the transformation to Ω_0 (*cf.* Section 5.4). The subscripts s and t are either nil, ξ, or η implying no differentiation, differentiation with respect to ξ, or differentiation with respect to η, respectively. Assuming that $\mathbf{N}$ has the form

$$\mathbf{N}^T = [N_1, N_2, \dots, N_{n_p}], \tag{6.1.1b}$$

then (6.1.1a) may be written in the more explicit form

$$\mathbf{Q} = \iint\limits_{\Omega_0} \alpha(\xi,\eta) \begin{bmatrix} (N_1)_s(N_1)_t & (N_1)_s(N_2)_t & & (N_1)_s(N_{n_p})_t \\ (N_2)_s(N_1)_t & (N_2)_s(N_2)_t & & (N_2)_s(N_{n_p})_t \\ & & \ddots & \\ (N_{n_p})_s(N_1)_t & (N_{n_p})_s(N_2)_t & & (N_{n_p})_s(N_{n_p})_t \end{bmatrix} \det(\mathbf{J}_e) d\xi d\eta. \tag{6.1.1c}$$

Integrals of the form (6.1.1b) may be evaluated exactly when the coordinate transformation is linear ($\mathbf{J}_e$ is constant) and the coefficients of the differential equation are constant (*cf.* Problem 1 at the end of this section). With certain coefficient functions and transformations it may be possible to evaluate (6.1.1b) exactly by symbolic integration; however, we'll concentrate on numerical integration because:

- it can provide exact results in simple situations (*e.g.*, when α and $\mathbf{J}_e$ are constants) and

- exact integration is not needed to achieve the optimal convergence rate of finite element solutions ([2, 9, 11], and Chapter 7).

Integration is often called *quadrature* in one dimension and *cubature* in higher dimensions; however, we'll refer to all numerical approximations as *quadrature rules.* We'll consider integrals and quadrature rules of the form

$$I = \iint_{\Omega_0} f(\xi, \eta) d\xi d\eta \approx \sum_{i=1}^{n} W_i f(\xi_i, \eta_i). \tag{6.1.2a}$$

where W_i, are the quadrature rule's *weights* and (ξ_i, η_i) are the *evaluation points*, $i = 1, 2, \dots, n$. Of course, we'll want to appraise the accuracy of the approximate integration and this is typically done by indicating those polynomials that are integrated exactly.

Definition 6.1.1. The integration rule (6.1.2a) is *exact to order* q if it is exact when $f(\xi, \eta)$ is any polynomial of degree q or less.

When the integration rule is exact to order q and $f(\xi, \eta) \in H^{q+1}(\Omega_0)$, the error

$$E = I - \sum_{i=1}^{n} W_i f(\xi_i, \eta_i) \tag{6.1.2b}$$

satisfies an estimate of the form

$$E \leq C||f(\cdot, \cdot)||_{q+1}. \tag{6.1.2c}$$

Example 6.1.1. Applying (6.1.2) to (6.1.1a) yields

$$\mathbf{Q} \approx \sum_{i=1}^{n} W_i \alpha(\xi_i, \eta_i) \mathbf{N}(\xi_i, \eta_i) \mathbf{N}^T(\xi_i, \eta_i) \det(\mathbf{J}_e(\xi_i, \eta_i)).$$

Thus, the integrand at the evaluation points is summed relative to the weights to approximate the given integral.

Problems

1. A typical term of an element stiffness or mass matrix has the form

$$\iint_{\Omega_0} \xi^i \eta^j d\xi d\eta, \qquad i, j, \geq 0.$$

Evaluate this integral when Ω_0 is the canonical square $[-1, 1] \times [-1, 1]$ and the canonical right 45° unit triangle.

6.2 One-Dimensional Gaussian Quadrature

Although we are primarily interested in two- and three-dimensional quadrature rules, we'll set the stage by studying one-dimensional integration. Thus, consider the one-dimensional equivalent of (6.1.2) on the canonical $[-1, 1]$ element

$$I = \int_{-1}^{1} f(\xi)d\xi = \sum_{i=1}^{n} W_i f(\xi_i) + E. \tag{6.2.1}$$

Most classical quadrature rules have this form. For example,the trapezoidal rule

$$I \approx f(-1) + f(1)$$

has the form (6.2.1) with $n = 2$, $W_1 = W_2 = 1$, $-\xi_1 = \xi_2 = 1$, and

$$E = -\frac{2f''(\sigma)}{3}, \qquad \sigma \in (-1, 1).$$

Similarly, Simpson's rule

$$I \approx \frac{1}{3}[f(-1) + 4f(0) + f(1)]$$

has the form (6.2.1) with $n = 3$, $W_1 = W_2/4 = W_3 = 1/3$, $-\xi_1 = \xi_3 = 1$, $\xi_2 = 0$, and

$$E = -\frac{f^{(iv)}(\sigma)}{90}, \qquad \sigma \in (-1, 1).$$

Gaussian quadrature is preferred to these Newton-Cotes formulas for finite element applications because they have fewer function evaluations for a given order. With Gaussian quadrature, the weights and evaluation points are determined so that the integration rule is exact ($E = 0$) to as high an order as possible. Since there are $2n$ unknown weights and evaluation points, we expect to be able to make (6.2.1) exact to order $2n - 1$. This problem has been solved [3, 6] and the evaluation points ξ_i, $i = 1, 2, \ldots , n$, are the roots of the Legendre polynomial of degree n (*cf.* Section 2.5). The weights W_i, $i = 1, 2, \ldots , n$, called *Christoffel weights*, are also known and are tabulated with the evaluation points in Table 6.2.1 for n ranging from 1 to 6. A more complete set of values appear in Abromowitz and Stegun [1].

Example 6.2.1. The derivation of the two-point ($n = 2$) Gauss quadrature rule is given as Problem 1 at the end of this section. From Table 6.2.1 we see that $W_1 = W_2 = 1$ and $-\xi_1 = \xi_2 = 1/\sqrt{3}$. Thus, the quadrature rule is

$$\int_{-1}^{1} f(\xi)d\xi \approx f(-1/\sqrt{3}) + f(1/\sqrt{3}).$$

This formula is exact to order three; thus the error is proportional to the fourth derivative of f (*cf.* Theorem 6.2.1, Example 6.2.4, and Problem 2 at the end of this section).

n	$\pm\xi_i$	W_i
1	0.00000 00000 00000	2.00000 00000 00000
2	0.57735 02691 89626	1.00000 00000 00000
3	0.00000 00000 00000	0.88888 88888 88889
	0.77459 66692 41483	0.55555 55555 55556
4	0.33998 10435 84856	0.65214 51548 62546
	0.86113 63115 94053	0.34785 48451 37454
5	0.00000 00000 00000	0.56888 88888 88889
	0.53846 93101 05683	0.47862 86704 99366
	0.90617 98459 38664	0.23692 68850 56189
6	0.23861 91860 83197	0.46791 39345 72691
	0.66120 93864 66265	0.36076 15730 48139
	0.93246 95142 03152	0.17132 44923 79170

Table 6.2.1: Christoffel weights W_i and roots ξ_i, $i = 1, 2, \ldots, n$, for Legendre polynomials of degrees 1 to 6 [1].

Example 6.2.2. Consider evaluating the integral

$$I = \int_0^1 e^{-x^2} dx = \frac{\sqrt{\pi}}{2}\text{erf}(1) = 0.74682413281243 \tag{6.2.2}$$

by Gauss quadrature. Let us transform the integral to $[-1, 1]$ using the mapping

$$\xi = 2x - 1$$

to get

$$I = \frac{1}{2}\int_{-1}^{1} e^{-(\frac{1+\xi}{2})^2} d\xi.$$

The two-point Gaussian approximation is

$$I \approx \tilde{I} = \frac{1}{2}[e^{-(\frac{1-1/\sqrt{3}}{2})^2} + e^{-(\frac{1+1/\sqrt{3}}{2})^2}].$$

Other approximations follow in similar order.

Errors $I - \tilde{I}$ when I is approximated by Gaussian quadrature to obtain $\tilde{I}$ appear in Table 6.2.2 for n ranging from 1 to 6. Results using the trapezoidal and Simpson's rules are also presented. The two- and three-point Gaussian rules have higher orders than the corresponding Newton-Cotes formulas and this leads to smaller errors for this example.

n	Gauss Rules Error	Newton Rules Error
1	3.198(- 2)	
2	-2.294(- 4)	-6.288(- 2)
3	-9.549(- 6)	3.563(- 4)
4	3.353(- 7)	
5	-6.046(- 9)	
6	7.772(-11)	

Table 6.2.2: Errors in approximating the integral of Example 6.2.2 by Gauss quadrature, the trapezoidal rule ($n = 2$, right) and Simpson's rule ($n = 3$, right). Numbers in parentheses indicate a power of ten.

Example 6.2.3. Composite integration formulas, where the domain of integration $[a, b]$ is divided into N subintervals of width

$$\Delta x_j = x_j - x_{j-1}, \qquad j = 1, 2, \ldots, N,$$

are not needed in finite element applications, except, perhaps, for postprocessing. However, let us do an example to illustrate the convergence of a Gaussian quadrature formula. Thus, consider

$$I = \int_a^b f(x)dx = \sum_{j=1}^{n} I_j$$

where

$$I_j = \int_{x_{j-1}}^{x_j} f(x)dx.$$

The linear mapping

$$x = x_{j-1}\frac{1-\xi}{2} + x_j\frac{1+\xi}{2}$$

transforms $[x_{j-1}, x_j]$ to $[-1, 1]$ and

$$I_j = \frac{\Delta x_j}{2}\int_{-1}^{1} f(x_{j-1}\frac{1-\xi}{2} + x_j\frac{1+\xi}{2})d\xi.$$

Approximating I_j by Gauss quadrature gives

$$I_j \approx \frac{\Delta x_j}{2}\sum_{i=1}^{n} W_i f(x_{j-1}\frac{1-\xi_i}{2} + x_j\frac{1+\xi_i}{2}).$$

We'll approximate (6.2.2) using composite two-point Gauss quadrature; thus,

$$I_j = \frac{\Delta x_j}{2}[e^{-(x_{j-1/2}-\Delta x_j/(2\sqrt{3}))^2} + e^{-(x_{j-1/2}+\Delta x_j/(2\sqrt{3}))^2}],$$

where $x_{j-1/2} = (x_j + x_{j-1})/2$. Assuming a uniform partition with $\Delta x_j = 1/N$, $j = 1, 2, \ldots, N$, the composite two-point Gauss rule becomes

$$I \approx \frac{1}{2N} \sum_{j=1}^{n} [e^{-(x_{j-1/2} - 1/(2N\sqrt{3}))^2} + e^{-(x_{j-1/2} + 1/(2N\sqrt{3}))^2}].$$

The composite Simpson's rule,

$$I \approx \frac{1}{3N} [1 + 4 \sum_{i=1,3}^{N-1} e^{-x_j} + 2 \sum_{i=2,4}^{N-2} e^{-x_j} + e^{-1}]$$

on $N/2$ subintervals of width $2\Delta x$ has an advantage relative to the composite Gauss rule since the function evaluations at the even-indexed points combine.

The number of function evaluations and errors when (6.2.2) is solved by the composite two-point Gauss and Simpson's rules are recorded in Table 6.2.3. We can see that both quadrature rules are converging as $O(1/N^4)$ ([6], Chapter 7). The computations were done in single precision arithmetic as opposed to those appearing in Table 6.2.2, which were done in double precision. With single precision, round-off error dominates the computation as N increases beyond 16 and further reductions of the error are impossible. With function evaluations defined as the number of times that the exponential is evaluated, errors for the same number of function evaluations are comparable for Gauss and Simpson's rule quadrature. As noted earlier, this is due to the combination of function evaluations at the ends of even subintervals. Discontinuous solution derivatives at inter-element boundaries would prevent such a combination with finite element applications.

N	Gauss Rules		Simpson's Rule	
	Fn. Eval.	Abs. Error	Fn. Eval.	Abs. Error
2	4	0.208(- 4)	3	0.356(- 3)
4	8	0.161(- 5)	5	0.309(- 4)
8	16	0.358(- 6)	9	0.137(- 5)
16	32	0.364(- 5)	17	0.244(-5)

Table 6.2.3: Comparison of composite two-point Gauss and Simpson's rule approximations for Example 6.2.3. The absolute error is the magnitude of the difference between the exact and computational result. The number of times that the exponential function is evaluated is used as a measure of computational effort.

As we may guess, estimates of errors for Gauss quadrature use the properties of Legendre polynomials (*cf.* Section 2.5). Here is a typical result.

Theorem 6.2.1. *Let $f(\xi) \in C^{2n}[-1,1]$, then the quadrature rule (6.2.1) is exact to order $2n-1$ if ξ_i, $i = 1, 2, \dots, n$, are the roots of $P_n(\xi)$, the nth-degree Legendre polynomial, and the corresponding Christoffel weights satisfy*

$$W_i = \frac{1}{P_n'(\xi_i)} \int_{-1}^{1} \frac{P_n(\xi)}{\xi - \xi_i} d\xi, \qquad i = 1, 2, \dots, n. \tag{6.2.3a}$$

Additionally, there exists a point $\zeta \in (-1,1)$ such that

$$E = \frac{f^{(2n)}(\zeta)}{2n!} \int_{-1}^{1} \prod_{i=1}^{n} (\xi - \xi_i)^2 d\xi. \tag{6.2.3b}$$

Proof. *cf.* [6], Sections 7.3, 4. □

Example 6.2.4. Using the entries in Table 6.2.1 and (6.2.3b), the discretization error of the two-point ($n = 2$) Gauss quadrature rule is

$$E = \frac{f^{iv}(\zeta)}{4!} \int_{-1}^{1} (\xi + \frac{1}{\sqrt{3}})^2 (\xi - \frac{1}{\sqrt{3}})^2 d\xi = \frac{f^{iv}(\zeta)}{135}, \qquad \zeta \in (-1,1).$$

Problems

1. Calculate the weights W_1 and W_2 and the evaluation points ξ_1 and ξ_2 so that the two-point Gauss quadrature rule

$$\int_{-1}^{1} f(x) \approx W_1 f(\xi_1) + W_2 f(\xi_2)$$

   is exact to as high an order as possible. This should be done by a direct calculation without using the properties of Legendre polynomials.

2. Lacking the precise information of Theorem 6.2.1, we may infer that the error in the two-point Gauss quadrature rule is proportional to the fourth derivative of $f(\xi)$ since cubic polynomials are integrated exactly. Thus,

$$E = C f^{iv}(\zeta), \qquad \zeta \in (-1,1).$$

   We can determine the error coefficient C by evaluating the formula for any function $f(x)$ whose fourth derivative does not depend on the location of the unknown point ζ. In particular, any quartic polynomial has a constant fourth derivative; hence, the value of ζ is irrelevant. Select an appropriate quartic polynomial and show that $C = 1/135$ as in Example 6.2.4.

6.3 Multi-Dimensional Quadrature

Integration on square elements usually relies on tensor products of the one-dimensional formulas illustrated in Section 6.2. Thus, the application of (6.2.1) to a two-dimensional integral on a canonical $[-1, 1] \times [-1, 1]$ square element yields the approximation

$$I = \int_{-1}^{1}\int_{-1}^{1} f(\xi,\eta)d\xi d\eta \approx \int_{-1}^{1}\sum_{i=1}^{n} W_i f(\xi_i,\eta)d\eta = \sum_{i=1}^{n} W_i \int_{-1}^{1} f(\xi_i,\eta)d\eta$$

and

$$I = \int_{-1}^{1}\int_{-1}^{1} f(\xi,\eta)d\xi d\eta \approx \sum_{i=1}^{n}\sum_{j=1}^{n} W_i W_j f(\xi_i,\eta_j). \tag{6.3.1}$$

Error estimates follow the one-dimensional analysis.

Tensor-product formulas are not optimal in the sense of using the fewest function evaluations for a given order. Exact integration of a quintic polynomial by (6.3.1) would require $n = 3$ or a total of 9 points. A complete quintic polynomial in two dimensions has 21 monomial terms; thus, a direct (non-tensor-product) formula of the form

$$I = \int_{-1}^{1}\int_{-1}^{1} f(\xi,\eta)d\xi d\eta \approx \sum_{i=1}^{n} W_i f(\xi_i,\eta_i)$$

could be made exact with only 7 points. The 21 coefficients W_i, ξ_i, η_i, $i = 1, 2, \ldots, 7$, could potentially be determined to exactly integrate all of the monomial terms.

Non-tensor-product formulas are complicated to derive and are not known to very high orders. Orthogonal polynomials, as described in Section 6.2, are unknown in two and three dimensions. Quadrature rules are generally derived by a method of undetermined coefficients. We'll illustrate this approach by considering an integral on a canonical right 45° triangle

$$I = \iint_{\Omega_0} f(\xi,\eta)d\xi d\eta = \sum_{i=1}^{n} W_i f(\xi_i,\eta_i) + E. \tag{6.3.2}$$

Example 6.3.1. Consider the one-point quadrature rule

$$\iint_{\Omega_0} f(\xi,\eta)d\xi d\eta = W_1 f(\xi_1,\eta_1) + E. \tag{6.3.3}$$

Since there are three unknowns W_1, ξ_1, and η_1, we expect (6.3.3) to be exact for any linear polynomial. Integration is a linear operator; hence, it suffices to ensure that (6.3.3) is exact for the monomials 1, ξ, and η. Thus,

- If $f(\xi,\eta) = 1$:
$$\int_0^1 \int_0^{1-\xi} (1) d\eta d\xi = \frac{1}{2} = W_1.$$
- If $f(\xi,\eta) = \xi$:
$$\int_0^1 \int_0^{1-\xi} (\xi) d\eta d\xi = \frac{1}{6} = W_1 \xi_1.$$
- If $f(\xi,\eta) = \eta$:
$$\int_0^1 \int_0^{1-\xi} (\eta) d\eta d\xi = \frac{1}{6} = W_1 \eta_1.$$

The solution of this system is $W_1 = 1/2$ and $\xi_1 = \eta_1 = 1/3$; thus, the one-point quadrature rule is

$$\iint\limits_{\Omega_0} f(\xi,\eta) d\xi d\eta = \frac{1}{2} f(\frac{1}{3},\frac{1}{3}) + E. \tag{6.3.4}$$

As expected, the optimal evaluation point is the centroid of the triangle.

A bound on the error E may be obtained by expanding $f(\xi,\eta)$ in a Taylor's series about some convenient point $(\xi_0,\eta_0) \in \Omega_0$ to obtain

$$f(\xi,\eta) = p_1(\xi,\eta) + R_1(\xi,\eta) \tag{6.3.5a}$$

where

$$p_1(\xi,\eta) = f(\xi_0,\eta_0) + [(\xi-\xi_0)\frac{\partial}{\partial \xi} + (\eta-\eta_0)\frac{\partial}{\partial \eta}] f(\xi_0,\eta_0) \tag{6.3.5b}$$

and

$$R_1(\xi,\eta) = \frac{1}{2}[(\xi-\xi_0)\frac{\partial}{\partial \xi} + (\eta-\eta_0)\frac{\partial}{\partial \eta}]^2 f(\theta,\omega), \qquad (\theta,\omega) \in \Omega_0. \tag{6.3.5c}$$

Integrating (6.3.5a) using (6.3.4)

$$E = \iint\limits_{\Omega_0} [p_1(\xi,\eta) + R_1(\xi,\eta)] d\xi d\eta - \frac{1}{2}[p_1(\frac{1}{3},\frac{1}{3}) + R_1(\frac{1}{3},\frac{1}{3})].$$

Since (6.3.4) is exact for linear polynomials

$$E = \iint\limits_{\Omega_0} R_1(\xi,\eta) d\xi d\eta - \frac{1}{2} R_1(\frac{1}{3},\frac{1}{3}).$$

Not being too precise, we take an absolute value of the above expression to obtain

$$|E| \leq \iint\limits_{\Omega_0} |R_1(\xi,\eta)| d\xi d\eta + \frac{1}{2} |R_1(\frac{1}{3},\frac{1}{3})|.$$

For the canonical element, $|\xi - \xi_0| \leq 1$ and $|\eta - \eta_0| \leq 1$; hence,

$$|R_1(\xi, \eta)| \leq 2 \max_{|\boldsymbol{\kappa}|=2} ||D^{\boldsymbol{\kappa}} f||_{\infty,0}$$

where

$$||f||_{\infty,0} = \max_{(\xi,\eta)\in\Omega_0} |f(\xi, \eta)|.$$

Since the area of Ω_0 is $1/2$,

$$|E| \leq 2 \max_{|\boldsymbol{\kappa}|=2} ||D^{\boldsymbol{\kappa}} f||_{\infty,0}. \tag{6.3.6}$$

Errors for other quadrature formulas follow the same derivation ([6], Section 7.7).

Two-dimensional integrals on triangles are conveniently expressed in terms of triangular coordinates as

$$\iint_{\Omega_e} f(x, y)dxdy = A_e \sum_{i=1}^{n} W_i f(\zeta_1^i, \zeta_2^i, \zeta_3^i) + E \tag{6.3.7}$$

where $(\zeta_1^i, \zeta_2^i, \zeta_3^i)$ are the triangular coordinates of evaluation point i and A_e is the area of triangle e. Symmetric quadrature formulas for triangles have appeared in several places. Hammer *et al.* [5] developed formulas on triangles, tetrahedra, and cones. Dunavant [4] presents formulas on triangles which are exact to order 20; however, some formulas have evaluation points that are outside of the triangle. Sylvester [10] developed tensor-product formulas for triangles. We have listed some quadrature rules in Table 6.3.1 that also appear in Dunavant [4], Strang and Fix [9], and Zienkiewicz [12]. A multiplication factor M indicates the number of permutations associated with an evaluation point having a weight W_i. The factor $M = 1$ is associated with an evaluation point at the triangle's centroid $(1/3, 1/3, 1/3)$, $M = 3$ indicates a point on a median line, and $M = 6$ indicates an arbitrary point in the interior. The factor p indicates the order of the quadrature rule; thus, $E = O(h^{p+1})$ where h is the maximum edge length of the triangle.

Example 6.3.2. Using the data in Table 6.3.1 with (6.3.7), the three-point quadrature rule on the canonical triangle is

$$\iint_{\Omega_0} f(\xi, \eta)d\xi d\eta = \frac{1}{6}[f(2/3, 1/6, 1/6) + f(1/6, 1/6, 2/3) + f(1/6, 2/3, 1/6)] + E.$$

The multiplicative factor of $1/6$ arises because the area of the canonical element is $1/2$ and all of the weights are $1/3$. The quadrature rule can be written in terms of the canonical variables by setting $\zeta_2 = \xi$ and $\zeta_3 = \eta$ (*cf.* (4.2.6) and (4.2.7)). The discretization error associated with this quadrature rule is $O(h^3)$.

n	W_i	ζ_1^i	ζ_2^i, ζ_3^i	M	p
1	1.000000000000000	0.333333333333333	0.333333333333333		1
			0.333333333333333	1	
3	0.333333333333333	0.666666666666667	0.166666666666667		2
			0.166666666666667	3	
4	-0.562500000000000	0.333333333333333	0.333333333333333		3
			0.333333333333333	1	
	0.520833333333333	0.600000000000000	0.200000000000000		
			0.200000000000000	3	
6	0.109951743655322	0.816847572980459	0.091576213509771		4
			0.091576213509771	3	
	0.223381589678011	0.108103018168070	0.445948490915965		
			0.445948490915965	3	
7	0.225000000000000	0.333333333333333	0.333333333333333		5
			0.333333333333333	1	
	0.125939180544827	0.797426985353087	0.101286507323456		
			0.101286507323456	3	
	0.132394152788506	0.059715871789770	0.470142064105115		
			0.470142064105115	3	
12	0.050844906370207	0.873821971016996	0.063089014491502		6
			0.063089014491502	3	
	0.116786275726379	0.501426509658179	0.249286745170910		
			0.249286745170910	3	
	0.082851075618374	0.636502499121399	0.310352451033785		
			0.053145049844816	6	
13	-0.149570044467670	0.333333333333333	0.333333333333333		7
			0.333333333333333	1	
	0.175615257433204	0.479308067841923	0.260345966079038		
			0.260345966079038	3	
	0.053347235608839	0.869739794195568	0.065130102902216		
			0.065130102902216	3	
	0.077113760890257	0.638444188569809	0.312865496004875		
			0.486903154253160	6	

Table 6.3.1: Weights and evaluation points for integration on triangles [4].

Quadrature rules on tetrahedra have the form

$$\iiint_{\Omega_e} f(x,y,z)dxdydz = V_e \sum_{i=1}^{n} W_i f(\zeta_1^i, \zeta_2^i, \zeta_3^i, \zeta_4^i) + E \tag{6.3.8}$$

where V_e is the volume of Element e and $(\zeta_1^i, \zeta_2^i, \zeta_3^i, \zeta_4^i)$ are the tetrahedral coordinates of evaluation point i. Quadrature rules are presented by Jinyun [7] for methods to order six and by Keast [8] for methods to order eight. Multiplicative factors are such that $M = 1$ for an evaluation point at the centroid $(1/4, 1/4, 1/4, 1/4)$, $M = 4$ for points on the median line through the centroid and one vertex, $M = 6$ for points on a line between opposite midsides, $M = 12$ for points in the plane containing an edge an and opposite midside, and $M = 24$ for points in the interior (Figure 6.3.1).

n	W_i	ζ_1^i, ζ_2	ζ_3, ζ_4	M	p
1	1.000000000000000	0.250000000000000	0.250000000000000		1
		0.250000000000000	0.250000000000000	1	
4	0.250000000000000	0.585410196624969	0.138196601125011		2
		0.138196601125011	0.138196601125011	4	
5	-0.800000000000000	0.250000000000000	0.250000000000000		3
		0.250000000000000	0.250000000000000	1	
	0.450000000000000	0.500000000000000	0.166666666666667		
		0.166666666666667	0.166666666666667	4	
11	-0.013155555555556	0.250000000000000	0.250000000000000		4
		0.250000000000000	0.250000000000000	1	
	0.007622222222222	0.785714285714286	0.071428571428571		
		0.071428571428571	0.071428571428571	4	
	0.024888888888889	0.399403576166799	0.399403576166799		
		0.100596423833201	0.100596423833201	6	
15	0.030283678097089	0.250000000000000	0.250000000000000		5
		0.250000000000000	0.250000000000000	1	
	0.006026785714286	0.000000000000000	0.333333333333333		
		0.333333333333333	0.333333333333333	4	
	0.011645249086029	0.727272727272727	0.090909090909091		
		0.090909090909091	0.090909090909091	4	
	0.010949141561386	0.066550153573664	0.066550153573664		
		0.433449846426336	0.433449846426336	6	

Table 6.3.2: Weights and evaluation points for integration on tetrahedra [7, 8].

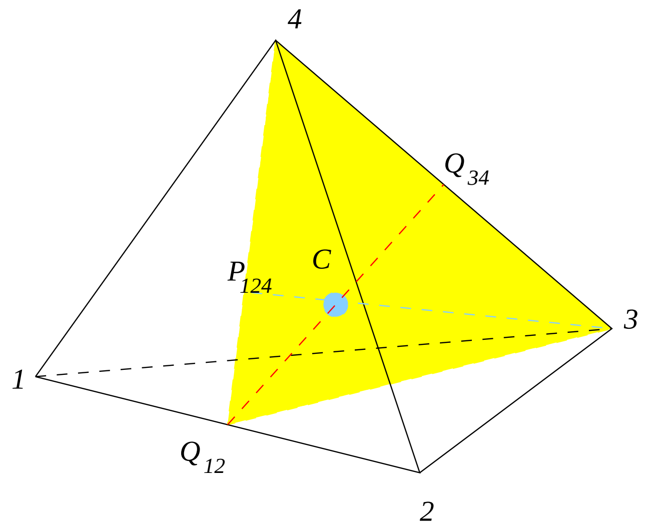


Figure 6.3.1: Some symmetries associated with the tetrahedral quadrature rules of Table 6.3.2. An evaluation point with $M = 1$ is at the centroid (C), one with $M = 4$ is on a line through a vertex and the centroid (*e.g.*, line $3 - P_{134}$), one with $M = 6$ is on a line between two midsides (*e.g.*, line $Q_{12} - Q_{34}$), and one with $M = 12$ is in a plane through two vertices and an opposite midside (*e.g.*, plane $3 - 4 - Q_{12}$)

Problems

1. Derive a three-point Gauss quadrature rule on the canonical right 45° triangle that is accurate to order two. In order to simplify the derivation, use symmetry arguments to conclude that the three points have the same weight and that they are symmetrically disposed on the medians of the triangle. Show that there are two possible formulas: the one given in Table 6.3.1 and another one. Find both formulas.

2. Show that the mapping

$$\xi = \frac{1+u}{2}, \qquad \eta = \frac{(1-u)(1+v)}{4}$$

transforms the integral (6.3.2) from the triangle Ω_0 to one on the square $-1 \leq u, v \leq 1$. Find the resulting integral and show how to approximate it using a tensor-product formula.

Bibliography

[1] M. Abromowitz and I.A. Stegun. *Handbook of Mathematical Functions*, volume 55 of *Applied Mathematics Series.* National Bureau of Standards, Gathersburg, 1964.

[2] S.C. Brenner and L.R. Scott. *The Mathematical Theory of Finite Element Methods.* Springer-Verlag, New York, 1994.

[3] R.L. Burden and J.D. Faires. *Numerical Analysis.* PWS-Kent, Boston, fifth edition, 1993.

[4] D.A. Dunavant. High degree efficient symmetrical Gaussian quadrature rules for the triangle. *International Journal of Numerical Methods in Engineering*, 21:1129–1148, 1985.

[5] P.C. Hammer, O.P. Marlowe, and A.H. Stroud. Numerical integration over simplexes and cones. *Mathematical Tables and other Aids to Computation*, 10:130–137, 1956.

[6] E. Isaacson and H.B. Keller. *Analysis of Numerical Methods.* John Wiley and Sons, New York, 1966.

[7] Y. Jinyun. Symmetric Gaussian quadrature formulae for tetrahedronal regions. *Computer Methods in Applied Mechanics and Engineering*, 43:349–353, 1984.

[8] P. Keast. Moderate-degree tetrahedral quadrature formulas. *Computer Methods in Applied Mechanics and Engineering*, 55:339–348, 1986.

[9] G. Strang and G. Fix. *Analysis of the Finite Element Method.* Prentice-Hall, Englewood Cliffs, 1973.

[10] P. Sylvester. Symmetric quadrature formulae for simplexes. *Mathematics of Computation*, 24:95–100, 1970.

[11] R. Wait and A.R. Mitchell. *The Finite Element Analysis and Applications.* John Wiley and Sons, Chichester, 1985.

[12] O.C. Zienkiewicz. *The Finite Element Method.* McGraw-Hill, New York, third edition, 1977.

Chapter 7

Analysis of the Finite Element Method

7.1 Introduction

Finite element theory is embedded in a very elegant framework that enables accurate *a priori* and a posteriori estimates of discretization errors and convergence rates. Unfortunately, a large portion of the theory relies on a knowledge of functional analysis, which has not been assumed in this material. Instead, we present the relevant concepts and key results without proof and cite sources of a more complete treatment. Once again, we focus on the model Galerkin problem: find $u \in H_0^1$ satisfying

$$A(v,u) = (v,f), \qquad \forall v \in H_0^1, \tag{7.1.1a}$$

where

$$(v,f) = \iint_{\Omega} vf\,dxdy, \tag{7.1.1b}$$

$$A(v,u) = \iint_{\Omega} [p(v_x u_x + v_y u_y) + qvu]dxdy, \tag{7.1.1c}$$

where the two-dimensional domain Ω has boundary $\partial\Omega = \partial\Omega_E \cup \partial\Omega_N$. For simplicity, we have assumed trivial essential and natural boundary data on $\partial\Omega_E$ and $\partial\Omega_N$, respectively.

Finite element solutions $U \in S_0^N$ of (7.1.1) satisfy

$$A(V,U) = (V,f), \qquad \forall V \in S_0^N, \tag{7.1.2}$$

where S_0^N is a finite-dimensional subspace of H_0^1.

As described in Chapter 2, error analysis typically proceeds in two steps:

1. showing that U is optimal in the sense that the error $u - U$ satisfies

$$\|u - U\| = \min_{W \in S_E^N} \|u - W\| \tag{7.1.3}$$

   in an appropriate norm, and

2. finding an upper bound for the right-hand side of (7.1.3).

The appropriate norm to use with (7.1.3) for the model problem (7.1.1) is the strain energy norm

$$\|v\|_A = \sqrt{A(v, v)}. \tag{7.1.4}$$

The finite element solution might not satisfy (7.1.3) with other norms and/or problems. For example, finite element solutions are not optimal in any norm for non-self-adjoint problems. In these cases, (7.1.3) is replaced by the weaker statement

$$\|u - U\| \leq C \min_{W \in S_0^N} \|u - W\|, \tag{7.1.5}$$

$C > 1$. Thus, the solution is "nearly best" in the sense that it only differs by a constant from the best possible solution in the space.

Upper bounds of the right-hand sides of (7.1.3) or (7.1.5) are obtained by considering the error of an interpolant W of u. Using Theorems 2.6.4 and 4.6.5, for example, we could conclude that

$$\|u - W\|_s \leq Ch^{p+1-s} \|u\|_{p+1}, \qquad s = 0, 1, \tag{7.1.6}$$

if S^N consists of complete piecewise polynomials of degree p with respect to a sequence of uniform meshes (*cf.* Definition 4.6.1) and $u \in H^{p+1}$. The bound (7.1.6) can be combined with either (7.1.3) or (7.1.5) to provide an estimate of the error and convergence rate of a finite element solution.

The Sobolev norm on H^1 and the strain energy norm (7.1.4) are equivalent for the model problem (7.1.1) and we shall use this with (7.1.3) and (7.1.6) to construct error estimates. Prior to continuing, you may want to review Sections 2.6, 3.2, and 4.6.

A priori finite element discretization errors, obtained as described, do not account for such "perturbations" as

1. using numerical integration,

2. interpolating Dirichlet boundary conditions by functions in S^N, and

3. approximating $\partial\Omega$ by piecewise-polynomial functions.

These effects will have to be appraised. Additionally, the *a priori* error estimates supply information on convergence rates but are difficult to use for quantitative error information. *A posteriori* error estimates, which use the computed solution, provide more practical accuracy appraisals.

7.2 Convergence and Optimality

While keeping the model problem (7.1.1) in mind, we will proceed in a slightly more general manner by considering a Galerkin problem of the form (7.1.1a) with a strain energy $A(v,u)$ that is a symmetric bilinear form (*cf.* Definitions 3.2.2, 3) and is also continuous and coercive.

Definition 7.2.1. A bilinear form $A(v,u)$ is *continuous* in H^s if there exists a constant $\alpha > 0$ such that

$$|A(v,u)| \leq \alpha \|u\|_s \|v\|_s, \qquad \forall u,v \in H^s. \tag{7.2.1}$$

Definition 7.2.2. A bilinear form $A(u,v)$ is *coercive* (H^s *– elliptic* or *positive definite*) in H^s if there exists a constant $\beta > 0$ such that

$$A(u,u) \geq \beta \|u\|_s^2, \qquad \forall u \in H^s. \tag{7.2.2}$$

Continuity and coercivity of $A(v,u)$ can be used to establish the existence and uniqueness of solutions to the Galerkin problem (7.1.1a). These results follow from the Lax-Milgram Theorem. We'll subsequently prove a portion of this result, but more complete treatments appear elsewhere [6, 12, 13, 15]. We'll use examples to provide insight into the meanings of continuity and coercivity.

Example 7.2.1. Consider the variational eigenvalue problem: determine nontrivial $u \in H_0^1$ and $\lambda \in [0,\infty)$ satisfying

$$A(u,v) = \lambda(u,v), \qquad \forall v \in H_0^1.$$

When $A(v,u)$ is the strain energy for the model problem (7.1.1), smooth solutions of this variational problem also satisfy the differential eigenvalue problem

$$-(pu_x)_x - (pu_y)_y + qu = \lambda u, \qquad (x,y) \in \Omega,$$

$$u = 0, \qquad (x,y) \in \partial\Omega_E, \qquad u_{\mathbf{n}} = 0, \qquad (x,y) \in \partial\Omega_N.$$

where $\mathbf{n}$ is the unit outward normal to $\partial\Omega$.

Letting λ_r and u^r, $r \geq 1$, be an eigenvalue-eigenfunction pair and using the variational statement with $v = u = u^r$, we obtain the *Rayleigh quotient*

$$\lambda_r = \frac{A(u^r, u^r)}{(u^r, u^r)}, \qquad r \geq 1.$$

Since this result holds for all r, we have

$$\lambda_1 = \min_{r \geq 1} \frac{A(u^r, u^r)}{(u^r, u^r)}$$

where λ_1 is the minimum eigenvalue. (As indicated in Problem 1, this result can be extended.)

Using the Rayleigh quotient with (7.2.2), we have

$$\lambda_r \geq \frac{\beta \|u^r\|_s^2}{\|u^r\|_0^2}, \qquad r \geq 1.$$

Since $\|u^r\|_s \geq \|u^r\|_0$, we have

$$\lambda_r \geq \beta > 0, \qquad r \geq 1.$$

Thus, $\beta \leq \lambda_r$, $r \geq 1$, and, in particular, $\beta \leq \lambda_1$.

Using (7.2.1) in conjunction with the Rayleigh quotient implies

$$\lambda_r \leq \frac{\alpha \|u^r\|_s^2}{\|u^r\|_0^2}, \qquad r \geq 1.$$

Combining the two results,

$$\beta \frac{\|u^r\|_s^2}{\|u^r\|_0^2} \leq \lambda_r \leq \alpha \frac{\|u^r\|_s^2}{\|u^r\|_0^2}, \qquad r \geq 1.$$

Thus, β provides a lower bound for the minimum eigenvalue and α provides a bound for the maximum growth rate of the eigenvalues in H^s.

Example 7.2.2. Solutions of the Dirichlet problem

$$-u_{xx} - u_{yy} = f(x, y), \qquad (x, y) \in \Omega, \qquad u = 0, \qquad (x, y) \in \partial\Omega,$$

satisfy the Galerkin problem (7.1.1) with

$$A(v, u) = \iint_{\Omega} \nabla v \cdot \nabla u dx dy, \qquad \nabla u = [u_x, u_y]^T.$$

An application of Cauchy's inequality reveals

$$|A(v, u)| = |\iint_{\Omega} \nabla v \cdot \nabla u dx dy| \leq \|\nabla v\|_0 \|\nabla u\|_0.$$

where

$$\|\nabla u\|_0^2 = \iint_{\Omega} (u_x^2 + u_y^2)dxdy.$$

Since $\|\nabla u\|_0 \leq \|u\|_1$, we have

$$|A(v,u)| \leq \|v\|_1 \|u\|_1.$$

Thus, (7.2.1) is satisfied with $s = 1$ and $\alpha = 1$, and the strain energy is continuous in H^1.

Establishing that $A(v,u)$ is coercive in H^1 is typically done by using Friedrichs's first inequality which states that there is a constant $\gamma > 0$ such that

$$\|\nabla u\|_0^2 \geq \gamma \|u\|_0^2. \tag{7.2.3}$$

Now, consider the identity

$$A(u,u) = \|\nabla u\|_0^2 = (1/2)\|\nabla u\|_0^2 + (1/2)\|\nabla u\|_0^2$$

and use (7.2.3) to obtain

$$A(u,u) \geq (1/2)\|\nabla u\|_0^2 + (1/2)\gamma\|u\|_0^2 \geq \beta\|u\|_1^2$$

where $\beta = (1/2)\max(1,\gamma)$. Thus, (7.2.2) is satisfied with $s = 1$ and $A(u,v)$ is coercive (H^1-elliptic).

Continuity and coercivity of the strain energy reveal the finite element solution U to be nearly the best approximation in S^N

Theorem 7.2.1. *Let $A(v,u)$ be symmetric, continuous, and coercive. Let $u \in H_0^1$ satisfy (7.1.1a) and $U \in S_0^N \subset H_0^1$ satisfy (7.1.2). Then*

$$\|u - U\|_1 \leq \frac{\alpha}{\beta}\|u - V\|_1, \qquad \forall V \in S_0^N, \tag{7.2.4a}$$

with α and β satisfying (7.2.1) and (7.2.2).

Remark 1. Equation (7.2.4a) may also be expressed as

$$\|u - U\|_1 \leq C \inf_{V \in S_0^N} \|u - V\|_1. \tag{7.2.4b}$$

Thus, continuity and H^1-ellipticity give us a bound of the form (7.1.5).

Proof. cf. Problem 2 at the end of this section. □

The bound (7.2.4) can be improved when $A(v,u)$ has the form (7.1.1c).

Theorem 7.2.2. *let $A(v, u)$ be a symmetric, continuous, and coercive bilinear form; $u \in H_0^1$ minimize*

$$I[w] = A(w, w) - 2(w, f), \qquad \forall w \in H_0^1; \tag{7.2.5}$$

and S_0^N be a finite-dimensional subspace of H_0^1. Then

1. *The minimum of $I[W]$ and $A(u - W, u - W)$, $\forall W \in S_0^N$, are achieved by the same function U.*

2. *The function U is the orthogonal projection of u onto S_0^N with respect to strain energy, i.e.,*

$$A(V, u - U) = 0, \qquad \forall V \in S_0^N. \tag{7.2.6}$$

3. *The minimizing function $U \in S_0^N$ satisfies the Galerkin problem*

$$A(V, U) = (V, f), \qquad \forall V \in S_0^N. \tag{7.2.7}$$

In particular, if S_0^N is the whole of H_0^1

$$A(v, u) = (v, f), \qquad \forall v \in H_0^1. \tag{7.2.8}$$

Proof. Our proof will omit several technical details, which appear in, *e.g.*, Wait and Mitchell [21], Chapter 6.

Let us begin with (7.2.7). If U minimizes $I[W]$ over S_0^N then for any ϵ and any $V \in S_0^N$

$$I[U] \leq I[U + \epsilon V].$$

Using (7.2.5),

$$I[U] \leq A(U + \epsilon V, U + \epsilon V) - 2(U + \epsilon V, f)$$

or

$$I[U] \leq I[U] + 2\epsilon[A(V, U) - (V, f)] + \epsilon^2 A(V, V)$$

or

$$0 \leq 2\epsilon[A(V, U) - (V, f)] + \epsilon^2 A(V, V).$$

This inequality must hold for all possible ϵ of either sign; thus, (7.2.7) must be satisfied. Equation (7.2.8) follows by repeating these arguments with S_0^N replaced by H_0^1.

Next, replace v in (7.2.8) by $V \in S_0^N \subset H_0^1$ and subtract (7.2.7) to obtain (7.2.6).

In order to prove Conclusion 1, consider the identity

$$A(u - U - V, u - U - V) = A(u - U, u - U) - 2A(u - U, V) + A(V, V).$$

Using (7.2.6)

$$A(u-U, u-U) = A(u-U-V, u-U-V) - A(V,V).$$

Since $A(V,V) \geq 0$,

$$A(u-U, u-U) \leq A(u-U-V, u-U-V), \qquad \forall V \in S_0^N.$$

Equality only occurs when $V = 0$; therefore, U is the unique minimizing function. □

Remark 2. We proved a similar result for one-dimensional problems in Theorems 2.6.1, 2.

Remark 3. Continuity and coercivity did not appear in the proof; however, they are needed to establish existence, uniqueness, and completeness. Thus, we never proved that $\lim_{N\to\infty} U = u$. A complete analysis appears in Wait and Mitchell [21], Chapter 6.

Remark 4. The strain energy $A(v,u)$ not need be symmetric. A proof without this restriction appears in Ciarlet [13].

Corollary 7.2.1. *With the assumptions of Theorem 7.2.2,*

$$A(u-U, u-U) = A(u,u) - A(U,U). \tag{7.2.9}$$

Proof. cf. Problem 3 at the end of this section. □

In Section 4.6, we obtained *a priori* estimates of interpolation errors under some mesh uniformity assumptions. Recall (*cf.* Definition 4.6.1), that we considered a family of finite element meshes Δ_h which became finer as $h \to 0$. The uniformity condition implied that all vertex angles were bounded away from 0 and π and that all aspect ratios were bounded away from 0 as $h \to 0$. Uniformity ensured that transformations from the physical to the computational space were well behaved. Thus, with uniform meshes, we were able to show (*cf.* Theorem 4.6.5) that the error in interpolating a function $u \in H^{p+1}$ by a complete polynomial W of degree p satisfies

$$\|u-W\|_s \leq Ch^{p+1-s}\|u\|_{p+1}, \qquad s = 0,1. \tag{7.2.10a}$$

The norm on the right can be replaced by the seminorm

$$|u|_{p+1}^2 = \sum_{|\boldsymbol{\kappa}|=p+1} \|D^{\boldsymbol{\kappa}} u\|_0^2 \tag{7.2.10b}$$

to produce a more precise estimate, but this will not be necessary for our present application. If singularities are present so that $u \in H^{q+1}$ with $q < p$ then, instead of (7.2.10a), we find

$$\|u-W\|_1 \leq Ch^q\|u\|_{q+1}. \tag{7.2.10c}$$

With optimality (or near optimality) established and interpolation error estimates available, we can establish convergence of the finite element method.

Theorem 7.2.3. *Suppose:*

1. *$u \in H_0^1$ and $U \in S_0^N \subset H_0^1$ satisfy (7.2.8) and (7.2.7), respectively*

2. *$A(v, u)$ is a symmetric, continuous, and H^1-elliptic bilinear form;*

3. *S_0^N consists of complete piecewise-polynomial functions of degree p with respect to a uniform family of meshes Δ_h; and*

4. *$u \in H_0^1 \cap H^{p+1}$.*

Then

$$\|u - U\|_1 \leq Ch^p \|u\|_{p+1} \tag{7.2.11a}$$

and

$$A(u - U, u - U) \leq Ch^{2p} \|u\|_{p+1}^2. \tag{7.2.11b}$$

Proof. From Theorem 7.2.2

$$A(u - U, u - U) = \inf_{V \in S_0^N} A(u - V, u - V) \leq A(u - W, u - W)$$

where W is an interpolant of u. Using (7.2.1) with $s = 1$ and v and u replaced by $u - W$ yields

$$A(u - W, u - W) \leq \alpha \|u - W\|_1^2.$$

Using the interpolation estimate (7.2.10a) with $s = 1$ yields (7.2.11b). In order to prove (7.2.11a), use (7.2.2) with $s = 1$ to obtain

$$\beta \|u - U\|_1^2 \leq A(u - U, u - U).$$

The use of (7.2.11b) and a division by β yields (7.2.11a). □

Since the H^1 norm dominates the L^2 norm, (7.2.11a) trivially gives us an error estimate in L^2 as

$$\|u - U\|_0 \leq Ch^p \|u\|_{p+1}.$$

This estimate does not have an optimal rate since the interpolation error (7.2.10a) is converging as $O(h^{p+1})$. Getting the correct rate for an L^2 error estimate is more complicated than it is in H^1. The proof is divided into two parts.

Lemma 7.2.1. *(Aubin-Nitsche) Under the assumptions of Theorem 7.2.3, let $\gamma(x,y) \in H_0^1$ be the solution of the "dual problem"*

$$A(v,\gamma) = (v,e), \qquad \forall v \in H_0^1, \tag{7.2.12a}$$

where

$$e = \frac{u-U}{\|u-U\|_0}. \tag{7.2.12b}$$

Let $\Gamma \in S_0^N$ be an interpolant of γ, then

$$\|u-U\|_0 \leq \alpha \|u-U\|_1 \|\gamma - \Gamma\|_1. \tag{7.2.12c}$$

Proof. Set $V = \Gamma$ in (7.2.6) to obtain

$$A(\Gamma, u-U) = 0. \tag{7.2.13}$$

Take the L^2 inner product of (7.2.12b) with $u-U$ to obtain

$$\|u-U\|_0 = (e, u-U).$$

Setting $v = u-U$ in (7.2.12a) and using the above relation yields

$$\|u-U\|_0 = A(u-U, \gamma).$$

Using (7.2.13)

$$\|u-U\|_0 = A(u-U, \gamma - \Gamma).$$

Now use the continuity of $A(v,u)$ in H^1 ((7.2.1) with $s=1$) to obtain (7.2.12c). □

Since we have an estimate for $\|u-U\|_1$, estimating $\|u-U\|_0$ by (7.2.12c) requires an estimate of $\|\gamma-\Gamma\|_1$. This, of course, will be done by interpolation; however, use of (7.2.10a) requires knowledge of the smoothness of γ. The following lemma provides the necessary *a priori* bound.

Lemma 7.2.2. *Let $A(u,v)$ be a symmetric, H^1-elliptic bilinear form and u be the solution of (7.2.8) on a smooth region Ω. Then*

$$\|u\|_2 \leq C\|f\|_0. \tag{7.2.14}$$

Remark 5. This result seems plausible since the underlying differential equation is of second order, so the second derivatives should have the same smoothness as the right-hand side f. The estimate might involve boundary data; however, we have assumed trivial conditions. Let's further assume that $\partial\Omega_E$ is not nil to avoid non-uniqueness issues.

Proof. Strang and Fix [18], Chapter 1, establish (7.2.14) in one dimension. Johnson [14], Chapter 4, obtain a similar result. □

With preliminaries complete, here is the main result.

Theorem 7.2.4. *Given the assumptions of Theorem 7.2.3, then*

$$\|u - U\|_0 \leq Ch^{p+1}\|u\|_{p+1}. \tag{7.2.15}$$

Proof. Applying (7.2.14) to the dual problem (7.2.12a) yields

$$\|\gamma\|_2 \leq C\|e\|_0 = C,$$

since $\|e\|_0 = 1$ according to (7.2.12b). With $\gamma \in H_2$, we may use (7.2.10c) with $q = s = 1$ to obtain

$$\|\gamma - \Gamma\|_1 \leq Ch\|\gamma\|_2 = Ch.$$

Combining this estimate with (7.2.11a) and (7.2.12c) yields (7.2.15). □

Problems

1. Show that the function u that minimizes

$$\lambda = \min_{w \in H_0^1,\ \|w\|_0 \neq 0} \frac{A(w,w)}{(w,w)}$$

is u^1, the eigenfunction corresponding to the minimum eigenvalue λ_1 of $A(v,u) = \lambda(v,u)$.

2. Assume that $A(v,u)$ is a symmetric, continuous, and H^1-elliptic bilinear form and, for simplicity, that $u, v \in H_0^1$.

   2.1. Show that the strain energy and H^1 norms are equivalent in the sense that

$$\beta\|u\|_1^2 \leq A(u,u) \leq \alpha\|u\|_1^2, \qquad \forall u \in H_0^1.$$

   where α and β satisfy (7.2.1) and (7.2.2).

   2.2. Prove Theorem 7.2.1.

3. Prove Corollary 7.2.1 to Theorem 7.2.2.

7.3 Perturbations

In this section, we examine the effects of perturbations due to numerical integration, interpolated boundary conditions, and curved boundaries.

7.3.1 Quadrature Perturbations

With numerical integration, we determine U^* as the solution of

$$A_*(V, U^*) = (V, f)_*, \qquad \forall V \in S_0^N, \tag{7.3.1a}$$

instead of determining U by solving (7.2.8). The approximate strain energy $A_*(V, U)$ or L^2 inner product $(V, f)_*$ reflect the numerical integration that has been used. For example, consider the loading

$$(V, f) = \sum_{e=1}^{N_\Delta} (V, f)_e, \qquad (V, f)_e = \iint_{\Omega_e} V(x, y) f(x, y) dx dy$$

where Ω_e is the domain occupied by element e in a mesh of N_Δ elements. Using an n-point quadrature rule (*cf.* (6.1.2a)) on element e, we would approximate (V, f) by

$$(V, f)_* = \sum_{e=1}^{N_\Delta} (V, f)_{e,*} \tag{7.3.1b}$$

where

$$(V, f)_{e,*} = \sum_{k=1}^{n} W_k V(x_k, y_k) f(x_k, y_k). \tag{7.3.1c}$$

The effects of transformations to a canonical element have not been shown for simplicity and a similar formula applies for $A_*(V, U)$.

Deriving an estimate for the perturbation introduced by (7.3.1a) is relatively simple if $A(V, U)$ and $A_*(V, U)$ are continuous and coercive.

Theorem 7.3.1. *Suppose that $A(v, u)$ and $A_*(V, U)$ are bilinear forms with A being continuous and A_* being coercive in H^1; thus, there exists constants α and β such that*

$$|A(u, v)| \leq \alpha \|u\|_1 \|v\|_1, \qquad \forall u, v \in H_0^1, \tag{7.3.2a}$$

and

$$A_*(U, U) \geq \beta \|U\|_1^2, \qquad \forall U \in S_0^N. \tag{7.3.2b}$$

Then

$$\begin{aligned}\|u - U^*\|_1 \quad &\leq \quad C\{\|u - V\|_1 + \sup_{W \in S_0^N} \frac{|A(V, W) - A_*(V, W)|}{\|W\|_1} + \\ &\sup_{W \in S_0^N} \frac{|(W, f) - (W, f)_*|}{\|W\|_1}\}, \qquad \forall V \in S_0^N.\end{aligned} \tag{7.3.3}$$

Proof. Using the triangular inequality

$$\|u - U^*\|_1 = \|u - V + V - U^*\|_1 \leq \|u - V\|_1 + \|W\|_1 \tag{7.3.4a}$$

where

$$W = U^* - V. \tag{7.3.4b}$$

Using (7.3.2b) and (7.3.4b)

$$\beta\|W\|_1^2 \leq A_*(U^* - V, W) = A_*(U^*, W) - A_*(V, W).$$

Using (7.3.1a) with V replaced by W to eliminate $A_*(U^*, W)$, we get

$$\beta\|W\|_1^2 \leq (f, W)_* - A_*(V, W).$$

Adding the exact Galerkin equation (7.2.8) with v replaced by W

$$\beta\|W\|_1^2 \leq (f, W)_* - (f, W) + A(u, W) - A_*(V, W).$$

Adding and subtracting $A(V, W)$ and taking an absolute value

$$\beta\|W\|_1^2 \leq |(f, W)_* - (f, W)| + |A(u - V, W)| + |A(V, W) - A_*(V, W)|.$$

Now, using the continuity condition (7.3.2a) with u replaced by $u - V$ and v replaced by W, we obtain

$$\beta\|W\|_1^2 \leq |(f, W)_* - (f, W)| + \alpha\|u - V\|_1\|W\|_1 + |A(V, W) - A_*(V, W)|.$$

Dividing by $\beta\|W\|_1$

$$\|W\|_1 \leq \frac{1}{\beta}\{\alpha\|u - V\|_1 + \frac{|(f, W)_* - (f, W)|}{\|W\|_1} + \frac{|A(V, W) - A_*(V, W)|}{\|W\|_1}\}.$$

Combining the above inequality with (7.3.4a), maximizing the inner product ratios over W, and choosing C as the larger of $1 + \alpha/\beta$ or $1/\beta$ yields (7.3.3). $\square$

Remark 1. Since the error estimate (7.3.3) is valid for all $V \in S_0^N$ it can be written in the form

$$\begin{aligned}\|u - U^*\|_1 \quad \leq \quad & C \inf_{V \in S_0^N}\{\|u - V\|_1 + \sup_{W \in S_0^N} \frac{|A(V, W) - A_*(V, W)|}{\|W\|_1} + \\ & \sup_{W \in S_0^N} \frac{|(W, f) - (W, f)_*|}{\|W\|_1}\}.\end{aligned} \tag{7.3.5}$$

To bound (7.3.3) or (7.3.5) in terms of a mesh parameter h, we use standard interpolation error estimates (*cf.* Sections 2.6 and 4.6) for the first term and numerical integration error estimates (*cf.* Chapter 6) for the latter two terms. Estimating quadrature errors is relatively easy and the following typical result includes the effects of transforming to a canonical element.

Theorem 7.3.2. *Let $\mathbf{J}(\xi,\eta)$ be the Jacobian of a transformation from a computational (ξ,η)-plane to a physical (x,y)-plane and let $W \in S_0^N$. Relative to a uniform family of meshes Δ_h, suppose that $\det(\mathbf{J}(\xi,\eta))W_x(\xi,\eta)$ and $\det(\mathbf{J}(\xi,\eta))W_y(\xi,\eta)$ are piecewise polynomials of degree at most r_1 and $\det(\mathbf{J}(\xi,\eta))W(\xi,\eta)$ is a piecewise polynomial of degree at most r_0. Then:*

1. *If a quadrature rule is exact (in the computational plane) for all polynomials of degree at most $r_1 + r$,*

$$\frac{|A(V,W) - A_*(V,W)|}{\|W\|_1} \leq Ch^{r+1}\|V\|_{r+2}, \qquad \forall V, W \in S_0^N, \tag{7.3.6a}$$

2. *If a quadrature rule is exact for all polynomials of degree at most $r_0 + r - 1$,*

$$\frac{|(f,W) - (f,W)_*|}{\|W\|_1} \leq Ch^{r+1}\|f\|_{r+1}, \qquad \forall W \in S_0^N. \tag{7.3.6b}$$

Proof. cf. Wait and Mitchell [21], Chapter 6, or Strang and Fix [18], Chapter 4. □

Example 7.3.1. Suppose that the coordinate transformation is linear so that $\det(\mathbf{J}(\xi,\eta))$ is constant and that S_0^N consists of piecewise polynomials of degree at most p. In this case, $r_1 = p-1$ and $r_0 = p$. The interpolation error in H^1 is

$$\|u - V\|_1 = O(h^p).$$

Suppose that the quadrature rule is exact for polynomials of degree ρ or less. Thus, $\rho = r_1 + r$ or $r = \rho - p + 1$ and (7.3.6a) implies that

$$\frac{|A(V,W) - A_*(V,W)|}{\|W\|_1} \leq Ch^{\rho-p+2}\|V\|_{\rho-p+3}, \qquad \forall V, W \in S_0^N.$$

With $\rho = r_0 + r - 1$ and $r_0 = p$, we again find $r = \rho - p + 1$ and, using (7.3.6b),

$$\frac{|(f,W) - (f,W)_*|}{\|W\|_1} \leq Ch^{\rho-p+2}\|f\|_{\rho-p+2}, \qquad \forall W \in S_0^N.$$

- If $\rho = 2(p-1)$ so that $r = p-1$ then the above perturbation errors are $O(h^p)$. Hence, all terms in (7.3.3) or (7.3.5) have the same order of accuracy and we conclude that

$$\|u - U^*\|_1 = O(h^p).$$

  This situation is regarded as optimal. If the coefficients of the differential equation are constant and, as is the case here, the Jacobian is constant, this result is equivalent to integrating the differentiated terms in the strain energy exactly (*cf.*, *e.g.*, (7.1.1c)).

- If $\rho > 2(p-1)$ so that $r > p-1$ then the error in integration is higher order than the $O(h^p)$ interpolation error; however, the interpolation error dominates and
$$\|u - U^*\|_1 = O(h^p).$$
The extra effort in performing the numerical integration more accurately is not justified.

- If $\rho < 2(p-1)$ so that $r < p-1$ then the integration error dominates the interpolation error and determines the order of accuracy as
$$\|u - U^*\|_1 = O(h^{\rho-p+2}).$$
In particular, convergence does not occur if $\rho \leq p-2$.

Let us conclude this example by examining convergence rates for piecewise-linear (or bilinear) approximations ($p = 1$). In this case, $r_1 = 0$, $r_0 = 1$, and $r = \rho$. Interpolation errors converge as $O(h)$. The optimal order of accuracy of the quadrature rule is $\rho = 0$, *i.e.*, only constant functions need be integrated exactly. Performing the integration more accurately yields no improvement in the convergence rate.

Example 7.3.2. Problems with variable Jacobians are more complicated. Consider the term
$$\det(\mathbf{J}(\xi,\eta))W_x(\xi,\eta) = J(W_\xi \xi_x + W_\eta \eta_x)$$
where $J = \det(\mathbf{J}(\xi,\eta))$. The metrics ξ_x and η_x are obtained from the inverse Jacobian
$$\mathbf{J}^{-1} = \begin{bmatrix} \xi_x & \xi_y \\ \eta_x & \eta_y \end{bmatrix} = \frac{1}{J}\begin{bmatrix} y_\eta & -x_\eta \\ -y_\xi & x_\xi \end{bmatrix}.$$
In particular, $\xi_x = y_\eta / J$ and $\eta_x = -y_\xi / J$ and
$$\det(\mathbf{J})W_x = W_\xi y_\eta - W_\eta y_\xi.$$

Consider an isoparametric transformation of degree p. Such triangles or quadrilaterals in the computational plane have curved sides of piecewise polynomials of degree p in the physical plane. If W is a polynomial of degree p then W_x has degree $p-1$. Likewise, x and y are polynomials of degree p in ξ and η. Thus, y_η and y_ξ also have degrees $p-1$. Therefore, JW_x and, similarly, JW_y have degrees $r_1 = 2(p-1)$. With J being a polynomial of degree $2(p-1)$, we find JW to be of degree $r_0 = 3p-2$.

For the quadrature errors (7.3.6) to have the same $O(h^p)$ rate as the interpolation error, we must have $r = p-1$ in (7.3.6a,b). Thus, according to Theorem 7.3.2, the order ρ of the quadrature rules in the (ξ,η)-plane should be
$$\rho = r_1 + r = 2(p-1) + (p-1) = 3(p-1)$$

for (7.3.6a) and

$$\rho = r_0 + r - 1 = (3p-2) + (p-1) - 1 = 4(p-1)$$

for (7.3.6b). These results are to be compared with the order of $2(p-1)$ that was needed with the piecewise polynomials of degree p and linear transformations considered in Example 7.3.1. For quadratic transformations and approximations ($p = 2$), we need third- and fourth-order quadrature rules for $O(h^2)$ accuracy.

7.3.2 Interpolated Boundary Conditions

Assume that integration is exact and the boundary $\partial\Omega$ is modeled exactly, but Dirichlet boundary data is approximated by a piecewise polynomial in S^N, *i.e.*, by a polynomial having the same degree p as the trial and test functions. Under these conditions, Wait and Mitchell [21], Chapter 6, show that the error in the solution U of a Galerkin problem with interpolated boundary conditions satisfies

$$\|u - U\|_1 \leq C\{h^p\|u\|_{p+1} + h^{p+1/2}\|u\|_{p+1}\}. \tag{7.3.7}$$

The first term on the right is the standard interpolation error estimate. The second term corresponds to the perturbation due to approximating the boundary condition. As usual, computation is done on a uniform family of meshes Δ_h and u is smooth enough to be in H^{p+1}. Brenner and Scott [12], Chapter 8, obtain similar results under similar conditions when interpolation is performed at the Lobatto points on the boundary of an element. The Lobatto polynomial of degree p is defined on $[-1, 1]$ as

$$L_p(\xi) = \frac{d^{p-2}}{d\xi^{p-2}}(1-\xi^2)^{p-1}, \qquad \xi \in [-1,1], \qquad p \geq 2.$$

These results are encouraging since the perturbation in the boundary data is of slightly higher order than the interpolation error. Unfortunately, if the domain Ω is not smooth and, *e.g.*, contains corners solutions will not be elements of H^{p+1}. Less is known in these cases.

7.3.3 Perturbed Boundaries

Suppose that the domain Ω is replaced by a polygonal domain $\tilde{\Omega}$ as shown in Figure 7.3.1. Strang and Fix [18], analyze second-order problems with homogeneous Dirichlet data of the form: determine $u \in H_0^1$ satisfying

$$A(v, u) = (v, f), \qquad \forall v \in H_0^1, \tag{7.3.8a}$$

where functions in H_0^1 satisfy $u(x,y) = 0$, $(x,y) \in \partial\Omega$. The finite element solution $U \in \tilde{S}_0^N$ satisfies

$$A(V,U) = (V,f), \qquad \forall V \in \tilde{S}_0^N, \tag{7.3.8b}$$

where functions in $\tilde{S}_0^N$ vanish on $\partial\tilde{\Omega}$. (Thus, $\tilde{S}_0^N$ is not a subspace of H_0^1.)

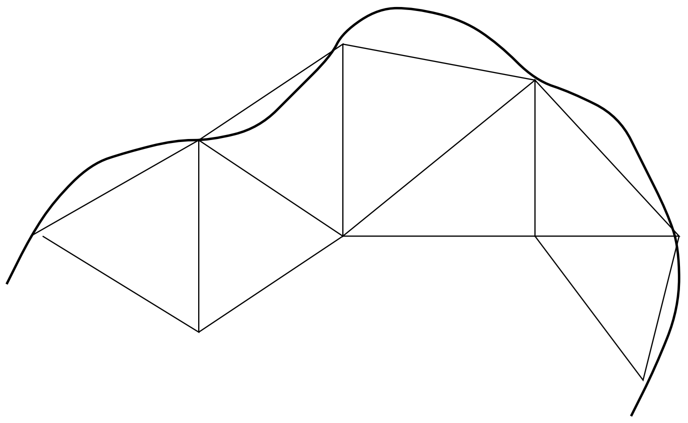


Figure 7.3.1: Approximation of a curved boundary by a polygon.

For piecewise linear polynomial approximations on triangles they show that $\|u - U\|_1 = O(h)$ and for piecewise quadratic approximations $\|u - U\|_1 = O(h^3/2)$. The poor accuracy with quadratic polynomials is due to large errors in a narrow "boundary layer" near $\partial\Omega$. Large errors are confined to the boundary layer and results are acceptable elsewhere. Wait and Mitchell [21], Chapter 6, quote other results which prove that $\|u - U\|_1 = O(h^p)$ for *pth* degree piecewise polynomial approximations when the distance between $\partial\Omega$ and $\partial\tilde{\Omega}$ is $O(h^{p+1})$. Such is the case when $\partial\Omega$ is approximated by *p th* degree piecewise-polynomial interpolation.

7.4 A Posteriori Error Estimation

In previous sections of this chapter, we considered *a priori error estimates*. Thus, we can, without computation, infer that finite element solutions converge at a certain rate depending on the exact solution's smoothness. Error bounds are expressed in terms of unknown constants which are difficult, if not impossible, to estimate. Having computed a finite element solution, it is possible to obtain *a posteriori error estimates* which give more quantitative information about the accuracy of the solution. Many error estimation techniques are available and before discussing any, let's list some properties that a good *a posteriori* error estimation procedure should possess.

- The error estimate should give an accurate measure of the discretization error for a wide range of mesh spacings and polynomial degrees.

- The procedure should be inexpensive relative to the cost of obtaining the finite element solution. This usually means that error estimates should be calculated using only local computations, which typically require an effort comparable to the cost of generating the stiffness matrix.

- A technique that provides estimates of pointwise errors which can subsequently be used to calculate error measures in several norms is preferable to one that only works in a specific norm. Pointwise error estimates and error estimates in local (elemental) norms may also provide an indications as to where solution accuracy is insufficient and where refinement is needed.

A posteriori error estimates can roughly be divided into four categories.

1. *Residual error estimates.* Local finite element problems are created on either an element or a subdomain and solved for the error estimate. The data depends on the residual of the finite element solution.

2. *Flux-projection error estimates.* A new flux is calculated by post processing the finite element solution. This flux is smoother than the original finite element flux and an error estimate is obtained from the difference of the two fluxes.

3. *Extrapolation error estimates.* Two finite element solutions having different orders or different meshes are compared and their differences used to provide an error estimate.

4. *Interpolation error estimates.* Interpolation error bounds are used with estimates of the unknown constants.

The four techniques are not independent but have many similarities. Surveys of error estimation procedures [7, 20] describe many of their properties, similarities, and differences. Let us set the stage by briefly describing two simple extrapolation techniques. Consider a one-dimensional problem for simplicity and suppose that an approximate solution $U_h^p(x)$ has been computed using a polynomial approximation of degree p on a mesh of spacing h (Figure 7.4.1). Suppose that we have an *a priori* interpolation error estimate of the form

$$u(x) - U_h^p(x) = C_{p+1}h^{p+1} + O(h^{p+2}).$$

We have assumed that the exact solution $u(x)$ is smooth enough for the error to be expanded in h to $O(h^{p+2})$. The leading error constant C_{p+1} generally depends on (unknown) derivatives of u. Now, compute a second solution with spacing $h/2$ (Figure 7.4.1) to obtain

$$u(x) - U_{h/2}^p(x) = C_{p+1}(\frac{h}{2})^{p+1} + O(h^{p+2}).$$

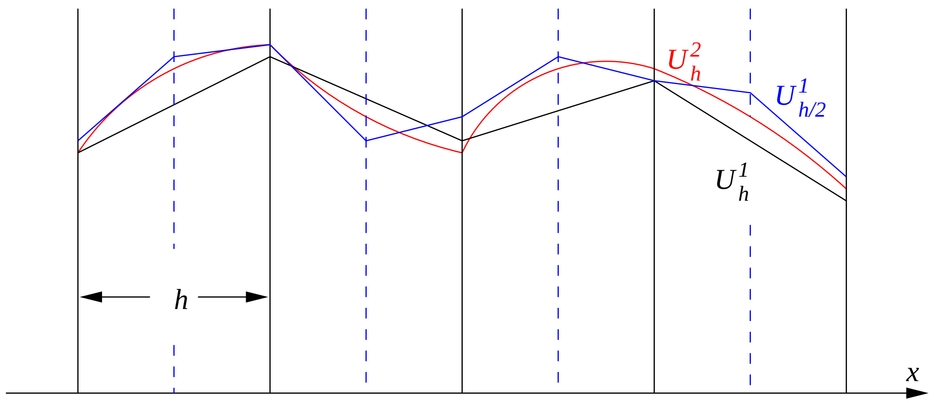


Figure 7.4.1: Solutions U_h^1 and $U_{h/2}^1$ computed on meshes having spacing h and $h/2$ with piecewise linear polynomials ($p = 1$) and a third solution U_h^2 computed on a mesh of spacing h with a piecewise quadratic polynomial ($p = 2$).

Subtracting the two solutions we eliminate the unknown exact solution and obtain

$$U_{h/2}^p(x) - U_h^p(x) = C_{p+1}h^{p+1}(1 - \frac{1}{2^p+1}) + O(h^{p+2}).$$

Neglecting the higher-order terms, we obtain an approximation of the discretization error as

$$C_{p+1}h^{p+1} \approx \frac{U_{h/2}^p(x) - U_h^p(x)}{1 - 1/2^{p+1}}.$$

Thus, we have an estimate of the discretization error of the coarse-mesh solution as

$$u(x) - U_h^p(x) \approx \frac{U_{h/2}^p(x) - U_h^p(x)}{1 - 1/2^{p+1}}.$$

The technique is called *Richardson's extrapolation* or *h-extrapolation*. It can also be used to obtain error estimates of the fine-mesh solution. The cost of obtaining the error estimate is approximately twice the cost of obtaining the solution. In two and three dimensions the cost factors rise to, respectively, four and eight times the solution cost. Most would consider this to be excessive. The only way of justifying the procedure is to consider the fine-mesh solution as being the result and the coarse-mesh solution as furnishing the error estimate. This strategy only furnishes an error estimate on the coarse mesh.

Another strategy for obtaining an error estimate by extrapolation is to compute a second solution using a higher-order method (Figure 7.4.1), *e.g.*,

$$u(x) - U_h^{p+1} = C_{p+2}h^{p+2} + O(h^{p+3}).$$

Now, use the identity

$$u(x) - U_h^p(x) = [u(x) - U_h^{p+1}(x)] + [U_h^{p+1}(x) - U_h^p].$$

The first term on the right is the $O(h^{p+1})$ error of the higher-order solution and, hence, can be neglected relative to the second term. Thus, we obtain the approximation

$$u(x) - U_h^p(x) \approx U_h^{p+1}(x) - U_h^p(x).$$

The difference between the lower- and higher-order solutions furnish an estimate of the error of the lower-order solution. The technique is called *order embedding* or *p-extrapolation*. There is no error estimate for the higher-order solution, but some use it without an error estimate. This strategy, called *local extrapolation*, can be dangerous near singularities. Unless there are special properties of the scheme that can be exploited, the work involved in obtaining the error estimate is comparable to the work of obtaining the solution. With a hierarchical embedding, computations needed for the lower-order method are also needed for the higher-order method and, hence, need not be repeated.

The extrapolation techniques just described are typically too expensive for use as error estimates. We'll develop a residual-based error estimation procedure that follows Bank (*cf.* [8], Chapter 7) and uses many of the ideas found in order embedding. We'll follow our usual course of presenting results for the model problem

$$-\nabla \cdot p\nabla u + qu = -(pu_x)_x - (pu_y)_y + qu = f(x,y), \qquad (x,y) \in \Omega, \tag{7.4.1a}$$

$$u(x,y) = \alpha, \qquad (x,y) \in \partial\Omega_E, \qquad pu_{\mathbf{n}}(x,y) = \beta, \qquad (x,y) \in \partial\Omega_N; \tag{7.4.1b}$$

however, results apply more generally. Of course, the Galerkin form of (7.4.1) is: determine $u \in H_E^1$ such that

$$A(v,u) = (v,f) + < v, \beta >, \qquad \forall v \in H_0^1, \tag{7.4.2a}$$

where

$$(v,f) = \iint_\Omega vf\,dxdy, \tag{7.4.2b}$$

$$A(v,u) = \iint_\Omega [p\nabla v \cdot \nabla u + qvu]dxdy, \tag{7.4.2c}$$

and

$$< v,u > = \int_{\partial\Omega_N} vuds. \tag{7.4.2d}$$

Similarly, the finite element solution $U \in S_E^N \subset H_E^1$ satisfies

$$A(V,U) = (V,f) + < V, \beta >, \qquad \forall V \in S_0^N. \tag{7.4.3}$$

We seek an error estimation technique that only requires local (element level) mesh computations, so let's construct a local Galerkin problem on element e by integrating (7.4.1a) over Ω_e and applying the divergence theorem to obtain: determine $u \in H^1(\Omega_e)$ such that

$$A_e(v, u) = (v, f)_e + < v, pu_{\mathbf{n}} >_e, \qquad \forall v \in H^1(\Omega_e), \tag{7.4.4a}$$

where

$$(v, f)_e = \iint_{\Omega_e} vf dxdy, \tag{7.4.4b}$$

$$A_e(v, u) = \iint_{\Omega_e} [p\nabla v \cdot \nabla u + qvu] dxdy, \tag{7.4.4c}$$

and

$$< v, u >_e = \int_{\partial\Omega_e} vuds. \tag{7.4.4d}$$

As usual, Ω_e is the domain of element e, s is a coordinate along $\partial\Omega_e$, and $\mathbf{n}$ is a unit outward normal to $\partial\Omega_e$.

Let

$$u = U + e, \tag{7.4.5}$$

where $e(x, y)$ is the discretization error of the finite element solution, and substitute (7.4.5) into (7.4.4a) to obtain

$$A_e(v, e) = (v, f)_e - A_e(v, U) + < v, pu_{\mathbf{n}} >_e, \qquad \forall v \in H^1(\Omega_e). \tag{7.4.6}$$

Equation (7.4.6), of course, cannot be solved because (*i*) v, u, and e are elements of an infinite-dimensional space and (*ii*) the flux $pu_{\mathbf{n}}$ is unknown on $\partial\Omega_e$. We could obtain a finite element solution of (7.4.6) by approximating e and v by E and V in a finite-dimensional subspace $\tilde{S}^N(\Omega_e)$ of $H^1(\Omega_e)$. Thus,

$$A_e(V, E) = (V, f)_e - A_e(V, U) + < V, pu_{\mathbf{n}} >_e, \qquad \forall V \in \tilde{S}^N(\Omega_e). \tag{7.4.7}$$

We will discuss selection of $\tilde{S}^N$ momentarily. Let us first prescribe the flux $pu_{\mathbf{n}}$ appearing in the last term of (7.4.7). The simplest possibility is to use an average flux obtained from $pU_{\mathbf{n}}$ across the element boundary, *i.e.*,

$$A_e(V, E) = (V, f)_e - A_e(V, U) + < V, \frac{(pU_{\mathbf{n}})^+ + (pU_{\mathbf{n}})^-}{2} >_e, \qquad \forall V \in \tilde{S}^N(\Omega_e), \tag{7.4.8}$$

where superscripts $+$ and $-$, respectively, denote values of $pU_{\mathbf{n}}$ on the exterior and interior of $\partial\Omega_e$.

Equation (7.4.8) is a local Neumann problem for determining the error approximation E on each element. No assembly and global solution is involved. Some investigators prefer to apply the divergence theorem to the second term on the right to obtain

$$A_e(V,E) = (V,r)_e - < V, (pU_{\mathbf{n}})^- >_e + < V, \frac{(pU_{\mathbf{n}})^+ + (pU_{\mathbf{n}})^-}{2} >_e$$

or

$$A_e(V,E) = (V,r)_e + < V, \frac{(pU_{\mathbf{n}})^+ - (pU_{\mathbf{n}})^-}{2} >_e \tag{7.4.9a}$$

where

$$r(x,y) = f + \nabla \cdot p\nabla U - qU \tag{7.4.9b}$$

is the residual. This form involves jumps in the flux across element boundaries.

Now let us select the error approximation space $\tilde{S}^N$. Choosing $\tilde{S}^N = S^N$ does not work since there are no errors in the solution subspace. Bank [10] chose $\tilde{S}^N$ as a space of discontinuous polynomials of the same degree p used for the solution space S_E^N; however, the algebraic system for E resulting from (7.4.8) or (7.4.9) could be ill-conditioned when the basis is nearly continuous. A better alternative is to select $\tilde{S}^N$ as a space of piecewise $p+1$ *st*-degree polynomials when S_E^N is a space of p *th* degree polynomials. Hierarchical bases (*cf.* Sections 2.5 and 4.4) are the most efficient to use in this regard. Let us illustrate the procedure by constructing error estimates for a piecewise bilinear solution on a mesh of quadrilateral elements. The bilinear shape functions for a canonical 2×2 square element are

$$N_{i,j}^1(\xi,\eta) = \bar{N}_i(\xi)\bar{N}_j(\eta), \qquad i,j = 1,2, \tag{7.4.10a}$$

where

$$\bar{N}_1(\xi) = \frac{1-\xi}{2}, \qquad \bar{N}_2(\xi) = \frac{1+\xi}{2}. \tag{7.4.10b}$$

The four second-order hierarchical shape functions are

$$N_{3,j}^2(\xi,\eta) = \bar{N}_j(\eta)\bar{N}_3^2(\xi), \qquad j = 1,2, \tag{7.4.11a}$$

$$N_{i,3}^2(\xi,\eta) = \bar{N}_i(\xi)\bar{N}_3^2(\eta), \qquad i = 1,2, \tag{7.4.11b}$$

where

$$\bar{N}_3^2(\xi) = \frac{3(\xi^2-1)}{2\sqrt{6}}. \tag{7.4.11c}$$

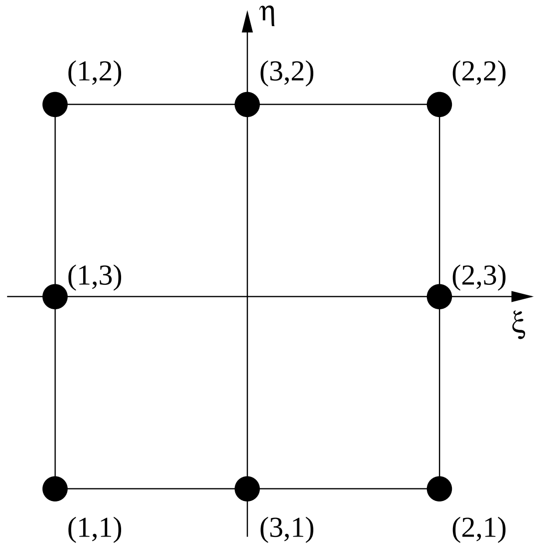


Figure 7.4.2: Nodal placement for bilinear and hierarchical biquadratic shape functions on a canonical 2×2 square element.

Node indexing is given in Figure 7.4.2

The restriction of a piecewise bilinear finite element solution U to the square canonical element is

$$U(\xi, \eta) = \sum_{i=1}^{2} \sum_{j=1}^{2} c_{ij}^{1} N_{ij}^{1}(\xi, \eta). \tag{7.4.12}$$

Using either (7.4.8) or (7.4.9), the restriction of the error approximation E to the canonical element is the second-order hierarchical function

$$E(\xi, \eta) = \sum_{i=1}^{2} \sum_{j=1}^{2} c_{ij}^{2} N_{ij}^{1}(\xi, \eta) + \sum_{i=1}^{2} d_{i3}^{2} N_{i3}^{2}(\xi, \eta) + \sum_{j=1}^{2} d_{3j}^{2} N_{3j}^{2}(\xi, \eta). \tag{7.4.13}$$

The local problems (7.4.8) or (7.4.9) are transformed to the canonical element and solved for the eight unknowns, c_{ij}^2, $i, j = 1, 2$, d_{i3}^2, $i = 1, 2$, d_{3j}^2, $j = 1, 2$, using the test functions $V = N_{ij}^k$, $i, j = 1, 2, 3$, $k = 1, 2$.

Several simplifications and variations are possible. One of these may be called *vertex superconvergence* which implies that the solution at vertices converges more rapidly than it does globally. Vertex superconvergence has been rigorously established in certain circumstances (*e.g.*, for uniform meshes of square elements), but it seems to hold more widely than current theory would suggest. In the present context, vertex superconvergence implies that the bilinear vertex solution c_{ij}^1, $i, j = 1, 2$, converges at a higher rate than the solution elsewhere on Element e. Thus, the error at the vertices c_{ij}^2, $i, j = 1, 2$, may be neglected relative to d_{i3}^2, $i = 1, 2$, and d_{3j}^2, $j = 1, 2$. With this simplification, (7.4.13) becomes

$$E(\xi,\eta) = \sum_{i=1}^{2} d_{i3}^2 N_{i3}^2(\xi,\eta) + \sum_{j=1}^{2} d_{3j}^2 N_{3j}^2(\xi,\eta). \tag{7.4.14}$$

Thus, there are four unknowns d_{13}^2, d_{23}^2, d_{31}^2, and d_{32}^2 per element. This technique may be carried to higher orders. Thus, if S_E^N contains complete polynomials of degree p, $\tilde{S}^N$ only contains the hierarchical correction of order $p+1$. All lower-order terms are neglected in the error estimation space.

The performance of an error estimate is typically appraised in a given norm by computing an *effectivity index* as

$$\Theta = \frac{\|E(x,y)\|}{\|e(x,y)\|}. \tag{7.4.15}$$

Ideally, the effectivity index should not differ greatly from unity for a wide range of mesh spacings and polynomial degrees. Bank and Weiser [11] and Oden *et al.* [17] studied the error estimation procedure (7.4.8) with the simplifying assumption (7.4.14) and were able to establish upper bounds of the form $\Theta \leq C$ in the strain energy norm

$$\|e\|_A, = \sqrt{A(e,e)}.$$

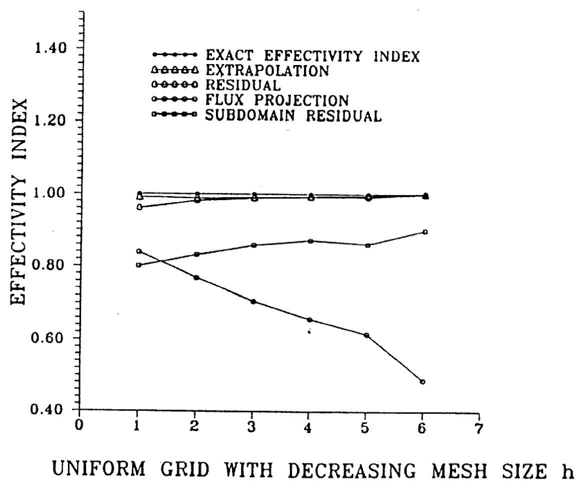
They could not, however, show that the estimation procedure was *asymptotically correct* in the sense that $\Theta \to 1$ under mesh refinement or order enrichment.

Example 7.4.1. Strouboulis and Haque [19] study the properties of several different error estimation procedures. We report results for the residual error estimation procedure (7.4.8, 7.4.14) on the "Gaussian Hill" problem. This problem involves a Dirichlet problem for Poisson's equation on an equilateral triangle having the exact solution

$$u(x,y) = 100e^{-1.5[(x-4.5)^2+(y-2.6)^2]}.$$

Errors are shown in Figure 7.4.3 for unifom p-refinement on a mesh of uniform triangular elements having an edge length of 0.25 and for uniform h-refinement with $p = 2$. "Extrapolation" refers to the p-refinement procedure described earlier in this section. This order embedding technique appears to produce accurate error estimates for all polynomial degrees and mesh spacings. The "residual" error estimation procedure is (7.4.8) with errors at vertices neglected and the hierarchical corrections of order $p+1$ forming $\tilde{S}^N$ (7.4.14). The procedure does well for even-degree approximations, but less well for odd-degree approximations.

From (7.4.8), we see that the error estimate E is obtained by solving a Neumann problem. Such problems are only solvable when the edge loading (the flux average across



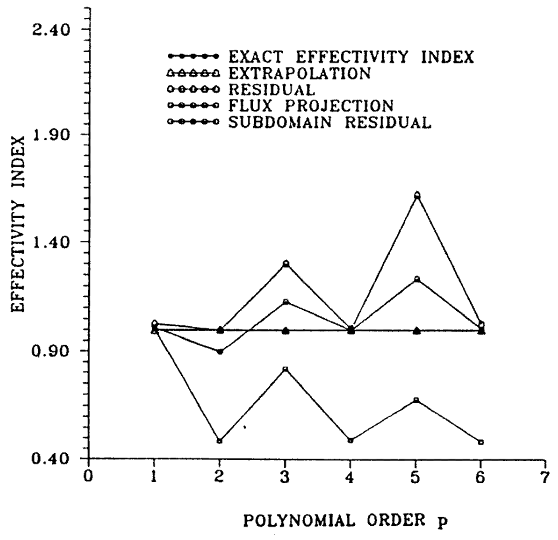


Figure 7.4.3: Effectivity indices for several error estimation procedures using uniform h-refinement (left) and p-refinement (right) for the Gaussian Hill Problem [19] of Example 7.4.1.

element edges) is equilibrated. The flux averaging used in (7.4.8) is, apparently, not sufficient to ensure this when p is odd. We'll pursue some remedies to this problem later in this section, but, first, let us look at another application.

Example 7.4.2. Aiffa [4] considers the nonlinear parabolic problem

$$u_t + qu^2(u-1) = \frac{u_{xx}+u_{yy}}{2}, \qquad (x,y) \in (0,1)\times(0,1), \qquad t>0,$$

with the inital and Dirichlet boundary conditions specified so that the exact solution is

$$u(x,y,t) = \frac{1}{1+e^{\sqrt{q/2}(x+y-t\sqrt{q/2})}}.$$

He estimates the spatial discretization error using the residual estimate (7.4.8) neglecting the error at vertices. The error estimation space $\tilde{S}^N$ consists of the hierarchical corrections of degree $p+1$; however, some lower-degree hierarchical terms are used in some cases. This is to provide a better equilibration of boundary terms and improve results. although this is a time-dependent problem, which we haven't studied yet, Aiffa [4] keeps the temporal errors small to concentrate on spatial error estimation. With $q = 500$, Aiffa's [4] effectivity indices in H^1 at $t = 0.06$ are presented in Table 7.4.1 for computations performed on uniform meshes of N_Δ triangles with polynomial degrees p ranging from 1 to 4.

The results with $\tilde{S}^N$ consisting only of hierarchical corrections of degree $p+1$ are reasonable. Effectivity indices are in excess of 0.9 for the lower-degree polynomials $p =$

p	$\tilde{S}^N$	N_Δ			
		8	32	128	512
1	2	1.228	1.066	1.019	1.005
2	3	0.948	0.993	0.998	0.999
3	4	0.951	0.938	0.938	0.938
	4, 2	3.766	1.734	1.221	1.039
4	5	0.650	0.785	0.802	0.803
	5, 3	0.812	0.911	0.920	0.925

Table 7.4.1: Effectivity indices in H^1 at $t = 0.06$ for Example 7.4.2. The degrees of the hierarchical modes used for $\tilde{S}^N$ are indicated in that column [4].

$1, 2$, but degrade with increasing polynomial degree. The addition of a lower (third) degree polynomial correction has improved the error estimates with $p = 4$; however, a similar tactic provided little improvement with $p = 3$. These results and those of Strouboulis and Haque [19] show that the performance of *a posteriori* error estimates is still dependent on the problem being solved and on the mesh used to solve it.

Another way of simplifying the error estimation procedure (7.4.8) and of understanding the differences between error estimates for odd- and even-order finite element solutions involves a profound, but little known, result of Babuška (*cf.* [1, 2, 3, 9, 22, 23]). Concentrating on linear second-order elliptic problems on rectangular meshes, Babuška indicates that asymptotically (as mesh spacing tends to zero) errors of odd-degree finite element solutions occur near element edges while errors of even-degree solutions occur in element interiors. These findings suggest that error estimates may be obtained by neglecting errors in element interiors for odd-degree polynomials and neglecting errors on element boundaries for even-degree polynomials.

Thus, for piecewise odd-degree approximations, we could neglect the area integrals on the right-hand sides of (7.4.8) or (7.4.9a) and calculate an error estimate by solving

$$A_e(V, E) = < V, \frac{(pU_{\mathbf{n}})^+ + (pU_{\mathbf{n}})^-}{2} >_e, \qquad \forall V \in \tilde{S}^N. \tag{7.4.16a}$$

or

$$A_e(V, E) = < V, \frac{(pU_{\mathbf{n}})^+ - (pU_{\mathbf{n}})^-}{2} >_e, \qquad \forall V \in \tilde{S}^N. \tag{7.4.16b}$$

For piecewise even-degree approximations, the boundary terms in (7.4.8) or (7.4.9a) can be neglected to yield

$$A_e(V, E) = (V, f)_e - A_e(V, U), \qquad \forall V \in \tilde{S}^N. \tag{7.4.17a}$$

or

$$A_e(V,E) = (V,r)_e, \qquad \forall V \in \tilde{S}^N. \tag{7.4.17b}$$

Yu [22, 23] used these arguments to prove asymptotic convergence of error estimates to true errors for elliptic problems. Adjerid *et al.* [2, 3] obtained similar results for transient parabolic systems. Proofs, in both cases, apply to a square region with square elements of spacing $h = 1/\sqrt{N_\Delta}$. A typical result follows.

Theorem 7.4.1. *Let* $u \in H_E^1 \cap H^{p+2}$ *and* $U \in S_E^N$ *be solutions of (7.4.2) using complete piecewise-bi-polynomial functions of order* p.

1. *If* p *is an odd positive integer then*

$$\|e(\cdot,\cdot)\|_1^2 = \|E(\cdot,\cdot)\|_1^2 + O(h^{2p+1}) \tag{7.4.18a}$$

where

$$\|E\|_1^2 = \frac{h^2}{16(2p+1)} \sum_{e=1}^{N_\Delta} \sum_{i=1}^{2} \sum_{k=1}^{4} [U_{x_i}(\mathbf{P}_{k,e})]_i^2, \tag{7.4.18b}$$

$\mathbf{P}_{k,e}$, $k = 1,2,3,4$, *are the coordinates of the vertices of* Ω_e, *and* $[f(\mathbf{P})]_i$ *denotes the jump in* $f(\mathbf{x})$ *in the direction* x_i, $i = 1,2$, *at the point* $\mathbf{P}$.

2. *If* p *is a positive even integer then (7.4.18a) is satisfied with*

$$A_e(V_i,E) = (V,f)_e - A_e(V_i,U), \tag{7.4.18c}$$

where

$$E(x_1,x_2) = b_{1,e}\Phi_e^{p+1}(x_1) + b_{2,e}\Phi_e^{p+1}(x_2), \tag{7.4.18d}$$

$$V_i(x_1,x_2) = x_i \frac{\Phi_e^{p+1}(x_1)}{x_1}\frac{\Phi_e^{p+1}(x_2)}{x_2}, \qquad i = 1,2, \tag{7.4.18e}$$

and $\Phi_e^m(x)$ *is the mapping of the hierarchical basis function*

$$\bar{N}_3^m(\xi) = \sqrt{\frac{2m-1}{2}} \int_{-1}^{\xi} P_{m-1}(\zeta)d\zeta \tag{7.4.18f}$$

from $[-1,1]$ *to the appropriate edge of* Ω_e.

Proof. cf. Adjerid *et al.* [2, 3] and Yu [22, 23]. Coordinates are written as $\mathbf{x} = [x_1, x_2]^T$ instead of (x,y) to simplify notation within summations. The hierarchical basis element (7.4.18f) is consistent with prior usage. Thus, the subscript 3 refers to a midside node as indicated in Figure 7.4.2. □

Remark 1. The error estimate for even-degree approximations has different trial and test spaces. The functions $V_i(x_1, x_2)$ vanish on $\partial\Omega_e$. Each function is the product of a "bubble function" $\Phi_e^{p+1}(x_1)\Phi_e^{p+1}(x_2)$ biased by a variation in either the x_1 or the x_2 direction. As an example, consider the test functions on the canonical element with $p = 2$. Restricting (7.4.18e) to the canonical element $-1 \leq \xi_1, \xi_2 \leq 1$, we have

$$V_i(\xi_1, \xi_2) = \xi_i \frac{\bar{N}_3^3(\xi_1)}{\xi_1} \frac{\bar{N}_3^3(\xi_2)}{\xi_2}, \qquad i = 1, 2.$$

Using (7.4.18f) with $m = 3$ or (2.5.8),

$$\bar{N}_3^3(\xi) = \frac{5}{2\sqrt{10}} \xi(\xi^2 - 1).$$

Thus,

$$V_i(\xi_1, \xi_2) = \frac{5\xi_i}{8}(\xi_1^2 - 1)(\xi_2^2 - 1), \qquad i = 1, 2.$$

Remark 2. Theorem 7.4.1 applies to tensor-product bi-polynomial bases. Adjerid *et al.* [1] show how this theorem can be modified for use with hierarchical bases.

Example 7.4.3. Adjerid *et al.* [2] solve the nonlinear parabolic problem of Example 7.4.2 with $q = 20$ on uniform square meshes with p ranging from 1 to 4 using the error estimates (7.4.18a,b) and (7.4.18a,c-f). Temporal errors were controlled to be negligible relative to spatial errors; thus, we need not be concerned that this is a parabolic and not an elliptic problem. The exact H^1 errors and effectivity indices at $t = 0.5$ are presented in Table 7.4.2. Approximate errors are within ten percent of actual for all but one mesh and appear to be converging at the same rate as the actual errors under mesh refinement.

p	$N_\Delta = 100$		400		900		1600	
	$\|\|e\|\|_1/\|\|u\|\|_1$	Θ	$\|\|e\|\|_1/\|\|u\|\|_1$	Θ	$\|\|e\|\|_1/\|\|u\|\|_1$	Θ	$\|\|e\|\|_1/\|\|u\|\|_1$	Θ
1	0.262(-1)	0.949	0.129(-1)	0.977	0.858(-2)	0.985	0.643(-2)	0.989
2	0.872(-3)	0.995	0.218(-3)	0.999	0.963(-4)	0.999	0.544(-4)	1.000
3	0.278(-4)	0.920	0.348(-5)	0.966	0.103(-5)	0.979	0.436(-6)	0.979
4	0.848(-6)	0.999	0.530(-7)	1.000	0.105(-7)	1.000	0.331(-8)	1.000

Table 7.4.2: Errors and effectivity indices in H^1 for Example 7.4.3 on N_Δ-element uniform meshes with piecewise bi-p polynomial bases. Numbers in parentheses indicate a power of ten.

The error estimation procedures (7.4.8) and (7.4.9) use average flux values on $\partial\Omega_e$. As noted, data for such (local) Neumann problems cannot be prescribed arbitrarily. Let us examine this further by concentrating on (7.4.9) which we write as

$$A_e(V, E) = (V, r)_e + < V, R >_e \tag{7.4.19a}$$

where the elemental residual r was defined by (7.4.9b) and the boundary residual is

$$R = \kappa[(pU_{\mathbf{n}})^+ - (pU_{\mathbf{n}})^-]. \tag{7.4.19b}$$

The function κ on $\partial\Omega_e$ was taken as 1/2 to obtain (7.4.9a); however, this may not have been a good idea for reasons suggested in Example 7.4.1.

Recall (*cf.* Section 3.1) that smooth solutions of the weak problem (7.4.19) satisfy the Neumann problem

$$-\nabla \cdot p\nabla E + qE = r, \qquad (x,y) \in \Omega_e, \tag{7.4.20a}$$

$$pE_{\mathbf{n}} = R, \qquad (x,y) \in \partial\Omega_e. \tag{7.4.20b}$$

Solutions of (7.4.20) only exist when the data R and r satisfy the *equilibrium condition*

$$\iint_{\Omega_e} r(x,y)dxdy + \int_{\partial\Omega_e} R(s)ds = 0. \tag{7.4.20c}$$

This condition will most likely not be satisfied by the choice of $\kappa = 1/2$. Ainsworth and Oden [5] describe a relatively simple procedure that requires the solution of the Poisson problem

$$-\Delta\omega_e = r, \qquad (x,y) \in \Omega_e, \tag{7.4.21a}$$

$$\frac{\partial\omega_e}{\partial\mathbf{n}} = R, \qquad (x,y) \in \partial\Omega_e - \partial\Omega_E, \tag{7.4.21b}$$

$$\omega_e = 0, \qquad (x,y) \in \partial\Omega_E. \tag{7.4.21c}$$

The error estimate is

$$||E||_A^2 = \sum_{e=1}^{N_\Delta} A_e(\omega_e, \omega_e). \tag{7.4.21d}$$

The function κ is approximated by a piecewise-linear polynomial in a coordinate s on $\partial\Omega_e$ and may be determined explicitly prior to solving (7.4.21). Let us illustrate the effect of this equilibrated error estimate.

Example 7.4.4. Oden [16] considers a "cracked panel" as shown in Figure 7.4.4 and determines u as the solution of

$$A(v,u) = \iint_{\Omega} (v_x u_x + v_y u_y)dxdy = 0.$$

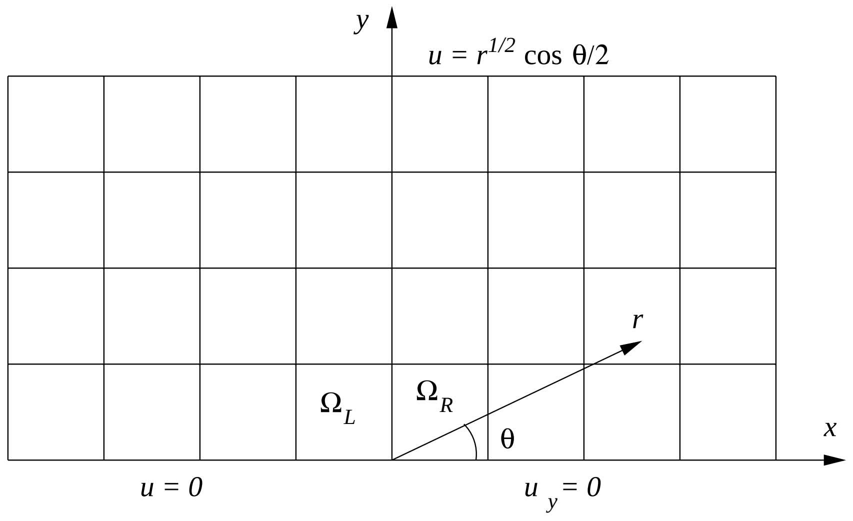


Figure 7.4.4: Cracked panel used for Example 7.4.4.

p	$1/h$	$\Theta(\Omega_L)$		$\Theta(\Omega_R)$		$\Theta(\Omega)$	
		With Balancing	Without Balancing	With Balancing	Without Balancing	With Balancing	Without Balancing
1	32	1.135	0.506	0.879	1.429	1.017	1.049
1	64	1.118	0.498	0.888	1.443	1.012	1.044
2	32	1.162	0.578	0.835	1.175	1.008	0.921

Table 7.4.3: Local and global effectivity indices for Example 7.4.4 using (7.4.21) with and without equilibration.

The essential boundary condition

$$u(r,\theta) = r^{1/2}\cos\theta/2$$

is prescribed on all boundaries except $x > 0$, $y = 0$. Thus, the solution of the Galerkin problem will satisfy the natural boundary condition $u_y = 0$ there. These conditions have been chosen so that the exact solution is the specified essential boundary condition. This solution is singular since $u_r \sim r^{-1/2}$ near the origin ($r = 0$).

Results for the effectivity indices in strain energy for the entire region and for the two elements, Ω_L and Ω_R, adjacent to the singularity are shown in Table 7.4.3. Computations were performed on a square grid with uniform spacing h in each coordinate direction (Figure 7.4.4). Piecewise linear and quadratic polynomials were used as finite element bases.

Local effectivity indices on Ω_L and Ω_R are not close to unity and don't appear to be converging as either the mesh spacing is refined or p is increased. Global effectivity indices are near unity. Convergence to unity is difficult to appraise with the limited data.

At this time, the field of *a posteriori* error estimation is still emerging. Error estimates for problems with singularities are not generally available. The performance of error estimates is dependent on both the problem, the mesh, and the basis. Error estimates for realistic nonlinear and transient problems are just emerging. Verfürth [20] provides an exceelent survey of methods and results.

Bibliography

[1] S. Adjerid, B. Belguendouz, and J.E. Flaherty. A posteriori finite element error estimation for diffusion problems. Technical Report 9-1996, Scientific Computation Research Center, Rensselaer Polytechnic Institute, Troy, 1996. SIAM Journal on Scientific Computation, to appear.

[2] S. Adjerid, J.E. Flaherty, and I. Babuška. A posteriori error estimation for the finite element method-of-lines solution of parabolic problems. *Mathematical Models and Methods in Applied Science*, 9:261–286, 1999.

[3] S. Adjerid, J.E. Flaherty, and Y.J. Wang. A posteriori error estimation with finite element methods of lines for one-dimensional parabolic systems. *Numererishe Mathematik*, 65:1–21, 1993.

[4] M. Aiffa. *Adaptive hp-Refinement Methods for Singularly-Perturbed Elliptic and Parabolic Systems*. PhD thesis, Rensselaer Polytechnic Institute, Troy, 1997.

[5] M. Ainsworth and J.T. Oden. A unified approach to a posteriori error estimation using element residual methods. *Numeriche Mathematik*, 65:23–50, 1993.

[6] O. Axelsson and V.A. Barker. *Finite Element Solution of Boundary Value Problems*. Academic Press, Orlando, 1984.

[7] I. Babuška, T. Strouboulis, and C.S. Upadhyay. A model study of the quality of a-posteriori estimators for linear elliptic problems. Part Ia: Error estimation in the interior of patchwise uniform grids of triangles. Technical Report BN-1147, Institute for Physical Science and Technology, University of Maryland, College Park, 1993.

[8] I. Babuška, O.C. Zienkiewicz, J. Gago, and E.R. de A. Oliveira, editors. *Accuracy Estimates and Adaptive Refinements in Finite Element Computations*. John Wiley and Sons, Chichester, 1986.

[9] I. Babuška and D. Yu. Asymptotically exact a-posteriori error estimator for biquadratic elements. Technical Report BN-1050, Institute for Physical Science and Technology, University of Maryland, College Park, 1986.

[10] R.E. Bank. *PLTMG: A Software Package for Solving Elliptic Partial Differential Equations. Users' Guide 6.0.* SIAM, Philadelphia, 1980.

[11] R.E. Bank and A. Weiser. Some a posteriori error estimators for elliptic partial differential equations. *Mathematics of Computation*, 44:283–302, 1985.

[12] S.C. Brenner and L.R. Scott. *The Mathematical Theory of Finite Element Methods.* Springer-Verlag, New York, 1994.

[13] P.G. Ciarlet. *The Finite Element Method for Elliptic Problems.* North-Holland, Amsterdam, 1978.

[14] C. Johnson. *Numerical Solution of Partial Differential Equations by the Finite Element method.* Cambridge, Cambridge, 1987.

[15] J. Necas. *Les Méthods Directes en Théorie des Equations Elliptiques.* Masson, Paris, 1967.

[16] J.T. Oden. Topics in error estimation. Technical report, Rensselaer Polytechnic Institute, Troy, 1992. Tutorial at the Workshop on Adaptive Methods for Partial Differential Equations.

[17] J.T. Oden, L. Demkowicz, W. Rachowicz, and T.A. Westermann. Toward a universal h-p adaptive finite element strategy, part 2: A posteriori error estimation. *Computer Methods in Applied Mechanics and Engineering*, 77:113–180, 1989.

[18] G. Strang and G. Fix. *Analysis of the Finite Element Method.* Prentice-Hall, Englewood Cliffs, 1973.

[19] T. Strouboulis and K.A. Haque. Recent experiences with error estimation and adaptivity, Part I: Review of error estimators for scalar elliptic problems. *Computer Methods in Applied Mechanics and Engineering*, 97:399–436, 1992.

[20] R. Verfürth. *A Review of Posteriori Error Estimation and Adaptive Mesh-Refinement Techniques.* Teubner-Wiley, Stuttgart, 1996.

[21] R. Wait and A.R. Mitchell. *The Finite Element Analysis and Applications.* John Wiley and Sons, Chichester, 1985.

[22] D.-H. Yu. Asymptotically exact a-posteriori error estimator for elements of bi-even degree. *Mathematica Numerica Sinica*, 13:89–101, 1991.

[23] D.-H. Yu. Asymptotically exact a-posteriori error estimator for elements of bi-odd degree. *Mathematica Numerica Sinica*, 13:307–314, 1991.

Chapter 8

Adaptive Finite Element Techniques

8.1 Introduction

The usual finite element analysis would proceed from the selection of a mesh and basis to the generation of a solution to an accuracy appraisal and analysis. Experience is the traditional method of determining whether or not the mesh and basis will be optimal or even adequate for the analysis at hand. Accuracy appraisals typically require the generation of a second solution on a finer mesh or with a different method and an *ad hoc* comparison of the two solutions. At least with *a posteriori* error estimation (*cf.* Section 7.4), accuracy appraisals can accompany solution generation at a lower cost than the generation of a second solution.

Adaptive procedures try to automatically refine, coarsen, or relocate a mesh and/or adjust the basis to achieve a solution having a specified accuracy in an optimal fashion. The computation typically begins with a trial solution generated on a coarse mesh with a low-order basis. The error of this solution is appraised. If it fails to satisfy the prescribed accuracy, adjustments are made with the goal of obtaining the desired solution with minimal effort. For example, we might try to reduce the discretization error to its desired level using the fewest degrees of freedom. While adaptive finite element methods have been studied for nearly twenty years [4, 5, 8, 13, 15, 18, 21, 36, 41], surprising little is known about optimal strategies. Common procedures studied to date include

- local refinement and/or coarsening of a mesh (*h-refinement*),
- relocating or moving a mesh (*r-refinement*), and
- locally varying the polynomial degree of the basis (*p-refinement*).

These strategies may be used singly or in combination. We may guess that r-refinement alone is generally not capable of finding a solution with a specified accuracy. If the mesh is too coarse, it might be impossible to achieve a high degree of precision without adding more elements or altering the basis. R-refinement is more useful with transient problems where elements move to follow an evolving phenomena. By far, h-refinement is the most popular [5, 13, 15, 18, 21, 41]. It can increase the convergence rate, particularly when singularities are present (*cf.* [6, 33] or Example 8.2.1). In some sense p-refinement is the most powerful. Exponential convergence rates are possible when solutions are smooth [8, 36, 40]. When combined with h-refinement, these high rates are also possible when singularities are present [31, 32, 36]. The use of p-refinement is most natural with a hierarchical basis, since portions of the stiffness and mass matrices and load vector will remain unchanged when increasing the polynomial degree of the basis.

A posteriori error estimates provide accuracy appraisals that are necessary to terminate an adaptive procedure. However, optimal strategies for deciding where and how to refine or move a mesh or to change the basis are rare. In Section 7.4, we saw that *a posteriori* error estimates in a particular norm were computed by summing their elemental contributions as

$$\|E\|^2 = \sum_{e=1}^{N_\Delta} \|E\|_e^2 \tag{8.1.1}$$

where N_Δ is the number of elements in the mesh and $\|E\|_e^2$ is the restriction of the error estimate $\|E\|^2$ to Element e. The most popular method of determining where adaptivity is needed is to use $\|E\|_e$ as an *enrichment indicator*. Thus, we assume that large errors come from regions where the local error estimate $\|E\|_e$ is large and this is where we should refine or concentrate the mesh and/or increase the method order. Correspondingly, the mesh would be coarsened or the polynomial degree of the basis lowered in regions where $\|E\|_e$ is small. This is the strategy that we'll follow (*cf.* Section 8.2); however, we reiterate that there is no proof of the optimality of enrichment in the vicinity of the largest local error estimate.

Enrichment indicators other than local error estimates have been tried. The use of solution gradients is popular. This is particularly true of fluid dynamics problems where error estimates are not readily available [14, 16, 17, 19].

In this chapter, we'll examine h-, p-, and hp-refinement. Strategies using r-refinement will be addressed in Chapter 9.

8.2 *h*-Refinement

Mesh refinement strategies for elliptic (steady) problems need not consider coarsening. We can refine an initially coarse mesh until the requested accuracy is obtained. This strategy might not be optimal and won't be, for example, if the coarse mesh is too fine in some regions. Nevertheless, we'll concentrate on refinement at the expense of coarsening. We'll also focus on two-dimensional problems to avoid the complexities of three-dimensional geometry.

8.2.1 Structured Meshes

Let us first consider adaptivity on structured meshes and then examine unstructured-mesh refinement. Refinement of an element of a structured quadrilateral-element mesh by bisection requires mesh lines running to the boundaries to retain the four-neighbor structure (*cf.* the left of Figure 8.2.1). This strategy is simple to implement and has been used with finite difference computation [42]; however, it clearly refines many more elements than necessary. The customary way of avoiding the excess refinement is to introduce *irregular nodes* where the edges of a refined element meet at the midsides of a coarser one (*cf.* the right of Figure 8.2.1). The mesh is no longer structured and our standard method of basis construction would create discontinuities at the irregular nodes.

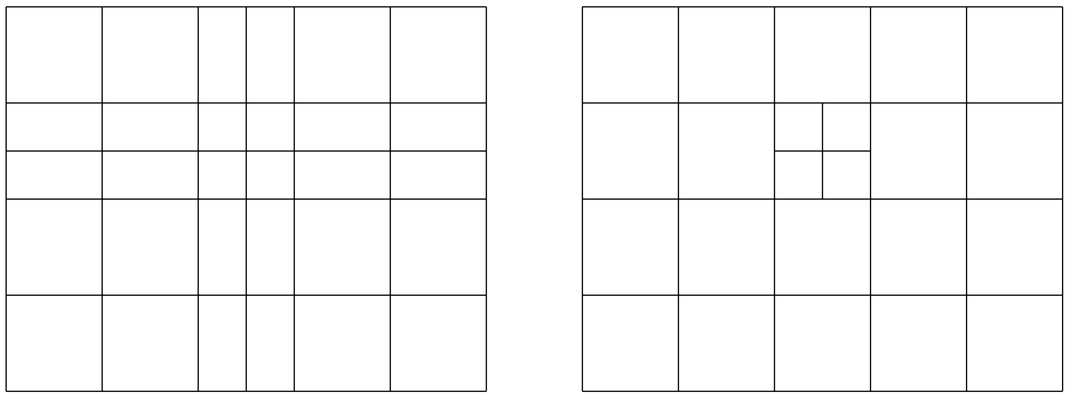


Figure 8.2.1: Bisection of an element of a structured rectangular-element mesh creating mesh lines running between the boundaries (left). The mesh lines are removed by creating irregular nodes (right).

The usual strategy of handling continuity at irregular nodes is to *constrain the basis.* Let us illustrate the technique for a piecewise-bilinear basis. The procedure for higher-order piecewise polynomials is similar. Thus, consider an edge between Vertices 1 and 2 containing an irregular node 3 as shown in Figure 8.2.2. For simplicity, assume that the elements are $h \times h$ squares and that those adjacent to Edge 1-2 are indexed 1, 2, and 3 as shown in the figure. For convenience, let's also place a Cartesian coordinate system at Vertex 2.

We proceed as usual, constructing shape functions on each element. Although not really needed for our present development, those bilinear shape functions that are nonzero on Edge 1-2 follow.

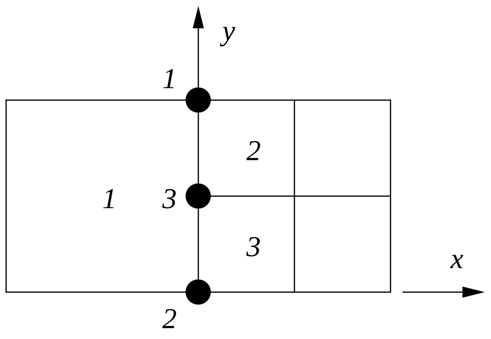


Figure 8.2.2: Irregular node at the intersection of a refined element.

- On Element 1:

$$N_{11} = (\frac{h+x}{h})(\frac{y}{h}), \qquad N_{21} = (\frac{h+x}{h})(\frac{h-y}{h}).$$

- On Element 2:

$$N_{12} = (\frac{h/2-x}{h/2})(\frac{y-h/2}{h/2}), \qquad N_{32} = (\frac{h/2-x}{h/2})(\frac{h-y}{h/2}).$$

- On Element 3:

$$N_{23} = (\frac{h/2-x}{h/2})(\frac{h/2-y}{h/2}), \qquad N_{33} = (\frac{h/2-x}{h/2})(\frac{y}{h/2}).$$

As in Chapter 2, the second subscript on N_{je} denotes the element index.

The restriction of U on Element 1 to Edge 1-2 is

$$U(x,y) = c_1 N_{11}(x,y) + c_2 N_{21}(x,y).$$

Evaluating this at Node 3 yields

$$U(x_3,y_3) = \frac{c_1+c_2}{2}, \qquad x < 0.$$

The restriction of U on Elements 2 and 3 to Edge 1-2 is

$$U(x,y) = \begin{cases} c_1 N_{12}(x,y) + c_3 N_{32}(x,y), & \text{if } y \geq h/2 \\ c_2 N_{23}(x,y) + c_3 N_{33}(x,y), & \text{if } y < h/2 \end{cases}.$$

In either case, we have

$$U(x_3,y_3) = c_3, \qquad x > 0.$$

Equating the two expressions for $U(x_3,y_3)$ yields the *constraint condition*

$$c_3 = \frac{c_1+c_2}{2}. \tag{8.2.1}$$

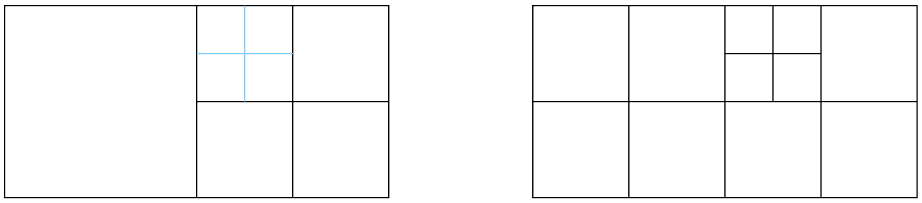


Figure 8.2.3: The one-irregular rule: the intended refinement of an element to create two irregular nodes on an edge (left) necessitates refinement of a neighboring element to have no more than one irregular node per element edge (right).

Thus, instead of determining c_3 by Galerkin's method, we constrain it to be determined as the average of the solutions at the two vertices at the ends of the edge. With the piecewise-bilinear basis used for this illustration, the solution along an edge containing an irregular node is a linear function rather than a piecewise-linear function.

Software based on this form of adaptive refinement has been implemented for elliptic [27] and parabolic [1] systems. One could guess that difficulties arise when there are too many irregular nodes on an edge. To overcome this, software developers typically use Bank's [9, 10] "one-irregular" and "three-neighbor" rules. The one-irregular rule limits the number of irregular nodes on an element edge to one. The impending introduction of a second irregular node on an edge requires refinement of a neighboring element as shown in Figure 8.2.3. The three-neighbor rule states that any element having irregular nodes on three of its four edges must be refined.

A modified quadtree (Section 5.2) can be used to store the mesh and solution data. Thus, let the root of a tree structure denote the original domain Ω. With a structured grid, we'll assume that Ω is square, although it could be obtained by a mapping of a distorted region to a square (Section 5.2). The elements of the original mesh are regarded as offspring of the root (Figure 8.2.4). Elements introduced by adaptive refinement are obtained by bisection and are regarded as offspring of the elements of the original mesh. This structure is depicted in Figure 8.2.4. Coarsening can be done by "pruning" refined quadrants. It's customary, but not essential, to assume that elements cannot be removed (by coarsening) from the original mesh [3].

Irregular nodes can be avoided by using *transition elements* as shown in Figure 8.2.5. The strategy on the right uses triangular elements as a transition between the coarse and fine elements. If triangular elements are not desirable, the transition element on the left uses rectangles but only adds a mid-edge shape functions at Node 3. There is no node at the midpoint of Edge 4-5. The shape functions on the transition element are

$$N_{11} = (\frac{h+x}{h})(\frac{y-h/2}{h/2}), \qquad N_{21} = (\frac{h+x}{h})(\frac{h/2-y}{h/2}),$$

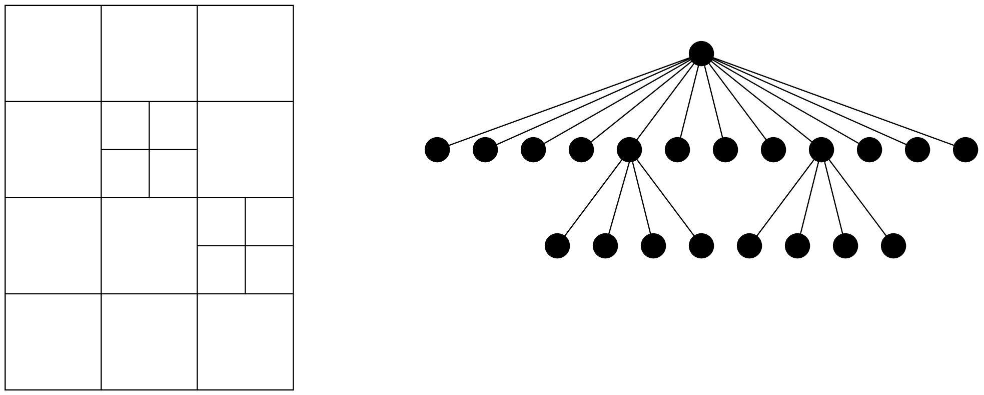


Figure 8.2.4: Original structured mesh and the bisection of two elements (left). The tree structure used to represent this mesh (right).

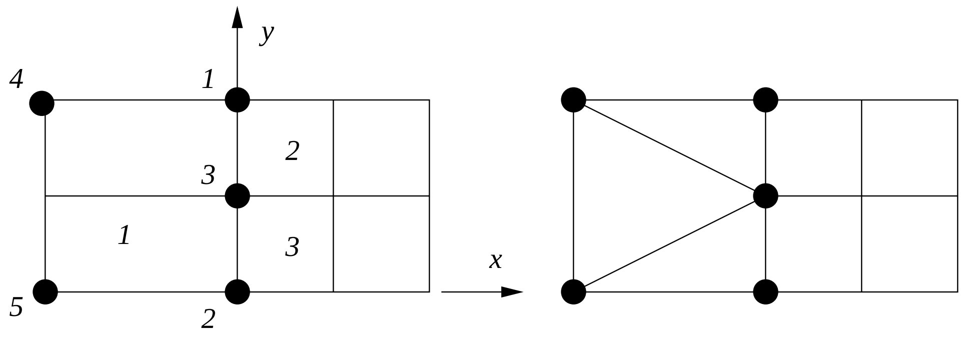


Figure 8.2.5: Transition elements between coarse and fine elements using rectangles (left) and triangles (right).

$$N_{31} = (\frac{h+x}{h}) \begin{cases} (\frac{y}{h/2}), & \text{if } 0 \leq y \leq h/2 \\ (\frac{h-y}{h/2}), & \text{if } h/2 \leq y \leq h \end{cases},$$

$$N_{41} = (\frac{-x}{h})(\frac{y}{h}), \qquad N_{51} = (\frac{-x}{h})(\frac{h-y}{h}).$$

Again, the origin of the coordinate system is at Node 2. Those shape functions associated with nodes on the right edge are piecewise-bilinear on Element 1, whereas those associated with nodes on the left edge are linear.

Berger and Oliger [12] considered structured meshes with structured mesh refinement, but allowed elements of finer meshes to overlap those of coarser ones (Figure 8.2.6). This method has principally used with adaptive finite difference computation, but it has had some use with finite element methods [29].

8.2.2 Unstructured Meshes

Computation with triangular-element meshes has been done since the beginning of adaptive methods. Bank [9, 11] developed the first software system PLTMG, which solves

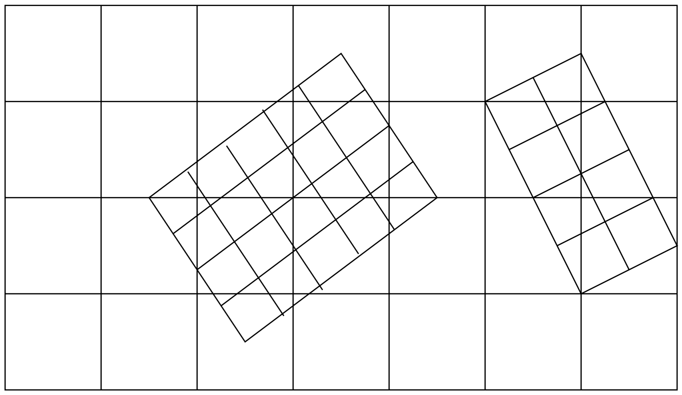


Figure 8.2.6: Composite grid construction where finer grids overlap elements of coarser ones.

our model problem with a piecewise-linear polynomial basis. It uses a multigrid iterative procedure to solve the resulting linear algebraic system on the sequence of adaptive meshes. Bank uses uniform bisection of a triangular element into four smaller elements. Irregular nodes are eliminated by dividing adjacent triangles sharing a bisected edge in two (Figure 8.2.7). Triangles divided to eliminate irregular nodes are called "green triangles" [10]. Bank imposes one-irregular and three-neighbor rules relative to green triangles. Thus, *e.g.*, an intended second bisection of a vertex angle of a green triangle would not be done. Instead, the green triangle would be uniformly refined (Figure 8.2.8) to keep angles bounded away from zero as the mesh is refined.

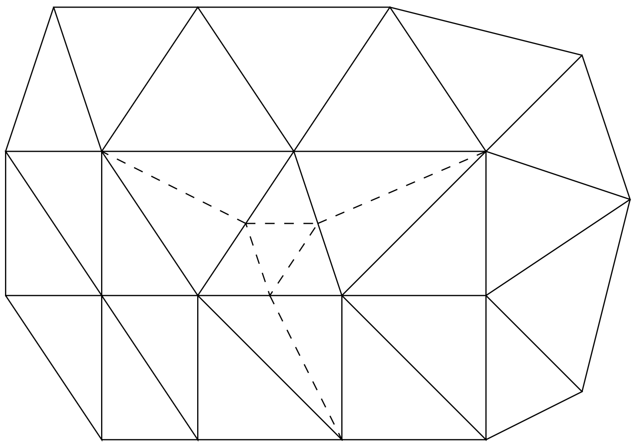


Figure 8.2.7: Uniform bisection of a triangular element into four and the division of neighboring elements in two (shown dashed).

Rivara [34, 33] developed a mesh refinement algorithm based on bisecting the longest edge of an element. Rivara's procedure avoids irregular nodes by additional refinement as described in the algorithm of Figure 8.2.9. In this procedure, we suppose that elements

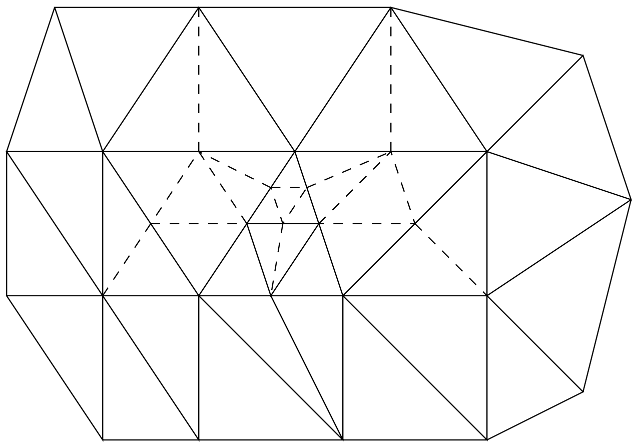


Figure 8.2.8: Uniform refinement of green triangles of the mesh shown in Figure 8.2.7 to avoid the second bisection of vertex angles. New refinements are shown as dashed lines.

of a sub-mesh δ of mesh Δ_h are scheduled for refinement. All elements of δ are bisected by their longest edges to create a mesh Δ_h^1, which may contain irregular nodes. Those elements e of Δ_h^1 that contain irregular nodes are placed in the set ρ^1. Elements of ρ^1 are bisected by their longest edge to create two triangles. This bisection may create another node Q that is different from the original irregular node P of element e. If so, P and Q are joined to produce another element (Figure 8.2.10). The process is continued until all irregular nodes are removed.

```
procedure rivara($\Delta_h$, $\delta$)
   Obtain $\Delta_h^1$ by bisecting all triangles of $\delta$ by their longest edges
   Let $\rho^1$ contain those elements of $\Delta_h^1$ having irregular nodes
   $i := 1$
   while $\rho_i$ is not $\emptyset$ do
      Let $e \in \rho_i$ have an irregular node $P$ and bisect $e$ by its longest edge
      Let $Q$ be the intersection point of this bisection
      if  $P \neq Q$ then
         Join $P$ and $Q$
      end if
      Let $\Delta_h^{i+1}$ be the mesh created by this process
      Let $\rho^{i+1}$ be the set of elements in $\Delta_h^{i+1}$ with irregular nodes
      i := i + 1
   end while
return $\Delta_h^i$
```

Figure 8.2.9: Rivara's mesh bisection algorithm.

Rivara's [33] algorithm has been proven to terminate with a regular mesh in a finite number of steps. It also keep angles bounded away from 0 and π. In fact, if α is the

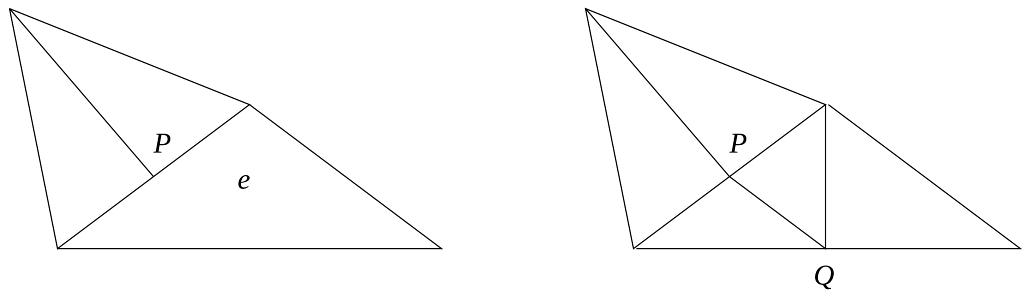


Figure 8.2.10: Elimination of an irregular node P (left) as part of Rivara's algorithm shown in Figure 8.2.9 by dividing the longest edge of Element e and connecting vertices as indicated.

smallest angle of any triangle in the original mesh, the smallest angle in the mesh obtained after an arbitrary number of applications of the algorithm of Figure 8.2.10 is no smaller than $\alpha/2$ [35]. Similar procedures were developed by Sewell [37] and used by Mitchell [28] by dividing the newest vertex of a triangle.

Tree structures can be used to represent the data associated with Bank's [10] and Rivara's [33] procedures. As with structured-mesh computation, elements introduced by refinement are regarded as offspring of coarser parent elements. The actual data representations vary somewhat from the tree described earlier (Figure 8.2.4) and readers seeking more detail should consult Bank [10] or Rivara [34, 33]. With tree structures, any coarsening may be done by pruning "leaf" elements from the tree. Thus, those elements nested within a coarser parent are removed and the parent is restored as the element. As mentioned earlier, coarsening beyond the original mesh is not allowed. The process is complex. It must be done without introducing irregular nodes. Suppose, for example, that the quartet of small elements (shown with dashed lines) in the center of the mesh of Figure 8.2.8 were scheduled for removal. Their direct removal would create three irregular nodes on the edges of the parent triangle. Thus, we would have to determine if removal of the elements containing these irregular nodes is justified based on error-indication information. If so, the mesh would be coarsened to the one shown in Figure 8.2.11. Notice that the coarsened mesh of Figure 8.2.11 differs from mesh of Figure 8.2.7 that was refined to create the mesh of Figure 8.2.8. Hence, refinement and coarsening may not be reversible operations because of their independent treatment of irregular nodes.

Coarsening may be done without a tree structure. Shephard *et al.* [38] describe an "edge collapsing" procedure where the vertex at one end of an element edge is "collapsed" onto the one at the other end. Aiffa [2] describes a two-dimensional variant of this procedure which we reproduce here. Let P be the polygonal region composed of the union of elements sharing Vertex V_0 (Figure 8.2.12). Let $V_1, V_2, \ldots, V_k$ denote the vertices on the k triangles containing V_0 and suppose that error indicators reveal that these elements may

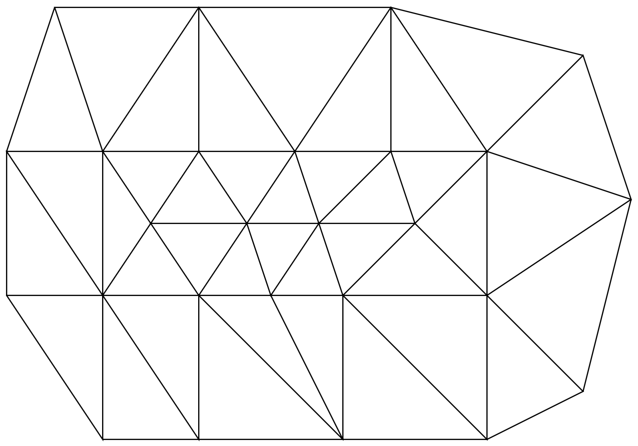


Figure 8.2.11: Coarsening of a quartet of elements shown with dashed lines in Figure 8.2.8 and the removal of surrounding elements to avoid irregular nodes.

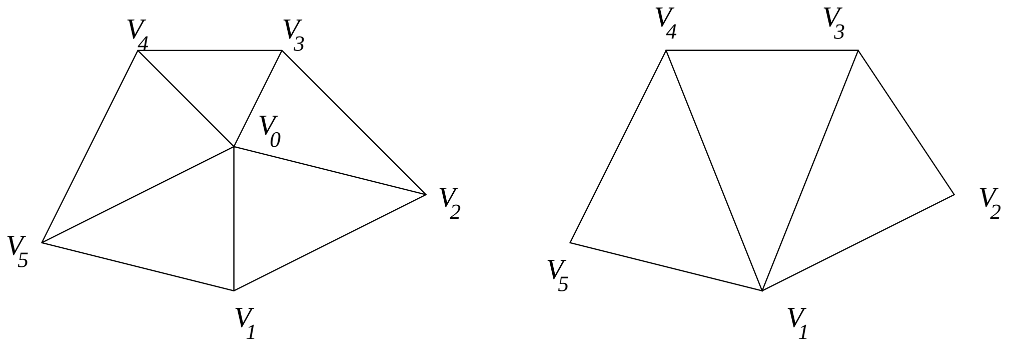


Figure 8.2.12: Coarsening of a polygonal region (left) by collapsing Vertex V_0 onto V_1 (right).

be coarsened. The strategy of collapsing V_0 onto one of the vertices V_j, $j = 1, 2, \ldots, k$, is done by deleting all edges connected to V_0 and then re-triangulating P by connecting V_j to the other vertices of P (*cf.* the right of Figure 8.2.12). Vertex V_0 is called the *collapsed vertex* and V_j is called the *target vertex*.

Collapsing has to be evaluated for topological compatibility and geometric validity before it is performed. Checking for geometric validity prevents situations like the one shown in Figure 8.2.13 from happening. A collapse is topologically incompatible when, *e.g.*, V_0 is on $\partial\Omega$ and the target vertex V_j is within Ω. Assuming that V_0 can be collapsed, the target vertex is chosen to be the one that maximizes the minimum angle of the resulting re-triangulation of P. Aiffa [2] does no collapsing when the smallest angle that would be produced by collapsing is smaller than a prescribed minimum angle. This might result in a mesh that is finer than needed for the specified accuracy. In this case, the minimum angle restriction could be waived when V_0 has been scheduled for coarsening more than a prescribed number of times. Suppose that the edges h_{1e}, h_{2e}, h_{3e} of an element e are indexed such that $h_{1e} \leq h_{2e} \leq h_{3e}$, then the smallest angle α_{1e} of Element e may be calculated as

$$\sin \alpha_{1e} = \frac{2A_e}{h_{2e}h_{3e}}$$

where A_e is the area of Element e.

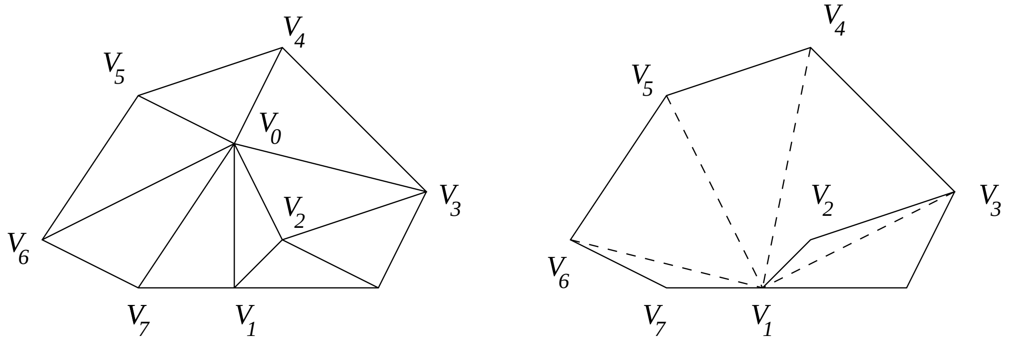


Figure 8.2.13: A situation where the collapse of Vertex V_0 (left) creates an invalid mesh (right).

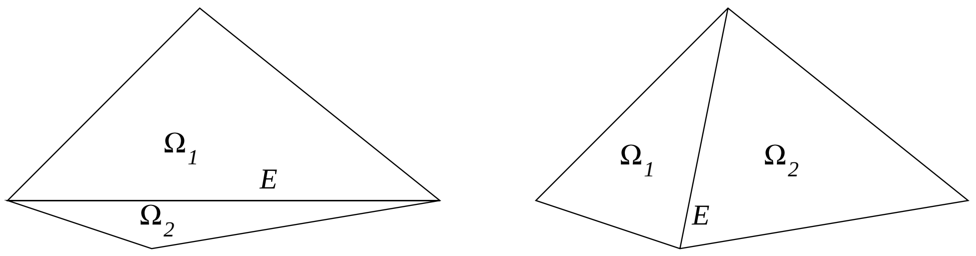


Figure 8.2.14: Swapping an edge of a pair of elements (left) to improve element shape (right).

The shape of elements containing small or large angles that were created during refinement or coarsening may be improved by *edge swapping*. This procedure operates on pairs of triangles Ω_1 and Ω_2 that share a common edge E. If $Q = \Omega_1 \cup \Omega_2$, edge swapping occurs deleting Edge E and re-triangulating Q by connecting the vertices opposite to Edge E (Figure 8.2.14). Swapping can be regarded as a refinement of Edge E followed by a collapsing of this new vertex onto a vertex not on Edge E. As such, we recognize that swapping will have to be checked for mesh validity and topological compatibility. Of course, it will also have to provide an improved mesh quality.

8.2.3 Refinement Criteria

Following the introductory discussion of error estimates in Section 8.1, we assume the existence of a set of refinement indicators ϵ_e, $e = 1, 2, \ldots, N_\Delta$, which are large where refinement is desired and small where coarsening is appropriate. As noted, these might be the restriction of a global error estimate to Element e

$$\epsilon_e^2 = \|E\|_e^2 \tag{8.2.2}$$

or an *ad hoc* refinement indicator such as the magnitude of the solution gradient on the element. In either case, how do we use this error information to refine the mesh. Perhaps the simplest approach is to refine a fixed percentage of elements having the largest error indicators, *i.e.*, refine all elements e satisfying

$$\epsilon_e \geq \lambda \max_{1 \leq j \leq N_\Delta} \epsilon_j. \tag{8.2.3}$$

A typical choice of the parameter $\lambda \in [0, 1]$ is 0.8.

We can be more precise when an error estimate of the form (8.1.1) with indicators given by (8.2.2) is available. Suppose that we have an *a priori* error estimate of the form

$$\|e\| \leq Ch^p. \tag{8.2.4a}$$

After obtaining an *a posteriori* error estimate $\|E\|$ on a mesh with spacing h, we could compute an estimate of the error constant C as

$$C \approx \frac{\|E\|}{h^p}. \tag{8.2.4b}$$

The mesh spacing parameter h may be taken as, *e.g.*, the average element size

$$h = \sqrt{\frac{A}{N_\Delta}} \tag{8.2.4c}$$

where A is the area of Ω.

Suppose that adaptivity is to be terminated when $\|E\| \approx \tau$ where τ is a prescribed tolerance. Using (8.2.4a), we would like to construct an enriched mesh with a spacing parameter $\tilde{h}$ such that

$$C\tilde{h}^p \approx \tau.$$

Using the estimate of C computed by (8.2.4b), we have

$$\frac{\tilde{h}}{h} \approx \left(\frac{\tau}{\|E\|}\right)^{1/p}. \tag{8.2.5a}$$

Thus, using (8.2.4c), an enriched mesh of

$$\tilde{N}_\Delta = \frac{\tilde{h}^2}{A} \approx \frac{h^2}{A}\left(\frac{\tau}{\|E\|}\right)^{2/p} \tag{8.2.5b}$$

elements will reduce $||E||$ to approximately τ.

Having selected an estimate of the number of elements $\tilde{N}_\Delta$ to be in the enriched mesh, we have to decide how to refine the current mesh in order to attain the prescribed tolerence. We may do this by *equidistributing* the error over the mesh. Thus, we would like each element of the enriched mesh to have approximately the same error. Using (8.1.1), this implies that

$$||\tilde{E}||_e^2 \approx \frac{\tau^2}{\tilde{N}_\Delta}$$

where $||\tilde{E}||_e$ is the error indicator of Element e of the enriched mesh. Using this notion, we divide the error estimate $||E||_e^2$ by a factor n so that

$$\frac{||E||_e^2}{n} \approx \frac{\tau^2}{\tilde{N}_\Delta}.$$

Thus, each element of the current mesh is divided into n segments such that

$$\frac{n}{\tilde{N}_\Delta} \approx \left(\frac{||E||_e}{\tau}\right)^2. \tag{8.2.6}$$

In practice, n and N_Δ may be rounded up or increased slightly to provide a measure of assurance that the error criterion will be satisfied after the next adaptive solution. The mesh division process may be implemented by repeated applications of a mesh-refinement algorithm without solving the partial differential equation in between. Thus, with bisection [34, 33], the elemental error estimate would be halved on each bisected element. Refinement would then be repeated until (8.2.6) is satisfied.

The error estimation process (8.2.6) works with coarsening when $n < 1$; however, neighboring elements would have to suggest coarsening as well.

Example 8.2.1 Rivara [33] solves Laplace's equation

$$u_{xx} + u_{yy} = 0, \qquad (x, y) \in \Omega,$$

where Ω is a regular hexagon inscribed in a unit circle. The hexagon is oriented with one vertex along the positive x-axis with a "crack" through this vertex for $0 \le x \le 1$, $y = 0$. Boundary conditions are established to be homogeneous Neumann conditions on the x-axis below the crack and

$$u(r, \theta) = r^{1/4} \sin\frac{\theta}{4}$$

everywhere else. This function is also the exact solution of the problem expressed in a polar frame eminating from the center of the hexagon. The solution has a singularity at the origin due to the "re-entrant" angle of 2π at the crack tip and the change in boundary conditions from Dirichlet to Neumann. The solution was computed with a piecewise-linear finite element basis using quasi-uniform and adaptive h-refinement. A residual error estimation procedure similar to those described in Section 7.4 was used to appraise solution accuracy [33]. Refinement followed (8.2.3).

The results shown in Figure 8.2.15 indicate that the uniform mesh is converging as $O(N^{-1/8})$ where N is the number of degrees of freedom. We have seen (Section 7.2) that uniform h-refinement converges as

$$\|e\|_1 \le C_1 h^{\min(p,q)} = C_2 N^{-\min(p,q)/2} \tag{8.2.7}$$

where $q > 0$ depends on the solution smoothness and, in two dimensions, $N \propto h^2$. For linear elliptic problems with geometric singularities, $q = \pi/\omega$ where ω is the maximum interior angle on $\partial\Omega$. For the hexagon with a crack, the interior angles would be $\pi/3$, $2\pi/3$, and 2π. The latter is the largest angle; hence, $q = 1/2$. Thus, with $p = 1$, convergence should occur at an $O(N^{-1/4})$ rate; however, the actual rate is lower (Figure 8.2.15).

The adaptive procedure has restored the $O(N^{-1/2})$ convergence rate that one would expect of a problem without singularities. In general, optimal adaptive h-refinement will converge as [6, 43]

$$\|e\|_1 \le C_1 h^p = C_2 N^{-p/2}. \tag{8.2.8}$$

8.3 *p*- and *hp*-Refinement

With p-refinement, the mesh is not changed but the order of the finite element basis is varied locally over the domain. As with h-refinement, we must ensure that the basis remains continuous at element boundaries. A situation where second- and fourth-degree hierarchical bases intersect along an edge between two square elements is shown on the left of Figure 8.3.1. The second-degree approximation (shown at the top left) consists of a bilinear shape function at each vertex and a second-degree correction on each edge. The fourth-degree approximation (bottom left) consists of bilinear shape functions at each vertex, second, third and fourth-degree corrections on each edge, and a fourth-degree bubble function associated with the centroid (*cf.* Section 4.4). The maximum degree of the polynomial associated with a mesh entity is identified on the figure. The second- and fourth-degree shape functions would be incompatible (discontinuous) across the common edge between the two elements. This situation can be corrected by constraining the edge functions to the lower-degree (two) basis of the top element as shown in the center

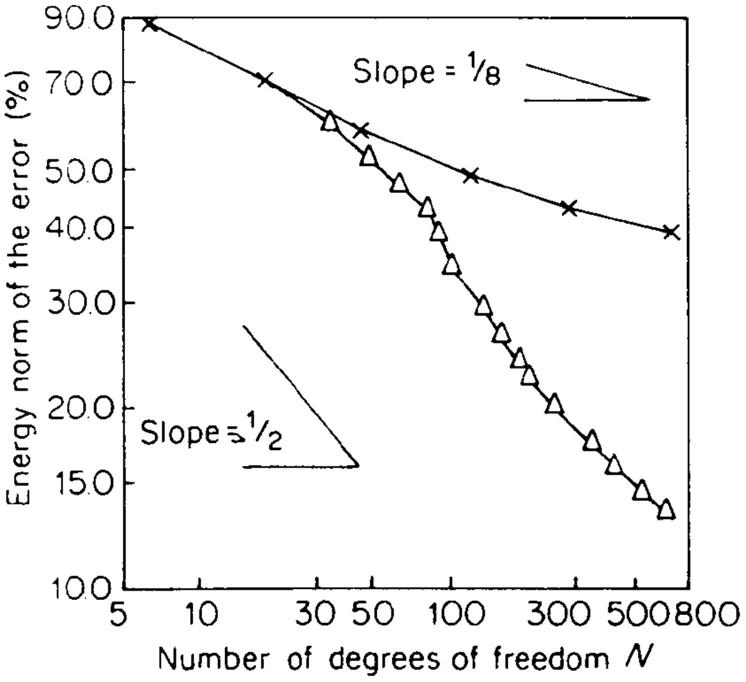


Figure 8.2.15: Solution of Example 8.2.1 by uniform ($\times$) and adaptive (Δ) h-refinement [33].

portion of the figure or by adding third- and fourth-order edge functions to the upper element as shown on the right of the figure. Of the two possibilities, the addition of the higher degree functions is the most popular. Constraining the space to the lower-degree polynomial could result in a situation where error criteria satisfied on the element on the lower left of Figure 8.3.1 would no longer be satisfied on the element in the lower-center portion of the figure.

Remark 1. The incompatibility problem just described would not arise with the hierarchical data structures defined in Section 5.3 since edge functions are blended onto all elements containing the edge and, hence, would always be continuous.

Szabó [39] developed a strategy for the adaptive variation of p by constructing error estimates of solutions with local degrees p, $p-1$, and $p-2$ on Element e and extrapolating to get an error estimates for solutions of higher degrees. With a hierarchical basis, this is straightforward when $p > 2$. One could just use the differences between higher- and lower-order solutions or an error estimation procedure as described in Section 7.4. When $p = 2$ on Element e, local error estimates of solutions having degrees 2 and 1 are linearly extrapolated. Szabo [39] began by generating piecewise-linear ($p = 1$) and piecewise-quadratic ($p = 2$) solutions everywhere and extrapolating the error estimates. Flaherty and Moore [20] suggest an alternative when $p = 1$. They obtain a "lower-order" piecewise

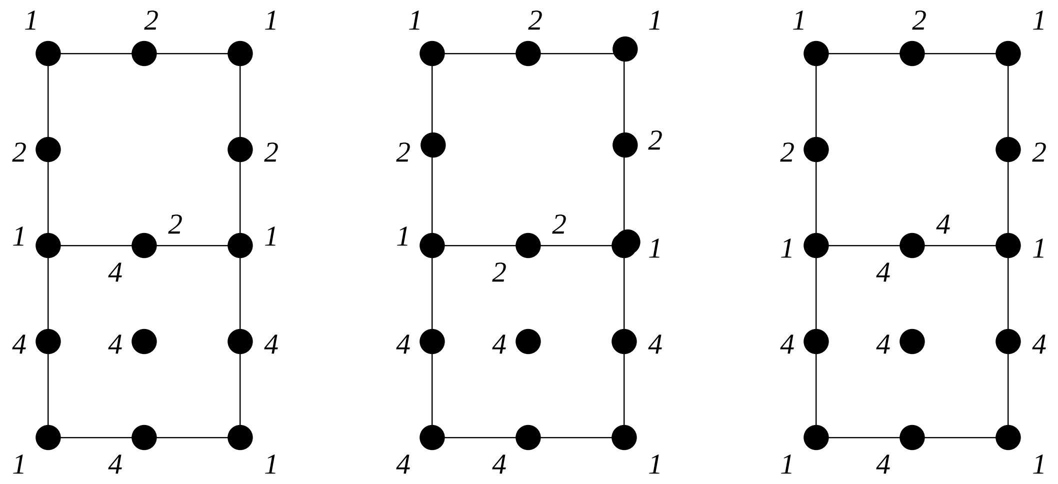


Figure 8.3.1: Second- and fourth-degree hierarchical shape functions on two square elements are incompatible across the common edge between elements (left). This can be corrected by removing the third- and fourth-degree edge functions from the lower element (center) or by adding third- and fourth-degree edge functions to the upper element (right). The maximum degree of the shape function associated with a mesh entity is shown in each case.

constant ($p = 0$) solution by using the value of the piecewise-linear solution at the center of Element e. The difference between these two "solutions" furnishes an error estimate which, when used with the error estimate for the piecewise-linear solution, is linearly extrapolated to higher values of p.

Having estimates of discretization errors as a function of p on each element, we can use a strategy similar to (8.2.6) to select a value of p to reduce the error on an element to its desired level. Often, however, a simpler strategy is used. As indicated earlier, the error estimate $\|E\|_e$ should be of size τ/N_Δ on each element of the mesh. When enrichment is indicated, *e.g.*, when $\|E\| > \tau$, we can increase the degree of the polynomial representation by one on any element e where

$$\epsilon_e > \lambda_R \frac{\tau}{N_\Delta}. \tag{8.3.1a}$$

The parameter ϵ_e is an enrichment indicator on Element e, which may be $\|E\|_e$, and $\lambda_R \approx 1.1$. If coarsening is done, the degree of the approximation on Element e can be reduced by one when

$$\epsilon_e < \lambda_C h_e \frac{\tau}{N_\Delta} \tag{8.3.1b}$$

where h_e is the longest edge of Element e and $\lambda_C \approx 0.1$.

The convergence rate of the p version of the finite element method is exponential when the solution has no singularities. For problems with singularities, p-refinement converges as

$$\|e\| \leq CN^{-q} \tag{8.3.2}$$

where $q > 0$ depends on the solution smoothness [22, 23, 24, 25, 26]. (The parameter q is intended to be generic and is not necessarily the same as the one appearing in (8.2.7)). With singularities, the performance of the p version of the finite element method depends on the mesh. Performance will be better when the mesh has been graded near the singularity.

This suggests combining h- and p-refinement. Indeed, when proper mesh refinement is combined with an increase of the polynomial degree p, the convergence rate is exponential

$$\|e\| \leq Ce^{-q_1 N^{q_2}} \tag{8.3.3}$$

where q_1 and q_2 are positive constants that depend on the smoothness of the exact solution and the finite element mesh. Generating the correct mesh is crucial and its construction is only known for model problems [22, 23, 24, 25, 26]. Oden *et al.* [30] developed a strategy for hp-refinement that involved taking three solution steps followed by an extrapolation. Some techniques do not attempt to adjust the mesh and the order at the same time, but, rather, adjust either the mesh or the order. We'll illustrate one of these, but first cite the more explicit version of the error estimate (8.2.7) given by Babuška and Suri [7]

$$\|e\|_1 \leq C\frac{h^{\min(p,q)}}{p^q}\|u\|_{\min(p,q)+1}. \tag{8.3.4}$$

The mesh must satisfy the uniformity condition, the polynomial-degree is uniform, and $u \in H^{q+1}$. In this form, the constant C is independent of h and p. This result and the previous estimates indicate that it is better to increase the polynomial degree when the solution u is smooth (q is large) and to reduce h near singularities. Thus, a possible strategy would be to increase p in smooth high-error regions and refine the mesh near singularities. We, therefore, need a method of estimating solution smoothness and Aiffa [2] does this by computing the ratio

$$\rho_e = \begin{cases} \epsilon_e(p)/\epsilon_e(p-1), & \text{if } \epsilon_e(p-1) \neq 0 \\ 0, & \text{otherwise} \end{cases} \tag{8.3.5}$$

where p is the polynomial degree on Element e. An argument has been added to the error indicator on Element e to emphasize its dependence on the local polynomial degree. As described in Section 8.2, $\epsilon(p-1)$ can be estimated from the part of U involving the hierarchal corrections of degree p. Now

- If $\rho_e < 1$, the error estimate is decreasing with increasing polynomial degree. If enrichment were indicated on Element e, p-refinement would be the preferred strategy.

- If $\rho_e \geq 1$ the recommended strategy would be h-refinement.

Aiffa [2] selects p-refinement if $\rho_e \leq \gamma$ and h-refinement if $\rho_e > \gamma$, with $\gamma \approx 0.6$. Adjustments have to made when $p = 1$ [2]. Coarsening is done by vertex collapsing when all elements surrounding a vertex have low errors [2].

Example 8.3.1 Aiffa [2] solves the nonlinear parabolic partial differential equation

$$u_t - \sigma u^2(1-u) = \frac{u_{xx} + u_{yy}}{2}, \qquad (x,y) \in \Omega, \qquad t > 0,$$

with the initial and Dirichlet boundary data defined so that the exact solution on the square $\Omega = \{(x,y)|0 < x, y < 2\}$ is

$$u(x,y,t) = \frac{1}{1 + e^{\sqrt{\sigma/2}(x+y-t\sqrt{\sigma/2})}}$$

Although this problem is parabolic, Aiffa [2] kept the temporal error small so that spatial errors dominate.

Aiffa [2] solved this problem with $\sigma = 500$ by adaptive h-, p-, and hp-refinement for a variety of spatial error tolerances. The initial mesh for h-refinement contained 32 triangular elements and used piecewise-quadratic ($p = 2$) shape functions. For p-refinement, the mesh contained 64 triangles with p varying from 1 to 5. The solution with adaptive hp-refinement was initiated with 32 elements and $p = 1$, The convergence history of the three adaptive strategies is reported in Figure 8.3.2.

The solution with h-refinement appears to be converging at an algebraic rate of approximately $N^{-0.95}$, which is close to the theoretical rate (*cf.* (8.2.7)). There are no singularities in this problem and the adaptive p- and hp-refinement methods appear to be converging at exponential rates.

This example and the material in this chapter give an introduction to the essential ideas of adaptivity and adaptive finite element analysis. At this time, adaptive software is emerging. Robust and reliable error estimation procedures are only known for model problems. Optimal enrichment strategies are just being discovered for realistic problems.

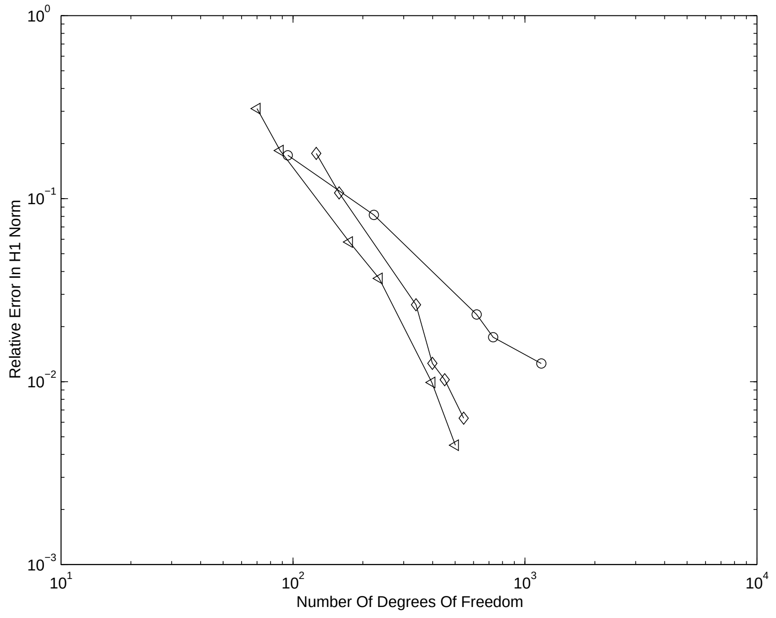


Figure 8.3.2: Errors vs. the number of degrees of freedom N for Example 8.3.1 at $t = 0.05$ using adaptive h-, p- and hp-refinement ($\circ$, $\diamond$, and $\triangleright$, respectively).

Bibliography

[1] S. Adjerid and J.E. Flaherty. A local refinement finite element method for two-dimensional parabolic systems. *SIAM Journal on Scientific and Statistical Computing*, 9:792–811, 1988.

[2] M. Aiffa. *Adaptive hp-Refinement Methods for Singularly-Perturbed Elliptic and Parabolic Systems*. PhD thesis, Rensselaer Polytechnic Institute, Troy, 1997.

[3] D.C. Arney and J.E. Flaherty. An adaptive mesh moving and local refinement method for time-dependent partial differential equations. *ACM Transactions on Mathematical Software*, 16:48–71, 1990.

[4] I. Babuška, J. Chandra, and J.E. Flaherty, editors. *Adaptive Computational Methods for Partial Differential Equations*, Philadelphia, 1983. SIAM.

[5] I. Babuška, J.E. Flaherty, W.D. Henshaw, J.E. Hopcroft, J.E. Oliger, and T. Tezduyar, editors. *Modeling, Mesh Generation, and Adaptive Numerical Methods for Partial Differential Equations*, volume 75 of *The IMA Volumes in Mathematics and its Applications*, New York, 1995. Springer-Verlag.

[6] I. Babuška, A. Miller, and M. Vogelius. Adaptive methods and error estimation for elliptic problems of structural mechanics. In I. Babuška, J. Chandra, and J.E. Flaherty, editors, *Adaptive Computational Methods for Partial Differential Equations*, pages 57–73, Philadelphia, 1983. SIAM.

[7] I. Babuška and Suri. The optimal convergence rate of the p-version of the finite element method. *SIAM Journal on Numerical Analysis*, 24:750–776, 1987.

[8] I. Babuška, O.C. Zienkiewicz, J. Gago, and E.R. de A. Oliveira, editors. *Accuracy Estimates and Adaptive Refinements in Finite Element Computations*. John Wiley and Sons, Chichester, 1986.

[9] R.E. Bank. The efficient implementation of local mesh refinement algorithms. In I. Babuška, J. Chandra, and J.E. Flaherty, editors, *Adaptive Computational Methods for Partial Differential Equations*, pages 74–81, Philadelphia, 1983. SIAM.

[10] R.E. Bank. *PLTMG: A Software Package for Solving Elliptic Partial Differential Equations. Users' Guide 7.0*, volume 15 of *Frontiers in Applied Mathematics.* SIAM, Philadelphia, 1994.

[11] R.E. Bank, A.H. Sherman, and A. Weiser. Refinement algorithms and data structures for regular local mesh refinement. In *Scientific Computing*, pages 3–17, Brussels, 1983. IMACS/North Holland.

[12] M.J. Berger and J. Oliger. Adaptive mesh refinement for hyperbolic partial differential equations. *Journal of Computational Physics*, 53:484–512, 1984.

[13] M.W. Bern, J.E. Flaherty, and M. Luskin, editors. *Grid Generation and Adaptive Algorithms*, volume 113 of *The IMA Volumes in Mathematics and its Applications*, New York, 1999. Springer.

[14] R. Biswas, K.D. Devine, and J.E. Flaherty. Parallel adaptive finite element methods for conservation laws. *Applied Numerical Mathematics*, 14:255–284, 1994.

[15] K. Clark, J.E. Flaherty, and M.S. Shephard, editors. *Applied Numerical Mathematics*, volume 14, 1994. Special Issue on Adaptive Methods for Partial Differential Equations.

[16] K. Devine and J.E. Flaherty. Parallel adaptive hp-refinement techniques for conservation laws. *Applied Numerical Mathematics*, 20:367–386, 1996.

[17] M. Dindar, M.S. Shephard, J.E. Flaherty, and K. Jansen. Adaptive cfd analysis for rotorcraft aerodynamics. Computer Methods in Applied Mechanics Engineering, submitted, 1999.

[18] D.B. Duncan, editor. *Applied Numerical Mathematics*, volume 26, 1998. Special Issue on Grid Adaptation in Computational PDEs: Theory and Applications.

[19] J.E. Flaherty, R. Loy, M.S. Shephard, B.K. Szymanski, J. Teresco, and L. Ziantz. Adaptive local refinement with octree load-balancing for the parallel solution of three-dimensional conservation laws. *Parallel and Distributed Computing*, 1998. to appear.

[20] J.E. Flaherty and P.K. Moore. Integrated space-time adaptive hp-refinement methods for parabolic systems. *Applied Numerical Mathematics*, 16:317–341, 1995.

[21] J.E. Flaherty, P.J. Paslow, M.S. Shephard, and J.D. Vasilakis, editors. *Adaptive methods for Partial Differential Equations*, Philadelphia, 1989. SIAM.

[22] W. Gui and I. Babuška. The h, p, and h-p version of the finite element method in 1 dimension. Part 1: The error analysis of the p-version. *Numerische Mathematik*, 48:557–612, 1986.

[23] W. Gui and I. Babuška. The h, p, and h-p version of the finite element method in 1 dimension. Part 2: The error analysis of the h- and h-p-version. *Numerische Mathematik*, 48:613–657, 1986.

[24] W. Gui and I. Babuška. The h, p, and h-p version of the finite element method in 1 dimension. Part 3: The adaptive h-p-version. *Numerische Mathematik*, 48:658–683, 1986.

[25] W. Guo and I. Babuška. The h-p version of the finite element method. Part 1: The basic approximation results. *Computational Mechanics*, 1:1–20, 1986.

[26] W. Guo and I. Babuška. The h-p version of the finite element method. Part 2: General results and applications. *Computational Mechanics*, 1:21–41, 1986.

[27] C. Mesztenyi and W. Szymczak. FEARS user's manual for UNIVAC 1100. Technical Report Note BN-991, Institute for Physical Science and Technology, University of Maryland, College Park, 1982.

[28] W.R. Mitchell. *Unified Multilevel Adaptive Finite Element Methods for Elliptic Problems*. PhD thesis, University of Illinois at Urbana-Champagne, Urbana, 1988.

[29] P.K. Moore and J.E. Flaherty. Adaptive local overlapping grid methods for parabolic systems in two space dimensions. *Journal of Computational Physics*, 98:54–63, 1992.

[30] J.T. Oden, W. Wu, and M. Ainsworth. Three-step h-p adaptive strategy for the incompressible Navier-Stokes equations. In I. Babuška, J.E. Flaherty, W.D. Henshaw, J.E. Hopcroft, J.E. Oliger, and T. Tezduyar, editors, *Modeling, Mesh Generation, and Adaptive Numerical Methods for Partial Differential Equations*, volume 75 of *The IMA Volumes in Mathematics and its Applications*, pages 347–366, New York, 1995. Springer-Verlag.

[31] W. Rachowicz, J.T. Oden, and L. Demkowicz. Toward a universal h-p adaptive finite element strategy, Part 3, design of h-p meshes. *Computer Methods in Applied Mechanics and Engineering*, 77:181–212, 1989.

[32] E. Rank and I. Babuška. An expert system for the optimal mesh design in the hp-version of the finite element method. *International Journal of Numerical methods in Engineering*, 24:2087–2106, 1987.

[33] M.C. Rivara. Design and data structures of a fully adaptive multigrid finite element software. *ACM Transactions on Mathematical Software*, 10:242–264, 1984.

[34] M.C. Rivara. Mesh refinement processes based on the generalized bisection of simplices. *SIAM Journal on Numerical Analysis*, 21:604–613, 1984.

[35] I.G. Rosenberg and F. Stenger. A lower bound on the angles of triangles constructed by bisecting the longest side. *Mathematics of Computation*, 29:390–395, 1975.

[36] C. Schwab. *P- And Hp- Finite Element Methods: Theory and Applications in Solid and Fluid Mechanics.* Numerical Mathematics and Scientific Computation. Clarendon, London, 1999.

[37] E.G. Sewell. *Automatic Generation of Triangulations for Piecewise Polynomial Approximation.* PhD thesis, Purdue University, West Lafayette, 1972.

[38] M.S. Shephard, J.E. Flaherty, C.L. Bottasso H.L. de Cougny, and C. Özturan. Parallel automatic mesh generation and adaptive mesh control. In M. Papadrakakis, editor, *Solving Large Scale Problems in Mechanics: Parallel and Distributed Computer Applications*, pages 459–493, Chichester, 1997. John Wiley and Sons.

[39] B. Szabó. Mesh design for the p-version of the finite element method. *Computer Methods in Applied Mechanics and Engineering*, 55:181–197, 1986.

[40] B. Szabó and I. Babuška. *Finite Element Analysis.* John Wiley and Sons, New York, 1991.

[41] R. Verfürth. *A Review of Posteriori Error Estimation and Adaptive Mesh-Refinement Techniques.* Teubner-Wiley, Stuttgart, 1996.

[42] H. Zhang, M.K. Moallemi, and V. Prasad. A numerical algorithm using multizone grid generation for multiphase transport processes with moving and free boundaries. *Numerical Heat Transfer*, B29:399–421, 1996.

[43] O.C. Zienkiewicz and J.Z. Zhu. Adaptive techniques in the finite element method. *Communications in Applied Numerical Methods*, 4:197–204, 1988.

Chapter 9

Parabolic Problems

9.1 Introduction

The finite element method may be used to solve time-dependent problems as well as steady ones. This effort involves both parabolic and hyperbolic partial differential systems. Problems of parabolic type involve diffusion and dissipation while hyperbolic problems are characterized by conservation of energy and wave propagation. Simple one-dimensional heat conduction and wave propagation equations will serve as model problems of each type.

Example 9.1.1. The one-dimensional heat conduction equation

$$u_t = pu_{xx}, \qquad a < x < b, \qquad t > 0, \tag{9.1.1a}$$

where p is a positive constant called the diffusivity, is of parabolic type. Initial-boundary value problems consist of determining $u(x,t)$ satisfying (9.1.1a) given the initial data

$$u(x,0) = u^0(x), \qquad a \leq x \leq b, \tag{9.1.1b}$$

and appropriate boundary data, *e.g.*,

$$pu_x(a,t) + \gamma_0 u(a,t) = \beta_0(t), \qquad pu_x(b,t) + \gamma_1 u(b,t) = \beta_1(t). \tag{9.1.1c}$$

As with elliptic problems, boundary conditions without the pu_x term are called Dirichlet conditions; those with $\gamma_i = 0$, $i = 0, 1$, are Neumann conditions; and those with both terms present are called Robin conditions. The problem domain is open in the time direction t; thus, unlike elliptic systems, this problem is evolutionary and computation continues in t for as long as there is interest in the solution.

Example 9.1.2. The one-dimensional wave equation

$$u_{tt} = c^2 u_{xx}, \qquad a < x < b, \qquad t > 0, \tag{9.1.2a}$$

where c is a constant called the wave speed, is a hyperbolic partial differential equation. Initial-boundary value problems consist of determining $u(x,t)$ satisfying (9.1.2a) given the initial data

$$u(x,0) = u^0(x), \qquad u_t(x,0) = \dot{u}^0(x), \qquad a \leq x \leq b, \tag{9.1.2b}$$

and boundary data of the form (9.1.1c). Small transverse vibrations of a taut string satisfy the wave equation. In this case, $u(x,t)$ is the transverse displacement of the string and $c^2 = T/\rho$, T being the applied tension and ρ being the density of the string.

We'll study parabolic problems in this chapter and hyperbolic problems in the next. We shall see that there are two basic finite element approaches to solving time-dependent problems. The first, called the *method of lines*, uses finite elements in space and ordinary differential equations software in time. The second uses finite element methods in both space and time. We'll examine the method of lines approach first and then tackle space-time finite element methods.

9.2 Semi-Discrete Galerkin Problems: The Method of Lines

Let us consider a parabolic problem of the form

$$u_t + \mathcal{L}[u] = f(x,y), \qquad (x,y) \in \Omega, \qquad t > 0, \tag{9.2.1a}$$

where $\mathcal{L}$ is a second-order elliptic operator. In two dimensions, u would be a function of x, y, and t and $\mathcal{L}[u]$ could be the very familiar

$$\mathcal{L}[u] = -(pu_x)_x - (pu_y)_y + qu. \tag{9.2.1b}$$

Appropriate initial and boundary conditions would also be needed, *e.g.*,

$$u(x,y,0) = u^0(x,y), \qquad (x,y) \in \Omega \cup \partial\Omega, \tag{9.2.1c}$$

$$u(x,y,t) = \alpha(x,y,t), \qquad (x,y) \in \partial\Omega_E, \tag{9.2.1d}$$

$$pu_{\mathbf{n}} + \gamma u = \beta, \qquad (x,y) \in \partial\Omega_N. \tag{9.2.1e}$$

We construct a Galerkin formulation of (9.2.1) in space in the usual manner; thus, we multiply (9.2.1a) by a suitable test function v and integrate the result over Ω to obtain

$$(v, u_t) + (v, \mathcal{L}[u]) = (v, f).$$

As usual, we apply the divergence theorem to the second-derivative terms in $\mathcal{L}$ to reduce the continuity requirements on u. When $\mathcal{L}$ has the form of (9.2.1b), the Galerkin problem consists of determining $u \in H^1_E \times (t > 0)$ such that

$$(v, u_t) + A(v, u) = (v, f) + < v, \beta - \gamma u >, \qquad \forall v \in H^1_0, \qquad t > 0. \tag{9.2.2a}$$

The L^2 inner product, strain energy, and boundary inner product are, as with elliptic problems,

$$(v, f) = \iint_\Omega v f dx dy, \tag{9.2.2b}$$

$$A(v, u) = \iint_\Omega [p(v_x u_x + v_y u_y) + vqu] dx dy, \tag{9.2.2c}$$

and

$$< v, pu_{\mathbf{n}} > = \int_{\partial\Omega_N} v p u_{\mathbf{n}} ds. \tag{9.2.2d}$$

The natural boundary condition (9.2.1e) has been used to replace $pu_{\mathbf{n}}$ in the boundary inner product. Except for the presence of the (v, u_t) term, the formulation appears to the same as for an elliptic problem.

Initial conditions for (9.2.2a) are usually determined by projection of the initial data (9.2.1c) either in L^2

$$(v, u) = (v, u^0), \qquad \forall v \in H^1_0, \qquad t = 0, \tag{9.2.3a}$$

or in strain energy

$$A(v, u) = A(v, u^0), \qquad \forall v \in H^1_0, \qquad t = 0. \tag{9.2.3b}$$

Example 9.2.1. We analyze the one-dimensional heat conduction problem

$$u_t = (pu_x)_x + f(x, t), \qquad 0 < x < 1, \qquad t > 0,$$

$$u(x, 0) = u^0(x), \qquad 0 \leq x \leq 1,$$

$$u(0, t) = u(1, t) = 0, \qquad t > 0,$$

thoroughly in the spirit that we did in Chapter 1 for a two-point boundary value problem.

A Galerkin form of this heat-conduction problem consists of determining $u \in H^1_0$ satisfying

$$(v, u_t) + A(v, u) = (v, f), \qquad \forall v \in H^1_0, \qquad t > 0,$$

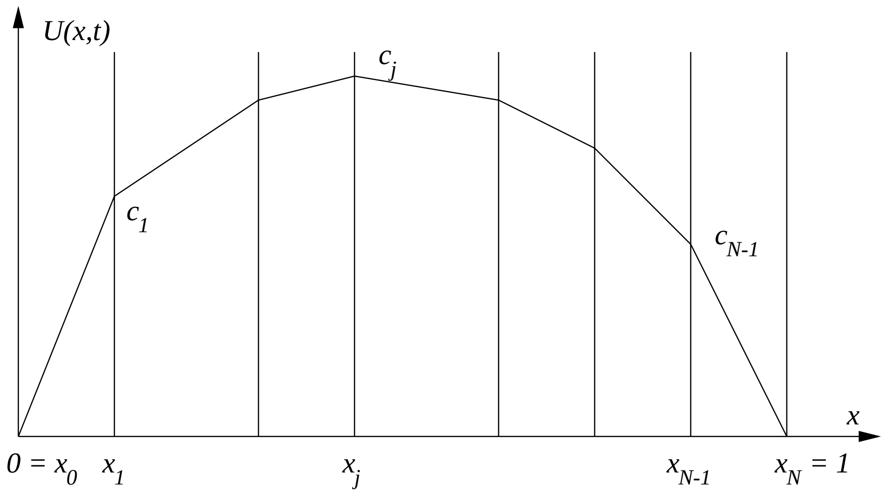


Figure 9.2.1: Mesh for the finite element solution of Example 9.2.1.

$$(v, u) = (v, u^0), \qquad \forall v \in H_0^1, \qquad t = 0,$$

where

$$A(v, u) = \int_0^1 v_x p u_x dx.$$

Boundary terms of the form (9.2.2d) disappear because $v = 0$ at $x = 0, 1$ with Dirichlet data.

We introduce a mesh on $0 \leq x \leq 1$ as shown in Figure 9.2.1 and choose an approximation U of u in a finite-dimensional subspace S_0^N of H_0^1 having the form

$$U(x, t) = \sum_{j=1}^{N-1} c_j(t) \phi_j(x).$$

Unlike steady problems, the coefficients c_j, $j = 1, 2, \ldots, N-1$, depend on t. The Galerkin finite element problem is to determine $U \in S_0^N$ such that

$$(\phi_j, U_t) + A(\phi_j, U) = (\phi_j, f), \qquad t > 0,$$

$$(\phi_j, U) = (\phi_j, u^0), \qquad t = 0, \qquad j = 1, 2, \ldots, N-1.$$

Let us chose a piecewise-linear polynomial basis

$$\phi_k(x) = \begin{cases} \frac{x - x_{k-1}}{x_k - x_{k-1}}, & \text{if } x_{k-1} < x \leq x_k \\ \frac{x_{k+1} - x}{x_{k+1} - x_k}, & \text{if } x_k < x \leq x_{k+1} \\ 0, & \text{otherwise} \end{cases}.$$

This problem is very similar to the one-dimensional elliptic problem considered in Section 1.3, so we'll skip several steps and also construct the discrete equations by vertices rather than by elements.

Since ϕ_j has support on the two elements containing node j we have

$$A(\phi_j, U) = \int_{x_{j-1}}^{x_j} \phi_j' p U_x dx + \int_{x_j}^{x_{j+1}} \phi_j' p U_x dx$$

where $(\ )' = d(\ )/dx$. Substituting for ϕ_j and U_x

$$A(\phi_j, U) = \int_{x_{j-1}}^{x_j} \frac{1}{h_j} p(x)(\frac{c_j - c_{j-1}}{h_j})dx + \int_{x_j}^{x_{j+1}} -\frac{1}{h_{j+1}} p(x)(\frac{c_{j+1} - c_j}{h_{j+1}})dx$$

where

$$h_j = x_j - x_{j-1}.$$

Using the midpoint rule to evaluate the integrals, we have

$$A(\phi_j, U) \approx \frac{p_{j-1/2}}{h_j}(c_j - c_{j-1}) - \frac{p_{j+1/2}}{h_{j+1}}(c_{j+1} - c_j)$$

where $p_{j-1/2} = p(x_{j-1/2})$.

Similarly,

$$(\phi_j, U_t) = \int_{x_{j-1}}^{x_j} \phi_j U_t dx + \int_{x_j}^{x_{j+1}} \phi_j U_t dx$$

or

$$(\phi_j, U_t) = \int_{x_{j-1}}^{x_j} \phi_j(\dot{c}_{j-1}\phi_{j-1} + \dot{c}_j\phi_j)dx + \int_{x_j}^{x_{j+1}} \phi_j(\dot{c}_j\phi_j + \dot{c}_{j+1}\phi_{j+1})dx$$

where $(\dot{\ }) = d(\ )/dt$. Since the integrands are quadratic functions of x they may be integrated exactly using Simpson's rule to yield

$$(\phi_j, U_t) = \frac{h_j}{6}(\dot{c}_{j-1} + 2\dot{c}_j) + \frac{h_{j+1}}{6}(2\dot{c}_j + \dot{c}_{j+1}).$$

Finally,

$$(\phi_j, f) \approx \int_{x_{j-1}}^{x_j} \phi_j f(x)dx + \int_{x_j}^{x_{j+1}} \phi_j f(x)dx.$$

Although integration of order one would do, we'll, once again, use Simpson's rule to obtain

$$(\phi_j, f) \approx \frac{h_j}{6}(2f_{j-1/2} + f_j) + \frac{h_{j+1}}{6}(f_j + 2f_{j+1/2}).$$

We could replace $f_{j-1/2}$ by the average of f_{j-1} and f_j to obtain a similar formula to the one obtained for (ϕ_j, U_t); thus,

$$(\phi_j, f) \approx \frac{h_j}{6}(f_{j-1} + 2f_j) + \frac{h_{j+1}}{6}(2f_j + f_{j+1}).$$

Combining these results yields the discrete finite element system

$$\frac{h_j}{6}(\dot{c}_{j-1} + 2\dot{c}_j) + \frac{h_{j+1}}{6}(2\dot{c}_j + \dot{c}_{j+1}) + \frac{p_{j-1/2}}{h_j}(c_j - c_{j-1}) - \frac{p_{j+1/2}}{h_j + 1/2}(c_{j+1} - c_j)$$

$$= \frac{h_j}{6}(f_{j-1} + 2f_j) + \frac{h_{j+1}}{6}(2f_j + f_{j+1}), \qquad j = 1, 2, \ldots, N-1.$$

(We have dropped the $\approx$ and written the equation as an equality.)

If p is constant and the mesh spacing h is uniform, we obtain

$$\begin{aligned} \frac{h}{6}(\dot{c}_{j-1} + 4\dot{c}_j + \dot{c}_{j+1}) - \frac{p}{h}(c_{j-1} - 2c_j + c_{j+1}) &= \frac{h}{6}(f_{j-1} + 4f_j + f_{j+1}), \\ & \quad j = 1, 2, \ldots, N-1. \end{aligned}$$

The discrete systems may be written in matrix form and, for simplicity, we'll do so for the constant coefficient, uniform mesh case to obtain

$$\mathbf{M}\dot{\mathbf{c}} + \mathbf{K}\mathbf{c} = \mathbf{l} \tag{9.2.4a}$$

where

$$\mathbf{M} = \frac{h}{6}\begin{bmatrix} 4 & 1 & & & \\ 1 & 4 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & 4 & 1 \\ & & & 1 & 4 \end{bmatrix}, \tag{9.2.4b}$$

$$\mathbf{K} = \frac{p}{h}\begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{bmatrix}, \tag{9.2.4c}$$

$$\mathbf{l} = \frac{h}{6}\begin{bmatrix} f_0 + 4f_1 + f_2 \\ f_1 + 4f_2 + f_3 \\ \vdots \\ f_{N-2} + 4f_{N-1} + f_N \end{bmatrix}, \tag{9.2.4d}$$

$$\mathbf{c} = [c_1, c_2, \ldots, c_{N-1}]^T. \tag{9.2.4e}$$

The matrices **M**, **K**, and **l** are the global mass matrix, the global stiffness matrix, and the global load vector. Actually, **M** has little to do with mass and should more correctly be called a global dissipation matrix; however, we'll stay with our prior terminology. In practical problems, element-by-element assembly should be used to construct global matrices and vectors and not the nodal approach used here.

The discrete finite element system (9.2.4) is an implicit system of ordinary differential equations for $\dot{\mathbf{c}}$. The mass matrix **M** can be "lumped" by a variety of tricks to yield an explicit ordinary differential system. One such trick is to approximate (ϕ_j, U_t) by using the right-rectangular rule on each element to obtain

$$(\phi_j, U_t) = \int_{x_{j-1}}^{x_j} \phi_j(\dot{c}_{j-1}\phi_{j-1} + \dot{c}_j\phi_j)dx + \int_{x_j}^{x_{j+1}} \phi_j(\dot{c}_j\phi_j + \dot{c}_{j+1}\phi_{j+1})dx \approx hc_j.$$

The resulting finite element system would be

$$h\mathbf{I}\dot{\mathbf{c}} + \mathbf{K}\mathbf{c} = \mathbf{l}.$$

Recall (*cf.* Section 6.3), that a one-point quadrature rule is satisfactory for the convergence of a piecewise-linear polynomial finite element solution.

With the initial data determined by L^2 projection onto S_E^N, we have

$$(\phi_j, U(\cdot, 0)) = (\phi_j, u^0), \qquad j = 1, 2, \ldots, N-1.$$

Numerical integration will typically be needed to evaluate (ϕ_j, u^0) and we'll approximate it in the manner used for the loading term (ϕ_j, f). Thus, with uniform spacing, we have

$$\mathbf{M}\mathbf{c}(0) = \mathbf{u}^0 = \frac{h}{6}\begin{bmatrix} u_0^0 + 4u_1^0 + u_2^0 \\ u_1^0 + 4u_2^0 + u_3^0 \\ \vdots \\ u_{N-2}^0 + 4u_{N-1}^0 + u_N^0 \end{bmatrix}. \tag{9.2.4f}$$

If the initial data is consistent with the trivial Dirichlet boundary data, *i.e.*, if $u^0 \in H_0^1$ then the above system reduces to

$$c_j(0) = u^0(x_j), \qquad j = 1, 2, 3, \ldots, N-1.$$

Had we solved the wave equation (9.1.2) instead of the heat equation (9.1.1) using a piecewise-linear finite element basis, we would have found the discrete system

$$\mathbf{M}\ddot{\mathbf{c}} + \mathbf{K}\mathbf{c} = \mathbf{0} \tag{9.2.5}$$

with p in (9.2.4c) replaced by c^2.

The resulting initial value problems (IVPs) for the ordinary differential equations (ODEs) (9.2.4a) or (9.2.5) typically have to be integrated numerically. There are several excellent software packages for solving IVPs for ODEs. When such ODE software is used with a finite element or finite difference spatial discretization, the resulting procedure is called the *method of lines.* Thus, the nodes of the finite elements appear to be "lines" in the time direction and, as shown in Figure 9.2.2 for a one-dimensional problem, the temporal integration proceeds along these lines.

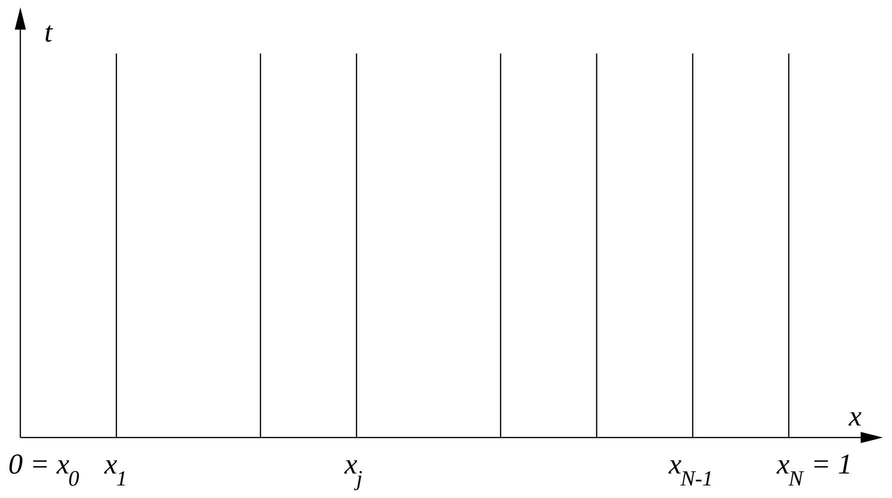


Figure 9.2.2: “Lines” for a method of lines integration of a one-dimensional problem.

Using the ODE software, solutions are calculated in a series of time steps $(0, t_1]$, $(t_1, t_2]$, $\ldots$. Methods fall into two types. Those that only require knowledge of the solution at time t_n in order to obtain a solution at time t_{n+1} are called *one-step methods*. Correspondingly, methods that require information about the solution at t_n and several times prior to t_n are called *multistep methods*. Excellent texts on the subject are available [2, 6, 7, 8]. One-step methods are Runge-Kutta methods while the common multistep methods are Adams or backward difference methods. Software based on these methods automatically adjusts the time steps and may also automatically vary the order of accuracy of a class of methods in order to satisfy a prescribed local error tolerance, minimize computational cost, and maintain numerical efficiency.

The choice of a one-step or multistep method will depend on several factors. Generally, Runge-Kutta methods are preferred when time integration is simple relative to the spatial solution. Multistep methods become more efficient for complex nonlinear problems. Implicit Runge-Kutta methods may be efficient for problems with high-frequency oscillations. The ODEs that arise from the finite element discretization of parabolic problems are “stiff” [2, 8] so backward difference methods are the preferred multistep methods.

Most ODE software [2, 7, 8] addresses first-order IVPs of the explicit form

$$\dot{\mathbf{y}}(t) = \mathbf{f}(t, \mathbf{y}(t)), \qquad \mathbf{y}(0) = \mathbf{y}^0. \tag{9.2.6}$$

Second-order systems such as (9.2.5) would have to be written as a first-order system by, *e.g.*, letting

$$\mathbf{d} = \dot{\mathbf{c}}$$

and, hence, obtaining

$$\begin{bmatrix} \dot{\mathbf{c}} \\ \mathbf{M}\dot{\mathbf{d}} \end{bmatrix} = \begin{bmatrix} \mathbf{d} \\ -\mathbf{K}\mathbf{c} \end{bmatrix}.$$

Unfortunately, systems having the form of (9.2.4a) or the one above are implicit and would require inverting or lumping $\mathbf{M}$ in order to put them into the standard explicit form (9.2.6). Inverting $\mathbf{M}$ is not terribly difficult when $\mathbf{M}$ is constant or independent of t; however, it would be inefficient for nonlinear problems and impossible when $\mathbf{M}$ is singular. The latter case can occur when, *e.g.*, a heat conduction and a potential problem are solved simultaneously.

Codes for differential-algebraic equations (DAEs) directly address the solution of implicit systems of the form

$$\mathbf{f}(t, \mathbf{y}(t), \dot{\mathbf{y}}(t)) = \mathbf{0}, \qquad \mathbf{y}(0) = \mathbf{y}^0. \tag{9.2.7}$$

One of the best of these is the code DASSL written by Petzold [3]. DASSL uses variable-step, variable-order backward difference methods to solve problems without needing $\mathbf{M}^{-1}$ to exist.

Let us illustrate these concepts by applying some simple one-step schemes to problems having the forms (9.2.1) or (9.2.4). However, implementation of these simple methods is only justified in certain special circumstances. In most cases, it is far better to use existing ODE software in a method of lines framework.

For simplicity, we'll assume that all boundary data is homogeneous so that the boundary inner product in (9.2.2a) vanishes. Selecting a finite-dimensional space $S_0^N \subset H_0^1$, we then determine U as the solution of

$$(V, U_t) + A(V, U) = (V, f), \qquad \forall v \in S_0^N. \tag{9.2.8}$$

Evaluation leads to ODEs having the form of (9.2.4a) regardless of whether or not the system is one-dimensional or the coefficients are constant. The actual matrices $\mathbf{M}$ and $\mathbf{K}$ and load vector $\mathbf{l}$ will, of course, differ from those of Example 9.2.1 in these cases. The systems (9.2.4a) or (9.2.8) are called *semi-discrete Galerkin equations* because time has not yet been discretized.

We discretize time into a sequence of time slices $(t_n, t_{n+1}]$ of duration Δt with $t_n = n\Delta t$, $n = 0, 1, \ldots$. For this discussion, no generality is lost by considering uniform time steps. Let:

- $u(x, t_n)$ be the exact solution of the Galerkin problem (9.2.2a) at $t = t_n$.
- $U(x, t_n)$ be the exact solution of the semi-discrete Galerkin problem (9.2.8) at $t = t_n$.
- $U^n(x)$ be the approximation of $U(x, t_n)$ obtained by ODE software.

- $c_j(t_n)$ be the Galerkin coefficient at $t = t_n$; thus, for a one-dimensional problem

$$U(x, t_n) = \sum_{j=1}^{N-1} c_j(t_n)\phi_j(x).$$

  For a Lagrangian basis, $c_j(t_n) = U(x_j, t_n)$.

- c_j^n be the approximation of $c_j(t_n)$ obtained by ODE software. For a one-dimensional problem

$$U^n(x) = \sum_{j=1}^{N-1} c_j^n \phi_j(x).$$

We suppose that all solutions are known at time t_n and that we seek to determine them at time t_{n+1}. The simplest numerical scheme for doing this is the forward Euler method where (9.2.8) is evaluated at time t_n and

$$U_t(x, t_n) \approx \frac{U^{n+1}(x) - U^n(x)}{\Delta t}. \qquad (9.2.9)$$

A simple Taylor's series argument reveals that the local discretization error of such an approximation is $O(\Delta t)$. Substituting (9.2.9) into (9.2.8) yields

$$(V, \frac{U^{n+1} - U^n}{\Delta t}) + A(V, U^n) = (V, f^n), \qquad \forall v \in S_0^N. \qquad (9.2.10a)$$

Evaluation of the inner products leads to

$$\mathbf{M}\frac{\mathbf{c}^{n+1} - \mathbf{c}^n}{\Delta t} + \mathbf{K}^n\mathbf{c}^n = \mathbf{l}^n. \qquad (9.2.10b)$$

We have allowed the stiffness matrix and load vector to be functions of time. The mass matrix would always be independent of time for differential equations having the explicit form of (9.2.1a) as long as the spatial finite element mesh does not vary with time. The ODEs (9.2.10a,b) are implicit unless $\mathbf{M}$ is lumped. If lumping were used and, *e.g.*, $\mathbf{M} \approx h\mathbf{I}$ then $\mathbf{c}^{n+1}$ would be determined as

$$\mathbf{c}^{n+1} = \mathbf{c}^n + \frac{\Delta t}{h}[\mathbf{l}^n - \mathbf{K}^n\mathbf{c}^n].$$

Assuming that $\mathbf{c}^n$ is known, we can determine $\mathbf{c}^{n+1}$ by inverting $\mathbf{M}$.

Using the backward Euler method, we evaluate (9.2.8) at t_{n+1} and use the approximation (9.2.9) to obtain

$$(V, \frac{U^{n+1} - U^n}{\Delta t}) + A(V, U^{n+1}) = (V, f^{n+1}), \qquad \forall v \in S_0^N. \qquad (9.2.11a)$$

and

$$\mathbf{M}\frac{\mathbf{c}^{n+1}-\mathbf{c}^n}{\Delta t}+\mathbf{K}^{n+1}\mathbf{c}^{n+1}=\mathbf{l}^{n+1}. \tag{9.2.11b}$$

The backward Euler method is implicit regardless of whether or not lumping is used. Computation of $\mathbf{c}^{n+1}$ requires inversion of

$$\frac{1}{\Delta t}\mathbf{M}+\mathbf{K}^{n+1}.$$

The most popular of these simple schemes uses a weighted average of the forward and backward Euler methods with weights of $1-\theta$ and θ, respectively. Thus,

$$\begin{aligned}(V,\frac{U^{n+1}-U^n}{\Delta t})+(1-\theta)A(V,U^n)+\theta A(V,U^{n+1}) &= (1-\theta)(V,f^n)+\theta(V,f^{n+1}),\\ &\quad \forall V\in S_0^N. \qquad (9.2.12a)\end{aligned}$$

and

$$\mathbf{M}\frac{\mathbf{c}^{n+1}-\mathbf{c}^n}{\Delta t}+(1-\theta)\mathbf{K}^n\mathbf{c}^n+\theta\mathbf{K}^{n+1}\mathbf{c}^{n+1}=(1-\theta)\mathbf{l}^n+\theta\mathbf{l}^{n+1}. \tag{9.2.12b}$$

The forward and backward Euler methods are recovered by setting $\theta=0$ and 1, respectively.

Let us regroup terms involving $\mathbf{c}^n$ and $\mathbf{c}^{n+1}$ in (9.2.12b) to obtain

$$[\mathbf{M}+\theta\Delta t\mathbf{K}^{n+1}]\mathbf{c}^{n+1}=[\mathbf{M}-(1-\theta)\Delta t\mathbf{K}^n]\mathbf{c}^n+\Delta t[(1-\theta)\mathbf{l}^n+\theta\mathbf{l}^{n+1}]. \tag{9.2.12c}$$

Thus, determination of $\mathbf{c}^{n+1}$ requires inversion of

$$\mathbf{M}+\theta\Delta t\mathbf{K}^{n+1}.$$

In one dimension, this system would typically be tridiagonal as with Example 9.2.1. In higher dimensions it would be sparse. Thus, explicit inversion would never be performed. We would just solve the sparse system (9.2.12c) for $\mathbf{c}^{n+1}$.

Taylor's series calculations reveal that the global discretization error is

$$\|\mathbf{c}(t_n)-\mathbf{c}^n\|=O(\Delta t)$$

for almost all choices of $\theta\in[0,1]$ [6]. The special choice $\theta=1/2$ yields the Crank-Nicolson method which has a discretization error

$$\|\mathbf{c}(t_n)-\mathbf{c}^n\|=O(\Delta t^2).$$

The foregoing discussion involved one-step methods. Multistep methods are also used to solve time-dependent finite element problems and we'll describe them for an ODE in the implicit form (9.2.7). The popular backward difference formulas (BDFs) approximate $\mathbf{y}(t)$ in (9.2.7) by a k *th* degree polynomial $\mathbf{Y}(t)$ that interpolates $\mathbf{y}$ at the $k+1$ times t_{n+1-i}, $i = 0, 1, \ldots, k$. The derivative $\dot{\mathbf{y}}$ is approximated by $\dot{\mathbf{Y}}$. The Newton backward difference form of the interpolating is most frequently used to represent $\mathbf{Y}$ [2, 3], but since we're more familiar with Lagrangian interpolation we'll write

$$\mathbf{y}(t) \approx \mathbf{Y}(t) = \sum_{i=0}^{k} \mathbf{y}^{n+1-i} N_i(t), \qquad t \in (t_{n+1-k}, t_{n+1}], \tag{9.2.13a}$$

where

$$N_i(t) = \prod_{j=0, j\neq i}^{k} \frac{t - t_{n+1-j}}{t_{n+1-i} - t_{n+1-j}}. \tag{9.2.13b}$$

The basis (9.2.13b) is represented by the usual Lagrangian shape functions (*cf.* Section 2.4), so $N_i(t_{n+1-j}) = \delta_{ij}$.

Assuming $\mathbf{y}^{n+1-i}$, $i = 1, 2, \ldots, k$, to be known, the unknown $\mathbf{y}^{n+1}$ is determined by collocation at t_{n+1}. Thus, using (9.2.7)

$$\mathbf{f}(\mathbf{t_{n+1}}, \mathbf{Y}(\mathbf{t_{n+1}}), \dot{\mathbf{Y}}(\mathbf{t_{n+1}})) = \mathbf{0}. \tag{9.2.14}$$

Example 9.2.2. The simplest BDF formula is obtained by setting $k = 1$ in (9.2.13) to obtain

$$\mathbf{Y}(t) = \mathbf{y}^{n+1} N_0(t) + \mathbf{y}^n N_1(t),$$
$$N_0(t) = \frac{t - t_n}{t_{n+1} - t_n}, \qquad N_1(t) = \frac{t - t_{n+1}}{t_n - t_{n+1}},$$

Differentiating $\mathbf{Y}(t)$

$$\dot{\mathbf{Y}}(t) = \frac{\mathbf{y}^{n+1} - \mathbf{y}^n}{t_{n+1} - t_n};$$

thus, the numerical method (9.2.13) is the backward Euler method

$$\mathbf{f}(t_{n+1}, \mathbf{y}^{n+1}, \frac{\mathbf{y}^{n+1} - \mathbf{y}^n}{t_{n+1} - t_n}) = \mathbf{0}.$$

Example 9.2.3. The second-order BDF follows by setting $k = 2$ in (9.2.13) to get

$$\mathbf{Y}(t) = \mathbf{y}^{n+1} N_0(t) + \mathbf{y}^n N_1(t) + \mathbf{y}^{n-1} N_2(t)$$
$$N_0(t) = \frac{(t - t_n)(t - t_{n-1})}{2\Delta t^2}, \qquad N_1(t) = \frac{(t - t_{n+1})(t - t_{n-1})}{-\Delta t^2},$$
$$N_2(t) = \frac{(t - t_{n+1})(t - t_n)}{2\Delta t^2},$$

where time steps are of duration Δt.

Differentiating and setting $t = t_{n+1}$

$$\dot{N}_0(t_{n+1}) = \frac{3}{2\Delta t}, \qquad \dot{N}_1(t_{n+1}) = -\frac{2}{\Delta t}, \qquad \dot{N}_2(t_{n+1}) = \frac{1}{2\Delta t}.$$

Thus,

$$\dot{\mathbf{Y}}(t_{n+1}) = \frac{3\mathbf{y}^{n+1} - 4\mathbf{y}^n + \mathbf{y}^{n-1}}{2\Delta t}$$

and the second-order BDF is

$$\mathbf{f}(t_{n+1}, \mathbf{y}^{n+1}, \frac{3\mathbf{y}^{n+1} - 4\mathbf{y}^n + \mathbf{y}^{n-1}}{2\Delta t}) = \mathbf{0}.$$

Applying this method to (9.2.4a) yields

$$\mathbf{M}\frac{3\mathbf{c}^{n+1} - 4\mathbf{c}^n + \mathbf{c}^{n-1}}{2\Delta t} + \mathbf{K}^{n+1}\mathbf{c}^{n+1} = \mathbf{l}^{n+1}.$$

Thus, computation of $\mathbf{c}^{n+1}$ requires inversion of

$$\frac{\mathbf{M}}{\mathbf{2\Delta t}} + \mathbf{K}.$$

Backward difference formulas through order six are available [2, 3, 6, 7, 8].

9.3 Finite Element Methods in Time

It is, of course, possible to use the finite element method in time. This can be done on space-time triangular or quadrilateral elements for problems in one space dimension; on hexahedra, tetrahedra, and prisms in two space dimensions; and on four-dimensional parallelepipeds and prisms in three space dimensions. However, for simplicity, we'll focus on the time aspects of the space-time finite element method by assuming that the spatial discretization has already been performed. Thus, we'll consider an ODE system in the form (9.2.4a) and construct a Galerkin problem in time by multiplying it by a test function $\mathbf{w} \in L^2$ and integrating on $(t_n, t_{n+1}]$ to obtain

$$(\mathbf{w}, \mathbf{M}\dot{\mathbf{c}})_n + (\mathbf{w}, \mathbf{K}\mathbf{c})_n = (\mathbf{w}, \mathbf{l})_n, \qquad \forall \mathbf{w} \in L^2(t_n, t_{n+1}], \tag{9.3.1a}$$

where the L^2 inner product in time is

$$(\mathbf{w}, \mathbf{c})_n = \int_{t_n}^{t_{n+1}} \mathbf{w}^T \mathbf{c} dt. \tag{9.3.1b}$$

Only first derivatives are involved in (9.2.4a); thus, neither the trial space for $\mathbf{c}$ nor the test space for $\mathbf{w}$ have to be continuous. For our initial method, let us assume that $\mathbf{c}(t)$ is continuous at t_n. By assumption, $\mathbf{c}(t_n)$ is known in this case and, hence, $\mathbf{w}(t_n) = \mathbf{0}$.

Example 9.3.1. Let us examine the method that results when $\mathbf{c}(t)$ and $\mathbf{w}(t)$ are linear on $(t_n, t_{n+1}]$. We represent $\mathbf{c}(t)$ in the manner used for a spatial basis as

$$\mathbf{c}(\tau) \approx \mathbf{c}^n N_n(\tau) + \mathbf{c}^{n+1} N_{n+1}(\tau) \tag{9.3.2a}$$

where

$$N_n(\tau) = \frac{1-\tau}{2}, \qquad N_{n+1}(\tau) = \frac{1+\tau}{2} \tag{9.3.2b}$$

are hat functions in time and

$$\tau = \frac{2t - t_n - t_{n+1}}{\Delta t} \tag{9.3.2c}$$

defines the canonical element in time. The test function

$$\mathbf{w} = N_{n+1}(\tau)[1, 1, \ldots, 1]^T \tag{9.3.2d}$$

vanishes at t_n $(\tau = -1)$ and is linear on (t_n, t_{n+1}).

Transforming the integrals in (9.3.1a) to $(-1, 1)$ using (9.3.2c) and using (9.3.2a,b,d) yields

$$(\mathbf{w}, \mathbf{M}\dot{\mathbf{c}})_n = \frac{\Delta t}{2}\int_{-1}^{1} \frac{1+\tau}{2}\mathbf{M}\frac{\mathbf{c}^{n+1}-\mathbf{c}^n}{\Delta t}d\tau,$$

$$(\mathbf{w}, \mathbf{K}\mathbf{c})_n = \frac{\Delta t}{2}\int_{-1}^{1} \frac{1+\tau}{2}\mathbf{K}[\mathbf{c}^n\frac{1-\tau}{2} + \mathbf{c}^{n+1}\frac{1+\tau}{2}]d\tau.$$

(Again, we have written equality instead of $\approx$ for simplicity.) Assuming that $\mathbf{M}$ and $\mathbf{K}$ are independent of time, we have

$$(\mathbf{w}, \mathbf{M}\dot{\mathbf{c}})_n = \mathbf{M}\frac{\mathbf{c}^{n+1}-\mathbf{c}^n}{2},$$

$$(\mathbf{w}, \mathbf{K}\mathbf{c})_n = \frac{\Delta t}{6}\mathbf{K}(\mathbf{c}^n + 2\mathbf{c}^{n+1}).$$

Substituting these into (9.3.1a)

$$\mathbf{M}\frac{\mathbf{c}^{n+1}-\mathbf{c}^n}{2} + \frac{\Delta t}{6}\mathbf{K}(\mathbf{c}^n + 2\mathbf{c}^{n+1}) = \frac{\Delta t}{2}\int_{-1}^{1}\frac{1+\tau}{2}\mathbf{l}(\tau)d\tau \tag{9.3.3a}$$

or, if $\mathbf{l}$ is approximated like $\mathbf{c}$,

$$\mathbf{M}\frac{\mathbf{c}^{n+1}-\mathbf{c}^n}{2} + \frac{\Delta t}{6}\mathbf{K}(\mathbf{c}^n + 2\mathbf{c}^{n+1}) = \frac{\Delta t}{6}(\mathbf{l}^n + 2\mathbf{l}^{n+1}). \tag{9.3.3b}$$

Regrouping terms

$$[\mathbf{M} + \frac{2}{3}\Delta t\mathbf{K}]\mathbf{c}^{n+1} = [\mathbf{M} - \frac{1}{3}\Delta t\mathbf{K}]\mathbf{c}^n + \frac{1}{3}\Delta t[\mathbf{l}^n + 2\mathbf{l}^{n+1}], \tag{9.3.3c}$$

we see that the piecewise-linear Galerkin method in time is a weighted average scheme (9.2.12c) with $\theta = 2/3$. Thus, at least to this low order, there is not much difference between finite difference and finite element methods. Other similarities appear in Problem 1 at the end of this section.

Low-order schemes such as (9.2.12) are popular in finite element packages. Our preference is for BDF or implicit Runge-Kutta software that control accuracy through automatic time step and order variation. Implicit Runge-Kutta methods may be derived as finite element methods by using the Galerkin method (9.3.1) with higher-order trial and test functions. Of the many possibilities, we'll examine a class of methods where the trial function $\mathbf{c}(t)$ is discontinuous.

Example 9.3.2. Suppose that $\mathbf{c}(t)$ is a polynomial on $(t_n, t_{n+1}]$ with jump discontinuities at t_n, $n \geq 0$. When we need to distinguish left and right limits, we'll use the notation

$$\mathbf{c}^{n-} = \lim_{\epsilon \to 0} \mathbf{c}(t_n - \epsilon), \qquad \mathbf{c}^{n+} = \lim_{\epsilon \to 0} \mathbf{c}(t_n + \epsilon). \tag{9.3.4a}$$

With jumps at t_n, we'll have to be more precise about the temporal inner product (9.3.1b) and we'll define

$$(u, v)_{n-} = \lim_{\epsilon \to 0} \int_{t_n - \epsilon}^{t_{n+1} - \epsilon} uv dt, \qquad (u, v)_{n+} = \lim_{\epsilon \to 0} \int_{t_n + \epsilon}^{t_{n+1} - \epsilon} uv dt. \tag{9.3.4b}$$

The inner product $(u, v)_{n-}$ may be affected by discontinuities in functions at t_n, but $(u, v)_{n+}$ only involves integrals of smooth functions. In particular:

- $(u, v)_{n-} = (u, v)_{n+}$ when $u(t)$ and $v(t)$ are either continuous or have jump discontinuities at t_n;
- $(u, v)_{n-}$ exists and $(u, v)_{n+} = 0$ when either u or v are proportional to the delta function $\delta(t - t_n)$; and
- $(u, v)_{n-}$ doesn't exist while $(v, u)_{n+} = 0$ when both u and v are proportional to $\delta(t - t_n)$.

Suppose, for example, that $v(t)$ is continuous at t_n and $u(t) = \delta(t - t_n)$. Then

$$(u, v)_{n-} = \lim_{\epsilon \to 0} \int_{t_n - \epsilon}^{t_{n+1} - \epsilon} \delta(t - t_n) v(t) dt = v(t_n).$$

The delta function can be approximated by a smooth function that depends on ϵ as was done in Section 3.2 to help explain this result.

Let us assume that $\mathbf{w}(t)$ is continuous and write $\mathbf{c}(t)$ in the form

$$\mathbf{c}(t) = \mathbf{c}^{n-} + [\bar{\mathbf{c}}(t) - \mathbf{c}^{n-}] H(t - t_n) \tag{9.3.5a}$$

where

$$H(t) = \begin{cases} 1, & \text{if } t > 0 \\ 0, & \text{otherwise} \end{cases} \tag{9.3.5b}$$

is the *Heaviside function* and $\bar{\mathbf{c}}$ is a polynomial in t.

Differentiating

$$\dot{\mathbf{c}}(t) = [\bar{\mathbf{c}}(t) - \mathbf{c}^{n-}]\delta(t - t_n) + \dot{\bar{\mathbf{c}}}(t)H(t - t_n). \tag{9.3.5c}$$

With the interpretation that inner products in (9.3.1) are of type (9.3.4), assume that $\mathbf{w}(t)$ is continuous and use (9.3.5) in (9.3.1a) to obtain

$$\mathbf{w}^T(t_n)\mathbf{M}(t_n)(\mathbf{c}^{n+} - \mathbf{c}^{n-}) + (\mathbf{w}, \mathbf{M}\dot{\bar{\mathbf{c}}})_{n+} + (\mathbf{w}, \mathbf{K}\bar{c})_{n+} = (\mathbf{w}, \mathbf{l})_{n+}, \qquad \forall \mathbf{w} \in H^1. \tag{9.3.6}$$

The simplest discontinuous Galerkin method uses a piecewise constant ($p = 0$) basis in time. Such approximations are obtained from (9.3.5a) by selecting

$$\bar{\mathbf{c}}(t) = \mathbf{c}^{n+} = \mathbf{c}^{(n+1)-}.$$

Testing against the constant function

$$\mathbf{w}(t) = [1, 1, \ldots, 1]^T$$

and assuming that $\mathbf{M}$ and $\mathbf{K}$ are independent of t, (9.3.6) becomes

$$\mathbf{M}(\mathbf{c}^{(n+1)-} - \mathbf{c}^{n-}) + \mathbf{K}\mathbf{c}^{(n+1)-}\Delta t = \int_{t_n}^{t_{n+1}} \mathbf{l}(t)dt.$$

The result is almost the same as the backward Euler formula (9.2.11b) except that the load vector $\mathbf{l}$ is averaged over the time step instead of being evaluated at t_{n+1}.

With a linear ($p = 1$) approximation for $\bar{\mathbf{c}}(t)$, we have

$$\bar{\mathbf{c}}(t) = \mathbf{c}^{n+}N_n(t) + \mathbf{c}^{(n+1)-}N_{n+1}(t)$$

where N_{n+i}, $i = 0, 1$, are given by (9.3.2b). Selecting the basis for the test space as

$$\mathbf{w}_i(t) = N_{n+i}(t)[1, 1, \ldots, 1]^T, \qquad i = 0, 1,$$

assuming that $\mathbf{M}$ and $\mathbf{K}$ are independent of t, and substituting the above approximations into (9.3.6), we obtain

$$\mathbf{M}(\mathbf{c}^{n+} - \mathbf{c}^{n-}) + \frac{1}{2}\mathbf{M}(\mathbf{c}^{(n+1)-} - \mathbf{c}^{n+}) + \frac{\Delta t}{6}\mathbf{K}(2\mathbf{c}^{n+} + \mathbf{c}^{(n+1)-}) = \int_{t_n}^{t_{n+1}} N_n \mathbf{l}(t)dt$$

and

$$\frac{1}{2}\mathbf{M}(\mathbf{c}^{(n+1)-} - \mathbf{c}^{n+}) + \frac{\Delta t}{6}\mathbf{K}(\mathbf{c}^{n+} + 2\mathbf{c}^{(n+1)-}) = \int_{t_n}^{t_{n+1}} N_{n+1}\mathbf{l}(t)dt.$$

Simplifying the expressions and assuming that $\mathbf{l}(t)$ can be approximated by a linear function on (t_n, t_{n+1}) yields the system

$$\mathbf{M}(\frac{\mathbf{c}^{n+} + \mathbf{c}^{(n+1)-}}{2} - \mathbf{c}^{n-}) + \frac{\Delta t}{6}\mathbf{K}(2\mathbf{c}^{n+} + \mathbf{c}^{(n+1)-}) = \frac{\Delta t}{6}(2\mathbf{l}^n + \mathbf{l}^{(n+1)-}),$$

$$\mathbf{M}\frac{\mathbf{c}^{(n+1)-} - \mathbf{c}^{n+}}{2} + \frac{\Delta t}{6}\mathbf{K}(\mathbf{c}^{n+} + 2\mathbf{c}^{(n+1)-}) = \frac{\Delta t}{6}(\mathbf{l}^n + 2\mathbf{l}^{(n+1)-}).$$

This pair of equations must be solved simultaneously for the two unknown solution vectors $\mathbf{c}^{n+}$ and $\mathbf{c}^{(n+1)-}$. This is an implicit Runge-Kutta method.

Problems

1. Consider the Galerkin method in time with a continuous basis as represented by (9.3.1). Assume that the solution $\mathbf{c}(t)$ is approximated by the linear function (9.3.2a-c) on (t_n, t_{n+1}) as in Example 9.3.1, but do not assume that the test space $\mathbf{w}(t)$ is linear in time.

   1.1. Specifying
   $$\mathbf{w}(\tau) = \omega(\tau)[1, 1, \ldots, 1]^T$$
   and assuming that $\mathbf{M}$ and $\mathbf{K}$ are independent ot t, show that (9.3.1a) is the weighted average scheme
   $$[\mathbf{M} + \theta\Delta t\mathbf{K}]\mathbf{c}^{n+1} = [\mathbf{M} - (1-\theta)\Delta t\mathbf{K}]\mathbf{c}^n + \Delta t[(1-\theta)\mathbf{l}^n + \theta\mathbf{l}^{n+1}]$$
   with
   $$\theta = \frac{\int_{-1}^{1}\omega(\tau)N_{n+1}^1(\tau)d\tau}{\int_{-1}^{1}\omega(\tau)d\tau}.$$
   When different trial and test spaces are used, the Galerkin method is called a *Petrov-Galerkin method.*

   1.2. The entire effect of the test function $\omega(t)$ is isolated in the weighting factor θ. Furthermore, no integration by parts was performed, so that $\omega(t)$ need not be continuous. Show that the choices of $\omega(t)$ listed in Table 9.3.1 correspond to the cited methods.

2. The discontinuous Galerkin method may be derived by simultaneously discretizing a partial differential system in space and time on $\Omega \times (t-n-, t_{(n+1)-})$. This form may have advantages when solving problems with rapid dynamics since the mesh may be either moved or regenerated without concern for maintaining continuity

Scheme	ω	θ
Forward Euler (9.2.10b)	$\delta(1+\tau)$	0
Crank-Nicolson (9.2.12b)	$\delta(\tau)$	1/2
Crank-Nicolson (9.2.12b)	1	1/2
Backward Euler (9.2.11b)	$\delta(1-\tau)$	1
Galerkin (9.3.3)	$N_{n+1}^1(\tau)$	2/3

Table 9.3.1: Test functions ω and corresponding methods for the finite element solution of (9.2.4a) with a linear trial function.

between time steps. Using (9.2.2a) as a model spatial finite element formulation, assume that test functions $v(x, y, t)$ are continuous but that trial functions $u(x, y, t)$ have jump discontinuities at t_n. Assume Dirichlet boundary data and show that the space-time discontinuous Galerkin form of the problem is

$$\begin{aligned}(v, u_t)_{ST} + (v(\cdot, t_n), u(\cdot, t_{n+}) - u(\cdot, t_{n-})) \quad + \quad & A_{ST}(v, u) = (v, f)_{ST}, \\ & \forall v \in H_0^1(\Omega \times (t_{n+}, t_{(n+1)-})),\end{aligned}$$

where

$$(v, u)_{ST} = \int_{t_{n+}}^{t_{(n+1)-}} \iint_{\Omega} vu dx dy dt$$

and

$$A_{ST}(v, u) = (v_x, pu_x)_{ST} + (v_y, pu_y)_{ST} + (v, qu)_{ST}.$$

In this form, the finite element problem is solved on the three-dimensional strips $\Omega \times (t_{n-}, t_{(n+1)-})$, $n = 0, 1, \ldots$.

9.4 Convergence and Stability

In this section, we will study some theoretical properties of the discrete methods that were introduced in Sections 9.2 and 9.3. Every finite difference or finite element scheme for time integration should have three properties:

1. *Consistency*: the discrete system should be a good approximation of the differential equation.

2. *Convergence*: the solution of the discrete system should be a good approximation of the solution of the differential equation.

3. *Stability*: the solution of the discrete system should not be sensitive to small perturbations in the data.

Somewhat because they are open ended, finite difference or finite element approximations in time can be sensitive to small errors, *e.g.*, introduced by round off. Let us illustrate the phenomena for the weighted average scheme (9.2.12c)

$$[\mathbf{M} + \theta\Delta t\mathbf{K}]\mathbf{c}^{n+1} = [\mathbf{M} - (1-\theta)\Delta t\mathbf{K}]\mathbf{c}^{n} + \Delta t[(1-\theta)\mathbf{l}^{n} + \theta\mathbf{l}^{n+1}]. \tag{9.4.1}$$

We have assumed, for simplicity, that $\mathbf{K}$ and $\mathbf{M}$ are independent of time.

Sensitivity to small perturbations implies a lack of stability as expressed by the following definition.

Definition 9.4.1. A finite difference scheme is *stable* if a perturbation of size $\|\delta\|$ introduced at time t_n remains bounded for subsequent times $t \leq T$ and all time steps $\Delta t \leq \Delta t_0$.

We may assume, without loss of generality, that the perturbation is introduced at time $t = 0$. Indeed, it is common to neglect perturbations in the coefficients and confine the analysis to perturbations in the initial data. Thus, in using Definition 9.4.1, we consider the solution of the related problem

$$[\mathbf{M} + \theta\Delta t\mathbf{K}]\tilde{\mathbf{c}}^{n+1} = [\mathbf{M} - (1-\theta)\Delta t\mathbf{K}]\tilde{\mathbf{c}}^{n} + \Delta t[(1-\theta)\mathbf{l}^{n} + \theta\mathbf{l}^{n+1}],$$

$$\tilde{\mathbf{c}}^{0} = \mathbf{c}^{0} + \boldsymbol{\delta}.$$

Subtracting (9.4.1) from the perturbed system

$$[\mathbf{M} + \theta\Delta t\mathbf{K}]\boldsymbol{\delta}^{n+1} = [\mathbf{M} - (1-\theta)\Delta t\mathbf{K}]\boldsymbol{\delta}^{n}, \qquad \boldsymbol{\delta}^{0} = \boldsymbol{\delta}, \tag{9.4.2a}$$

where

$$\boldsymbol{\delta}^{n} = \tilde{\mathbf{c}}^{n} - \mathbf{c}^{n}. \tag{9.4.2b}$$

Thus, for linear problems, it suffices to apply Definition 9.4.1 to a homogeneous version of the difference scheme having the perturbation as its initial condition. With these restrictions, we may define stability in a more explicit form.

Definition 9.4.2. A linear difference scheme is *stable* if there exists a constant $C > 0$ which is independent of Δt and such that

$$\|\boldsymbol{\delta}^{n}\| < C\|\boldsymbol{\delta}^{0}\| \tag{9.4.3}$$

as $n \to \infty$, $\Delta t \to 0$, $t \leq T$.

Both Definitions 9.4.1 and 9.4.2 permit the initial perturbation to grow, but only by a bounded amount. Restricting the growth to finite times $t < T$ ensures that the definitions apply when the solution of the difference scheme $\mathbf{c}^n \to \infty$ as $n \to \infty$. When applying Definition 9.4.2, we may visualize a series of computations performed to time T with an increasing number of time steps M of shorter-and-shorter duration Δt such that $T = M\Delta t$. As Δt is decreased, the perturbations δ^n, $n = 1, 2, \ldots, M$, should settle down and eventually not grow to more than C times the initial perturbation.

Solutions of continuous systems are often stable in the sense that $\mathbf{c}(t)$ is bounded for all $t \geq 0$. In this case, we need a stronger definition of stability for the discrete system.

Definition 9.4.3. The linear difference scheme (9.4.1) is *absolutely stable* if

$$\|\boldsymbol{\delta}^n\| < \|\boldsymbol{\delta}^0\|. \tag{9.4.4}$$

Thus, perturbations are not permitted to grow at all.

Stability analyses of linear constant coefficient difference equations such as (9.4.2) involve assuming a perturbation of the form

$$\boldsymbol{\delta}^n = (\lambda)^n \mathbf{r}. \tag{9.4.5}$$

Substituting into (9.4.2a) yields

$$[\mathbf{M} + \theta\Delta t\mathbf{K}](\lambda)^{n+1}\mathbf{r} = [\mathbf{M} - (1-\theta)\Delta t\mathbf{K}](\lambda)^n\mathbf{r}.$$

Assuming that $\lambda \neq 0$ and $\mathbf{M} + \theta\Delta t\mathbf{K}$ is not singular, we see that λ is an eigenvalue and $\mathbf{r}$ is an eigenvector of

$$[\mathbf{M} + \theta\Delta t\mathbf{K}]^{-1}[\mathbf{M} - (1-\theta)\Delta t\mathbf{K}]\mathbf{r}_k = \lambda_k\mathbf{r}_k, \qquad k = 1, 2, \ldots, N. \tag{9.4.6}$$

Thus, $\boldsymbol{\delta}^n$ will have the form (9.4.5) with $\lambda = \lambda_k$ and $\mathbf{r} = \mathbf{r}_k$ when the initial perturbation $\boldsymbol{\delta}^0 = \mathbf{r}_k$. More generally, the solution of (9.4.2a) is the linear combination

$$\boldsymbol{\delta}^n = \sum_{k=1}^{N} \delta_k^0 (\lambda_k)^n \mathbf{r}_k \tag{9.4.7a}$$

when the initial perturbation has the form

$$\boldsymbol{\delta}^0 = \sum_{k=1}^{N} \delta_k^0 \mathbf{r}_k. \tag{9.4.7b}$$

Using (9.4.7a), we see that (9.4.2) will be absolutely stable when

$$|\lambda_k| \leq 1, \qquad k = 1, 2, \ldots, N. \tag{9.4.8}$$

The eigenvalues and eigenvectors of many tridiagonal matrices are known. Thus, the analysis is often straight forward for one-dimensional problems. Analyses of two- and three-dimensional problems are more difficult; however, eigenvalue-eigenvector pairs are known for simple problems on simple regions.

Example 9.4.1. Consider the eigenvalue problem (9.4.6) and rearrange terms to get

$$[\mathbf{M} + \theta\Delta t\mathbf{K}]\lambda_k \mathbf{r_k} = [\mathbf{M} - (1-\theta)\Delta t\mathbf{K}]\mathbf{r_k}$$

or

$$(\lambda_k - 1)\mathbf{M}\mathbf{r}_k = -[\lambda_k\theta + (1-\theta)]\Delta t\mathbf{K}\mathbf{r}_k$$

or

$$-\mathbf{K}\mathbf{r}_k = \mu_k\mathbf{M}\mathbf{r}_k$$

where

$$\mu_k = \frac{\lambda_k - 1}{[\lambda_k\theta + (1-\theta)]\Delta t}$$

Thus, μ_k is an eigenvalue and $\mathbf{r}_k$ is an eigenvector of $-\mathbf{M}^{-1}\mathbf{K}$.

Let us suppose that $\mathbf{M}$ and $\mathbf{K}$ correspond to the mass and stiffness matrices of the one-dimensional heat conduction problem of Example 9.2.1. Then, using (9.2.4b,c), we have

$$-\frac{p}{h}\begin{bmatrix} 2 & -1 & & \\ -1 & 2 & -1 & \\ & & \ddots & \\ & & -1 & 2 \end{bmatrix}\begin{bmatrix} r_{k1} \\ r_{k2} \\ \vdots \\ r_{k,N-1} \end{bmatrix} = \frac{\mu_k h}{6}\begin{bmatrix} 4 & 1 & & \\ 1 & 4 & 1 & \\ & & \ddots & \\ & & 1 & 4 \end{bmatrix}\begin{bmatrix} r_{k1} \\ r_{k2} \\ \vdots \\ r_{k,N-1} \end{bmatrix}.$$

The diffusivity p and mesh spacing h have been assumed constant. Also, with Dirichlet boundary conditions, the dimension of this system is $N-1$ rather than N.

It is difficult to see in the above form, but writing this eigenvalue-eigenvector problem in component form

$$\frac{p}{h}(r_{j-1} - 2r_j + r_{j+1}) = \frac{\mu_k h}{6}(r_{j-1} + 4r_j + r_{j+1}), \qquad j = 1, 2, \ldots, N-1,$$

we may infer that the components of the eigenvector are

$$r_{kj} = \sin\frac{k\pi j}{N}, \qquad j = 1, 2, \ldots, N-1.$$

This guess of $\mathbf{r}_k$ may be justified by the similarity of the discrete eigenvalue problem to a continuous one; however, we will not attempt to do this. Assuming it to be correct, we substitute r_{kj} into the eigenvalue problem to find

$$\frac{p}{h}(\sin\frac{k\pi(j-1)}{N} - 2\sin\frac{k\pi j}{N} + \sin\frac{k\pi(j+1)}{N})$$

$$= \frac{\mu_k h}{6}(\sin\frac{k\pi(j-1)}{N} + 4\sin\frac{k\pi j}{N} + \sin\frac{k\pi(j+1)}{N}), \qquad j = 1, 2, \ldots, N-1.$$

But

$$\sin\frac{k\pi(j-1)}{N} + \sin\frac{k\pi(j+1)}{N} = 2\sin\frac{k\pi j}{N}\cos\frac{k\pi}{N}$$

and

$$\frac{p}{h}(\cos\frac{k\pi}{N} - 1)\sin\frac{k\pi j}{N} = \frac{\mu_k h}{6}(\cos\frac{k\pi}{N} + 2)\sin\frac{k\pi j}{N}.$$

Hence,

$$\mu_k = \left(\frac{6p}{h^2}\right)\left(\frac{\cos k\pi/N - 1}{\cos k\pi/N + 2}\right).$$

With $\cos k\pi/N$ ranging on $[-1, 1]$, we see that $-12p/h^2 \leq \mu_k \leq 0$. Determining λ_k in terms of μ_k

$$\lambda_k = \frac{1 + \mu_k(1-\theta)\Delta t}{1 - \mu_k\theta\Delta t} = 1 + \frac{\mu_k \Delta t}{1 - \mu_k\theta\Delta t}.$$

We would like $|\lambda_k| \leq 1$ for absolute stability. With $\mu_k \leq 0$, we see that the requirement that $\lambda_k \leq 1$ is automatically satisfied. Demanding the $\lambda_k \geq -1$ yields

$$|\mu_k|\Delta t(1 - 2\theta) \leq 2.$$

If $\theta \geq 1/2$ then $1 - 2\theta \leq 0$ and the above inequality is satisfied for all choices of μ_k and Δt. Methods of this class are *unconditionally absolutely stable*. When $\theta < 1/2$, we have to satisfy the condition

$$\frac{p\Delta t}{h^2} \leq \frac{1}{6(1-2\theta)}.$$

If we view this last relation as a restriction of the time step Δt, we see that the forward Euler method ($\theta = 0$) has the smallest time step. Since all other methods listed in Table 9.3.1 are unconditionally stable, there would be little value in using the forward Euler method without lumping the mass matrix. With lumping, the stability restriction of the forward Euler method actually improves slightly to $p\Delta t/h^2 \leq 1/2$.

Let us now turn to a more general examination of stability and convergence. Let's again focus on our model problem: determine $u \in H_0^1$ satisfying

$$(v, u_t) + A(v, u) = (v, f), \qquad \forall v \in H_0^1, \qquad t > 0, \tag{9.4.9a}$$

$$(v, u) = (v, u^0), \qquad \forall v \in H_0^1, \qquad t = 0. \tag{9.4.9b}$$

The semi-discrete approximation consists of determining $U \in S_0^N \subset H_0^1$ such that

$$(V, U_t) + A(V, U) = (V, f), \qquad \forall V \in S_0^N, \qquad t > 0, \tag{9.4.10a}$$

$$(V, U) = (V, u^0), \qquad \forall V \in S_0^N, \qquad t = 0. \tag{9.4.10b}$$

Trivial Dirichlet boundary data, again, simplifies the analysis.

Our first result establishes the absolute stability of the finite element solution of the semi-discrete problem (9.4.10) in the L^2 norm.

Theorem 9.4.1. *Let $\delta \in S_0^N$ satisfy*

$$(V, \delta_t) + A(V, \delta) = 0, \qquad \forall V \in S_0^N, \qquad t > 0, \tag{9.4.11a}$$

$$(V, \delta) = (V, \delta^0), \qquad \forall V \in S_0^N, \qquad t = 0. \tag{9.4.11b}$$

Then

$$\|\delta(\cdot, \cdot, t)\|_0 \le \|\delta^0\|_0, \qquad t > 0. \tag{9.4.11c}$$

Remark 1. With $\delta(x, t)$ being the difference between two solutions of (9.4.10a) satisfying initial conditions that differ by $\delta^0(x)$, the loading (V, f) vanishes upon subtraction (as with (9.4.2)).

Proof. Replace V in (9.4.11a) by δ to obtain

$$(\delta, \delta_t) + A(\delta, \delta) = 0,$$

or

$$\frac{1}{2}\frac{d}{dt}\|\delta\|_0^2 + A(\delta, \delta) = 0.$$

Integrating

$$\|\delta(\cdot, \cdot, t)\|_0^2 = \|\delta(\cdot, \cdot, 0)\|_0^2 - 2\int_0^t A(\delta, \delta)d\tau.$$

The result (9.4.11c) follows by using the initial data (9.4.11b) and the non-negativity of $A(\delta, \delta)$. □

We've discussed stability at some length, so now let us turn to the concept of convergence. Convergence analyses for semi-discrete Galerkin approximations parallels the lines of those for elliptic systems. Let us, as an example, establish convergence for piecewise-linear solutions of (9.4.10) to solutions of (9.4.9).

Theorem 9.4.2. *Let S_0^N consist of continuous piecewise-linear polynomials on a family of uniform meshes Δ_h characterized by their maximum element size h. Then there exists a constant $C > 0$ such that*

$$\max_{t \in (0,T]} \|u - U\|_0 \le C(1 + |\log \frac{T}{h^2}|)h^2 \max_{t \in (0,T]} \|u\|_2. \tag{9.4.12}$$

Proof. Create the auxiliary problem: determine $W \in S_0^N$ such that

$$-(V, W_\tau(\cdot,\cdot,\tau)) + A(V, W(\cdot,\cdot,\tau)) = 0, \qquad \forall V \in S_0^N, \qquad \tau \in (0,t), \tag{9.4.13a}$$

$$W(x,y,t) = E(x,y,t) = U(x,y,t) - \tilde{U}(x,y,t), \tag{9.4.13b}$$

where $\tilde{U} \in S_0^N$ satisfies

$$A(V, u(\cdot,\cdot,\tau) - \tilde{U}(\cdot,\cdot,\tau)) = 0, \qquad \forall V \in S_0^N, \qquad \tau \in (0,T]. \tag{9.4.13c}$$

We see that W satisfies a terminal value problem on $0 \le \tau \le t$ ant that $\tilde{U}$ satisfies an elliptic problem with τ as a parameter.

Consider the identity

$$\frac{d}{d\tau}(W,E) = (W_\tau, E) + (W, E_\tau).$$

Integrate and use (9.4.13b)

$$\|E(\cdot,\cdot,t)\|_0^2 = (W, E(\cdot,\cdot,0)) + \int_0^t [(W_\tau, E) + (W, E_\tau)]d\tau.$$

Use (9.4.13a) with V replaced by E

$$\|E(\cdot,\cdot,t)\|_0^2 = (W, E(\cdot,\cdot,0)) + \int_0^t [A(W, E) + (W, E_\tau)]d\tau. \tag{9.4.14}$$

Setting v in (9.4.9) and V in (9.4.10) to W and subtracting yields

$$(W, u_\tau - U_\tau) + A(W, u - U) = 0, \qquad \tau > 0,$$

$$(W, u - U)(0) = 0, \qquad \tau = 0.$$

Add these results to (9.4.14) and use (9.4.13b) to obtain

$$\|E(\cdot,\cdot,t)\|_0^2 = (W, \theta(\cdot,\cdot,0)) + \int_0^t [A(W,\theta) + (W, \theta_\tau)]d\tau,$$

where

$$\theta = u - \tilde{U}.$$

The first term in the integrand vanishes by virtue of (9.4.13c). The second term is integrated by parts to obtain

$$\|E(\cdot,\cdot,t)\|_0^2 = (W, \theta(\cdot,\cdot,t)) - \int_0^t (W_\tau, \theta)d\tau. \tag{9.4.15a}$$

This result can be simplified slightly by use of Cauchy's inequality ($|(W,V)| \le \|W\|_0 \|V\|_0$) to obtain

$$\|E(\cdot,\cdot,t)\|_0^2 \le \|W(\cdot,\cdot,t)\|_0 \|\theta(\cdot,\cdot,t)\|_0 + \int_0^t \|W_\tau\|_0 \|\theta\|_0 d\tau. \tag{9.4.15b}$$

Introduce a basis on S_0^N and write W in the standard form

$$W(x,y,\tau) = \sum_{j=0}^{N} c_j(\tau)\phi_j(x,y). \tag{9.4.16}$$

Substituting (9.4.16) into (9.4.13a) and following the steps introduced in Section 9.2, we are led to

$$-\mathbf{M}\dot{\mathbf{c}} + \mathbf{K}\mathbf{c} = \mathbf{0}, \tag{9.4.17a}$$

where

$$M_{ij} = (\phi_i, \phi_j), \tag{9.4.17b}$$

$$K_{ij} = A(\phi_i, \phi_j), \qquad i,j = 1,2,\ldots,N. \tag{9.4.17c}$$

Assuming that the stiffness matrix $\mathbf{K}$ is independent of τ, (9.4.17a) may be solved exactly to show that (*cf.* Lemmas 9.4.1 and 9.4.2 which follow)

$$\|W(\cdot,\cdot,\tau)\|_0 \le \|E(\cdot,\cdot,t)\|_0, \qquad 0 < \tau \le t, \tag{9.4.18a}$$

$$\int_0^t \|W_\tau\|_0 d\tau \le C(1 + |\log \frac{t}{h^2}|)\|E(\cdot,\cdot,t)\|_0. \tag{9.4.18b}$$

Equation (9.4.18a) is used in conjunction with (9.4.15b) to obtain

$$\|E(\cdot,\cdot,t)\|_0^2 \le (\|E(\cdot,\cdot,t)\|_0 + \int_0^t \|W_\tau\|_0 d\tau) \max_{\tau\in(0,t]} \|\theta(\cdot,\cdot,\tau)\|_0.$$

Now, using (9.4.18b)

$$\|E(\cdot,\cdot,t)\|_0 \le C(1 + |\log \frac{t}{h^2}|) \max_{\tau\in(0,t]} \|\theta(\cdot,\cdot,\tau)\|_0. \tag{9.4.19}$$

Writing

$$u - U = u - \tilde{U} + \tilde{U} - U = \theta - E$$

and taking an L^2 norm

$$\|u - U\|_0 \le \|\theta\|_0 + \|E\|_0.$$

Using (9.4.19)

$$\|u - U\|_0 \le C(1 + |\log \frac{t}{h^2}|) \max_{\tau \in (0,t]} \|\theta(\cdot,\cdot,\tau)\|_0. \qquad (9.4.20a)$$

Finally, since θ satisfies the elliptic problem (9.4.13c), we can use Theorem 7.2.4 to write

$$\|\theta(\cdot,\cdot,\tau)\|_0 \le Ch^2 \|u(\cdot,\cdot,\tau)\|_2. \qquad (9.4.20b)$$

Combining (9.4.20a) and (9.4.20b) yields the desired result (9.4.12). □

The two results that were used without proof within Theorem 9.4.2 are stated as Lemmas.

Lemma 9.4.1. *Under the conditions of Theorem 9.4.2, there exists a constant* $C > 0$ *such that*

$$A(V,V) \le \frac{C}{h^2}\|V\|_0^2, \qquad \forall V \in S_0^N. \qquad (9.4.21)$$

Proof. The result can be inferred from Example 9.2.1; however, a more formal proof is given by Johnson [9], Chapter 7. □

Instead of establishing (9.4.18b), we'll examine a slightly more general situation. Let $\mathbf{c}$ be the solution of

$$\mathbf{M}\dot{\mathbf{c}} + \mathbf{K}\mathbf{c} = \mathbf{0}, \qquad t > 0, \qquad \mathbf{c}(0) = \mathbf{c}^0. \qquad (9.4.22)$$

The mass and stiffness matrices $\mathbf{M}$ and $\mathbf{K}$ are positive definite, so we can diagonalize (9.4.22). In particular, let $\mathbf{\Lambda}$ be a diagonal matrix containing the eigenvalues of $\mathbf{M}^{-1}\mathbf{K}$ and $\mathbf{R}$ be a matrix whose columns are the eigenvectors of the same matrix, i.e.,

$$\mathbf{M}^{-1}\mathbf{K}\mathbf{R} = \mathbf{R}\mathbf{\Lambda}. \qquad (9.4.23a)$$

Further let

$$\mathbf{d}(t) = \mathbf{R}^{-1}\mathbf{c}(t). \qquad (9.4.23b)$$

Then (9.4.22) can be written in the diagonal form

$$\dot{\mathbf{d}} + \mathbf{\Lambda}\mathbf{d} = \mathbf{0} \qquad (9.4.24a)$$

by multiplying it by $(\mathbf{MR})^{-1}$ and using (9.4.23a,b). The initial conditions generally remain coupled through (9.4.23a,b), *i.e.*,

$$\mathbf{d}(0) = \mathbf{d}^0 = \mathbf{R}^{-1}\mathbf{c}^0. \qquad (9.4.24b)$$

With these preliminaries, we state the desired result.

Lemma 9.4.2. *If* $\mathbf{d}(t)$ *is the solution of (9.4.24) then*

$$|\dot{\mathbf{d}}| + |\mathbf{\Lambda d}| \leq \frac{C|\mathbf{d}^0|}{t}, \qquad t > 0, \tag{9.4.25a}$$

where $|\mathbf{d}| = \sqrt{\mathbf{d}^T\mathbf{d}}$. *If, in addition,*

$$\max_{\boldsymbol{\xi}\neq 0} \frac{|\mathbf{\Lambda}\boldsymbol{\xi}|}{|\boldsymbol{\xi}|} \leq \frac{C}{h^2} \tag{9.4.25b}$$

then

$$\int_0^T (|\dot{\mathbf{d}}| + |\mathbf{\Lambda d}|)dt \leq C(1 + |\log\frac{T}{h^2}|)|\mathbf{d}^0|. \tag{9.4.25c}$$

Proof. cf. Problem 1. □

Problems

1. Prove Lemma 9.4.2.

9.5 Convection-Diffusion Systems

Problems involving convection and diffusion arise in fluid flow and heat transfer. Let us consider the model problem

$$u_t + \boldsymbol{\omega} \cdot \nabla u = \nabla \cdot (p\nabla u) \tag{9.5.1a}$$

where $\boldsymbol{\omega} = [\omega_1, \omega_2]^T$ is a velocity vector. Written is scalar form, (9.5.1a) is

$$u_t + \omega_1 u_x + \omega_2 u_y = (pu_x)_x + (pu_y)_y. \tag{9.5.1b}$$

The vorticity transport equation of fluid mechanics has the form of (9.5.1). In this case, u would represent the vorticity of a two-dimensional flow.

If the magnitude of $\boldsymbol{\omega}$ is small relative to the magnitude of the diffusivity $p(x, y)$, then the standard methods that we have been studying work fine. This, however, is not the case in many applications and, as indicated by the following example, standard finite element methods can produce spurious results.

Example 9.5.1 [1]. Consider the steady, one-dimensional, convection-diffusion equation

$$-\epsilon u'' + u' = 0, \qquad 0 < x < 1, \tag{9.5.2a}$$

with Dirichlet boundary conditions

$$u(0) = 1, \qquad u(1) = 2. \tag{9.5.2b}$$

The exact solution of this problem is

$$u(x) = 1 + \frac{e^{-(1-x)/\epsilon} - e^{-1/\epsilon}}{1 - e^{-1/\epsilon}}. \tag{9.5.2c}$$

If $0 < \epsilon \ll 1$ then, as shown by the solid line in Figure 9.5.1, the solution features a boundary layer near $x = 1$. At points removed from an $O(\epsilon)$ neighborhood of $x = 1$, the solution is smooth with $u \approx 1$. Within the boundary layer, the solution rises sharply from its unit value to $u = 2$ at $x = 1$.

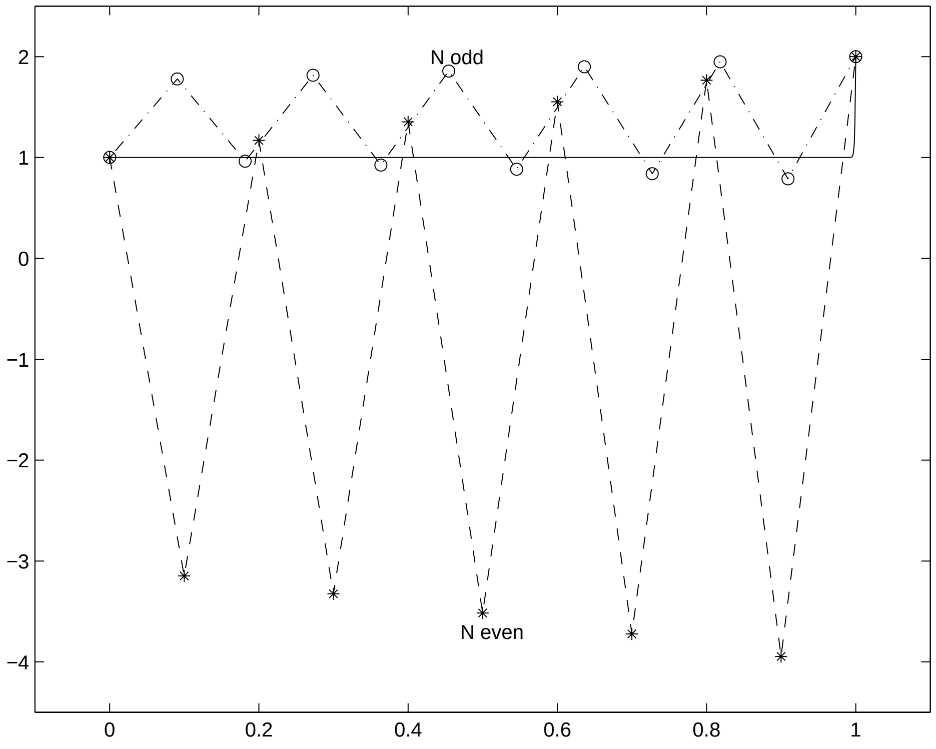


Figure 9.5.1: Solutions of (9.5.2) with $\epsilon = 10^{-3}$. The exact solution is shown as a solid line. Piecewise-linear Galerkin solutions with 10- and 11-element meshes are shown as dashed and dashed-dotted lines, respectively [1].

The term $\epsilon u''$ is diffusive while the term u' is convective. With a small diffusivity ϵ, convection dominates diffusion outside of the narrow $O(\epsilon)$ boundary layer. Within this layer, diffusion cannot be neglected and is on an equal footing with convection. This simple problem will illustrate many of the difficulties that arise when finite element methods are applied to convection-diffusion problems while avoiding the algebraic and geometric complexities of more realistic problems.

Let us divide $[0, 1]$ into N elements of width $h = 1/N$. Since the solution is slowly varying over most of the domain, we would like to choose h to be significantly larger than the boundary layer thickness. This could introduce large errors within the boundary layer which we assume can be reduced by local mesh refinement. This strategy is preferable to the alternative of using a fine mesh everywhere when the solution is only varying rapidly within the boundary layer.

Using a piecewise-linear basis, we write the finite element solution as

$$U(x) = \sum_{j=0}^{N} c_j \phi_j(x), \qquad c_0 = 1, \qquad c_N = 2, \tag{9.5.3a}$$

where

$$\phi_k(x) = \begin{cases} \frac{x-x_{k-1}}{x_k-x_{k-1}}, & \text{if } x_{k-1} < x \leq x_k \\ \frac{x_{k+1}-x}{x_{k+1}-x_k}, & \text{if } x_k < x \leq x_{k+1} \\ 0, & \text{otherwise} \end{cases}. \tag{9.5.3b}$$

The coefficients c_0 and c_N are constrained so that $U(x)$ satisfies the essential boundary conditions (9.5.2b).

The Galerkin problem for (9.5.2) consists of determining $U(x) \in S_0^N$ such that

$$\epsilon(\phi_i', U') + (\phi_i, U') = 0, \qquad i = 1, 2, \ldots, N-1. \tag{9.5.4a}$$

Since this problem is similar to Example 9.2.1, we'll omit the development and just write the inner products

$$(\phi_i', U') = \frac{\epsilon}{h}(c_{i-1} - 2c_i + c_{i+1}), \tag{9.5.4b}$$

$$(\phi_i, U') = \frac{c_{i+1} - c_{i-1}}{2}. \tag{9.5.4c}$$

Thus, the discrete finite element system is

$$(1 - \frac{h}{2\epsilon})c_{i+1} - 2c_i + (1 + \frac{h}{2\epsilon})c_{i-1} = 0, \qquad i = 1, 2, \ldots, N-1. \tag{9.5.4d}$$

The solution of this second-order, constant-coefficient difference equation is

$$c_i = 1 + \frac{1 - \beta^i}{1 - \beta^N}, \qquad i = 0, 1, \ldots, N, \tag{9.5.4e}$$

$$\beta = \frac{1 + h/2\epsilon}{1 - h/2\epsilon}. \tag{9.5.4f}$$

The quantity $h/2\epsilon$ is called the *cell Peclet* or *cell Reynolds number.* If $h/2\epsilon \ll 1$, then

$$\beta = 1 + \frac{h}{\epsilon} + O((\frac{h}{\epsilon})^2) = e^{h/\epsilon} + O((\frac{h}{\epsilon})^2),$$

which is the correct solution. However, if $h/2\epsilon \gg 1$, then $\beta \approx -1$ and

$$c_i \approx \begin{cases} 1, & \text{if } i \text{ is even} \\ 2, & \text{if } i \text{ is odd} \end{cases}$$

when N is odd and

$$c_i \approx \begin{cases} (N+i)/N, & \text{if } i \text{ is even} \\ O(1/\epsilon), & \text{if } i \text{ is odd} \end{cases}$$

when N is even. These two obviously incorrect solutions are shown with the correct results in Figure 9.5.1.

Let us try to remedy the situation. For simplicity, we'll stick with an ordinary differential equation and consider a two-point boundary value problem of the form

$$\mathcal{L}[u] = -\epsilon u'' + \omega u' + qu = f, \qquad 0 < x < 1, \tag{9.5.5a}$$

$$u(0) = u(1) = 0. \tag{9.5.5b}$$

Let us assume that $u, v \in H_0^1$ with u' and v' being continuous except, possibly, at $x = \xi \in (0,1)$. Multiplying (9.5.5a) by v and integrating the second derivative terms by parts yields

$$(v, \mathcal{L}[u]) = A(v,u) + [\epsilon u' v]_{x=\xi} \tag{9.5.6a}$$

where

$$A(v,u) = \epsilon(v', u') + (v, \omega u') + (v, qu), \tag{9.5.6b}$$

$$[Q]_{x=\xi} = \lim_{\delta \to 0} [Q(\xi + \delta) - Q(\xi - \delta)]. \tag{9.5.6c}$$

We must be careful because the "strain energy" $A(v,u)$ is not an inner product since $A(u,u)$ need not be positive definite. We'll use the inner product notation here for convenience.

Integrating the first two terms of (9.5.6b) by parts

$$(v, \mathcal{L}[u]) = (\mathcal{L}^*[v], u) - [\epsilon(v'u - u'v) + \omega v u]_{x=\xi}$$

or, since u and v are continuous

$$(v, \mathcal{L}[u]) = (\mathcal{L}^*[v], u) - [\epsilon(v'u - u'v)]_{x=\xi} \tag{9.5.7a}$$

The differential equation

$$\mathcal{L}^*[v] = -\epsilon v'' - (\omega v)' + qv. \tag{9.5.7b}$$

with the boundary conditions $v(0) = v(1) = 0$ is called the *adjoint problem* and the operator $\mathcal{L}^*[\;]$ is called the *adjoint operator.*

Definition 9.5.1. A *Green's function* $G(\xi, x)$ for the operator $\mathcal{L}[\ ]$ is the continuous function that satisfies

$$\mathcal{L}^*[G(\xi,x)] = -\epsilon G_{xx} - (\omega G)_x + qG = 0, \qquad x \in (0,\xi) \cup (\xi,1), \tag{9.5.8a}$$

$$G(\xi,0) = G(\xi,1) = 0 \tag{9.5.8b}$$

$$[G_x(\xi,x)]_{x=\xi} = -\frac{1}{\epsilon}. \tag{9.5.8c}$$

Evaluating (9.5.7a) with $v(x) = G(\xi,x)$ while using (9.5.5a, 9.5.8) and assuming that $u'(x) \in H^1(0,1)$ gives the familiar relationship

$$u(\xi) = (\mathcal{L}[u], G(\xi,\cdot)) = \int_0^1 G(\xi,x)f(x)dx. \tag{9.5.9a}$$

A more useful expression for our present purposes is obtained by combining (9.5.7a) and (9.5.6a) with $v(x) = G(\xi,x)$ to obtain

$$u(\xi) = A(u, G(\xi,\cdot)). \tag{9.5.9b}$$

As usual, Galerkin and finite element Galerkin problems for (9.5.5a) would consist of determining $u \in H_0^1$ or $U \in S_0^N \subset H_0^1$ such that

$$A(v,u) = (v,f), \qquad \forall v \in H_0^1, \tag{9.5.10a}$$

and

$$A(V,U) = (V,f), \qquad \forall v \in S_0^N. \tag{9.5.10b}$$

Selecting $v = V$ in (9.5.10a) and subtracting (9.5.10b) yields

$$A(V,e) = 0, \qquad \forall v \in S_0^N, \tag{9.5.10c}$$

where

$$e(x) = u(x) - U(x). \tag{9.5.10d}$$

Equation (9.5.9b) did not rely on the continuity of $u'(x)$; hence, it also holds when u is replaced by either U or e. Replacing u by e in (9.5.9b) yields

$$e(\xi) = A(e, G(\xi,\cdot)). \tag{9.5.11a}$$

Subtacting (9.5.10c)

$$e(\xi) = A(e, G(\xi, \cdot) - V). \tag{9.5.11b}$$

Assuming that $A(v, u)$ is continuous in H^1, we have

$$|e(\xi)| \leq C\|e\|_1 \|G(\xi, \cdot) - V\|_1. \tag{9.5.11c}$$

Expressions (9.5.11b,c) relate the local error at a point ξ to the global error. Equation (9.5.11c) also explains superconvergence. From Theorem 7.2.3 we know that $\|e\|_1 = O(h^p)$ when S^N consists of piecewise polynomials of degree p and $u \in H^{p+1}$. The test function V is also an element of S^N; however, $G(\xi, x)$ cannot be approximated to the same precision as u because it may be less smooth. To elaborate further, consider

$$\|G(\xi, \cdot) - V\|_1^2 = \sum_{j=1}^{N} \|G(\xi, \cdot) - V\|_{1,j}^2$$

where

$$\|u\|_{1,j}^2 = \int_{x_j - 1}^{x_j} [(u')^2 + u^2] dx.$$

If $\xi \in (x_{k-1}, x_k)$, $k = 1, 2, \ldots, N$, then the discontinuity in $G_x(\xi, x)$ occurs on some interval and $G(\xi, x)$ cannot be approximated to high order by V. If, on the other hand, $\xi = x_k$, $k = 0, 1, \ldots, N$, then the discontinuity in $G_x(\xi, x)$ is confined to the mesh and $G(\xi, x)$ is smooth on every subinterval. Thus, in this case, the Green's function can be approximated to $O(h^p)$ by the test function V and, using (9.5.11c), we have

$$u(x_k) = Ch^{2p}, \qquad k = 0, 1, \ldots, N. \tag{9.5.12}$$

The solution at the vertices is converging to a much higher order than it is globally.

Equation (9.5.11c) suggests that there are two ways of minimizing the pointwise error. The first is to have U be a good approximation of u and the second is to have V be a good approximation of $G(\xi, x)$. If the problem is not singularly perturbed, then the two conditions are the same. However, when $\epsilon \ll 1$, the behavior of the Green's function is hardly polynomial. Let us consider two simple examples.

Example 9.5.2 [5]. Consider (9.5.5) in the case when $\omega(x) > 0$, $x \in [0, 1]$. Balancing the first two terms in (9.5.5a) implies that there is a boundary layer near $x = 1$; thus, at points other than the right endpoint, the small second derivative terms in (9.5.5) may be neglected and the solution is approximately

$$\omega u_R' + qu_R = f, \qquad 0 < x < 1, \qquad u_R(0) = 0,$$

where u_R is called the *reduced solution.* Near $x = 1$ the reduced solution must be corrected by a *boundary layer* that brings it from its limiting value of $u_R(1)$ to zero. Thus, for $0 < \epsilon \ll 1$, the solution of (9.5.5) is approximately

$$u(x) \sim u_R(x) - u_R(1)e^{-(1-x)\omega(1)/\epsilon}.$$

Similarly, the Green's function (9.5.8) has boundary layers at $x = 0$ and $x = \xi^-$. At points other than these, the second derivative terms in (9.5.8a) may be neglected and the Green's function satisfies the reduced problem

$$-(\omega G_R)' + qG_R = 0, \qquad x \in (0,\xi) \cup (\xi,1), \qquad G_R(\xi,x) \in C(0,1), \qquad G_R(\xi,1) = 0.$$

Boundary layer jumps correct the reduced solution at $x = 0$ and $x = \xi$ and determine an asymptotic approximation of $G(\xi,x)$ as

$$G(\xi,x) \sim c(\xi) \begin{cases} G_R(\xi,x) - G_R(\xi,0)e^{-\omega(0)x/\epsilon}, & \text{if } x \le \xi \\ e^{-(x-\xi)\omega(\xi)/\epsilon}, & \text{if } x > \xi \end{cases}.$$

The function $c(\xi)$ is given in Flaherty and Mathon [5].

Knowing the Green's function, we can construct test functions that approximate it accurately. To be specific, let us write it as

$$G(\xi,x) = \sum_{j=1}^{N} G(\xi,x_j)\psi_j(x) \tag{9.5.13}$$

where $\psi_j(x)$, $j = 0,1,\ldots,N$, is a basis. Let us consider (9.5.5) and (9.5.8) with $\omega > 0$, $x \in [0,1]$. Approximating the Green's function for arbitrary ξ is difficult, so we'll restrict ξ to x_k, $k = 0,1,\ldots,N$, and establish the goal of minimizing the pointwise error of the solution. Mapping each subinterval to a canonical element, the basis $\psi_j(x)$, $x \in (x_{j-1},x_{j+1})$ is

$$\psi_j(x) = \hat{\psi}(\frac{x-x_j}{h}) \tag{9.5.14a}$$

where

$$\hat{\psi}(s) = \begin{cases} \frac{1-e^{-\rho(1+s)}}{1-e^{-\rho}}, & \text{if } -1 \le s < 0 \\ \frac{e^{-\rho s}-e^{-\rho}}{1-e^{-\rho}}, & \text{if } 0 \le s < 1 \\ 0, & \text{otherwise} \end{cases} \tag{9.5.14b}$$

where

$$\rho = \frac{h\bar{\omega}}{\epsilon} \tag{9.5.14c}$$

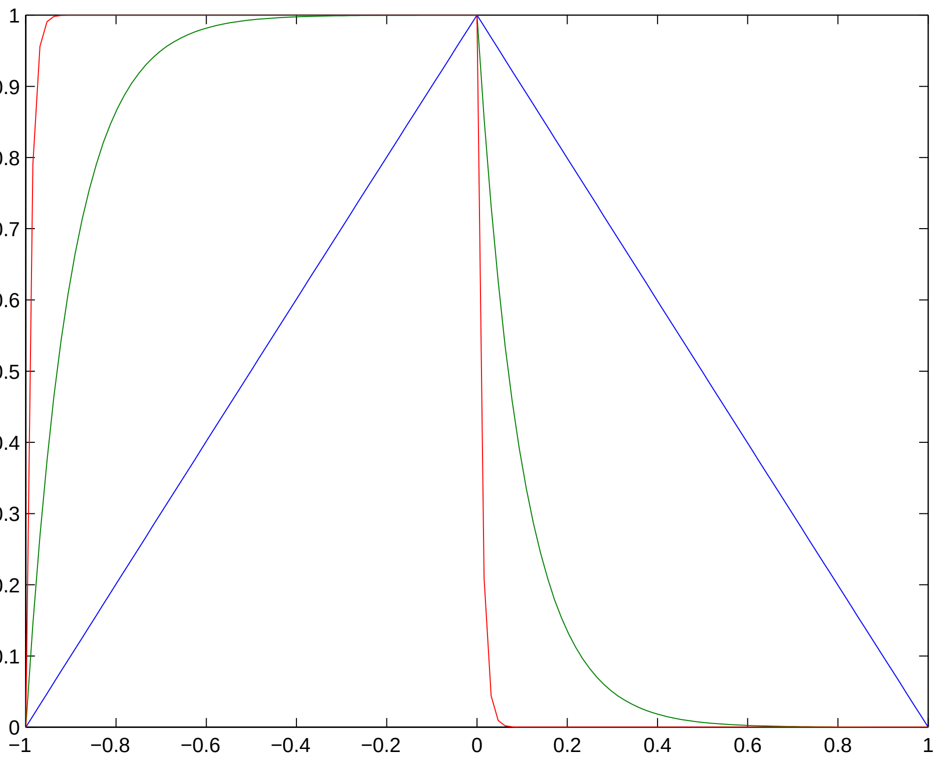


Figure 9.5.2: Canonical basis element $\hat{\psi}(s)$ for $\rho = 0$, 10, and 100 (increasing steepness).

is the cell *Peclet number*. The value of $\bar{\omega}$ will remain undefined for the moment. The canonical basis element $\hat{\psi}(s)$ is illustrated in Figure 9.5.2. As $\rho \to 0$ the basis (9.5.14b) becomes the usual piecewise-linear hat function

$$\hat{\psi}(s) = \frac{1}{2}\begin{cases} 1+s, & \text{if } -1 \le s < 0 \\ 1-s, & \text{if } 0 \le s < 1 \\ 0, & \text{otherwise} \end{cases}$$

As $\rho \to \infty$, (9.5.14b) becomes the piecewise-constant function

$$\hat{\psi}(s) = \begin{cases} 1, & \text{if } -1 < s \le 0 \\ 0, & \text{otherwise} \end{cases} .$$

The limits of this function are nonuniform at $s = -1, 0$.

We're now in a position to apply the Petrov-Galerkin method with $U \in S_0^N$ and $V \in \hat{S}_0^N$ to (9.5.5). The trial space S^N will consist of piecewise linear functions and, for the moment, the test space will remain arbitrary except for the assumptions

$$\psi_j(x) \in H^1[0,1], \qquad \psi_j(x_k) = \delta_{jk}, \qquad \int_{-1}^{1} \hat{\psi}(s)ds = 1, \qquad j,k = 1,2,\ldots,N-1. \tag{9.5.15}$$

The Petrov-Galerkin system for (9.5.5) is

$$\epsilon(\psi_i', U') + (\psi_i, \omega U') + (\psi_i, qU) = (\psi_i, f), \qquad i = 1, 2, \ldots, N-1. \tag{9.5.16}$$

Let us use node-by-node evaluation of the inner products in (9.5.16). For simplicity, we'll assume that the mesh is uniform with spacing h and that ω and q are constant. Then

$$\epsilon(\psi_i', U') = \frac{\epsilon}{h}\int_{-1}^{1} \hat{\psi}'(s)\hat{U}'(s)ds$$

where $\hat{U}(s)$ is the mapping of $U(x)$ onto the canonical element $-1 \leq s \leq 1$. With a piecewise linear basis for $\hat{U}$ and the properties noted in (9.5.15) for ψ_j, we find

$$\epsilon(\psi_i', U') = -\frac{\epsilon}{h}\delta^2 c_i. \tag{9.5.17a}$$

We've introduced the central difference operator

$$\delta c_i = c_{i+1/2} - c_{i-1/2} \tag{9.5.17b}$$

for convenience. Thus,

$$\delta^2 c_i = \delta(\delta c_i) = c_{i+1} - 2c_i + c_{i-1}. \tag{9.5.17c}$$

Considering the convective term,

$$\omega(\psi_i, U') = \omega\int_{-1}^{1} \hat{\psi}(s)\hat{U}'(s)ds = \omega(\mu\delta - \gamma\delta^2/2)c_i \tag{9.5.18a}$$

where μ is the averaging operator

$$\mu c_i = (c_{i+1/2} + c_{i-1/2})/2. \tag{9.5.18b}$$

Thus,

$$\mu\delta c_i = \mu(\delta c_i) = (c_{i+1} - c_{i-1})/2. \tag{9.5.18c}$$

Additionally,

$$\gamma = -\int_{0}^{1} [\hat{\psi}(s) - \hat{\psi}(-s)]ds \tag{9.5.18d}$$

Similarly

$$q(\psi_i, U) = qh\int_{-1}^{1} \hat{\psi}(s)\hat{U}(s)ds = qh(1 - \beta\mu\delta + \alpha\delta^2/2)c_i \tag{9.5.19a}$$

where

$$\alpha = \int_{-1}^{1} |s|\hat{\psi}(s)ds, \tag{9.5.19b}$$

$$\beta = -\int_{-1}^{1} s\hat{\psi}(s)ds. \tag{9.5.19c}$$

Finally, if $f(x)$ is approximated by a piecewise-linear polynomial, we have

$$(\psi_i, f) \approx h(1 - \beta\mu\delta + \alpha\delta^2/2)f_i \tag{9.5.20}$$

where $f_i = f(x_i)$.

Substituting (9.5.17a), (9.5.18a), (9.5.19a), and (9.5.20) into (9.5.16) gives a difference equation for c_k, $k = 1, 2, \ldots, N-1$. Rather than facing the algebraic complexity, let us continue with the simpler problem of Example 9.5.1.

Example 9.5.3. Consider the boundary value problem (9.5.2). Thus, $q = f(x) = 0$ in (9.5.17-9.5.20) and we have

$$\epsilon(\psi_i', U') + \omega(\psi_i, U') = -\frac{\epsilon}{h}\delta^2 c_i + \omega(\mu\delta - \gamma\delta^2/2)c_i, \qquad i = 1, 2, \ldots, N-1, \tag{9.5.21a}$$

or, using (9.5.14c), (9.5.17c), and (9.5.18c)

$$-\frac{1}{2}(\gamma + \frac{2}{\rho})(c_{i+1} - 2c_i + c_{i-1}) + \frac{c_{i+1} - c_{i-1}}{2} = 0, \qquad i = 1, 2, \ldots, N-1. \tag{9.5.21b}$$

This is to be solved with the boundary conditions

$$c_0 = 1, \qquad c_N = 2. \tag{9.5.21c}$$

The exact solution of this second-order constant-coefficient difference equation is

$$c_i = 1 + \frac{1 - \zeta^i}{1 - \zeta^N}, \qquad i = 0, 1, \ldots, N. \tag{9.5.22a}$$

where

$$\zeta = \frac{\gamma + 2/\rho + 1}{\gamma + 2/\rho - 1}. \tag{9.5.22b}$$

In order to avoid the spurious oscillations found in Example 9.5.1, we'll insist that $\zeta > 0$. Using (9.5.22b), we see that this requires

$$\gamma > \text{sgn}\rho - \frac{2}{\rho}. \tag{9.5.22c}$$

Some specific choices of γ follow:

1. *Galerkin's method*, $\gamma = 0$. In this case,
$$\hat{\psi}(s) = \hat{\phi}(s) = \frac{1-|s|}{2}.$$
Using (9.5.22), this method is oscillation free when
$$\frac{2}{|\rho|} > 1.$$
From (9.5.14c), this requires $h < 2|\epsilon/\omega|$. For small values of $|\epsilon/\omega|$, this would be too restrictive.

2. *Il'in's scheme*. In this case, $\hat{\psi}(s)$ is given by (9.5.14b) and
$$\gamma = \coth\frac{\rho}{2} - \frac{2}{\rho}.$$
This scheme gives the exact solution at element vertices for all values of ρ. Either this result or the use of (9.5.22c) indicates that the solution will be oscillation free for all values of ρ. This choice of γ is shown with the function $1 - 2/\rho$ in Figure 9.5.3.

3. *Upwind differencing*, $\gamma = \text{sgn}\rho$. When $\rho > 0$, the shape function $\hat{\psi}(s)$ is the piecewise constant function
$$\hat{\psi}(s) = \begin{cases} 1, & \text{if } -1 < s \le 0 \\ 0, & \text{otherwise} \end{cases}.$$
This function is discontinuous; however, finite element solutions still converge. With $\gamma = 1$, (9.5.22b) becomes
$$\zeta = \frac{2(1+1/\rho)}{2/\rho}.$$
In the limit as $\rho \to \infty$, we have $\zeta \approx \rho$; thus, using (9.5.22a)
$$c_i \sim 1 - \rho^{-(N-i)}, \qquad i = 0, 1, \ldots, N, \qquad \rho \gg 1.$$
This result is a good asymptotic approximation of the true solution.

Examining (9.5.21) as a finite difference equation, we see that positive values of γ can be regarded as adding dissipation to the system.

This approach can also be used for variable-coefficient problems and for nonuniform mesh spacing. The cell Peclet number would depend on the local value of ω and the mesh spacing in this case and could be selected as
$$\rho_j = \frac{h_j \bar{\omega}_j}{\epsilon} \tag{9.5.23}$$

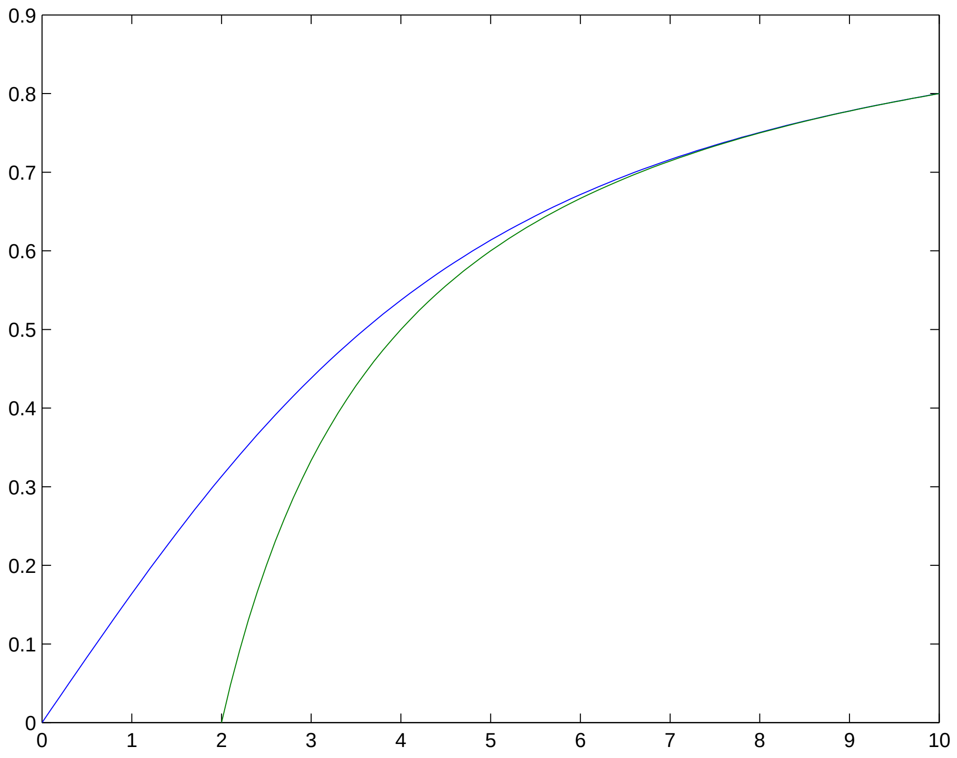


Figure 9.5.3: The upwinding parameter $\gamma = \coth \rho/2 - 2/\rho$ for Il'in's scheme (upper curve) and the function $1 - 2/\rho$ (lower curve) vs. ρ.

where $h_j = x_j - x_{j-1}$ and $\bar{\omega}_j$ is a characteristic value of $\omega(x)$ when $x \in [x_{j-1}, x_j)$, *e.g.*, $\bar{\omega}_j = \mu\omega_{j+1/2}$. Upwind differencing is too diffusive for many applications. Il'in's scheme offers advantages, but it is difficult to extend to problems other than (9.5.5).

The Petrov-Galerkin technique has also been applied to transient problems of ther form (9.5.1); however, the results of applying Il'in's scheme to transient problems have more diffusion than when it is applied to steady problems.

Example 9.5.4 [4]. Consider Burgers's equation

$$\epsilon u_{xx} - u u_x = 0, \qquad 0 < x < 1,$$

with the Dirichlet boundary conditions selected so that the exact solution is

$$u(x) = \tanh \frac{1-x}{\epsilon}.$$

Burgers's equation is often used as a test problem because it is a nonlinear problem with a known exact solution that has a behavior found in more complex problems. Flaherty [4] solved problems with $h/\epsilon = 6,500$ and $N = 20$ using upwind differencing and Il'in's scheme (the Petrov-Galerkin method with the exponential weighting given by (9.5.14b)).

h/ϵ	Maximum Error	
	Upwind	Exponential
6	0.124	0.0766
500	0.00200	0.00100

Table 9.5.1: Maximum pointwise errors for the solution of Example 9.5.4 using upwind differencing ($\gamma = \text{sgn}\rho$) and exponential weighting ($\gamma = \coth \rho/2 - 2/\rho$) [4].

The cell Peclet number (9.5.23) used

$$\bar{\omega}_j = \begin{cases} U(x_j), & \text{if } \mu U_{j-1/2} < 0 \\ \mu U(x_{j-1/2}), & \text{if } \mu U_{j-1/2} = 0 \\ U(x_j - 1), & \text{if } \mu U_{j-1/2} > 0 \end{cases}.$$

The nonlinear solution is obtained by iteration with the values of $U(x)$ evaluated at the beginning of an iterative step.

The results for the pointwise error

$$|e|_\infty = \max_{0 \le j \le N} |u(x_j) - U(x_j)|$$

are shown in Table 9.5.1. The value of $h/\epsilon = 6$ is approximately where the greatest difference between upwind differencing ($\gamma = \text{sgn}\rho$) and exponential weighting ($\gamma = \coth \rho/2 - 2/\rho$) exists. Differences between the two methods decrease for larger and smaller values of h/ϵ.

The solution of convection-diffusion problems is still an active research area and much more work is needed. This is especially the case in two and three dimensions. Those interested in additional material may consult Roos *et al.* [10].

Problems

1. Consider (9.5.5) when $\omega(x) \equiv$, $q(x) > 0$, $x \in [0, 1]$ [5].

   1.1. Show that the solution of (9.5.5) is asymptotically given by

   $$u(x) \approx \frac{f(x)}{q(x)} - u_R(0)e^{-x\sqrt{q(0)/\epsilon}} - u_R(1)e^{-(1-x)\sqrt{q(1)/\epsilon}}.$$

   Thus, the solution has $O(\sqrt{\epsilon})$ boundary layers at both $x = 0$ and $x = 1$.

   1.2. In a similar manner, show that the Green's function is asymptotically given by

   $$G(\xi, x) \sim \frac{1}{2[\epsilon^2 q(x)q(\xi)]^{1/4}} \begin{cases} e^{-(\xi-x)\sqrt{q(\xi)/\epsilon}}, & \text{if } x \le \xi \\ e^{-(x-\xi)\sqrt{q(\xi)/\epsilon}}, & \text{if } x > \xi \end{cases}.$$

   The Green's function is exponentially small away from $x = \xi$, where it has two boundary layers. The Green's function is also unbounded as $O(\epsilon^{-1/2})$ at $x = \xi$ as $\epsilon \to 0$.

Bibliography

[1] S. Adjerid, M. Aiffa, and J.E. Flaherty. Computational methods for singularly perturbed systems. In J.D. Cronin and Jr. R.E. O'Malley, editors, *Analyzing Multiscale Phenomena Using Singular Perturbation Methods*, volume 56 of *Proceedings of Symposia in Applied Mathematics*, pages 47–83, Providence, 1999. AMS.

[2] U.M. Ascher and L.R. Petzold. *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations.* SIAM, Philadelphia, 1998.

[3] K.E. Brenan, S.L Campbell, and L.R. Petzold. *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations.* North Holland, New York, 1989.

[4] J.E. Flaherty. A rational function approximation for the integration point in exponentially weighted finite element methods. *International Journal of Numerical Methods in Engineering*, 18:782–791, 1982.

[5] J.E. Flaherty and W. Mathon. Collocation with polynomial and tension splines for singularly-perturbed boundary value problems. *SIAM Journal on Scie3ntific and Statistical Computation*, 1:260–289, 1990.

[6] C.W. Gear. *Numerical Initial Value Problems in Ordinary Differential Equations.* Prentice Hall, Englewood Cliffs, 1971.

[7] E. Hairer, S.P. Norsett, and G. Wanner. *Solving Ordinary Differential Equations I: Nonstiff Problems.* Springer-Verlag, Berlin, second edition, 1993.

[8] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II: Stiff and Differential Algebraic Problems.* Springer-Verlag, Berlin, 1991.

[9] C. Johnson. *Numerical Solution of Partial Differential Equations by the Finite Element method.* Cambridge, Cambridge, 1987.

[10] H.-G. Roos, M. Stynes, and L. Tobiska. *Numerical Methods for Singularly Perturbed Differential Equations.* Springer-Verlag, Berlin, 1996.

Chapter 10

Hyperbolic Problems

10.1 Conservation Laws

We have successfully applied finite element methods to elliptic and parabolic problems; however, hyperbolic problems will prove to be more difficult. We got an inkling of this while studying convection-diffusion problems in Section 9.5. Conventional Galerkin methods required the mesh spacing h to be on the order of the diffusivity ϵ to avoid spurious oscillations. The convection-diffusion equation (9.5.1) changes type from parabolic to hyperbolic in the limit as $\epsilon \to 0$. The boundary layer also leads to a jump discontinuity in this limit. Thus, a vanishingly small mesh spacing will be required to avoid oscillations, at least when discontinuities are present. We'll need to overcome this limitation for finite element methods to be successful with hyperbolic problems.

Instead of the customary second-order scalar differential equation, let us consider hyperbolic problems as first-order vector systems. Let us confine our attention to conservation laws in one space dimension which typically have the form

$$\mathbf{u}_t + \mathbf{f}(\mathbf{u})_x = \mathbf{b}(x,t,\mathbf{u}), \tag{10.1.1a}$$

where

$$\mathbf{u}(x,t) = \begin{bmatrix} u_1(x,t) \\ u_2(x,t) \\ \vdots \\ u_m(x,t) \end{bmatrix}, \qquad \mathbf{f}(\mathbf{u}) = \begin{bmatrix} f_1(\mathbf{u}) \\ f_2(\mathbf{u}) \\ \vdots \\ f_m(\mathbf{u}) \end{bmatrix}, \qquad \mathbf{b}(x,t,\mathbf{u}) = \begin{bmatrix} b_1(x,t,\mathbf{u}) \\ b_2(x,t,\mathbf{u}) \\ \vdots \\ b_m(x,t,\mathbf{u}) \end{bmatrix} \tag{10.1.1b}$$

are m-dimensional density, flux, and load vectors, respectively. It's also convenient to write (10.1.1a) as

$$\mathbf{u}_t + \mathbf{A}(\mathbf{u})\mathbf{u}_x = \mathbf{b}(x,t,\mathbf{u}) \tag{10.1.2a}$$

where the system Jacobian is the $m \times m$ matrix

$$\mathbf{A}(\mathbf{u}) = \mathbf{f}(\mathbf{u})_{\mathbf{u}}. \tag{10.1.2b}$$

Equation (10.1.1a) is called the *conservative form* and (10.1.2a) is called the *convective form* of the partial differential system.

Conditions under which (10.1.1) and (10.1.2) are of hyperbolic type follow.

Definition 10.1.1. If $\mathbf{A}$ has m real and distinct eigenvalues $\lambda_1 < \lambda_2 < \ldots < \lambda_m$ and, hence, m linearly independent eigenvectors $\mathbf{p}^{(1)}, \mathbf{p}^{(2)}, \ldots, \mathbf{p}^{(m)}$, then (10.1.2a) is said to be *hyperbolic.*

Physical problems where dissipative effects can be neglected often lead to hyperbolic systems. Areas where these arise include acoustics, dynamic elasticity, electromagnetics, and gas dynamics. Here are some examples.

Example 10.1.1. The Euler equations for one-dimensional compressible inviscid flows satisfy

$$\rho_t + m_x = 0, \tag{10.1.3a}$$

$$m_t + (\frac{m^2}{\rho} + p)_x = 0, \tag{10.1.3b}$$

$$e_t + [(e+p)\frac{m}{\rho}]_x = 0. \tag{10.1.3c}$$

Here ρ, m, e, and p are, respectively, the fluid's density, momentum, internal energy, and pressure. The fluid velocity $u = m/\rho$ and the pressure is determined by an equation of state, which, for an ideal fluid is

$$p = (\gamma - 1)[e - \frac{m^2}{2\rho}], \tag{10.1.3d}$$

where γ is a constant. Equations (10.1.3a), (10.1.3b), and (10.1.3c) express the facts that the mass, momentum, and energy of the fluid are neither created nor destroyed and are, hence, *conserved.* We readily see that the system (10.1.3) has the form of (10.1.1) with

$$\mathbf{u} = \begin{bmatrix} \rho \\ m \\ e \end{bmatrix}, \qquad \mathbf{f}(\mathbf{u}) = \begin{bmatrix} m \\ m^2/\rho + p \\ (e+p)m/\rho \end{bmatrix}, \qquad \mathbf{b}(x,t,\mathbf{u}) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}. \tag{10.1.4}$$

Example 10.1.2. The deflection of a taut string has the form

$$u_{tt} = a^2 u_{xx} + q(x), \tag{10.1.5a}$$

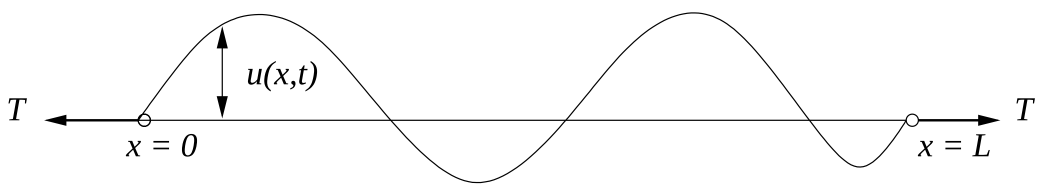


Figure 10.1.1: Geometry of the taut string of Example 10.1.2.

where $a^2 = T/\rho$ with T being the tension and ρ being the linear density of the string (Figure 10.1.1). The lateral loading $q(x)$ applied in the transverse direction could represent the weight of the string.

This second-order partial differential equation can be written as a first-order system of two equations in a variety of ways. Perhaps the most common approach is to let

$$u_1 = u_t, \qquad u_2 = au_x. \tag{10.1.5b}$$

Physically, $u_1(x,t)$ is the velocity and $u_2(x,t)$ is the stress at point x and time t in the string. Differentiating with respect to t while using (10.1.5a) and (10.1.5b) yields

$$(u_1)_t = u_{tt} = a^2 u_{xx} + q(x) = a(u_2)_x + q(x), \qquad (u_2)_t = au_{xt} = au_{tx} = a(u_1)_x.$$

Thus, the one-dimensional wave equation has the form of (10.1.1) with

$$\mathbf{u} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}, \qquad \mathbf{f}(\mathbf{u}) = \begin{bmatrix} -cu_2 \\ -cu_1 \end{bmatrix}, \qquad \mathbf{b}(x,t,\mathbf{u}) = \begin{bmatrix} q(x) \\ 0 \end{bmatrix}. \tag{10.1.5c}$$

In the convective form (10.1.2), we have

$$\mathbf{A} = \begin{bmatrix} 0 & -a \\ -a & 0 \end{bmatrix}. \tag{10.1.5d}$$

10.1.1 Characteristics

The behavior of the system (10.1.1) can be determined by diagonalizing the Jacobian (10.1.2b). This can be done for hyperbolic systems since $\mathbf{A}(\mathbf{u})$ has m distinct eigenvalues (Definition 10.1.1). Thus, let

$$\mathbf{P} = [\mathbf{p}^{(1)}, \mathbf{p}^{(2)}, \ldots, \mathbf{p}^{(m)}] \tag{10.1.6a}$$

and recall the eigenvalue-eigenvector relation

$$\mathbf{AP} = \mathbf{P\Lambda}, \tag{10.1.6b}$$

where

$$\mathbf{\Lambda} = \begin{bmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_m \end{bmatrix} \tag{10.1.6c}$$

Multiplying (10.1.2a) by $\mathbf{P}^{-1}$ and using (10.1.6b) gives

$$\mathbf{P}^{-1}\mathbf{u}_t + \mathbf{P}^{-1}\mathbf{A}\mathbf{u}_x = \mathbf{P}^{-1}\mathbf{u}_t + \mathbf{\Lambda}\mathbf{P}^{-1}\mathbf{u}_x = \mathbf{P}^{-1}\mathbf{b}.$$

Let

$$\mathbf{w} = \mathbf{P}^{-1}\mathbf{u} \tag{10.1.7}$$

so that

$$\mathbf{w}_t + \mathbf{\Lambda}\mathbf{w}_x = \mathbf{P}^{-1}\mathbf{u}_t + (\mathbf{P}^{-1})_t\mathbf{u} + \mathbf{\Lambda}[\mathbf{P}^{-1}\mathbf{u}_x + (\mathbf{P}^{-1})_x\mathbf{u}].$$

Using (10.1.7)

$$\mathbf{w}_t + \mathbf{\Lambda}\mathbf{w}_x = \mathbf{Q}\mathbf{w} + \mathbf{g}, \tag{10.1.8a}$$

where

$$\mathbf{Q} = [(\mathbf{P}^{-1})_t + \mathbf{\Lambda}(\mathbf{P}^{-1})_x]\mathbf{P}, \qquad \mathbf{g} = \mathbf{P}^{-1}\mathbf{b}. \tag{10.1.8b}$$

In component form, (10.1.8a) is

$$(w_i)_t + \lambda_i(w_i)_x = \sum_{j=1}^{m} q_{i,j}w_j + g_i, \qquad i = 1, 2, \ldots, m. \tag{10.1.8c}$$

Thus, the transformation (10.1.7) has uncoupled the differentiated terms of the original system (10.1.2a).

Consider the directional derivative of each component w_i, $i = 1, 2, \ldots, m$, of $\mathbf{w}$,

$$\frac{dw_i}{dt} = (w_i)_t + (w_i)_x\frac{dx}{dt}, \qquad i = 1, 2, \ldots, m,$$

in the directions

$$\frac{dx}{dt} = \lambda_i, \qquad i = 1, 2, \ldots, m, \tag{10.1.9a}$$

and use (10.1.8c) to obtain

$$\frac{dw_i}{dt} = \sum_{j=1}^{m} q_{i,j}w_j + g_i, \qquad i = 1, 2, \ldots, m. \tag{10.1.9b}$$

The curves (10.1.9a) are called the *characteristics* of the system (10.1.1, 10.1.2). The partial differential equations (10.1.2) may be solved by integrating the $2m$ ordinary differential equations (10.1.9a, 10.1.9b). This system is uncoupled through its differentiated terms but coupled through $\mathbf{Q}$ and $\mathbf{g}$. This method of solution is, quite naturally, called the method of characteristics. While we could develop numerical methods based on the method of characteristics, they are generally not efficient when $m > 2$.

Definition 10.1.2. The set of all points that determine the solution at a point $P(x_0, t_0)$ is called the *domain of dependence* of P.

Consider the arbitrary point $P(x_0, t_0)$ and the characteristics passing through it as shown in Figure 10.1.2. The solution $u(x_0, t_0)$ depends on the initial data on the interval $[A, B]$ and on the values of $\mathbf{b}$ in the region APB, bounded by $[A, B]$ and the characteristic curves $\dot{x} = \lambda_1$ and $\dot{x} = \lambda_m$. Thus, the region APB is the domain of dependence of P.

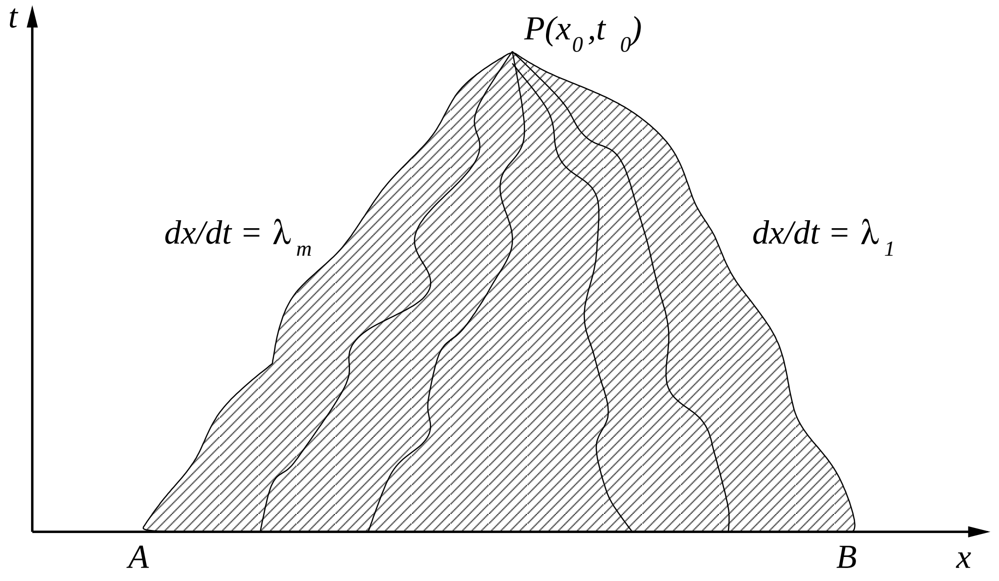


Figure 10.1.2: Domain of dependence of a point $P(x_0, t_0)$. The solution at P depends on the initial data on the line $[A, B]$ and the values of $\mathbf{b}$ within the region APB bounded by the characteristic curves $dx/dt = \lambda_1, \lambda_m$.

Example 10.1.3. Consider an initial value problem for the forced wave equation (10.1.5a) with the initial data

$$u(x, 0) = u^0(x), \qquad u_t(x, 0) = \dot{u}^0(x), \qquad -\infty < x < \infty.$$

Transforming (10.1.5a) using (10.1.5b) yields the first-order system (10.1.2) with $\mathbf{A}$ and $\mathbf{b}$ given by (10.1.5). Using (10.1.5b), The initial conditions become

$$u_1(x, 0) = \dot{u}^0(x), \qquad u_2(x, 0) = au_x^0(x), \qquad -\infty < x < \infty.$$

With $\mathbf{A}$ given by (10.1.5), we find its eigenvalues as $\lambda_{1,2} = \pm a$. Thus, the characteristics are

$$\dot{x} = \pm a,$$

and the eigenvectors are

$$\mathbf{P} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}.$$

Since $\mathbf{P}^{-1} = \mathbf{P}$, we may use (10.1.7) to determine the canonical variables as

$$w_1 = \frac{u_1 + u_2}{\sqrt{2}}, \qquad w_2 = \frac{u_1 - u_2}{\sqrt{2}}.$$

From (10.1.8), the canonical form of the problem is

$$(w_1)_t - a(w_1)_x = \frac{q}{\sqrt{2}}, \qquad (w_2)_t + a(w_2)_x = \frac{q}{\sqrt{2}}.$$

The characteristics integrate to

$$x = x_0 - at, \qquad x = x_0 + at,$$

and along the characteristics, we have

$$\frac{dw_k}{dt} = \frac{q}{\sqrt{2}}, \qquad k = 1, 2.$$

Integrating, we find

$$w_1(x,t) = w_1^0(x_0) + \frac{1}{\sqrt{2}} \int_0^t q(x_0 - a\tau) d\tau$$

or

$$w_1(x,t) = w_1^0(x_0) - \frac{1}{a\sqrt{2}} \int_{x_0}^{x_0 - at} q(\xi) d\xi.$$

It's usual to eliminate x_0 by using the characteristic equation to obtain

$$w_1(x,t) = w_1^0(x + at) - \frac{1}{a\sqrt{2}} \int_{x+at}^{x} q(\xi) d\xi.$$

Likewise

$$w_2(x,t) = w_2^0(x - at) + \frac{1}{a\sqrt{2}} \int_{x-at}^{x} q(\xi) d\xi.$$

The domain of dependence of a point $P(x_0, t_0)$ is shown in Figure 10.1.3. Using the bounding characteristics, it is the triangle connecting the points (x_0, t_0), $(x_0 - at_0, 0)$, and $(x_0 + at_0, 0)$. (Actually, with q being a function of x only, the domain of dependence only involves values of $q(x)$ on the subinterval $(x_0 - at_0, 0)$ to $(x_0 + at_0, 0)$.)

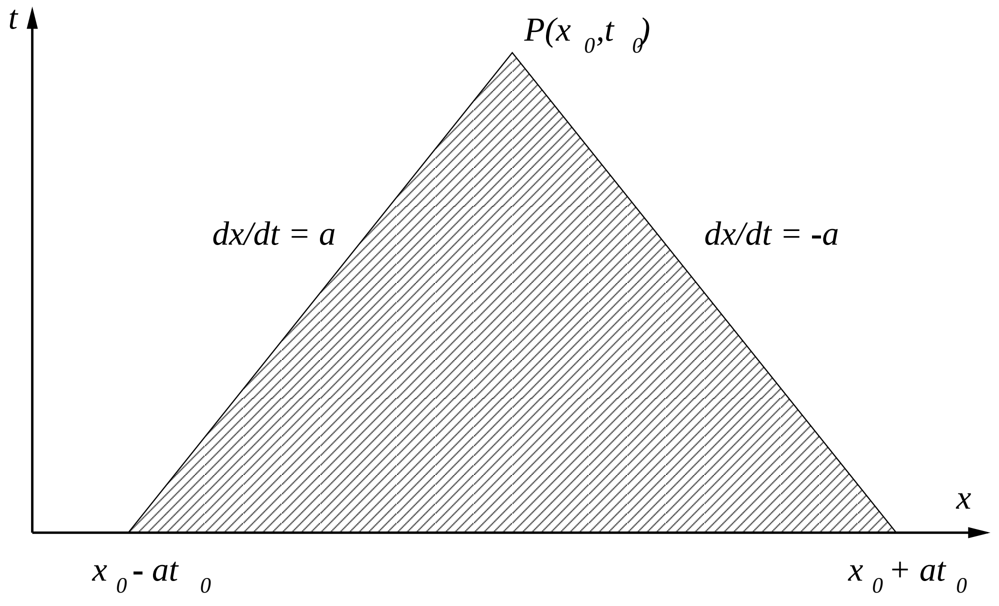


Figure 10.1.3: The domain of dependence of a point $P(x_0, t_0)$ for Example 10.1.3 is the triangle connecting the points P, $(x_0 - at_0, 0)$, and $(x_0 + at_0, 0)$.

Transforming back to the physical variables

$$u_1(x,t) = \frac{1}{\sqrt{2}}(w_1 + w_2) = \frac{1}{\sqrt{2}}[w_1^0(x+at) + w_2^0(x-at)] + \frac{1}{2a}\int_{x-at}^{x+at} q(\xi)d\xi,$$

$$u_2(x,t) = \frac{1}{\sqrt{2}}(w_1 - w_2) = \frac{1}{\sqrt{2}}[w_1^0(x+at) - w_2^0(x-at)] - \frac{1}{2a}[\int_{x+at}^{x} q(\xi)d\xi + \int_{x-at}^{x} q(\xi)d\xi].$$

Suppose, for simplicity, that $\dot{u}^0(x) = 0$, then

$$u_1(x,0) = 0 = \frac{1}{\sqrt{2}}[w_1^0(x) + w_2^0(x)],$$

$$u_2(x,0) = au_x^0(x) = \frac{1}{\sqrt{2}}[w_1^0(x) - w_2^0(x)].$$

Thus,

$$w_1^0(x) = -w_2^0(x) = \frac{au_x^0(x)}{\sqrt{2}},$$

and

$$u_1(x,t) = \frac{a}{2}[u_x^0(x+at) - u_x^0(x-at)] + \frac{1}{2a}\int_{x-at}^{x+at} q(\xi)d\xi,$$

$$u_2(x,t) = \frac{a}{2}[u_x^0(x+at) + u_x^0(x-at)] - \frac{1}{2a}[\int_{x+at}^{x} q(\xi)d\xi + \int_{x-at}^{x} q(\xi)d\xi].$$

Since $u_2 = au_x$, we can integrate to find the solution in the original variables. In order to simplify the manipulations, let's do this with $q(x) = 0$. In this case, we have

$$u_2(x,t) = \frac{a}{2}[u_x^0(x+at) + u_x^0(x-at)];$$

hence,

$$u(x,t) = \frac{1}{2}[u^0(x+at) + u^0(x-at)].$$

The solution for an initial value problem when

$$u^0(x) = \begin{cases} x+1, & \text{if } -1 \leq x \leq 0 \\ 1-x, & \text{if } 0 \leq x \leq 1 \\ 0, & \text{otherwise} \end{cases}$$

is shown in Figure 10.1.4. The initial data splits into two waves having half the initial amplitude and traveling in the positive and negative x directions with speeds a and $-a$, respectively.

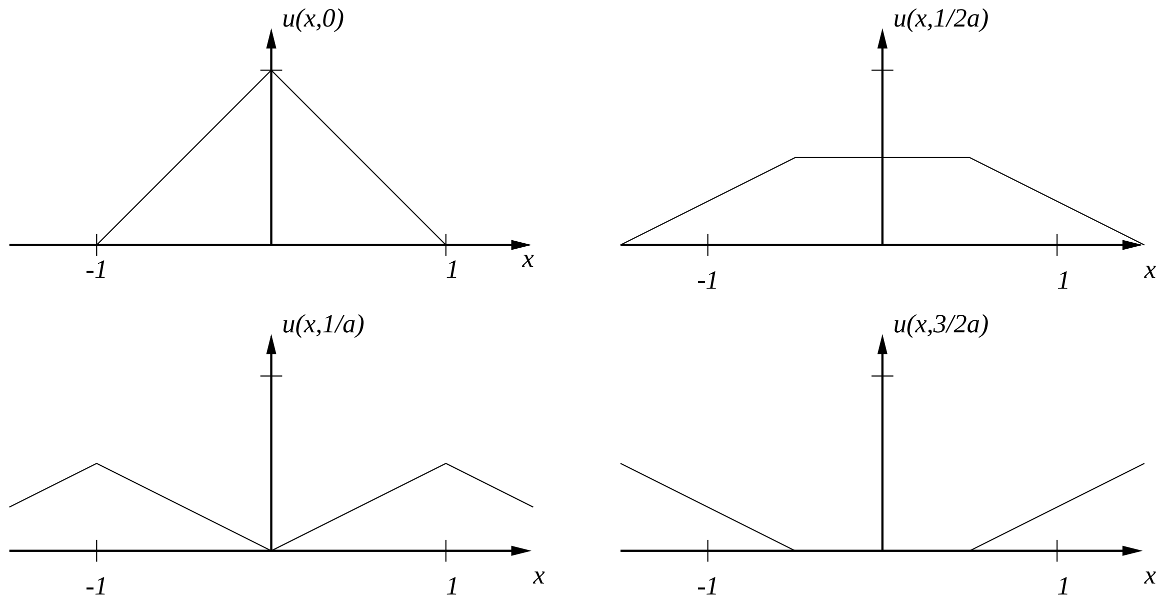


Figure 10.1.4: Solution of Example 10.1.3 at $t = 0$ (upper left), $1/2a$ (upper right), $1/a$ (lower left), and $3/2a$ (lower right).

10.1.2 Rankine-Hugoniot Conditions

For simplicity, let us neglect $\mathbf{b}(x, t, \mathbf{u})$ in (10.1.1a) and consider the integral form of the conservation law

$$\frac{d}{dt}\int_{\alpha}^{\beta} \mathbf{u} dx = -\mathbf{f}(\mathbf{u})|_{\alpha}^{\beta} = -\mathbf{f}(\mathbf{u}(\beta,t)) + \mathbf{f}(\mathbf{u}(\alpha,t)), \tag{10.1.10}$$

which states that the rate of change of $\mathbf{u}$ within the interval $\alpha \leq x \leq \beta$ is equal to the change in its flux through the boundaries $x = \alpha$, β.

If $\mathbf{f}$ and $\mathbf{u}$ are smooth functions, then (10.1.10) can be written as

$$\int_{\alpha}^{\beta} [\mathbf{u}_t + \mathbf{f}(\mathbf{u})_x] dx = 0.$$

If this result is to hold for all "control volumes" (α, β), the integrand must vanish, and, hence, (10.1.1a) and (10.1.10) are equivalent.

To further simplify matters, let confine our attention to the scalar conservation law

$$u_t + f(u)_x = 0 \tag{10.1.11a}$$

with

$$a(u) = \frac{df(u)}{du}, \tag{10.1.11b}$$

and

$$u_t + a(u)u_x = 0. \tag{10.1.11c}$$

The characteristic equation is

$$\frac{dx}{dt} = \lambda = a(u). \tag{10.1.12a}$$

The scalar equation (10.1.11c) is already in the canonical form (10.1.8a). We calculate the directional derivative on the characteristic as

$$\frac{du}{dt} = u_t dt + u_x \frac{dx}{dt} = u_t + a(u)u_x = 0. \tag{10.1.12b}$$

Thus, in this homogeneous scalar case, $u(x,t)$ is constant along the characteristic curve (10.1.9a).

For an initial value problem for (10.1.11a) on $-\infty < x < \infty$, $t > 0$, the solution would have to satisfy the initial condition

$$u(x,0) = u^0(x), \qquad -\infty < x < \infty. \tag{10.1.13}$$

Since u is constant along characteristic curves, it must have the same value that it had initially. Thus, $u = u^0(x_0) \equiv u_0^0$ along the characteristic that passes through $(x_0, 0)$. From (10.1.12a), we see that this characteristic satisfies the ordinary initial value problem

$$\frac{dx}{dt} = a(u_0^0), \qquad t > 0, \qquad x(0) = x_0. \tag{10.1.14}$$

Integrating, we determine that the characteristic is the straight line

$$x = x_0 + a(u_0^0)t. \tag{10.1.15}$$

This procedure can be repeated to trace other characteristics and thereby construct the solution.

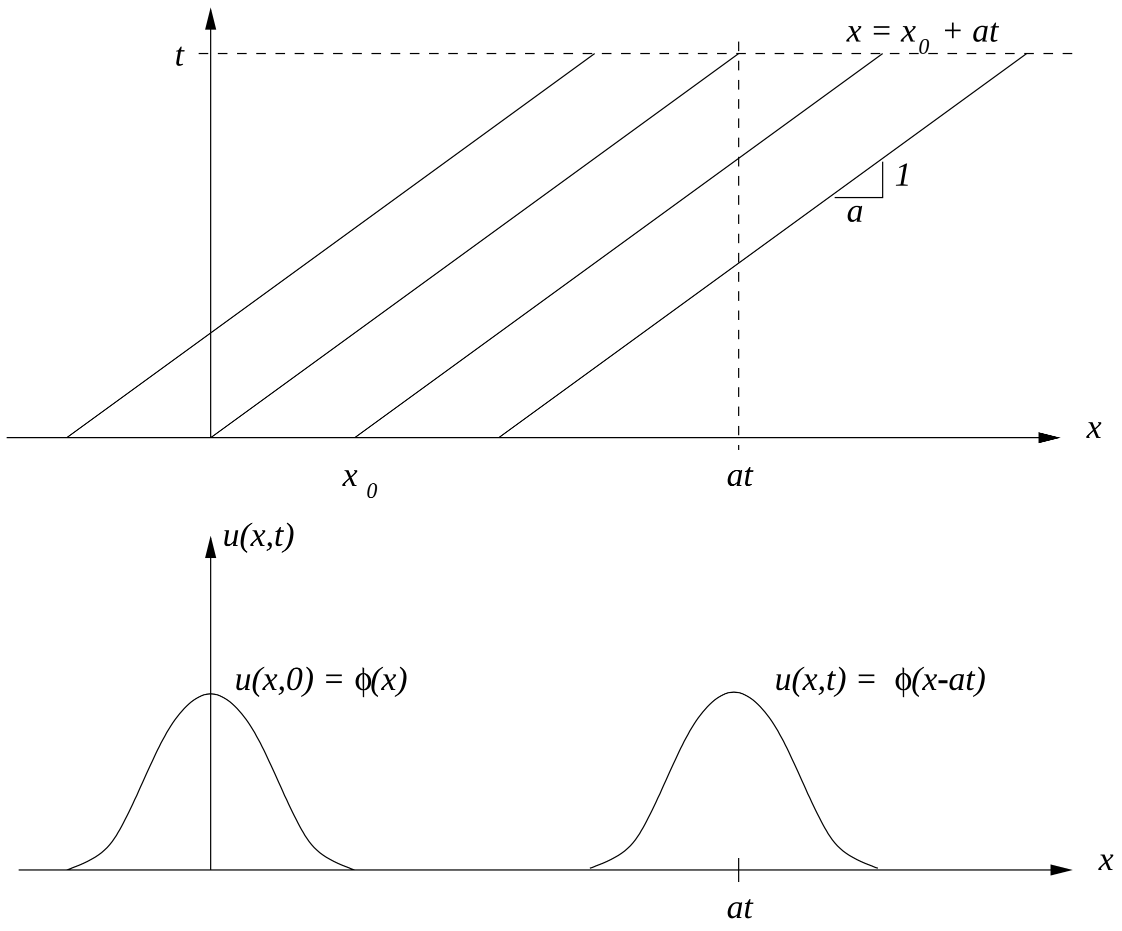


Figure 10.1.5: Characteristic curves and solution of the initial value problem (10.1.11a, 10.1.13) when a is a constant.

Example 10.1.4. The simplest case occurs when a is a constant and $f(u) = au$. All of the characteristics are parallel straight lines with slope $1/a$. The solution of the initial value problem (10.1.11a, 10.1.13) is $u(x,t) = u^0(x - at)$ and is, as shown in Figure 10.1.5, a wave that maintains its shape and travels with speed a.

Example 10.1.5. Setting $a(u) = u$ and $f(u) = u^2/2$ in (10.1.11a, 10.1.11b) yields the inviscid Burgers' equation

$$u_t + \frac{1}{2}(u^2)_x = 0. \tag{10.1.16}$$

Again, consider an initial value problem having the initial condition (10.1.13), so the characteristic is given by (10.1.15) with $a_0 = u(x_0, 0) = u^0(x_0)$, *i.e.*,

$$x = x_0 + u^0(x_0)t. \tag{10.1.17}$$

The characteristics are straight lines with a slope that depends on the value of the initial data; thus, the characteristic passing through the point $(x_0, 0)$ has slope $1/u^0(x_0)$.

The fact that the characteristics are not parallel introduces a difficulty that was not present in the linear problem of Example 10.1.4. Consider characteristics passing through $(x_0, 0)$ and $(x_1, 0)$ and suppose that $u^0(x_0) > u^0(x_1)$ for $x_1 > x_0$. Since the slope of the characteristic passing through $(x_0, 0)$ is less than the slope of the one passing through $(x_1, 0)$, the two characteristics will intersect at a point, say, P as shown in Figure 10.1.6. The solution would appear to be multivalued at points such as P.

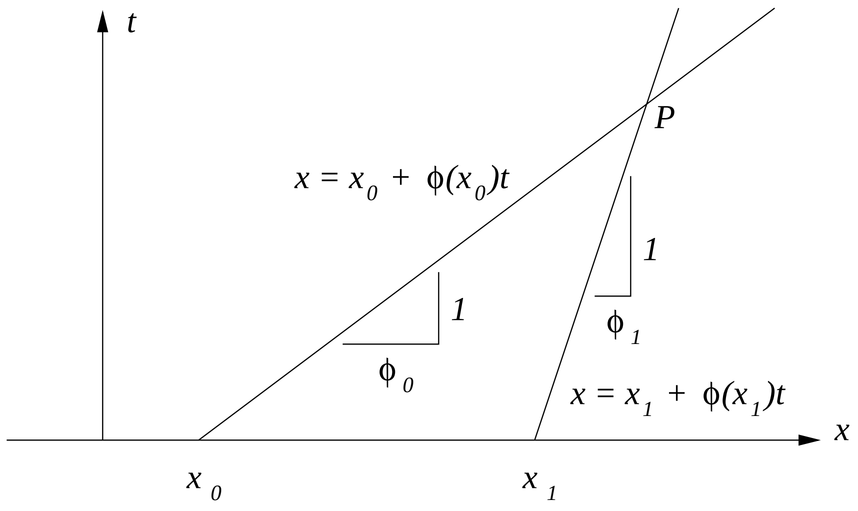


Figure 10.1.6: Characteristic curves for two initial points x_0 and x_1 for Burgers' equation (10.1.16). The characteristics intersect at a point P.

In order to clarify matters, let's examine the specific choice of u^0 given by Lax [20]

$$u^0(x) = \begin{cases} 1, & \text{if } x < 0 \\ 1 - x, & \text{if } 0 \leq x < 1 \\ 0, & \text{if } 1 \leq x \end{cases} . \tag{10.1.18}$$

Using (10.1.17), we see that the characteristic passing through the point $(x_0, 0)$ satisfies

$$x = \begin{cases} x_0 + t, & \text{if } x_0 < 0 \\ x_0 + (1 - x_0)t, & \text{if } 0 \leq x < 1 \\ x_0, & \text{if } 1 \leq x \end{cases} . \tag{10.1.19}$$

Several characteristics are shown in Figure 10.1.7. The characteristics first intersect at $t = 1$. After that, the solution would presumably be multivalued, as shown in Figure 10.1.8.

It's, of course, quite possible for multivalued solutions to exist; however, (*i*) they are not observed in physical situations and (*ii*) they do not satisfy (10.1.11a) in any classical sense. Discontinuous solutions are often observed in nature once characteristics of the corresponding conservation law model have intersected. They also do not satisfy

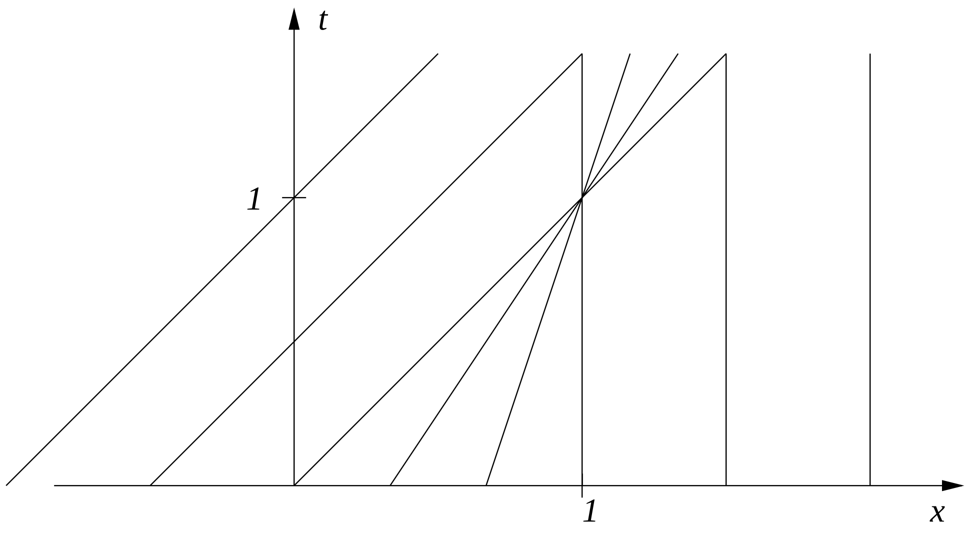


Figure 10.1.7: Characteristics for Burgers' equation (10.1.16) with initial data given by (10.1.18).

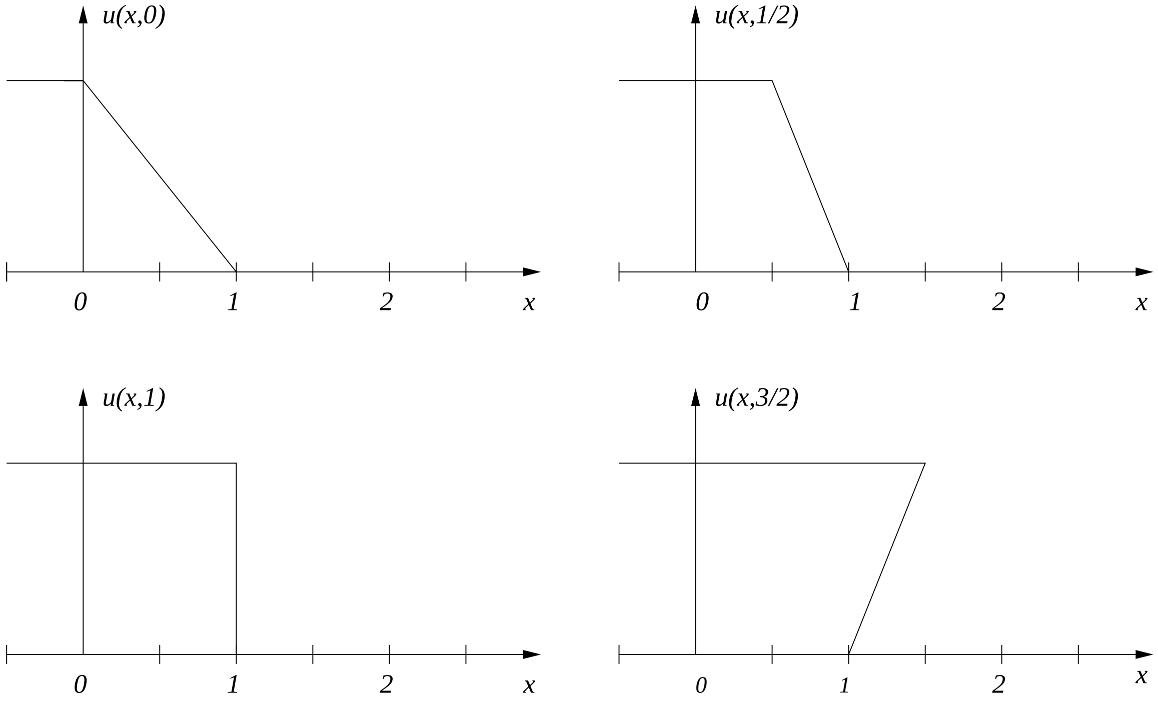


Figure 10.1.8: Multivalued solution of Burgers' equation (10.1.16) with initial data given by (10.1.18). The solution $u(x,t)$ is shown as a function of x for $t = 0$, 1/2, 1, and 3/2.

(10.1.11a), but they might satisfy the integral form of the conservation law (10.1.1). We examine the simplest case when two classical solutions satisfying (10.1.11a) are separated by a single smooth curve $x = \xi(t)$ across which $u(x,t)$ is discontinuous. For each $t > 0$ we assume that $\alpha < \xi(t) < \beta$ and let superscripts - and + denote conditions immediately to the left and right, respectively, of $x = \xi(t)$. Then, using (10.1.1), we have

$$\frac{d}{dt}\int_\alpha^\beta u dx = \frac{d}{dt}[\int_\alpha^{\xi^-} u dx + \int_{\xi^+}^\beta u dx] = -f(u)|_\alpha^\beta$$

or, differentiating the integrals

$$\int_\alpha^{\xi^-} u_t dx + u^-\dot{\xi}^- + \int_{\xi^+}^\beta u_t dx - u^+\dot{\xi}^+ = -f(u)|_\alpha^\beta.$$

The solution on either side of the discontinuity was assumed to be smooth, so (10.1.11a) holds in (α, ξ^-) and (ξ^+, β) and can be used to replace the integrals. Additionally, since ξ is smooth, $\dot{\xi}^- = \dot{\xi}^+ = \dot{\xi}$. Thus, we have

$$-f(u)|_\alpha^{\xi^-} + u^-\dot{\xi} - f(u)|_{\xi^+}^\beta - u^+\dot{\xi} = -f(u)|_\alpha^\beta,$$

or

$$\dot{\xi}(u^+ - u^-) = f(u^+) - f(u^-). \tag{10.1.20}$$

Let

$$[q] \equiv q^+ - q^- \tag{10.1.21a}$$

denote the jump in a quantity q and write (10.1.20) as

$$[u]\dot{\xi} = [f(u)]. \tag{10.1.21b}$$

Equation (10.1.21b) is called the *Rankine-Hugoniot jump condition* and the discontinuity is called a *shock wave*. We can use the Rankine-Hugoniot condition to find a discontinuous solution of Example 10.1.5.

Example 10.1.6. For $t < 1$, the discontinuous solution of (10.1.16, 10.1.18) is as given in Example 10.1.5. For $t \geq 1$, we hypothesize the existence of a single shock wave, passing through $(1, 1)$ in the (x, t)-plane. As shown in Figure 10.1.9, the solution of Example 10.1.5 can be used to infer that $u^- = 1$ and $u^+ = 0$. Thus, $f(u^-) = (u^-)^2/2 = 1/2$ and $f(u^+) = (u^+)^2/2 = 0$. Using (10.1.21b), the velocity of the shock wave is

$$\dot{\xi} = \frac{1}{2}.$$

Integrating, we find the shock location as

$$\xi = \frac{1}{2}t + c.$$

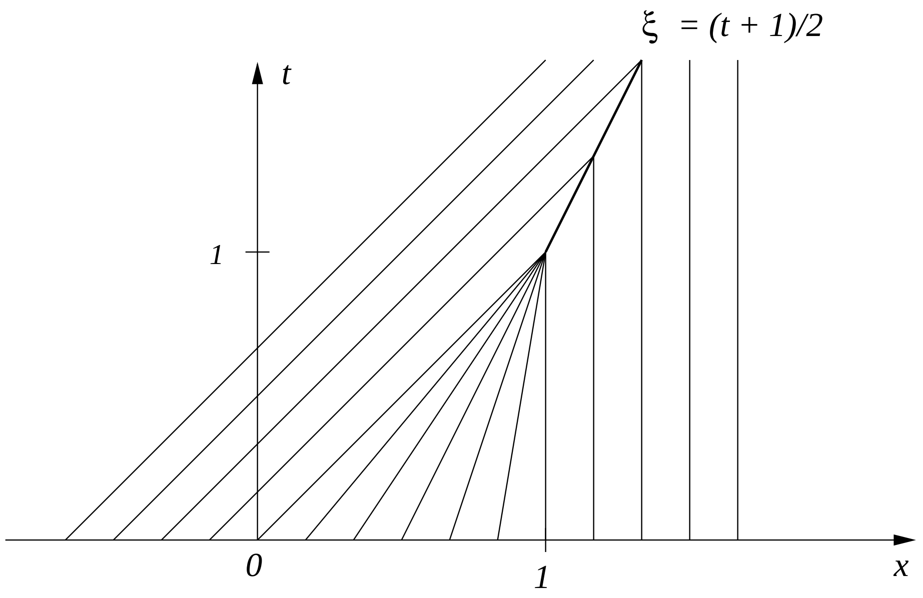


Figure 10.1.9: Characteristics and shock discontinuity for Example 10.1.6.

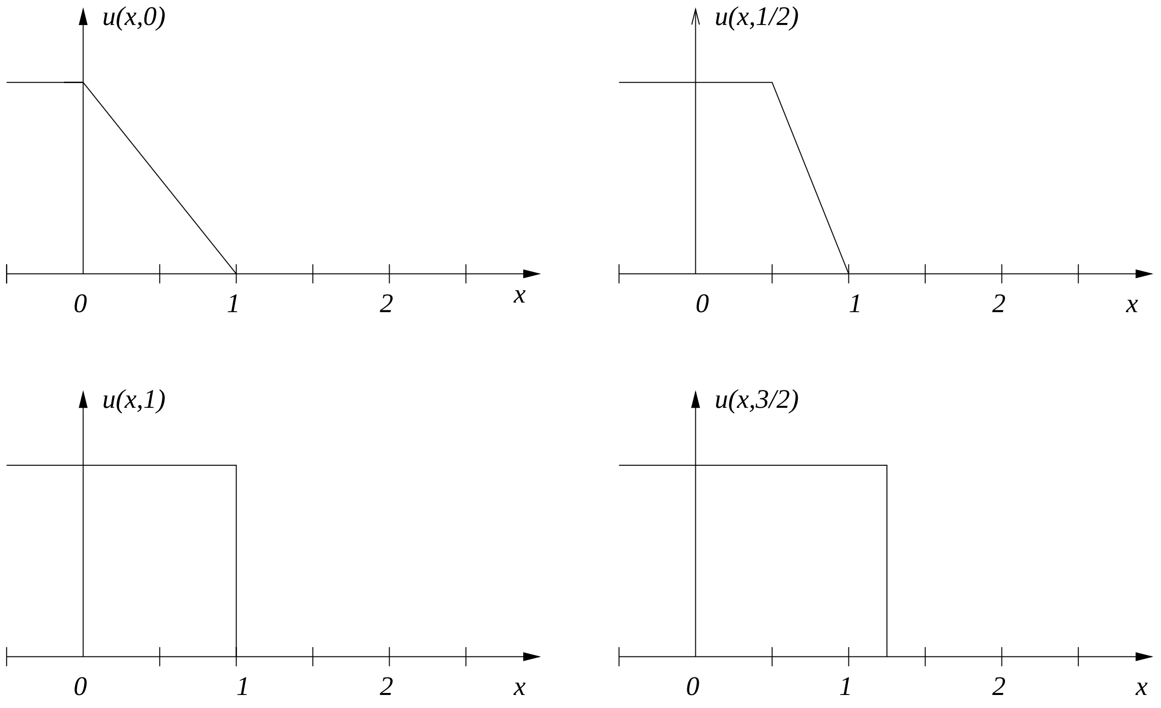


Figure 10.1.10: Solution $u(x, t)$ of Example 10.1.6 as a function of x at $t = 0$, $1/2$, 1, and $3/2$. The solution is discontinuous for $t > 1$.

Since the shock passes through $(1, 1)$, the constant of integration $c = 1/2$, and

$$\xi = \frac{1}{2}(t + 1). \tag{10.1.22}$$

The characteristics and shock wave are shown in Figure 10.1.9 and the solution $u(x,t)$ is shown as a function of x for several times in Figure 10.1.10.

Let us consider another problem for Burgers' equation with different initial conditions that will illustrate another structure that arises in the solution of nonlinear hyperbolic systems.

Example 10.1.7. Consider Burgers' equation (10.1.16) subject to the initial conditions

$$u^0(x) = \begin{cases} 0, & \text{if } x < 0 \\ x, & \text{if } 0 \le x < 1 \\ 1, & \text{if } 1 \le x \end{cases}. \tag{10.1.23}$$

Using (10.1.17) and (10.1.23), we see that the characteristic passing through $(x_0, 0)$ satisfies

$$x = \begin{cases} x_0, & \text{if } x < 0 \\ x_0(1+t), & \text{if } 0 \le x < 1 \\ x_0 + t, & \text{if } 1 \le x \end{cases}. \tag{10.1.24}$$

These characteristics, shown in Figure 10.1.11, may be used to verify that the solution, shown in Figure 10.1.12, is continuous. Additional considerations and difficulties with nonlinear hyperbolic systems are discussed in Lax [20].

Example 10.1.8. A Riemann problem is an initial value (Cauchy) problem for (10.1.1) with piecewise-constant initial data. Riemann problems play an important role in the numerical solution of conservation laws using both finite difference and finite element techniques. In this introductory section, let us illustrate a Riemann problem for the inviscid Burgers' equation (10.1.16). Thus, we apply the initial data

$$\mathbf{u}(x,0) = \begin{cases} \mathbf{u}_L, & \text{if } x < 0 \\ \mathbf{u}_R, & \text{if } x \ge 0 \end{cases}. \tag{10.1.25}$$

As in the previous two examples, we have to distinguish between two cases when $u_L > u_R$ and $u_L \le u_R$. The solution may be obtained by considering piecewise-linear continuous initial conditions as in Examples 10.1.6 and 10.1.7, but with the "ramp" extending from 0 to ϵ instead of from 0 to 1. We could then take a limit as $\epsilon \to 0$. The details are left to an exercise (Problem 1 at the end of this section).

When $u_L > u_R$, the characteristics emanating from points $x_0 < 0$ are the straight lines $x = x_0 + u_L t$ (*cf.* (10.1.17)). Those emanating from points $x_0 > 0$ are $x = x_0 + u_R t$. The characteristics cross immediately and a shock forms. Using (10.1.20), we see that the shock moves with speed $\dot{\xi} = (u_L + u_R)/2$. The solution is constant along the characteristics and, hence, is given by

$$u(x,t) = \begin{cases} u_L, & \text{if } x/t < (u_L + u_R)/2 \\ u_R, & \text{if } x/t \ge (u_L + u_R)/2 \end{cases}, \qquad u_L > u_R. \tag{10.1.26a}$$

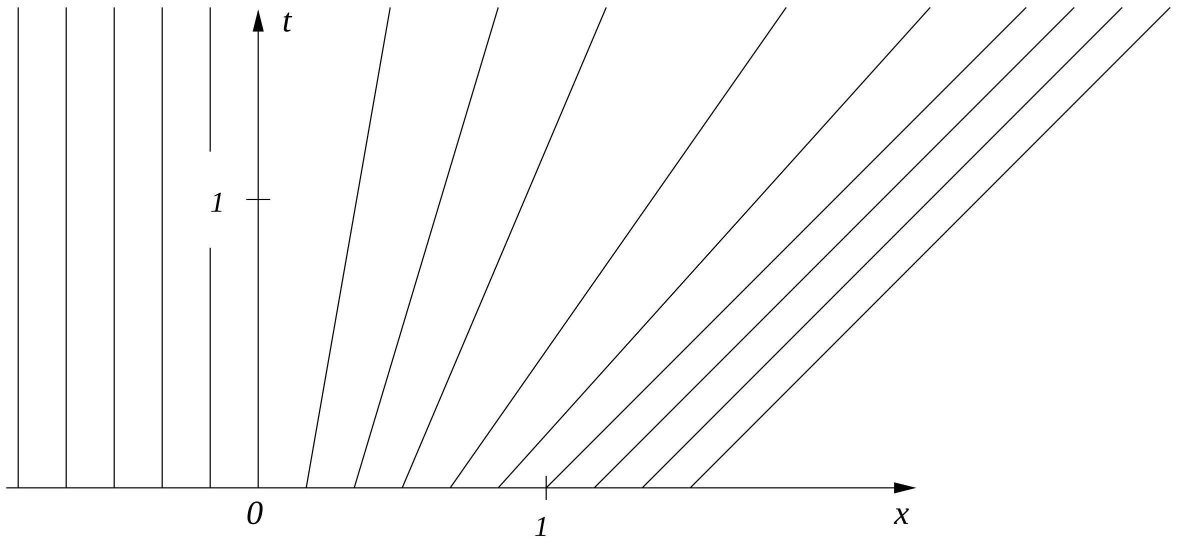


Figure 10.1.11: Characteristics for Example 10.1.7.

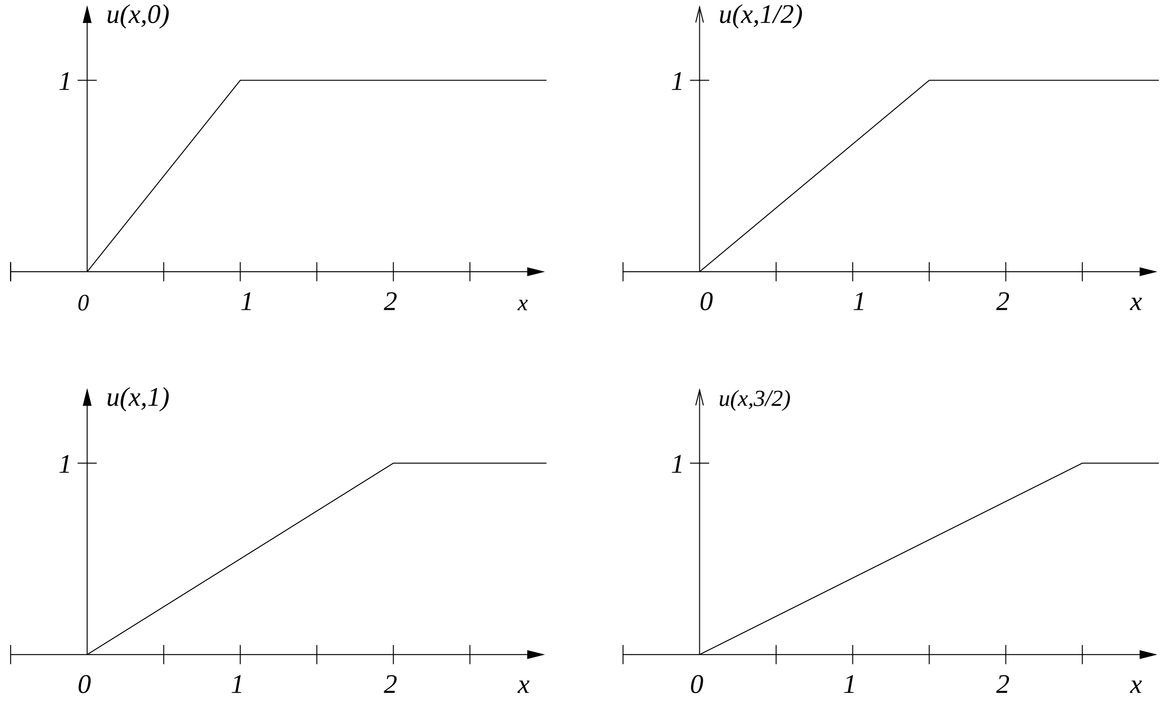


Figure 10.1.12: Solution $u(x, t)$ of Example 10.1.7 as a function of x at $t = 0$, $1/2$, 1, and $3/2$.

Several characteristics and the location of the shock are shown in Figure 10.1.13.

When $u_L \leq u_R$, the characteristics do not intersect. There is a region between the characteristic $x = u_L t$ emanating from $x_0 = 0^-$ and $x = u_R t$ emanating from $x_0 = 0^+$ where the initial conditions fail to determine the solution. As determined by either the limiting process suggested in Problem 1 or thermodynamic arguments using entropy considerations [20], no shock forms and the solution in this region is an expansion fan. Several characteristics are shown in Figure 10.1.13 and the expansion solution is given by

$$u(x,t) = \begin{cases} u_L, & \text{if } x/t < u_L \\ x/t, & \text{if } u_L \leq x/t < u_R \\ u_R, & \text{if } x/t \geq u_R \end{cases}, \qquad u_L \leq u_R. \tag{10.1.26b}$$

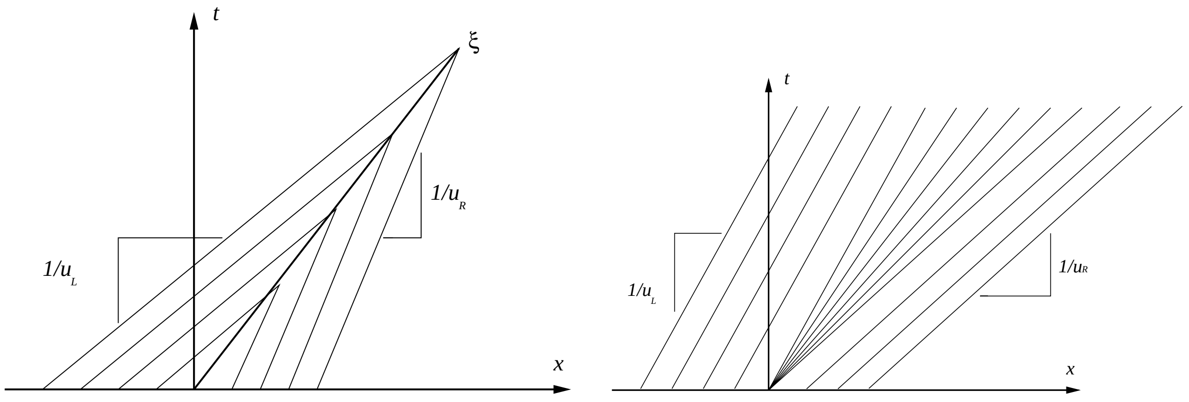


Figure 10.1.13: Shock (left) and expansion (right) wave characteristics of the Riemann problem of Example 10.1.8.

We conclude this example by examining the solution of the Riemann problem along the line $x = 0$. Characteristics for several choices of initial data are shown in Figure 10.1.14 and, by examining these and (10.1.26), we see that

$$u(0,t) = \begin{cases} u_L, & \text{if } u_L, u_R > 0 \\ u_R, & \text{if } u_L, u_R < 0 \\ 0, & \text{if } u_L < 0,\ u_R > 0 \\ u_L, & \text{if } u_L > 0,\ u_R < 0,\ (u_L + u_R)/2 > 0 \\ u_R, & \text{if } u_L > 0,\ u_R < 0,\ (u_L + u_R)/2 < 0 \end{cases}.$$

This data will be useful when constructing numerical schemes based on the solution of Riemann problems.

Problems

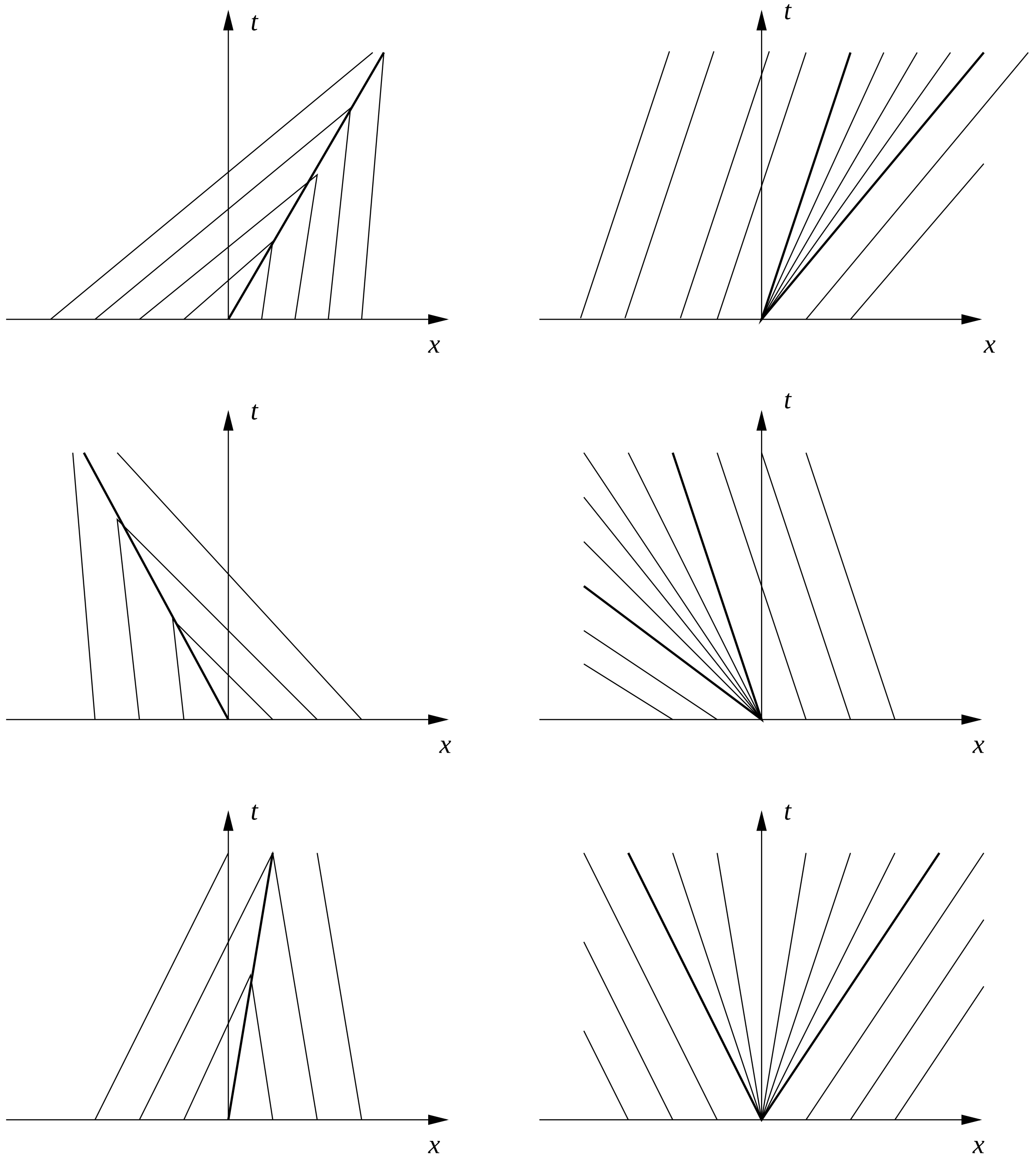


Figure 10.1.14: Characteristics of Riemann problems for Burgers' equation when $u_L, u_R > 0$ (top); $u_L, u_R < 0$ (center); $u_L > 0$, $u_R < 0$, $(u_L + u_R)/2 > 0$ (bottom left); and $u_L < 0, u_R > 0$ (bottom right).

1. Show that the solution of the Riemann problem (10.1.16, 10.1.25) is given by (10.1.26). You may begin by solving a problem with continuous initial data, *e.g.*,

$$u(x,0) = \begin{cases} u_L, & \text{if } x < -\epsilon \\ \frac{u_L}{2\epsilon}(\epsilon - x) + \frac{u_R}{2\epsilon}(\epsilon + x), & \text{if } -\epsilon < x \leq \epsilon \\ u_R, & \text{if } \epsilon < x \end{cases},$$

and take the limit as $\epsilon \to 0$.

10.2 Discontinuous Galerkin Methods

In Section 9.3, we examined the use of the discontinuous Galerkin method for time integration. We'll now examine it as a way of performing spatial discretization of conservation laws (10.1.1). The method might have some advantages when solving problems with discontinuous solutions. The discontinuous Galerkin method was first used for to solve an ordinary differential equation for neutron transport [21]. At the moment, it is very popular and is being used to solve ordinary differential equations [24, 19] and hyperbolic [5, 6, 7, 8, 12, 11, 13, 16], parabolic [14, 15], and elliptic [4, 3, 28] partial differential equations. A recent proceedings contains a complete and current survey of the method and its applications [10].

The discontinuous Galerkin method has a number of advantages relative to traditional finite element methods when used to discretize hyperbolic problems. We have already noted that it has the potential of sharply representing discontinuities. The piecewise continuous trial and test spaces make it unnecessary to impose interelement continuity. There is also a simple communication pattern between elements that makes it useful for parallel computation.

We'll begin by describing the method for conservation laws (10.1.1) in one spatial dimension. In doing this, we present a simple construction due to Cockburn and Shu [12] rather than the (more standard) approach [19] used in Section 9.3 for time integration. Using a method of lines formulation, let us divide the spatial region into elements (x_{j-1}, x_j), $j = 1, 2, \ldots, N$, and construct a local Galerkin problem on Element (x_{j-1}, x_j) in the usual manner by multiplying (10.1.1a) by a test function $\mathbf{v}$ and integrating to obtain

$$\int_{x_{j-1}}^{x_j} \mathbf{v}^T[\mathbf{u}_t + \mathbf{f}(\mathbf{u})_x]dx = 0. \qquad (10.2.1a)$$

The loading term $\mathbf{b}(x, t, \mathbf{u})$ in (10.1.1a) causes no conceptual or practical difficulties and we have neglected it to simplify the presentation.

Following the usual procedure, let us map (x_{j-1}, x_j) to the canonical element $(-1, 1)$ using the linear transformation

$$x = \frac{1-\xi}{2}x_{j-1} + \frac{1+\xi}{2}x_j. \qquad (10.2.1b)$$

Then, after integrating the flux term in (10.2.1a) by parts, we obtain

$$\frac{h_j}{2}\int_{-1}^{1} \mathbf{v}^T\mathbf{u}_t d\xi + \mathbf{v}^T\mathbf{f}(\mathbf{u})|_{-1}^{1} = \int_{-1}^{1} \mathbf{v}_\xi^T\mathbf{f}(\mathbf{u})d\xi \qquad (10.2.1c)$$

where

$$h_j = x_j - x_{j-1}. \qquad (10.2.1d)$$

Without a need to maintain interelement continuity, there are several options available for selecting a finite element basis. Let us choose one based on Legendre polynomials. As we shall see, this will produce a diagonal mass matrix without a need to use lumping. Thus, we select the approximation $\mathbf{U}_j(x,t)$ of $\mathbf{u}(x,t)$ on the mapping of (x_{j-1}, x_j) to the canonical element as

$$\mathbf{U}_j(\xi,t) = \sum_{k=0}^{p} \mathbf{c}_{kj}(t) P_k(\xi) \tag{10.2.2a}$$

where $\mathbf{c}_{kj}(t)$ is an m-vector and $P_k(\xi)$ is the Legendre polynomial of degree k in ξ. Recall (*cf.* Section 2.5), that the Legendre polynomials satisfy the orthogonality relation

$$\int_{-1}^{1} P_i(\xi)P_j(\xi)d\xi = \frac{2\delta_{ij}}{2i+1}, \qquad i,j \geq 0 \tag{10.2.2b}$$

are normalized as

$$P_i(1) = 1, \qquad i \geq 0, \tag{10.2.2c}$$

and satisfy the symmetry relation

$$P_i(\xi) = (-1)^i P_i(-\xi), \qquad i \geq 0. \tag{10.2.2d}$$

The first six Legendre polynomials are

$$\begin{aligned} P_0(\xi) &= 1, \qquad & P_1(\xi) &= \xi, \\ P_2(\xi) &= \frac{3\xi^2 - 1}{2}, \qquad & P_3(\xi) &= \frac{5\xi^3 - 3\xi}{2}, \\ P_4(\xi) &= \frac{35\xi^4 - 30\xi^2 + 3}{2}, \qquad & P_5(\xi) &= \frac{63\xi^5 - 70\xi^3 + 15\xi}{8}. \end{aligned} \tag{10.2.3}$$

These polynomials are illustrated in Figure 10.2.1). Additional information appears in Section 2.5 and Abromowitz and Stegen [1].

Substituting (10.2.2a) into (10.2.1c), testing against $P_i(\xi)$, and using (10.2.2b-d) yields

$$\begin{aligned} \frac{h_j \dot{\mathbf{c}}_{ij}}{2i+1} + \mathbf{f}(\mathbf{U}(x_j,t)) - (-1)^i \mathbf{f}(\mathbf{U}(x_{j-1},t)) &= \int_{-1}^{1} \frac{dP_i(\xi)}{d\xi} \mathbf{f}(\mathbf{U}_j(\xi,t)) d\xi, \\ i &= 1, 2, \ldots, p, \end{aligned} \tag{10.2.4a}$$

where $\dot{(\,)} = d(\,)/dt$.

Neighboring elements must communicate information to each other and, in this form of the discontinuous Galerkin element method, this is done through the boundary flux

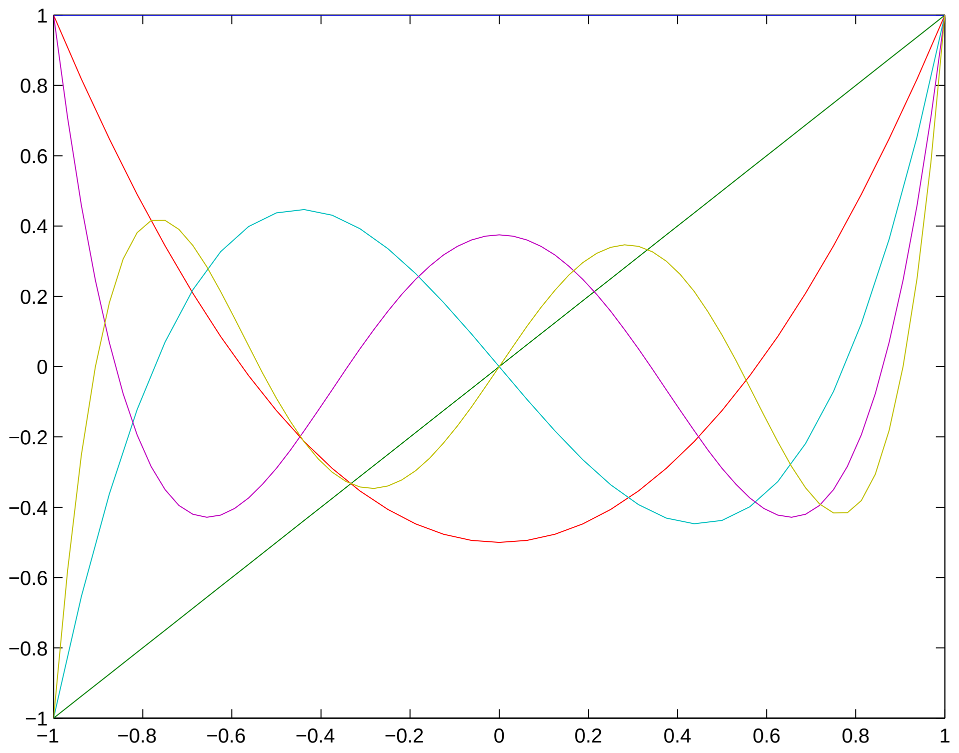


Figure 10.2.1: Legendre polynomials of degrees $p = 0, 1, \ldots, 5$.

terms. The usual practice is to replace the boundary flux terms $\mathbf{f}(\mathbf{U}(x_k, t)$, $k = j-1, j$, by a *numerical flux function*

$$\mathbf{f}(\mathbf{U}(x_k, t) \approx \mathbf{F}(\mathbf{U}_k(x_k, t)), \mathbf{U}_{k+1}(x_k, t)) \tag{10.2.4b}$$

that depends on the approximate solutions $\mathbf{U}_k$ and $\mathbf{U}_{k+1}$ on the two elements sharing the vertex at x_k. Cockburn and Shu [12] present several possible numerical flux functions. Perhaps, the simplest is the average

$$\mathbf{F}(\mathbf{U}_k(x_k, t)), \mathbf{U}_{k+1}(x_k, t)) = \frac{\mathbf{f}(\mathbf{U}_k(x_k, t)) + \mathbf{f}(\mathbf{U}_{k+1}(x_k, t))}{2}. \tag{10.2.5a}$$

Based on our work with convection-diffusion problems in Section 9.5, we might expect that some upwind considerations might be worthwhile. This happens to be somewhat involved for nonlinear vector systems. We'll postpone it and, instead, note that an upwind flux for a scalar problem is

$$F(U_k(x_k, t)), U_{k+1}(x_k, t)) = \begin{cases} f(U_k(x_k, t)), & \text{if } a(U_k(x_k, t)) + a(U_{k+1}(x_k, t)) > 0 \\ f(U_{k+1}(x_k, t)), & \text{if } a(U_k(x_k, t)) + a(U_{k+1}(x_k, t)) \leq 0 \end{cases} \tag{10.2.5b}$$

where

$$a(u) = f_u(u). \tag{10.2.5c}$$

A simple numerical flux that is relatively easy to apply to vector systems and employs upwind information is the *Lax-Friedrichs* function [12]

$$\begin{aligned}\mathbf{F}(\mathbf{U}_k(x_k,t),\mathbf{U}_{k+1}(x_k,t)) &= \frac{1}{2}[\mathbf{f}(\mathbf{U}_k(x_k,t))+\mathbf{f}(\mathbf{U}_{k+1}(x_k,t)) \\ &\quad -\lambda_{max}(\mathbf{U}_{k+1}(x_k,t)-\mathbf{U}_k(x_k,t))], \qquad (10.2.5\text{d})\end{aligned}$$

where λ_{max} is the maximum absolute eigenvalue of the Jacobian matrix $\mathbf{f}_{\mathbf{u}}(\mathbf{u})$, $\mathbf{u} \in [\mathbf{U}_k(x_k,t)),\mathbf{U}_{k+1}(x_k,t)]$.

Example 10.2.1. The simplest discontinuous Galerkin scheme uses piecewise-constant ($p=0$) solutions

$$\mathbf{U}_j(\xi,t)=\mathbf{c}_{0j}(t)P_0(\xi)=\mathbf{c}_{0j}.$$

In this case, (10.2.4a) becomes

$$h_j\dot{\mathbf{c}}_{0j}+\mathbf{f}(\mathbf{U}(x_j,t))-\mathbf{f}(\mathbf{U}(x_{j-1},t))=\mathbf{0}.$$

In this initial example, let's choose a scalar problem and evaluate the flux using the average (10.2.5a)

$$F(U_k(x_k,t)),U_{k+1}(x_k,t))=\frac{f(U_k(x_k,t))+f(U_{k+1}(x_k,t))}{2}=\frac{f(c_{0,k})+f(c_{0,k+1})}{2}$$

and upwind (10.2.5b)

$$F(U_k(x_k,t)),U_{k+1}(x_k,t))=\begin{cases} f(c_{0,k}), & \text{if } a(c_{0,k})+a(c_{0,k+1})>0 \\ f(c_{0,k+1}), & \text{if } a(c_{0,k})+a(c_{0,k+1})\le 0\end{cases}$$

numerical fluxes. With these flux choices, we have the ordinary differential systems

$$\dot{c}_{0j}+\frac{f(c_{0,j+1})-f(c_{0,j-1})}{2h_j}=0$$

and

$$\dot{c}_{0j}+\frac{(1-\delta_j)f(c_{0,j+1})+(1+\delta_j)f(c_{0,j})-(1-\delta_{j-1})f(c_{0,j})-(1+\delta_{j-1})f(c_{0,j-1})}{2h_j}=0$$

where

$$\delta_j=\operatorname{sgn}(a(c_{0,j})+a(c_{0,j+1})).$$

In the (simplest) case when $f(u)=au$ with a a positive constant, we have the two schemes

$$\dot{c}_{0j}+\frac{a(c_{0,j+1}-c_{0,j-1})}{2h_j}=0 \qquad j=0,1,\ldots,J,$$

and

$$\dot{c}_{0j}+\frac{a(c_{0,j}-c_{0,j-1})}{h_j}=0, \qquad j=0,1,\ldots,J.$$

Initial conditions for $c_{0j}(0)$ may be specified by interpolating the initial data at the center of each interval, *i.e.*, $c_{0,j}(0) = u^0(x_j - h_j/2)$, $j = 1, 2, \ldots, J$.

We use these two techniques to solve an initial value problem with $a = 1$ and

$$u^0(x,t) = \sin 2\pi x.$$

Thus, the exact solution is

$$u(x,t) = \sin 2\pi(x-t).$$

Piecewise-constant discontinuous Galerkin solutions with upwind and centered fluxes are shown at $t = 1$ in Figure 10.2.2. A 16-element uniform mesh was used and time integration was performed using the MATLAB Runge-Kutta procedure *ode*45. The solution with the upwind flux has greatly dissipated the solution after one period in time. The maximum error at cell centers

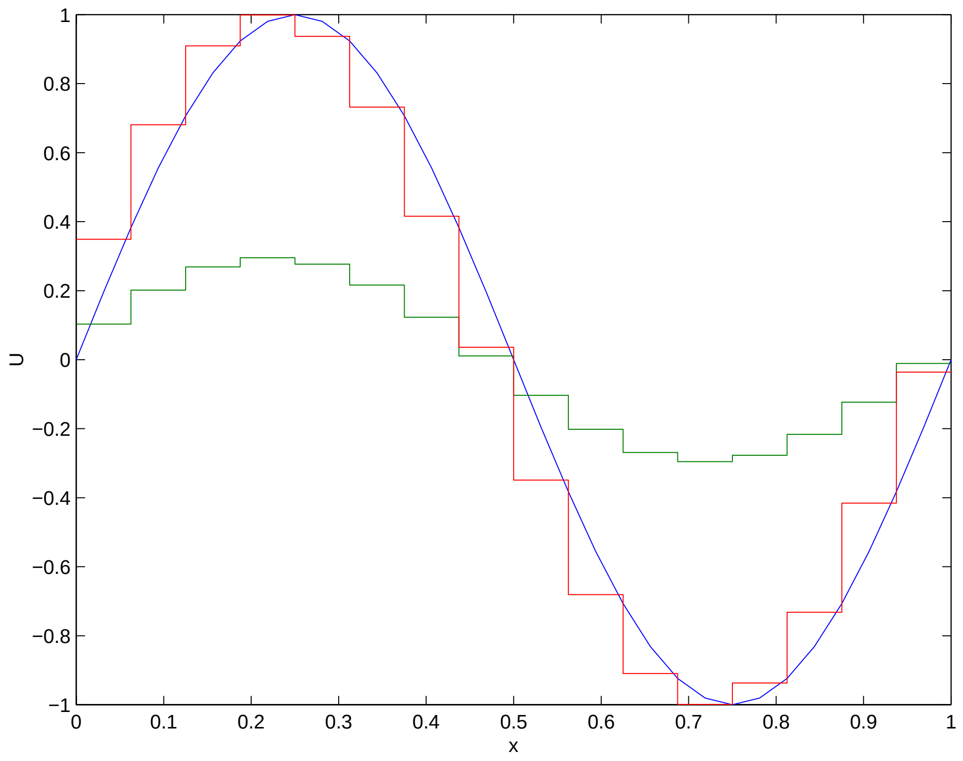


Figure 10.2.2: Exact and piecewise-constant discontinuous solutions of a linear kinematic wave equation with sinusoidal initial data at $t = 1$. Solutions with upwind and centered fluxes are shown. The solution using the upwind flux exhibits the most dissipation.

$$|e(\cdot,t)|_\infty := \max_{1\le j\le J} |u(x_j - h_j/2, t) - U(x_j - h_j/2, t)|$$

at $t = 1$ is shown in Table 10.2.1 on meshes with $J = 16$, 32, and 64 elements. Since the errors are decreasing by a factor of two for each mesh doubling, it appears that the upwind-flux solution is converging at a linear rate. Using similar reasoning, the centered solution appears to converge at a quadratic rate. The errors appear to be smallest at the downwind (right) end of each element. This superconvergence result has been known for some time [19] but other more general results were recently discovered [2].

J	Upwind $\|e\|_\infty$	Centered $\|e\|_\infty$
16	0.7036	0.1589
32	0.4597	0.0400
64	0.2653	0.0142

Table 10.2.1: Maximum errors for solutions of a linear kinematic wave equation with sinusoidal initial data at $t = 1$ using meshes with $J = 16$, 32, and 64 uniform elements. Solutions were obtained using upwind and centered fluxes.

As a second calculation, let's consider discontinuous initial data

$$u^0(x,t) = \begin{cases} 1, & \text{if } 0 \leq x < 1/2 \\ -1, & \text{if } 1/2 \leq x < 1 \end{cases} .$$

This data is extended periodically to the whole real line. Piecewise-constant discontinuous Galerkin solutions with upwind and centered fluxes are shown at $t = 1$ in Figure 10.2.3. The upwind solution has, once again, dissipated the initial square pulse. This time, however, the centered solution is exhibiting spurious oscillations. As with convection-dominated convection-diffusion equations, some upwinding will be necessary to eliminate spurious oscillations near discontinuities.

10.2.1 High-Order Discontinuous Galerkin Methods

The results of Example 10.2.1 are extremely discouraging. It would appear that we have to contend with either excessive diffusion or spurious oscillations. To overcome these choices, we investigate the use of the higher-order techniques offered by (10.2.4). With $\mathbf{c}_{ij}$ being an m-vector and i ranging from 0 to p, we have $p+1$ vector and $m(p+1)$ scalar unknowns on each element.

We will focus on the four major tasks: (*i*) evaluating the integral on the right side of (10.2.4a), (*ii*) performing the time integration (*iii*) defining the initial conditions, and (*iv*) evaluating the fluxes. The integral in (10.2.4a) will typically require numerical integration and the obvious choice is Gaussian quadrature as described in Chapter 6. This works fine and there is no need to discuss it further.

Time integration can be performed by either explicit or implicit techniques. The choice usually depends on the spread of the eigenvalues λ_i, $i = 1, 2, \ldots, m$, of the Jacobian $\mathbf{A}(\mathbf{u})$. If the eigenvalues are close to each other, explicit integration is fine. Stability

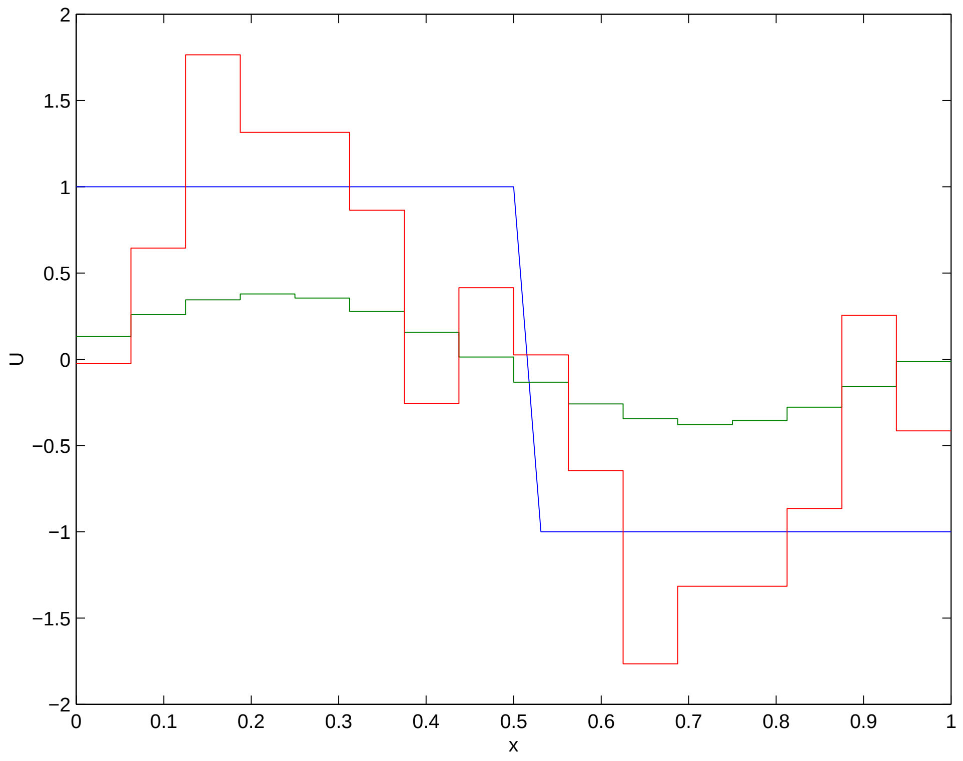


Figure 10.2.3: Exact and piecewise-constant discontinuous solutions of a linear kinematic wave equation with discontinuous initial data at $t = 1$. Solutions with upwind and centered fluxes are shown. The solution using the upwind flux is dissipative. The solution using the centered flux exhibits spurious oscillations.

is usually not a problem. An implicit scheme might be necessary when the eigenvalues are widely separated or when integrating (10.2.4) to a steady state. For explicit integration, Cockburn and Shu [12] recommend a total variation diminishing (TVD) Runge-Kutta scheme. However, Biswas *et al.* [8] found that classical Runge-Kutta formulas gave similar results. Second- and third-order and fourth- and fifth-order classical Runge-Kutta software was used for time integration of Example 10.2.1. If forward Euler integration of (10.2.4a) were used, we would have to solve the explicit system

$$\begin{aligned}\frac{h_j}{2i+1}\frac{\mathbf{c}_{ij}^{n+1}-\mathbf{c}_{ij}^{n}}{\Delta t} = -\mathbf{f}(\mathbf{U}^n(x_j)) + (-1)^i\mathbf{f}(\mathbf{U}^n(x_{j-1})) &= \int_{-1}^{1}\frac{dP_i(\xi)}{d\xi}\mathbf{f}(\mathbf{U}_j^n(\xi))d\xi,\\ i &= 1,2,\ldots,p.\end{aligned}$$

The notation is identical to that used in Chapter 9; thus, $\mathbf{U}^n(x)$ and $\mathbf{c}_{ij}^n$ are the approximations of $\mathbf{U}(x,t_n)$ and $\mathbf{c}_{ij}(t_n)$, respectively, produced by the time integration software and Δt is the time step. The forward Euler method is used for illustration because of its simplicity. The order of the temporal integration method should be comparable to p.

Initial conditions may be determined by L^2 projection as

$$\int_{-1}^{1} P_i(\xi)[\mathbf{U}_j(\xi,0) - \mathbf{u}^0(\xi)]d\xi = \mathbf{0}, \qquad i = 0,1,\ldots,p, \qquad j = 1,2,\ldots,J. \qquad (10.2.6)$$

One more difficulty emerges. Higher-order schemes for hyperbolic problems oscillate near discontinuities. This is a fundamental result that may be established by theoretical means (*cf.*, *e.g.*, Sod [25]). One technique for reduced these oscillations involves *limiting* the computed solution. Many limiting algorithms have been suggested but none are totally successful. We describe a procedure for limiting the slope $\partial\mathbf{U}_j(x,t)/\partial x$ of the solution that is widely used. With this approach, $\partial\mathbf{U}_j(x,t)/\partial x$ is modified so that:

1. the solution (10.2.2a) does not take on values outside of the adjacent grid averages (Figure 10.2.4, upper left);
2. local extrema are set to zero (Figure 10.2.4, upper right); and
3. the gradient is replaced by zero if its sign is not consistent with its neighbors (Figure 10.2.4, lower center).

Figure 10.2.4 illustrates these situations when the solution is a piecewise-linear ($p = 1$) function relative to the mesh.

A formula for accomplishing this limiting can be summarized concisely using the minimum modulus function as

$$\frac{\partial\mathbf{U}_{j,mod}(x_j,t)}{\partial x} = \text{minmod}(\frac{\partial\mathbf{U}_j(x_j,t)}{\partial x}, \nabla\mathbf{U}_j(x_{j-1/2},t), \Delta\mathbf{U}_j(x_{j-1/2},t)) \qquad (10.2.7a)$$

$$\frac{\partial\mathbf{U}_{j,mod}(x_{j-1},t)}{\partial x} = \text{minmod}(\frac{\partial\mathbf{U}_j(x_{j-1},t)}{\partial x}, \nabla\mathbf{U}_j(x_{j-1/2},t), \Delta\mathbf{U}_j(x_{j-1/2},t)) \qquad (10.2.7b)$$

where

$$\text{minmod}(a,b,c) = \begin{cases} \text{sgn}(a)\min(|a|,|b|,|c|), & \text{if } \text{sgn}(a) = \text{sgn}(b) = \text{sgn}(c) \\ 0, & \text{otherwise} \end{cases} \qquad (10.2.7c)$$

and ∇ and Δ are the backward and forward difference operators

$$\nabla\mathbf{U}_j(x_{j-1/2},t) = \mathbf{U}_j(x_{j-1/2},t) - \mathbf{U}_j(x_{j-3/2},t), \qquad (10.2.7d)$$

and

$$\Delta\mathbf{U}_j(x_{j-1/2},t) = \mathbf{U}_j(x_{j+1/2},t) - \mathbf{U}_j(x_{j-1/2},t). \qquad (10.2.7e)$$

With $\partial\mathbf{U}_{j,mod}(x_{j-1},t)/\partial x$ and $\partial\mathbf{U}_{j,mod}(x_j,t)/\partial x$, determined, (10.2.7a,b) are used to re-computed the coefficients in (10.2.2a) to reduce the oscillations. However, (10.2.7a,b)

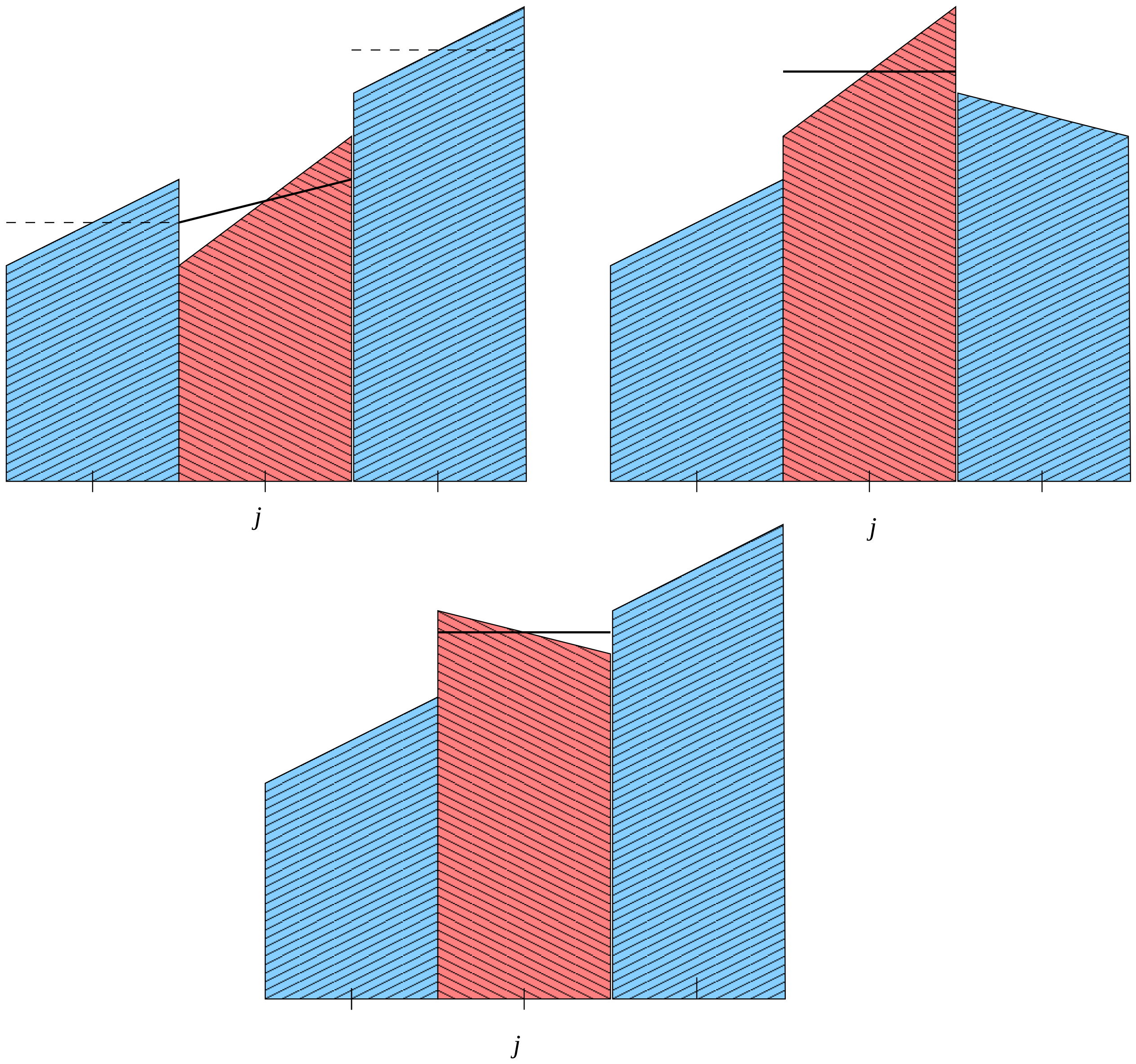


Figure 10.2.4: Solution limiting: reduce slopes to be within neighboring averages (upper left); set local extrema to zero (upper right); and set slopes to zero if they disagree with neighboring trends.

only provide two vector equations for modifying the p vector coefficients $\mathbf{c}_{ij,mod}(t)$, $i = 1, 2, \ldots, p$, in $\partial \mathbf{U}_j(x,t)/\partial x$. When $p = 1$, (10.2.7a,b) are identical and $\mathbf{c}_{1j,mod}(t)$ is uniquely determined. Likewise, when $p = 2$, the two conditions (10.2.7a,b) suffice to uniquely determine the modified coefficients $\mathbf{c}_{1j,mod}(t)$ and $\mathbf{c}_{2j,mod}(t)$. Equations (10.2.7a,b) are insufficient to determine the modified coefficients when $p > 2$ and Cockburn and Shu [12] suggested setting the higher-order coefficients $c_{ij,mod}(t)$, $i = 3, 4, \ldots, p$, to zero. This has the disturbing characteristic of "flattening" the solution near smooth extrema and reducing the order of accuracy. Biswas *et al.* [8] developed an adaptive limiter which applied the minimum modulus function (10.2.7c) to higher derivatives of $\mathbf{U}_j$. They began by limiting the *p th* derivative of $\mathbf{U}_j$ and worked downwards until either a derivative was not changed by the limiting or they modified all of the coefficients. Their procedure, called "moment limiting." is described further in their paper [8].

Example 10.2.2. Biswas *et al.* [8] solve the inviscid Burgers' equation (10.1.16) with the initial data

$$u(x,0) = \frac{1+\sin x}{2}.$$

This initial data steepens to form a shock which propagates in the positive x direction.

Biswas *et al.* [8] use an upwind numerical flux (10.2.5b) and solve problems on uniform meshes with $h = 1/32$ with $p = 0, 1, 2$. Time integration was done using classical Runge-Kutta methods of orders 1-3, respectively, for $p = 0, 1, 2$. Exact and computed solutions are shown in Figure 10.2.5. The piecewise polynomial functions used to represent the solution are plotted at eleven points on each subinterval.

The first-order solution ($p = 0$) shown at the upper left of Figure 10.2.5 is characteristically diffusive. The second-order solution ($p = 1$) shown at the upper right of Figure 10.2.5 has greatly reduced the diffusion while not introducing any spurious oscillations. The minimum modulus limiter (10.2.7) has flattened the solution near the shock as seen with the third-order solution ($p = 2$) shown at the lower left of Figure 10.2.5. There is a loss of (local) monotonicity near the shocks. (Average solution values are monotone and this is all that the limiter (10.2.7) was designed to produce.) The adaptive moment limiter of Biswas *et al.* [8] reduces the flattening and does a better job of preserving local monotonicity near discontinuities. The solution with $p = 2$ using this limiter is shown in the lower portion of Figure 10.2.5.

Example 10.2.3. Adjerid *et al.* [2] solve the nonlinear wave equation

$$u_{tt} - u_{xx} = u(2u^2 - 1) \tag{10.2.8a}$$

which can be written in the form (10.1.1a) as

$$(u_1)_t + (u_1)_x = u_2, \qquad (u_2)_t - (u_2)_x = u_1(2u_1^2 - 1) \tag{10.2.8b}$$

with $u_1 = u$. The initial and boundary conditions are such that the exact solution of (10.2.8a) is the solitary wave

$$u(x,t) = \operatorname{sech}(x \cosh\frac{1}{2} + t \sinh\frac{1}{2}) \tag{10.2.8c}$$

(*cf.* Figure 10.2.1).

Adjerid *et al.* [2] solved problems on $-\pi/3 < x < \pi/3$, $0 < t < 1$ by the discontinuous Galerkin method using polynomials of degrees $p = 0$ to 4. The solution at $t = 1$

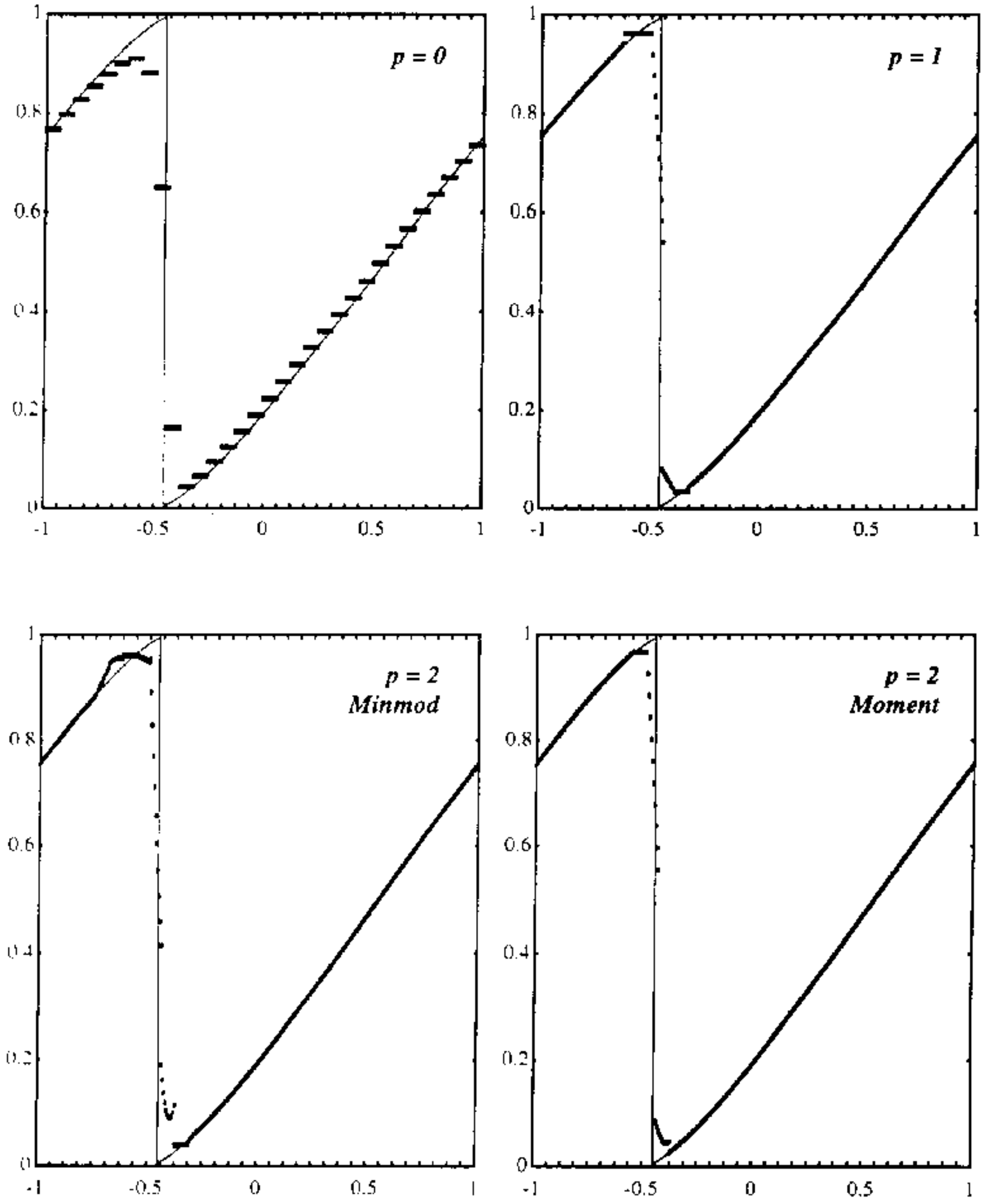


Figure 10.2.5: Exact (line) and discontinuous Galerkin solutions of Example 10.2.2 for $p = 0, 1, 2$, and $h = 1/32$. Solutions with the minmod limiter (10.2.7) and an adaptive moment limiter of Biswas *et al.* [8] are shown for $p = 2$.

performed with $p = 2$ and $J = 64$ is shown in Figure 10.2.1. The entire solitary wave is shown; however, the computation was performed on the center region $-\pi/3 < x < \pi/3$.

Discretization errors in the L^1 norm

$$\|e(\cdot,t)\| = \sum_{j=1}^{J} \int_{x_{j-1}}^{x_j} |U(x,t) - U_j(x,t)|dx$$

are presented for the solution u for various combinations of h and p in Table 10.2.2. Solutions of this nonlinear wave propagation problem appear to be converging as $O(h^{p+1})$ in the L_1 norm. This can be proven correct for smooth solutions of discontinuous Galerkin methods [2, 11, 12].

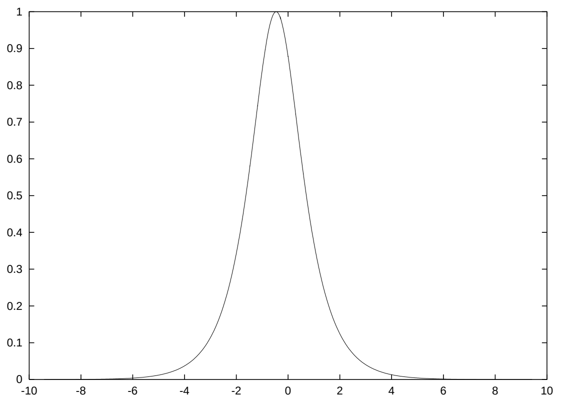


Figure 10.2.6: Solution of Example 10.2.3 at $t = 1$ obtained by the discontinuous Galerkin method with $p = 2$ and $N = 64$.

J	$p = 0$	$p = 1$	$p = 2$	$p = 3$	$p = 4$
8	2.16e-01	5.12e-03	1.88e-04	7.12e-06	3.67e-07
16	1.19e-01	1.19e-03	2.32e-05	4.38e-07	1.12e-08
32	6.39e-02	2.88e-04	2.90e-06	2.70e-08	3.55e-10
64	3.32e-02	7.06e-05	3.63e-07	1.68e-09	1.10e-11
128	1.69e-02	1.74e-05	4.53e-08	1.04e-10	3.49e-13
256	8.58e-03	4.34e-06	5.67e-09		

Table 10.2.2: Discretization errors at $t = 1$ as functions J and p for Example 10.2.3.

Evaluating numerical fluxes and using limiting for vector systems is more complicated than indicated by the previous scalar example. Cockburn and Shu [12] reported problems when applying limiting component-wise. At the price of additional computation, they applied limiting to the characteristic fields obtained by diagonalizing the Jacobian $\mathbf{f_u}$. Biswas *et al.* [8] proceeded in a similar manner. “Flux-vector splitting” may provide a compromise between the two extremes. As an example, consider the solution and flux vectors for the one-dimensional Euler equations of compressible flow (10.1.3). For this and related differential systems, the flux vector is a homogeneous function that may be expressed as

$$\mathbf{f}(\mathbf{u}) = \mathbf{A}\mathbf{u} = \mathbf{f}_{\mathbf{u}}(\mathbf{u})\mathbf{u}. \tag{10.2.9a}$$

Since the system is hyperbolic, the Jacobian $\mathbf{A}$ may be diagonalized as described in Section 10.1 to yield

$$\mathbf{f}(\mathbf{u}) = \mathbf{P}^{-1}\mathbf{\Lambda}\mathbf{P}\mathbf{u} \tag{10.2.9b}$$

where the diagonal matrix $\mathbf{\Lambda}$ contains the eigenvalues of $\mathbf{A}$

$$\mathbf{\Lambda} = \begin{bmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_m \end{bmatrix} = \begin{bmatrix} u-c & & \\ & u & \\ & & u+c \end{bmatrix}. \tag{10.2.9c}$$

The variable $c = \sqrt{\partial p/\partial \rho}$ is the speed of sound in the fluid. The matrix $\mathbf{\Lambda}$ can be decomposed into components

$$\mathbf{\Lambda} = \mathbf{\Lambda}^{+} + \mathbf{\Lambda}^{-} \tag{10.2.10a}$$

where $\mathbf{\Lambda}^{+}$ and $\mathbf{\Lambda}^{-}$ are, respectively, composed of the non-negative and non-positive components of $\mathbf{\Lambda}$

$$\lambda_i^{\pm} = \frac{\lambda_i \pm |\lambda_i|}{2}, \qquad i = 1, 2, \ldots, m. \tag{10.2.10b}$$

Writing the flux vector in similar fashion using (10.2.9)

$$\mathbf{f}(\mathbf{u}) = \mathbf{P}^{-1}(\mathbf{\Lambda}^{+} + \mathbf{\Lambda}^{-})\mathbf{P}\mathbf{u} = \mathbf{f}(\mathbf{u})^{+} + \mathbf{f}(\mathbf{u})^{-}. \tag{10.2.10c}$$

Split fluxes for the Euler equations were presented by Steger and Warming [26]. Van Leer [27] found an improvement that provided better performance near sonic and stagnation points of the flow. The split fluxes are evaluated by upwind techniques. Thus, at an interface $x = x_j$, $\mathbf{f}^{+}$ is evaluated using $\mathbf{U}_j(x_j, t)$ and $\mathbf{f}^{-}$ is evaluated using $\mathbf{U}_{j+1}(x_j, t)$.

Calculating fluxes based on the solution of Riemann problems is another popular way of specifying numerical fluxes for vector systems. To this end, let $\mathbf{w}(x/t, \mathbf{u}_L, \mathbf{u}_R)$ be the solution of a Riemann problem for (10.1.1a) with the peicewise-constant initial data (10.1.25). The solution of a Riemann problem "breaking" at (x_j, t_n) would be $\mathbf{w}((x-x_j)/(t-t_n), \mathbf{U}_j(x_j, t_n), \mathbf{U}_j(x_{j+1}, t_n))$. Using this, we would calculate the numerical flux at (x_j, t), $t > t_n$, as

$$\mathbf{F}(\mathbf{U}_j(x_j, t_n), \mathbf{U}_{j+1}(x_j, t_n)) = \mathbf{f}(\mathbf{w}(0, \mathbf{U}_j(x_j, t_n), \mathbf{U}_{j+1}(x_j, t_n)). \tag{10.2.11}$$

Example 10.2.4. Let us calculate the numerical flux based on the solution of a Riemann problem for Burgers' equation (10.1.16). Using the results of Example 10.1.8) we know that the solution of the appropriate Riemann problem is

$$w(0, U_j, U_{j+1}) = \begin{cases} U_j, & \text{if } U_j, U_{j+1} > 0 \\ U_{j+1}, & \text{if } U_j, U_{j+1} < 0 \\ 0, & \text{if } U_j < 0, U_{j+1} > 0 \\ U_j, & \text{if } U_j > 0, U_{j+1} < 0,\ (U_j + U_{j+1})/2 > 0 \\ U_{j+1}, & \text{if } U_j > 0, U_{j+1} < 0,\ (U_j + U_{j+1})/2 < 0 \end{cases}.$$

(The arguments of U_j and U_{j+1} are all (x_j, t_n). These have been omitted for clarity.) With $f(u) = u^2/2$ for Burgers' equation, we find the numerical flux

$$F(U_j, U_{j+1}) = \begin{cases} U_j^2/2, & \text{if } U_j, U_{j+1} > 0 \\ U_{j+1}^2/2, & \text{if } U_j, U_{j+1} < 0 \\ 0, & \text{if } U_j < 0,\ U_{j+1} > 0 \\ U_j^2/2, & \text{if } U_j > 0,\ U_{j+1} < 0,\ (U_j + U_{j+1})/2 > 0 \\ U_{j+1}^2/2, & \text{if } U_j > 0,\ U_{j+1} < 0,\ (U_j + U_{j+1})/2 < 0 \end{cases}.$$

Letting

$$u^+ = \max(u, 0), \qquad u^- = \min(u, 0),$$

we can write the numerical flux more concisely as

$$F(U_j, U_{j+1}) = \max[(U_j^+)^2/2, (U_{j+1}^-)^2/2].$$

When used with a piecewise-constant basis and forward Euler time integration, the resulting discontinuous Galerkin scheme is identical to Godunov's finite difference scheme [18]. This was the first difference scheme to be based on the solution of a Riemann problem. This early work and a subsequent work of Glimm [17] and Chorin [9] stimulated a great deal of interest in using Riemann problems to construct numerical flux functions. A summary of a large number of choices appears in Cockburn and Shu [12].

10.3 Multidimensional Discontinuous Galerkin Methods

Let us extend the discontinuous Galerkin method to multidimensional conservation laws of the form

$$\mathbf{u}_t + \nabla \cdot \boldsymbol{f}(\mathbf{u})_x = \mathbf{b}(x, y, z, t, \mathbf{u}), \qquad (x, y, z) \in \Omega, \qquad t > 0, \tag{10.3.1a}$$

where

$$\boldsymbol{f}(\mathbf{u}) = [\mathbf{f}(\mathbf{u}), \mathbf{g}(\mathbf{u}), \mathbf{h}(\mathbf{u})] \tag{10.3.1b}$$

and

$$\nabla \cdot \boldsymbol{f}(\mathbf{u}) = \mathbf{f}(\mathbf{u})_x + \mathbf{g}(\mathbf{u})_y + \mathbf{h}(\mathbf{u})_z. \tag{10.3.1c}$$

The solution $\mathbf{u}(x, y, z, t)$; componenets of the flux vector $\mathbf{f}(\mathbf{u})$, $\mathbf{g}(\mathbf{u})$, and $\mathbf{h}(\mathbf{u})$; and the loading $\mathbf{b}(x, y, z, t, \mathbf{u})$ are m-vectors and Ω is a bounde region of $\Re^3$. Boundary conditions must be prescribed on $\partial\Omega$ along characteristics that enter the region. We'll see what this means by example. Initial condtions prescribe

$$\mathbf{u}(x, y, z, 0) = \mathbf{0}, \qquad (x, y, z) \in \Omega \cup \partial\Omega. \tag{10.3.1d}$$

Following our analysis of Section 10.2, we partition Ω into a set of finite elements Ω_j, $j = 1, 2, \ldots, N_\Delta$, and construct a weak form of the problem on an element. This is done, as usual, by multiplying (10.3.1a) by a test function $\mathbf{v} \in L^2(\Omega_j)$, integrating over Ω_j, and applying the divergence theorem to the flux to obtain

$$(\mathbf{v}, \mathbf{u}_t)_j + < \mathbf{v}, \boldsymbol{f} \cdot \mathbf{n} >_j - (\nabla \mathbf{v}, \boldsymbol{f})_j = (\mathbf{v}, \mathbf{b})_j, \qquad \forall \mathbf{v} \in L^2(\Omega_j), \tag{10.3.2a}$$

where

$$(\mathbf{v}, \mathbf{u})_j = \int_{\Omega_j} \mathbf{v}^T \mathbf{u}\, dx dy dz, \tag{10.3.2b}$$

$$(\nabla \mathbf{v}, \boldsymbol{f})_j = \int_{\Omega_j} [\mathbf{v}_x^T \mathbf{f}(\mathbf{u}) + \mathbf{v}_y^T \mathbf{g}(\mathbf{u}) + \mathbf{v}_z^T \mathbf{h}(\mathbf{u})] dx dy dz, \tag{10.3.2c}$$

$$\boldsymbol{f} \cdot \mathbf{n} = \boldsymbol{f}_n = \mathbf{f}(\mathbf{u}) n_1 + \mathbf{g}(\mathbf{u}) n_2 + \mathbf{h}(\mathbf{u}) n_3, \tag{10.3.2d}$$

and

$$< \mathbf{v}, \boldsymbol{f} \cdot \mathbf{n} >_j = \int_{\partial\Omega_j} \mathbf{v}^T \boldsymbol{f}_n dS. \tag{10.3.2e}$$

The vector $\mathbf{n} = [n_1, n_2, n_3]^T$ is the unit outward normal vector to $\partial\Omega$ and dS is a surface infinitessimal on $\partial\Omega_j$.

Only the normal component of the flux is involved in (10.3.2); hence, its approximation on $\partial\Omega_j$ is the same as the one-dimensional problems of Section 10.2. Thus, the numerical normal flux function can be taken as a one-dimensional numerical flux using solution values on each side of $\partial\Omega_j$. In order to specify this more precisely, let $nb_{j,k}$, $k = 1, 2, \ldots, N_E$, denote the indices of the N_E elements sharing the bounding faces of Ω_j and let $\partial\Omega_{j,k}$, $k = 1, 2, \ldots, N_E$, be the faces of Ω_j (Figure 10.3.1). Then, we write (10.3.2a) in the more explicit form

$$(\mathbf{v}, \mathbf{u}_t)_j + \sum_{k=1}^{N_E} < \mathbf{v}, \boldsymbol{F}_n(\mathbf{U}_j, \mathbf{U}_{nb_{j,k}}) >_{j,k} - (\nabla \mathbf{v}, \boldsymbol{f})_j = (\mathbf{v}, \mathbf{b})_j, \qquad \forall \mathbf{v} \in L^2(\Omega_j). \tag{10.3.3}$$

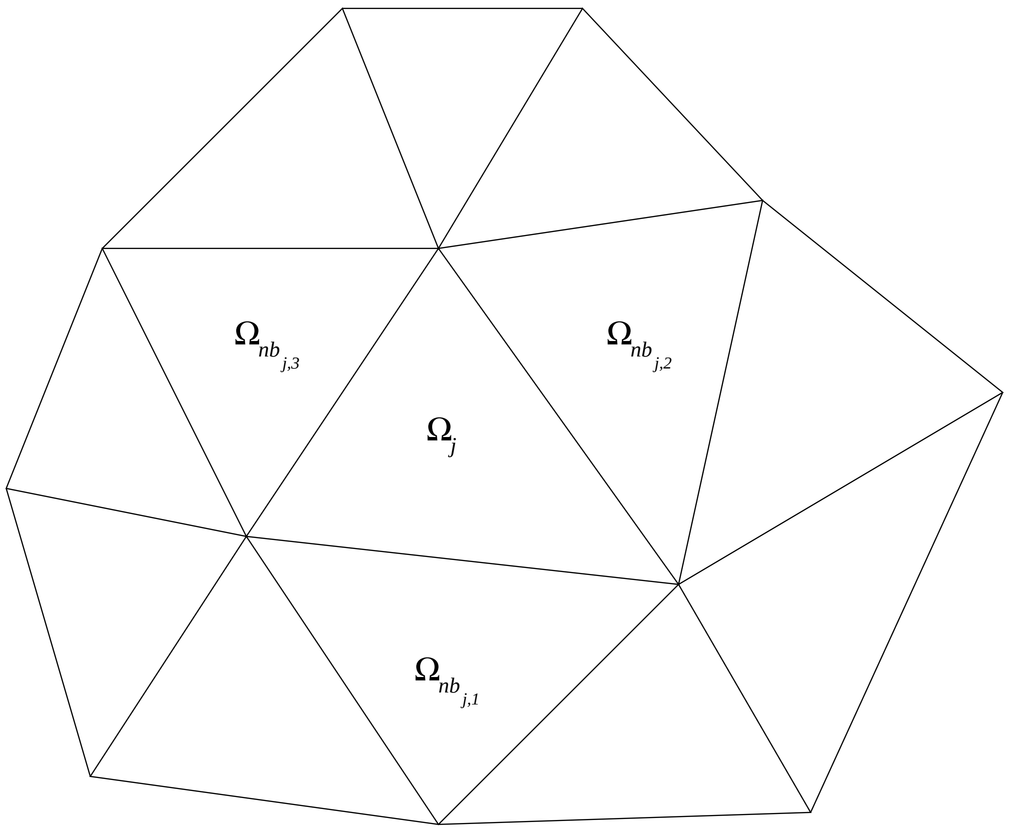


Figure 10.3.1: Element j and its neighboring elements indicating that the segments $\partial\Omega_{j,k}$, $k = 1, 2, \ldots, N_E$, .

Without the need to maintain inter-element continuity, virtually any polynomial basis can be used for the approximate solution $\mathbf{U}_j(x, y, z, t)$ on Ω_j. Tensor products of Legendre polynomials can provide a basis on square or cubic canonical elements, but these are unavailable for triangles and tetrahedra. Approximations on triangles and tetrahedra can use a basis of monomial terms. Focusing on two-dimensional problems on the canonical (right 45°) triangle, we write the finite element solution in the usual form

$$\mathbf{U}_j(x, y, t) = \sum_{k=1}^{n_p} \mathbf{c}_{kj} N_k(\xi, \eta), \tag{10.3.4}$$

where $n_p = (p + 1)(p + 2)/2$ is the number of monomial terms in a complete polynomial of degree p. A basis of monomial terms would set

$$N_1 = 1, \qquad N_2 = \xi, \qquad N_3 = \eta, \qquad \ldots, \qquad N_{n_p} = \eta^p. \tag{10.3.5}$$

All terms in the mass matrix can be evaluated by exact integration on the canonical triangle (*cf.* Problem 1 at the end of this section) as long as it has straight sides; however, without orthogonality, the mass matrix will not be diagonal. This is not a severe restriction since the mass matrix is independent of time and, thus, need only be inverted (factored) once. The ill-conditioning of the mass matrix at high p is a more important concern with the monomial basis (10.3.5).

Ill-conditioning can be reduced and the mass matrix diagonalized by extracting an orthogonal basis from the monomial basis (10.3.5). This can be done by the Gram-Schmidt orthogonalization process shown in Figure 10.3.2. The inner product and norm are defined in L^2 on the canonical element as

```
procedure gram($\mathbf{N}$)
   $\bar{N}_1 := N_1/||N_1||_{0,0}$
   for $k := 1$ to $n_p$ do
      $t := N_k - \sum_{i=1}^{k-1}(N_k, \bar{N}_i)_0 \bar{N}_i$
      $\bar{N}_k := t/||t||_{0,0}$
   bf end for
return $\bar{\mathbf{N}}$
```

Figure 10.3.2: Gram-Schmidt process to construct an orthogonal basis $\bar{N}_k$ $k = 1, 2, \dots, n_p$ from a basis of monomials N_k, $k = 1, 2, \dots, n_p$.

$$(u, v)_0 = \int_0^1 \int_0^{1-\xi} uv d\xi d\eta, \qquad ||u||_{0,0} = (u, u)_0^{1/2}. \tag{10.3.6a}$$

The result of the Gram-Schmidt process is a basis $\bar{N}_k$, $k = 1, 2, \dots, n_p$ that satisfies the orthogonality condition

$$(\bar{N}_i, \bar{N}_k) = \delta_{i,k}, \qquad i, k = 1, 2, \dots, n_p. \tag{10.3.6b}$$

The actual process can be done using symbolic computation using a computer algebra system such as *MAPLE* or *MATHEMATICA* (*cf.* Remacle *et al.* [22] and Problem 2 at the end of this section).

Example 10.3.1. We will illustrate some results using the discontinuous Galerkin method to solve two- and three-dimensional compressible flow problems involving the

Euler equations. This complex nonlinear system has the form of (10.3.1a) with

$$\mathbf{u} = \begin{bmatrix} \rho \\ m \\ n \\ l \\ e \end{bmatrix}, \qquad \boldsymbol{f}(\mathbf{u}) \;=\; [\mathbf{f}(\mathbf{u}), \mathbf{g}(\mathbf{u}), \mathbf{h}(\mathbf{u})] = \begin{bmatrix} m & n & l \\ m^2/\rho + p & nm/\rho & lm/\rho \\ mn/\rho & n^2/\rho + p & ln/\rho \\ ml/\rho & nl/\rho & l^2/\rho + p \\ (e+p)m/\rho & (e+p)n/\rho & (e+p)b/\rho \end{bmatrix},$$

$$\mathbf{b}(x, t, \mathbf{u}) \;=\; \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}. \tag{10.3.7a}$$

Here, ρ is the fluid density; m, n, and l are the Cartesian components of the momentum vector per unit volume; e is the total energy per unit volume; and p is the pressure, which must satisfy an equation of state of the form

$$p = (\gamma - 1)[e - (m^2 + n^2 + l^2)/2\rho]. \tag{10.3.7b}$$

This equation of state assumes an ideal fluid with gas constant γ.

Let us consider a classical Rayleigh-Taylor instability which has a heavy ($\rho = 2$) fluid above a light ($\rho = 1$) fluid (Figure 10.3.3). This hydrostatic configuration is unstable and any slight perturbation will cause the heavier fluid to fall and the lighter one to rise. The fluid motion is quite complex and Remacle *et al.* [22] simulated it using discontinuous Galerkin methods. They considered two-dimensional motion ($l = 0$, $\partial/\partial z = 0$ in (10.3.7)) with the initial perturbation

$$\rho = \begin{cases} 1, & \text{if } 0 \le y < 1/2 \\ 2, & \text{if } 1/2 \le y < 1 \end{cases}, \qquad p = \begin{cases} 3/2 - y, & \text{if } 0 \le y < 1/2 \\ 2(1-y), & \text{if } 1/2 \le y < 1 \end{cases}$$

$$u = \epsilon_x \sin 8\pi x \cos \pi y \sin^{\tau-1} \pi y, \qquad v = -\epsilon_y \cos 8 \sin^{\tau} \pi y.$$

Here u, v, and w are the Cartesian velocity components and $\gamma = 5/3$, $\tau = 6$, and ϵ_x and ϵ_y were chosen to be small. The boundary conditions specify that $u = 0$ on the sides and top and $v = 0$ on the bottom.

Solutions for the density ρ at $t = 1.8$ are shown in Figure 10.3 for computations with $p = 0$ to 3. The mesh used for all values of p is shown in Figure 10.3. The total number of vector degrees of freedom for two-dimensional discontinuous Galerkin methods is $N_{\Delta} n_p$. Since there are four unknowns per element (ρ, m, n, and e) for two-dimensional flows, there are 2016, 6048, 12096, and 20160 unknowns for degrees $p = 0$, 1, 2, and 3, respectively. Fluxes were evaluated using Roe's linearized flux approximation [23]. No limiting was used for this computation. A high-frequency filtering [22] was used to suppress oscillations in the vicinity of the interface separating the two fluids.

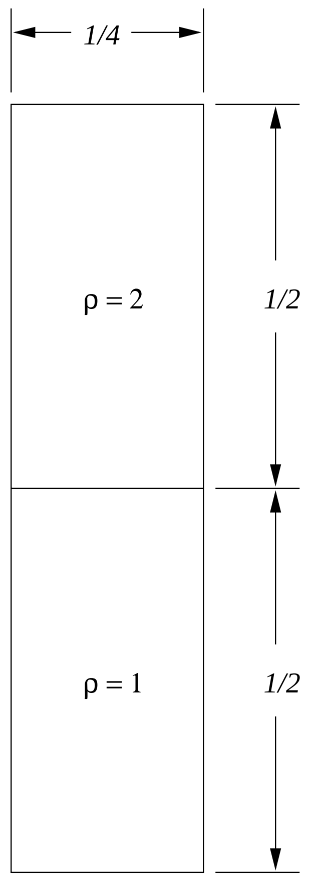


Figure 10.3.3: Configuration for the Rayleigh-Taylor instability of Example 10.3.1. There are solid walls on the bottom and sides and open flow at the top.

The results with $p = 0$ show very little structure of the solution. Those with $p = 1$ show more-and-more detail of the flow. There is no exact solution of this problem, so it is not possible to appraise the effects of using higher degree polynomials; however, solutions with more detail are assumed to be more correct.

Remacle *et al.* [22] also did computations using adaptive p-refinement. There is no error estimate available for the Euler equations, so they used an error indicator E_j on element j consisting of

$$E_j = \int_{\Omega_j} \nabla\rho \cdot \nabla\rho dV + \sum_{k=1}^{3} \left| \int_{\partial\Omega_j} (\rho_j - \rho_{nb_{j,k}}) dS \right| .$$

This can be shown [22] to be the length of the interface that separates the two fluids on Ω_j. Remacle *et al.* [22] increased the degree on elements where E_j was above the median of all error indicators. Results using this adaptive p-refinement strategy with p ranging from 1 to 3 are shown in Figure 10.3. The mesh used for these computations was

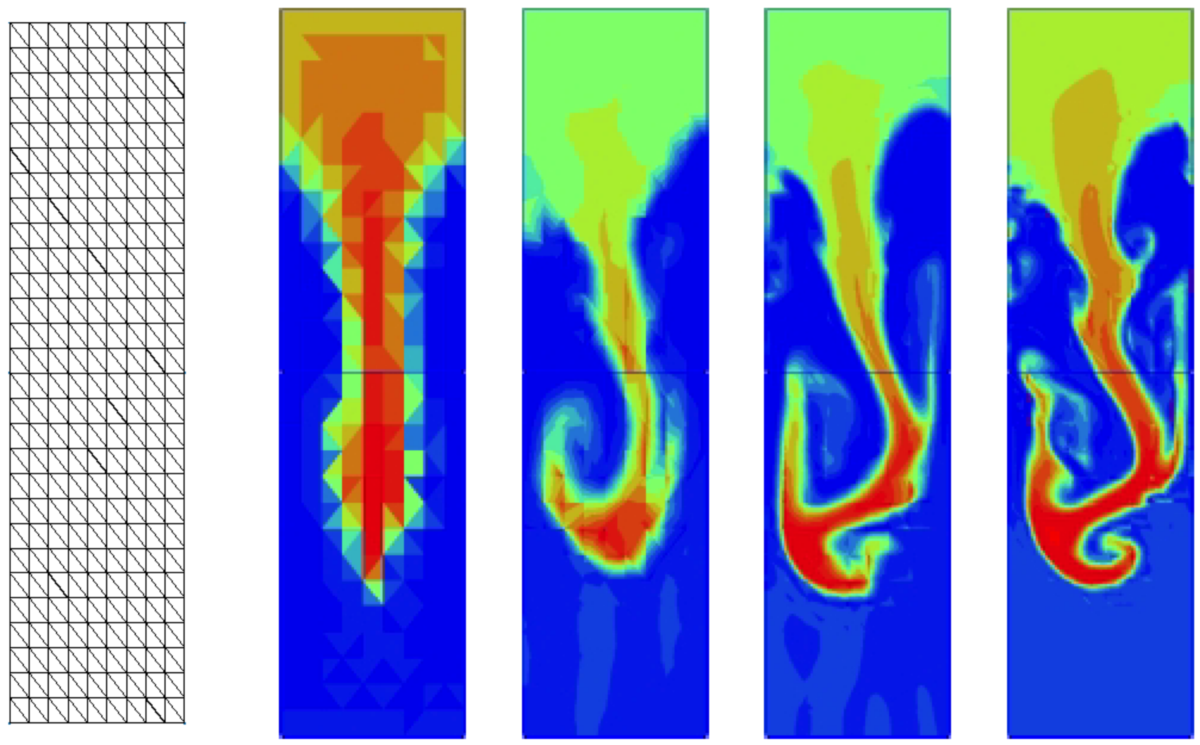


Figure 10.3.4: Densities for the Rayleigh-Taylor instability of Example 10.3.1 at $t = 1.8$ and $p = 0$ to 3. The mesh used for all computations is shown at the left.

a uniform bisection of each element of the mesh shown in Figure 10.3 into four elements.

Successive frames in Figure 10.3 show the selected values of p and the density ρ at $t = 0.75$, 1.2, and 1.5. The computations show the complex series of bifurcations that occur at the interface between the two fluids.)

Example 10.3.2. Flaherty *et al.* [16] solve a flow problem for the three-dimensional Euler equations (10.3.7) in a tube containing a vent (Figure 10.3) using a piecewise-constant discontinuous Galerkin method. A van Leer flux vector splitting (10.2.9 - 10.2.10) [27] was used to evaluate fluxes. No limiting is necessary with a first-order method. The main tube initially had a supersonic flow at a Mach number (ratio of the speed of the fluid to the speed of sound) of 1.23. There was no flow in the vent. At time $t = 0$ a hypothetical diaphragm between the main and vent cylinders is ruptured and the flow expands into the vent. Flaherty *et al.* citeFLS97 solve this problem using an adaptive h-refinement procedure. They used the magnitude of density jumps across element boundaries as a refinement indicator. Solutions for the Mach number at $t = 0$ and 10.1 are shown on the left of Figure 10.3 for a portion of the problem domain. The mesh used in each each case

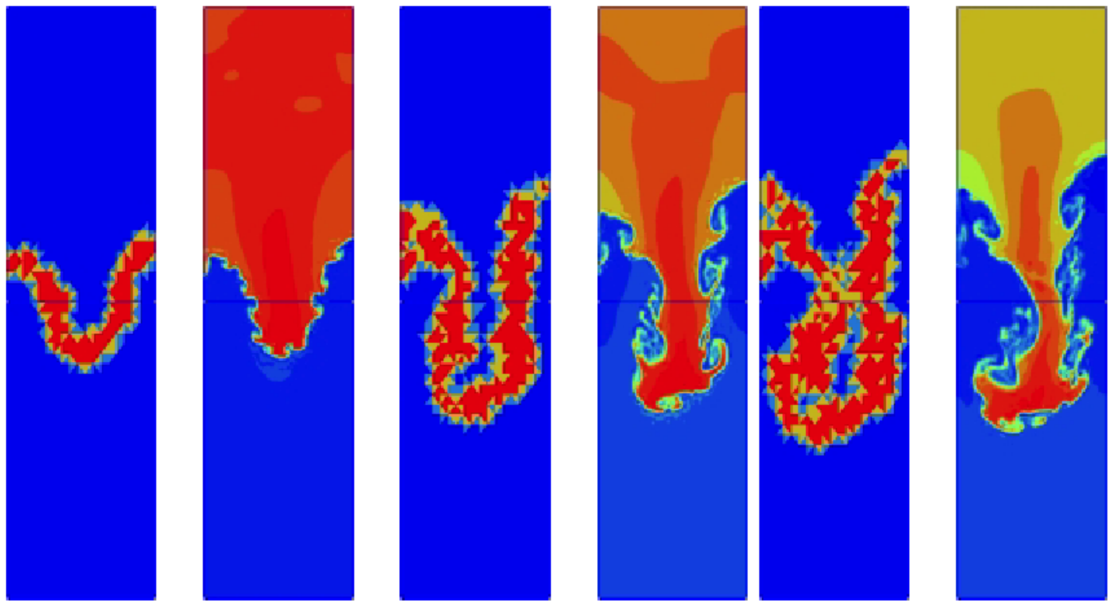


Figure 10.3.5: Density for the Rayleigh-Taylor instability of Example 10.1.1 at $t = 0.75$, 1.2, and 1.5 (left to right) obtained by adaptive p-refinement. The values of p used on each element are shown in the first, third, and fifth frames with blue denoting $p = 1$ and red denoting $p = 3$.

is shown on the right of the figure.

A shock forms on the downwind end of the vent tube and expansion forms on the upwind end. The mesh is largely concentrated in these regions where the rapid solution changes occur. The initial mesh consisted of 28,437 elements. This rose to more than 400,000 elements during the adaptive enrichment. This computation was done on 16 processors of a parallel computer. The coloring of the images on the right of Figure 10.3 indicates processor assignments.

The discontinuous Galerkin method is still evolving and many questions regarding flux evaluation, limiting, *a posteriori* error estimation, the treatment of diffusive problems, and its efficiency relative to standard finite element methods remain unanswered.

Problems

1. Construct a typical term in the mass matrix on the canonical element by integrating
$$\int_0^1 \int_0^{1-\xi} N_m(\xi, \eta) N_n(\xi, \eta) d\xi d\eta$$
using the basis of monomials (10.3.5).

2. Use the monomial basis (10.3.5) and the Gram-Schmidt process of Figure 10.3.2 to construct an orthogonal basis on the canonical right triangle for polynomials of

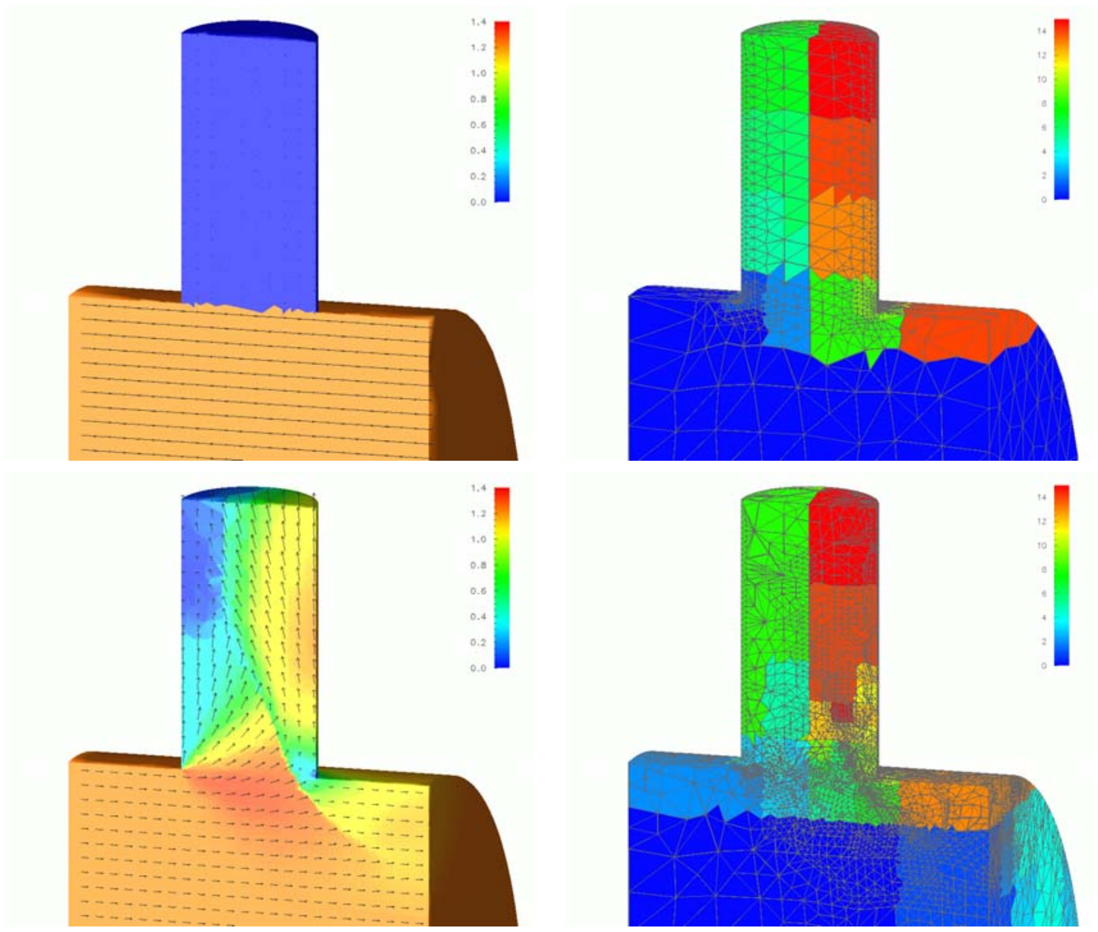


Figure 10.3.6: Mach contours (left) and adaptive meshes (right) used to solve the compressible flow problem of Example 10.3.2 at $t = 0$ (top) and $t = 10.1$ (bottom).

degree $p = 2$ or less.

Bibliography

[1] M. Abromowitz and I.A. Stegun. *Handbook of Mathematical Functions*, volume 55 of *Applied Mathematics Series*. National Bureau of Standards, Gathersburg, 1964.

[2] S. Adjerid, K.D. Devine, J.E. Flaherty, and L. Krivodonova. A posteriori error estimation for discontinuous Galerkin solutions of hyperbolic problems. In preparation, 2000.

[3] F. Bassi and S. Rebay. A high-order accurate discontinuous finite element method for the numerical solution of the compressible navier-stokes equations. *Journal of Computational Physics*, 131:267–279, 1997.

[4] C.E. Baumann and J.T. Oden. A discontinuous hp finite element method for convection-diffusion problems. to appear, 1999.

[5] K.S. Bey and J.T. Oden. hp-version discontinuous galerkin method for hyperbolic conservation laws. *Computer Methods in Applied Mechanics and Engineering*, 133:259–286, 1996.

[6] K.S. Bey, J.T. Oden, and A. Patra. hp-version discontinuous galerkin method for hyperbolic conservation laws: A parallel strategy. *International Journal of Numerical Methods in Engineering*, 38:3889–3908, 1995.

[7] K.S. Bey, J.T. Oden, and A. Patra. A parallel hp-adaptive discontinuous galerkin method for hyperbolic conservation laws. *Applied Numerical Mathematics*, 20:321–386, 1996.

[8] R. Biswas, K.D. Devine, and J.E. Flaherty. Parallel adaptive finite element methods for conservation laws. *Applied Numerical Mathematics*, 14:255–284, 1994.

[9] A.J. Chorin. Random choice solution of hyperbolic systems. *Journal of Computational Physics*, 25:517–533, 1976.

[10] B. Cockburn, G. Karniadakis, and C.-W. Shu, editors. *Discontinous Galerkin Methods Theory, Computation and Applications*, volume 11 of *Lecture Notes in Computational Science and Engineering*, Berlin, 2000. Springer.

[11] B. Cockburn, S.-Y. Lin, and C.-W. Shu. TVB Runge-Kutta local projection discontinuous finite element method for conservation laws III: One-dimensional systems. *Journal of Computational Physics*, 84:90–113, 1989.

[12] B. Cockburn and C.-W. Shu. TVB Runge-Kutta local projection discontinuous finite element method for conservation laws II: General framework. *Mathematics of Computation*, 52:411–435, 1989.

[13] K. Devine and J.E. Flaherty. Parallel adaptive hp-refinement techniques for conservation laws. *Applied Numerical Mathematics*, 20:367–386, 1996.

[14] K. Ericksson and C. Johnson. Adaptive finite element methods for parabolic problems I: A linear model problem. *SIAM Journal on Numerical Analysis*, 28:12–23, 1991.

[15] K. Ericksson and C. Johnson. Adaptive finite element methods for parabolic problems II: Optimal error estimates in $l_\infty l_2$ and $l_\infty l_\infty$. *SIAM Journal on Numerical Analysis*, 32:706–740, 1995.

[16] J.E. Flaherty, R. Loy, M.S. Shephard, B.K. Szymanski, J. Teresco, and L. Ziantz. Adaptive local refinement with octree load-balancing for the parallel solution of three-dimensional conservation laws. *Journal of Parallel and Distributed Computing*, 47:139–152, 1997.

[17] J. Glimm. Solutions in the large for nonlinear hyperbolic systems of equations. *Communications on Pure and Applied Mathematics*, 18:697–715, 1965.

[18] S.K. Godunov. A finite difference method for the numerical computation of discontinuous solutions of the equations of fluid dynamics. *Mat. Sbornik.*, 47:271–306, 1959.

[19] C. Johnson. Error estimates and adaptive time step control for a class of one step methods for stiff ordinary differential equations. *SIAM Journal on Numerical Analysis*, 25:908–926, 1988.

[20] P.D. Lax. *Hyperbolic Systems of Conservation Laws and the Mathematical Theory of Shock Waves.* Regional Conference Series in Applied Mathematics, No. 11. SIAM, Philadelphia, 1973.

[21] W.H. Reed and T.R. Hill. Triangular mesh methods for the neutron transport equation. Technical Report LA-UR-73-479, Los Alamos Scientific Laboratory, Los Alamos, 1973.

[22] J.-F. Remacle, J.E. Flaherty, and M.S. Shephard. Adaptive order discontinuous galerkin methods. In preparation, 2000.

[23] P.L. Roe. Approximate Riemann solvers, parameter vectors, and difference schemes. *Journal of Computational Physics*, 43:357–372, 1981.

[24] P. Le Saint and P. Raviart. On a finite element method for solving the newtron transport equations. In C. de Boor, editor, *Mathematical Aspects of Finite Elements in Partial Differential Equations*, pages 89–145, New York, 1974. Academic Press.

[25] G.A. Sod. *Numerical Methods in Fluid Dynamic*. Cambridge University Press, Cambridge, 1985.

[26] J.L Steger and R.F. Warming. Flux vector splitting of the inviscid gasdynamic equations with applications to finite difference methods. *Journal of Computational Physics*, 40:263–293, 1981.

[27] B. van Leer. Flux-vector splitting gor the Euler equations. *Lecture Notes in Physics*, 170:507–512, 1982.

[28] M.F. Wheeler. An elliptic collocation-finite element method with interior penalties. *SIAM Journal on Numerical Analysis*, 15:152–161, 1978.