Geometry-Based Three-Dimensional *hp*-Finite Element Modelling And Computations

by

Saikat Dey

A Thesis Submitted to the Graduate

Faculty of Rensselaer Polytechnic Institute

in Partial Fulfillment of the

Requirements for the Degree of

DOCTOR OF PHILOSOPHY

Major Subject: Civil Engineering

Approved by the

Examining Committee:

Mark S. Shephard, Thesis Advisor

Joseph E. Flaherty, Member

Jacob Fish, Member

Robert L. Spilker, Member

Rensselaer Polytechnic Institute

Troy, New York

January 1997

(For graduation May 1997)

Contents

List of Tables

List of Figures

Nomenclature

Domain Representation

Ω_v	Domain associated with the model $v, v = G, M$ where G signifies the geometric model and M signifies the mesh model.
Γ_v	Boudary of Ω_v defined as $\Gamma_v = \partial(\Omega_v)$.
$\overline{\Omega}_v$	Closure of domain associated with the model $v, v = G, M$ given by $(\Omega_v \cup \partial(\Omega_v))$.
M_i^d	Mesh topological entity i of dimension d, $d = 0$ is a vertex which represents a point in space, $d = 1$ is an edge which represents a 1-D locus of points, $d = 2$ is a face which represents a 2-D locus of points, $d = 3$ is a region which represents a 3-D locus of points.
G_i^d	Geometric model topological entity i of dimension d.
$\partial(M_i^d)$	Boundary of mesh topological entity M_i^d.
$\partial(G_i^d)$	Boundary of geometric model topological entity G_i^d.
$\overline{G}_i^d$	Closure of geometric model topological entity defined as $(G_i^d \cup \partial(G_i^d))$.
$\overline{M}_i^d$	Closure of mesh topological entity defined as $(M_i^d \cup \partial(M_i^d))$.
Ω_i^e	Domain of a finite element associated with a M_i^d.
Γ_i^e	Boundary of a finite element domain defined as $\Gamma_i^e = \partial(\Omega_i^e)$.
$\overline{\Omega}_i^e$	Closure of a finite element domain defined as $(\Omega_i^e \cup \Gamma_i^e)$.
$\sqsubset$	Classification symbol used to indicate the association of one or more entities from the mesh model, M with the geometric model, G.

Coordinate Systems

ξ_j	j-th component of parametric coordinates associated with M_i^d or Ω_i^e.
ζ_j	j-th component of parametric coordinates associated with G_i^d.
x_j	j-th component of parametric coordinate system of the partial differential equation.
$\oplus_i(\xi_j)$	Cyclic permutation of ξ_j, $j \neq i$.
$\ominus_i(\xi_j)$	Acyclic permutation of ξ_j, $j \neq i$.

Mesh Parameters

$h(\mathbf{x})$	Local mesh size.
$h_{min}(\mathbf{x})$	Local minimum mesh size prescribed as mesh edge length.
$h_{max}(\mathbf{x})$	Local maximum mesh size prescribed as mesh edge length.
$Q^k_{M_i^d}$	k-th quality metric for mesh entity M_i^d, $d \geq 2$.
$Q^k_{d,min}$	k-th quality metric minimum limit for mesh entity of dimension d, $d \geq 2$.

Sets

$\{v\}$	Unordered set.
$\{v\}_i$	i-th member of the unordered set $\{v\}$.
$\lfloor v \rfloor$	Ordered set.
$\lfloor v \rfloor_i$	i-th member of the ordered set $\lfloor v \rfloor$.

Finite Element Approximations

$u^{(h,p)}$	*hp* finite element approximation of unknown solution u using a mesh of size h and shape functions of polynomial order p.
$N^{(\alpha,p)}_{M_j^d}$	α-th hierarchic shape function of polynomial order p from $M_j^d \in \overline{\Omega}_i^e$.
N_i	Compact notation of i-th overall shape function when details of M_j^d and p are not important.
H^m	m-th *Sobolev* space of functions.
$\|u\|_m$	m-th *Sobolev* norm of u.
$\|u\|_e$	*Energy* norm of u.

Acknowledgments

Many people have helped and supported me through the several years of endeavor which culminated into this thesis.

I am grateful to Prof. Mark S. Shephard for his support and guidance throughout my doctoral research work. Mark has taught me the importance of not just doing good research but also the necessity of simple to follow and precise presentation of ones technical work. I thank him for his patience in going through the drafts of this thesis and providing valuable input to improve the overall presentation.

I thank the members of my thesis examination committee for their time and effort in reviewing this thesis. I am grateful to Prof. Joseph E. Flaherty for many insightful discussions that helped me get a better understanding of the mathematical aspects of the finite element method and numerical approximations in general.

The staff and my student colleagues at the Scientific Computation Research Center provided a helpful and stimulating research environment for which I am grateful.

I would like to acknowledge the financial support provided by the affiliates of the Scientific Computation Research Center, in particular, the Office of Naval Research and the Naval Research Laboratory. I also appreciate the many interactions I had with Dr. Joseph J. Shirron from NRL related to the application of the software technology developed as part of this thesis for solving problems in structural acoustics. Thanks are also due to Dr. Luise Couchman from NRL for her enthusiastic support of my doctoral research.

I would not be what I am today without the love, support, and understanding of all my family members. I shall be eternally grateful to all of them.

Abstract

Efficient computational methodologies for *hp*-finite element modeling and computation for the numerical solution of partial differential equations defined on curved three-dimensional domains are presented.

An algorithm to automatically detect and eliminate poorly shaped mesh entities resulting from extremely small geometric model features is described. The algorithm dramatically improves the quality of automatically generated meshes. An automated incremental algorithm based on local mesh modifications is developed to correct unacceptably distorted curvilinear mesh entities.

A novel decomposition of hierarchic shape functions, based on mesh topological adjacency, is developed that allows for the unified and consistent specification and evaluation of variable p-order C^0 finite element approximations over conforming meshes with mixed topology entities including line, triangle, quadrilateral, tetrahedron, hexahedron, wedge and pyramid elements.

A general and efficient scheme for evaluation of higher order element level integrals is presented based on direct polynomial approximation of the integrand followed by use of precomputed exact values of the resulting polynomial integrals. It is shown that to preserve the rate of convergence of the finite element solution error for second order elliptic boundary value problems using a p-th order basis, the integrand approximation must be at least accurate to order $p-1$. Blending schemes are used to construct mesh geometry mapping procedures that conform exactly to the domain boundary definition within a geometric modelling system. The concept of permutation of element parametric coordinates is developed to identify symmetries in the precomputed entries needed for elemental stiffness and mass matrices and elemental force vectors. Specific symmetries that reduce the number of precomputed entries which need to be stored are described.

Examples based on the solution of *Poisson's Equation* are presented to demonstrate applicability and efficiency of computational techniques developed in this thesis for the solution of partial differential equations in curved three-dimensional domains.

1. Background and Introduction

Computer based simulation is an important tool for scientific research and the solution of a wide range of industrial problems. The gamut of applications include the analysis and design of automotive and aircraft structures, semiconductor device simulations, study of human biomechanics, climate modelling, etc. A majority of computer simulations require numerical solution to the governing partial differential equations that mathematically describe the phenomena being studied. The efficiency, accuracy, and robustness of the simulations is therefore dependent on those of the underlying numerical solution procedure.

The finite element method has emerged as the tool of choice for solving partial differential equations on complex geometric domains for many applications. Its mathematical properties have been well established and there are mathematically rigorous error analysis techniques to accurately determine the error in the numerical solution in a choice of norms [20, 55, 62, 87, 89]. In addition to the partial differential equation(s), the complete specification of a problem requires a mathematical description of the problem domain, Ω_G, along with analysis attributes associated with Ω_G which describe the various boundary conditions and coefficients occurring in the partial differential equation(s). The finite element method constructs a piecewise approximation, typically polynomial[1], of the unknown solution field(s) over a discretized representation of the problem domain referred to as the mesh, Ω_M. Two parameters that can be varied to improve solution accuracy are the size of the finite elements, h, and the order of the basis functions, p, used in the approximation. The h-adaptive version of the method varies h while keeping p fixed, the p-adaptive version varies p while keeping h fixed, whereas the hp-adaptive version varies both h and p simultaneously to improve solution accuracy. Compared to only algebraic convergence rates possible using the h-adaptive version, the hp-adaptive version offers, for properly designed meshes, exponential rates of solution convergence [7, 38]. Various investigators have numerically demonstrated the computational optimality of the hp-adaptive version of the finite element method in achieving a numerical solution with a prescribed level of accuracy using a minimum amount of computational resources [3, 59, 89]. Recent works have also shown advantages of parallel hp-adaptive finite element analysis [25, 58, 61].

Exponential convergence of hp-approximations over curved domains with piecewise analytic boundary has been proven in [6]. However, efficient numerical realization of exponential convergence requires that the approximation of the domain geometry must be carefully controlled. Approximate geometry representation leads to approximate representation of boundary and initial conditions and to the approximate evaluation of element level integrals[2]. In order to sustain the exponential rate of convergence of *discretization error* as the hp-approximation is improved (by refining h and/or p), the errors in the approximation of the integrals and the boundary conditions, due to approximate geometry representation, must also decay exponentially. This requires that

1 Non-polynomial basis functions have also been used, see for example, [91].

2 Use of rational or other non-integrable basis functions can also contribute to integration error.

in a p-adaptive environment, the accuracy of geometry approximations used in approximating boundary conditions and that of geometry related terms appearing in element level integrals must vary with p.

Accurate domain geometry representation in an automated hp-adaptive analysis environment requires the ability to generate valid and acceptable curvilinear meshes. An effective implementation of the hp-adaptive strategy, therefore, must address the impact of domain geometry on the generation and adaptation of valid and acceptable curvilinear meshes as well as issues related to domain geometry approximation that influence the optimality of the analysis and error estimation (indication) process.

1.1 Scope and Outline of Thesis

The aim of this thesis is to develop the computational tools needed for the effective implementation of hp-adaptive techniques on complex geometric domains in three dimensions. Specific areas addressed include:

1. Automated generation and control of good quality hp-meshes directly from the mathematical description of the problem domain within a geometric modelling system.
2. A methodology for the specification and construction of variable order finite element approximations over conforming meshes.
3. Accurate geometry representation for element level integrals based on CAD data.
4. Computationally efficient integration of p-version matrices and vectors that do not destroy the optimality of the underlying hp-approximation.

The thesis is organized as follows: Chapter 2 overviews the components of a geometry based hp-adaptive framework and their interactions followed by a brief description of the geometry based mesh topological hierarchy underlying the framework. Chapter 3 deals with the issues in the automated generation of hp-meshes including controlling mesh quality in the presence of small geometric model features and ensuring validity and acceptability of curvilinear meshes. Chapter 4 describes the topology based specification and construction of variable order finite element approximations for conforming meshes. Chapter 5 discusses issues related to accurate geometry representation for higher-order finite element methods. Chapter 6 outlines a methodology for mesh geometry representation that is suitable for higher order approximations. Issues related to effective and efficient evaluation of element level integrals is discussed in Chapter 7. Finally, the thesis ends with a summary of contributions and a discussion of open research issues in Chapter 8.

2. Overview of Geometry Based Automated *hp*-Adaptive Framework

Adaptive *hp* methods are desired for their optimal convergence properties and reliability of the solution accuracy. Automating the entire *hp* adaptive process eliminates the labor intensive aspects of *hp* mesh generation and adaptation, and provides a computational environment with the flexibility of obtaining numerical solutions of varying accuracy based on the needs of specific applications with minimal user intervention. A two-dimensional *hp*-adaptive computational environment is described in [35]. In three dimensions, fully automated *h*-adaptive environments have been described in [9, 68]. Issues in the design and implementation of *hp*-adaptive finite element systems have also been discussed in [24, 60, 69]. Development of fully automated three-dimensional *hp*-adaptive computational environments is an active research area.

A fully automated *hp*-adaptive finite element computational environment must have a geometric modelling system to define the geometry of the problem domain, an automated mesh generator and adaptation module, analysis engine which supports variable *p*-order elements, and *a-posteriori* error estimation. A schematic representation of an automated *hp*-adaptive analysis environment that works off a geometry based problem specification is depicted in Figure 1. The problem input consists of:

1. The governing partial differential equation(s).
2. A mathematical description of the domain geometry, Ω_G, defined within a geometric modelling system, for example, Parasolid^{TM}[28] or Shapes^{TM} [94].
3. Analysis attributes[3] that describe boundary and initial condition(s) and coefficients associated with the partial differential equation(s).
4. The maximum allowable error in the finite element solution in a norm suitable for the problem.

The first step in the adaptive solution process is the generation of a starting mesh. Since the exact solution is not known *a-priori*, initial *h* and *p* specification for the mesh is usually based either on *a-priori* error estimates [63], or heuristics derived from known solution characteristics. The analysis engine computes the finite element solution based on the mesh and the polynomial order distribution specified. This involves computing the element level contributions to the global system of equations followed by the solution of the global system to determine the unknown solution coefficients. The *a-posteriori* error analysis module estimates the local and global error in the finite element solution in the appropriate norm(s). It then specifies local *h* and *p* refinements (coarsening) if the solution fails to meet the prescribed level of accuracy. The mesh adaptation module modifies the mesh size *h* and polynomial order *p* to produce the *hp*-mesh for the next adaptive iteration.

---

[3] The attributes needed to specify a particular problem are assumed to be directly associated with Ω_G. Although important in itself, a detailed description of the specifics of attribute storage and retrieval [76] is not necessary for the present discussion.

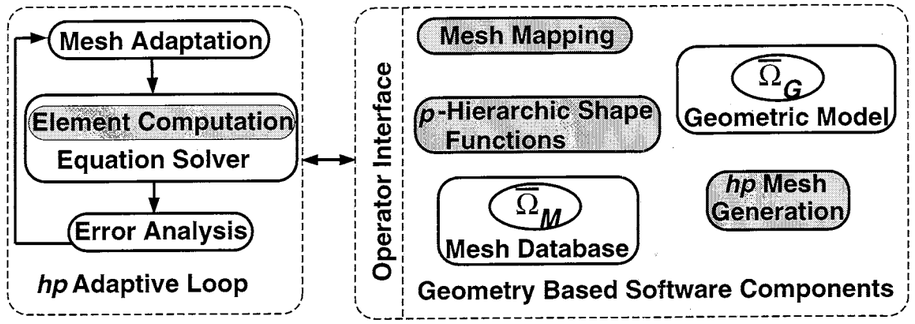


Figure 1. Geometry-based *hp*-adaptive computational environment.

The effective realization of an automated *hp*-adaptive environment for complex geometric domains requires the ability to specify and construct variable order (*p*-hierarchic) finite element approximations, construct and evaluate geometric mappings for curvilinear mesh entities, compute element level integrals, and solve the resulting system of equations. To be fully automated, all functional components must work directly off the geometric model description within the geometric modelling system as shown in Figure 1.

The focus of this thesis is limited to the shaded modules in Figure 1. Each module is built around a mesh topological hierarchy, described in the next section, that relates the mesh model to the geometric model.

2.1 Geometry Based Mesh Topological Hierarchy

Standard finite element data structures consisting of node point coordinates and element connectivity are not adequate for variable *p*-order meshes and should be replaced by data structures that allow the independent assignment of polynomial order over the elements [24, 50]. The basic structure used here for specifying a variable *p*-order mesh parallels the boundary representations used in geometric modeling systems. The structure assumes a general geometry-based specification of the problem being analyzed and relies on the relationship between the mesh and geometric specification of the domain to support several key functions in the construction of element stiffness matrices.

Although a number of schemes are possible for defining the closure of a geometric domain Ω_G [70], the most advantageous are boundary-based schemes denoted by

$$\overline{\Omega}_G(T_G, S_G) \tag{1}$$

where S_G represents the information defining the entity shapes and T_G represents the topological types and adjacencies[4] of the entities that define the domain. In addition to being unique, the use of topological entities and their associativities provides a convenient abstraction to define

[4] Adjacencies are the relationships between topological entities which bound each other, e.g., the edges that bound a face.

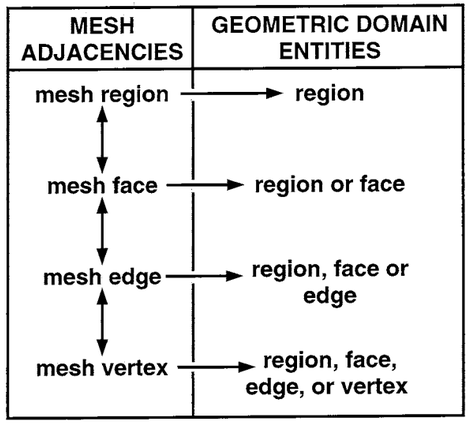


Figure 2. Mesh topological hierarchy.

the relationship between different models of the domain. In the present case, the two analysis domain models of concern are the geometric model and mesh model, which is a discretization of $\overline{\Omega}_G$. Boundary representations allow the convenient specification, with respect to the geometric domain, of the physical attributes for that analysis [76, 79]. Current computer aided design systems, furthermore, support a boundary representation of the domains defined within them. This allows the effective combination of these systems with automatic finite element analysis packages. A final advantage of recent boundary representations is their ability to properly represent non-manifold geometric domains commonly used for analysis [39, 53, 93].

Since the representation of the mesh domain, $\overline{\Omega}_M$, may not contain all of the necessary shape information of the geometric domain, $\overline{\Omega}_G$, it is critical to employ a representational scheme which can maintain the relationships between the two models. A boundary representation for the mesh consistent with that used for the geometric model is an ideal means to meet these needs [13].

Since individual finite elements are limited to simple regions that are bounded by simply connected faces, consideration of the topological entities for a mesh model can focus on the basic 0- to d-dimensional topological entities. For the three-dimensional case (d=3) these are:

$$T_M = \{M\{M^0\}, M\{M^1\}, M\{M^2\}, M\{M^3\}\} \tag{2}$$

where $M\{M^d\}$, $d = 0,1,2,3$, are, respectively, the set of vertices, edges, faces and regions defining the primary topological entities of the mesh model M. This structure maintains the relationship between the mesh entity and the geometric model entity to which it belongs [74, 79]. Figure 2 depicts a structure where the topological adjacencies between entities one geometric dimension apart are maintained, as well as the relationship of each mesh entity with respect to the geometric model. Alternative structures [13] that maintain knowledge of each topological entity can also meet the needs of the variable p-order mesh specification described here.

In addition to their crucial role in automated generation and adaptation of *hp*-meshes, the mesh topological adjacency relations [13] support a variety of operations critical to the effective

application of the finite element method, including equation ordering to minimize solution time, associating boundary loads with appropriate elements, and construction and evaluation of geometric mappings for various mesh entities. The use of the topological hierarchy in the specification of variable *p*-order meshes consists of the simple assignment of the desired polynomial order information with each topological entity in the mesh. Thus, each mesh vertex M_i^0, edge M_i^1, face M_i^2, and region M_i^3, can have its own polynomial order specification, including the possibility of different directional polynomial orders for faces and regions.

Although the full set of mesh adjacencies [13] used in defining topologically-based mesh data structures is not used here, it will be useful to specify the first-order adjacency sets of individual mesh entities

$$M_i^{d_i}\left\{M^{d_j}\right\}, \tag{3}$$

which defines the set of mesh entities of dimension d_j adjacent to mesh entity $M_i^{d_i}$. For example, the mesh vertices bounding mesh region i are denoted as $M_i^3\{M^0\}$, while the mesh edges that are bounded by mesh vertex i are denoted as $M_i^0\{M^1\}$. When employing a mesh description in terms of a topological hierarchy, the first-order adjacencies which constitute the closure of a finite element, $\overline{\Omega}_e$, of dimension d_e can be specified and evaluated as

$$\overline{\Omega}_e = \left\{M_e^{d_e}, \partial\left(M_e^{d_e}\right)\right\} = \left\{M_e^{d_e}, M_e^{d_e}\left\{M_j^{d_e-1}\right\}, \ldots, M_e^{d_e}\left\{M_j^0\right\}\right\}. \tag{4}$$

The concept of mesh entity classification and adjacency relations play a critical role in all aspects of automated *hp*-adaptive analysis addressed in this thesis. Mesh classification and adjacency information is essential in controlling quality and distortion of automatically generated *hp*-meshes. The definition of the closure of an element domain in equation (4) is used in the specification and construction of variable order finite element approximations. The construction of mesh geometry mapping uses the adjacency information and their evaluation utilizes the classification information to query relevant shape data from the geometric modelling system.

3. Automated Generation of *hp* Meshes

An essential requirement for an automated *hp*-adaptive procedure is the ability to generate and adapt, without user intervention, valid *hp*-meshes directly from the description of Ω_G contained in a geometric modelling system. There exist a number of approaches for automatic three dimension mesh generation [34, 51, 54, 66, 75, 92]. Two issues crucial to the automated generation of good quality curvilinear *hp*-meshes for real world Ω_G, (*i*) control of mesh quality in the presence of small geometric model features, and (*ii*) control of element distortion in curvilinear meshes, are addressed in the following two subsections.

3.1 Adverse Influence of Small Geometric Model Features on Quality of Automatically Generated Meshes

The quality of automatic generated meshes with straight edges and planar faces is typically measured in terms of one or more of the following, or similar, shape metrics [10, 17, 23]

1. **Aspect Ratio** measure:

   a. Normalized ratio of the inscribed sphere (circle) radius, r, to the circumscribed sphere (circle) radius, R,

$$Q^0_{M_i^d} \stackrel{def}{=} \frac{\lambda r}{R},\ d = 2,3 \tag{5}$$

   b. Normalized ratio of the minimum height, d_{min}, to the largest base length, l_{max},

$$Q^1_{M_i^d} \stackrel{def}{=} \frac{\kappa d_{min}}{l_{max}},\ d = 2,3 \tag{6}$$

   The meaning of r, R, d_{min} and l_{max} are depicted in Figure 3. λ and κ represent normalization constants.

2. **Angle** measure: Smallest and the largest dihedral (face) angle for mesh regions (faces), $\phi_{min}, \phi_{max} \in [0, 180]$ degrees which can be measured as the following normalized metrics:

   a. Small angle metric

$$Q^2_{M_i^d} \stackrel{def}{=} a\left(\frac{1 - cos(\phi_{min})}{2}\right),\ d = 2,3 \tag{7}$$

   b. Large angle metric

$$Q^3_{M_i^d} \stackrel{def}{=} b\left(\frac{1 + cos(\phi_{max})}{2}\right),\ d = 2,3 \tag{8}$$

   Where a and b are normalization constants.

Topology	λ	κ	a	b
Triangle	2	$\frac{2}{\sqrt{3}}$	4	$\frac{4}{3}$
Tetrahedron	3	$\sqrt{\frac{3}{2}}$	3	$\frac{3}{2}$

Table 1. Quality metric normalization factors for triangle and tetrahedron.

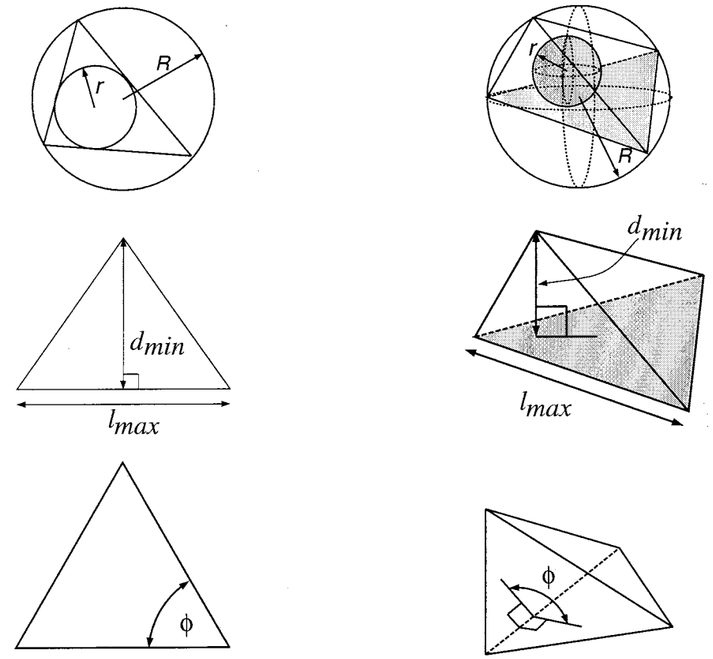


Figure 3. Quantities used in definition of entity quality metrics for 2D and 3D simplices.

Values of λ, κ, a, b for a triangle (d=2) and a tetrahedron (d=3) as given in Table 1 result in the following bounded quality metrics

$$0 \leq Q^k_{M^d_i} \leq 1,\ d = 2,3,\ \forall k. \tag{9}$$

The best value of the metrics is 1 and the worst value is 0.

The presence of small geometric model features leads to the creation of very small mesh entities adjacent to very large mesh entities resulting in poor aspect ratio and small angle metrics. The small angles and aspect ratio metrics influence the numerical conditioning of the resulting algebraic system of equations [47, 71]. Although important for maintaining theoretical rates of solution convergence [4, 48], the largest dihedral (face) angle is not considered here since its creation is not forced by the small geometric model features, but can result from the inability

of the mesh generation algorithm, in the presence of small geometric model features, to prevent *a-priori* the creation of large angles or to eliminate them in an *a-posteriori* mesh optimization procedure.

Definition and identification of small geometric model features is based upon comparison of size, and quality metrics, of mesh entities classified on the closure of a given geometric model entity to the user prescribed limits $h_{min}(\mathbf{x})$ and $Q^k_{d_i,min}$ for that particular model entity.

Definition: Small Geometric Model Feature – A geometric model entity, $G^{d_j}_j$, with $d_j > 0$, or a portion of it, is a small geometric model feature if the smallest feature dimension is smaller than the size that would allow the creation of mesh entities of the specified local mesh size which satisfy the mesh quality metric limit(s).

Small geometric model features make it impossible for automatic meshers to generate meshes that satisfy prescribed element shape limits

$$Q^k_{d,min} \leq Q^k_{M^d_i} \;\; \forall M^d_i, k;\; d \geq 2, k = 1..n_Q \tag{10}$$

in terms of n_Q quality metrics and simultaneously satisfy the mesh size limits in terms of mesh edge length

$$h_{min}(\mathbf{x}) \leq h(\mathbf{x}) \leq h_{max}(\mathbf{x}). \tag{11}$$

As an illustration of the problems faced by automatic mesh generators in the presence of small geometric model features, consider the geometric model of a turbine blade platform depicted in Figure 4. The geometric model has a small model edge $\frac{1}{5000}th$ the size of the largest dimension of the model bounding box. The model edge was created by the geometric modeler to patch up a gap in the model. The portion of the resulting mesh in the vicinity of the small model edge is shown in Figure 5(b). Notice that the presence of the small model edge leads to the creation of an extremely small mesh edge right next to much larger mesh edges resulting in sliver mesh faces and regions in the vicinity of the small model edge. These sliver entities have poor angles and aspect ratios and produce ill-conditioned stiffness matrices resulting in stiffening [71]. To improve the element shape measures to within acceptable limits while maintaining the explicit representation of the small geometric model feature requires the mesh to be refined to an extent that the size of all the mesh edges in the vicinity of the small model edge are of the order of the model edge size. This refinement leads to the violation of the mesh size prescription in the vicinity of the small geometric feature. Table 2 compares the element quality metrics before and after refinement in the vicinity of the small geometric model feature. Local refinement to improve the quality metrics marginally leads to a 33% increase in the number of mesh regions in the final mesh as shown in Figure 6. Thus, the presence of the small geometric feature results in poorly shaped mesh entities if mesh size is satisfied, or results in excessive number of mesh entities if shape quality metric(s) are to be improved.

To further highlight the undesirability of mesh refinement as a tool to improve mesh quality in the presence of small geometric model features consider the simple example shown in Figure 7. Notice that the presence of a slender model face in the geometric model results in the creation

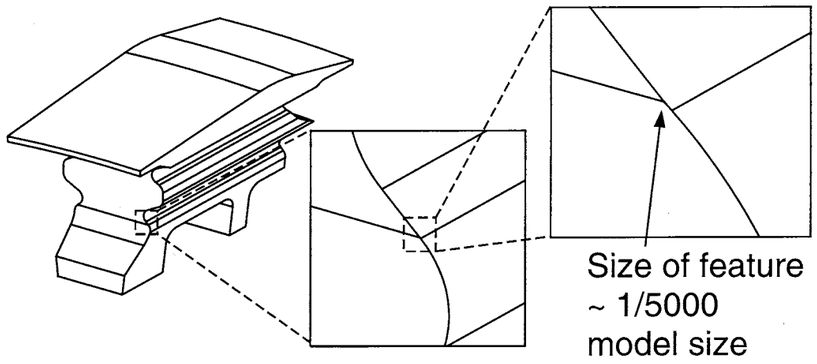
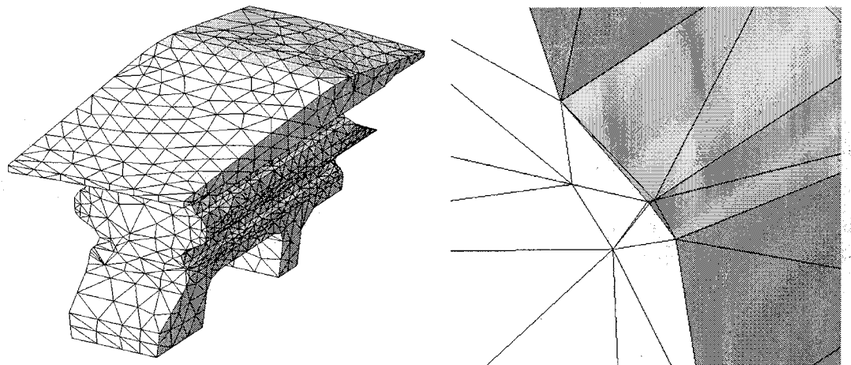


Figure 4. Industrial part example, geometric model with small model edge.

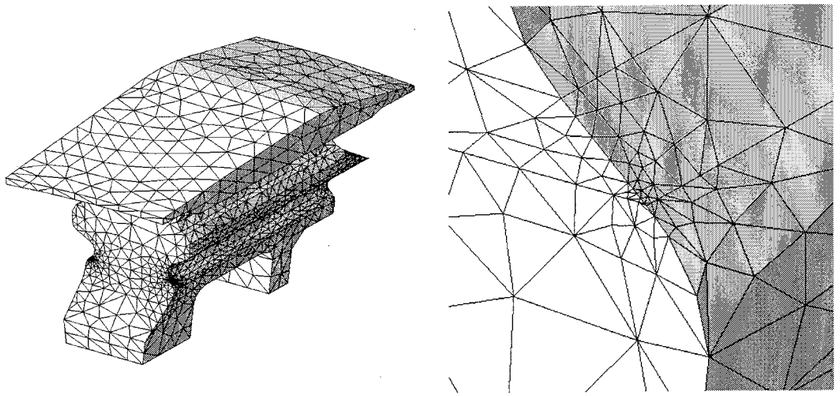


(a)

(b)

Figure 5. Initial mesh for industrial part example with 9767 tetrahedral regions with close-up of mesh in the vicinity of the small model edge.

of mesh faces and regions in the vicinity of the model face with a worst $Q^0_{M_i^3}$, $Q^1_{M_i^3}$ and $Q^2_{M_i^3}$ of 0.051, 0.022 and 3.0e-4 (1.146 degree) respectively. A thirteen-fold increase in the total number of mesh regions due to refinement in the vicinity of the slender model face is only able to improve the worst $Q^0_{M_i^3}$, $Q^1_{M_i^3}$ and $Q^2_{M_i^3}$ to 0.172, 0.13 and 0.02 (8.195 degree) respectively. The two examples demonstrate the adverse influence of the geometric model features on the quality of the resulting mesh. They also indicate that local refinement is unacceptable because of the large increase in number of mesh entities for a marginal increase in mesh quality.



(a) (b)

Figure 6. Mesh for industrial part example with 12950 tetrahedral regions with close-up of mesh in the vicinity of the small model edge.

Quality Metric	Initial Mesh	Refined Mesh
$Q^0_{M^3_i}$	0.02	0.03
$Q^1_{M^3_i}$	0.01	0.04
$Q^2_{M^3_i}$	2.87e-4 (1.12 degree)	2.1e-3 (3.0 degree)
Number of regions	9767	12950

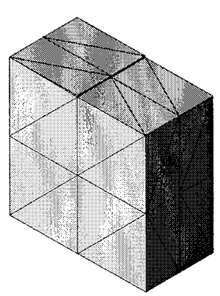
Table 2. Mesh refinement to improve element quality metrics for the industrial part shown in Figure 4.

3.1.1 Approach to Remove Adverse Influence of Small Geometric Model Features on Mesh Quality

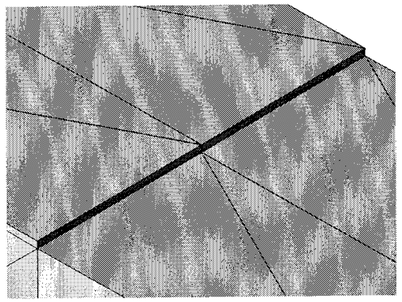
Two basic approaches can be adopted to eliminate the adverse influence of small geometric model features on the quality of automatically generated meshes. They are:

1. Modify the original geometric model Ω_G to eliminate all small geometric model features resulting in a modified geometric model Ω'_G. Use Ω'_G as the input to the automatic mesher.
2. Mesh the original geometric model Ω_G (including the small geometric model features) to obtain Ω_M. Employ specific local mesh modifications to Ω_M in the vicinity of the small geometric model features to eliminate poorly shaped mesh entities.

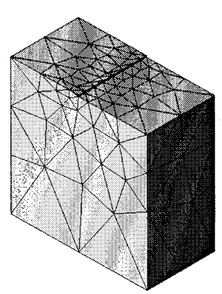
Although the first approach seems promising, it has several drawbacks. First, procedures to automatically detect "small" geometric model features, and to carry out suitable modifications of a geometric model of arbitrary complexity are complex and not known for all circumstances. Manual elimination of the small geometric features, when possible, is extremely time consuming



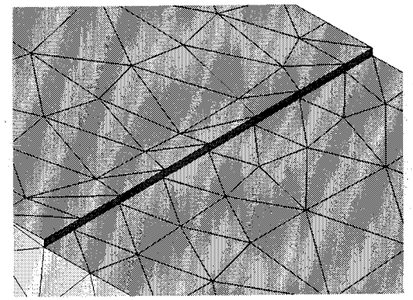
(a)



(b)



(c)

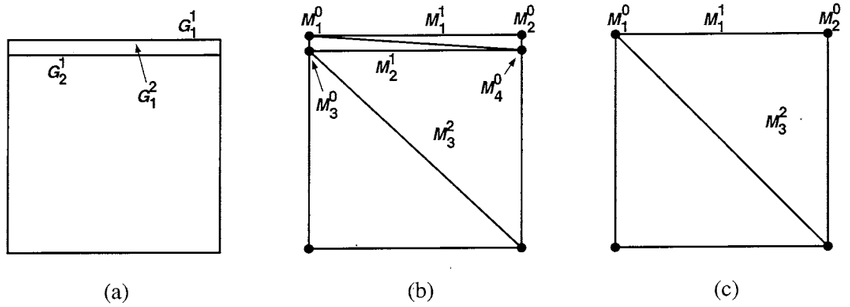


(d)

Figure 7. Refinement to improve element quality metric in the presence of small geometric model features.

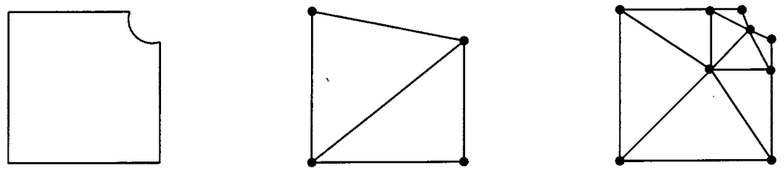
and error prone procedure. A second drawback comes from the fact that in an adaptive environment, the mesh must be able to relate back to the original geometric model. Adaptive mesh refinements must be done against the original geometric model to ensure validity of the process, and the ability to improve the mesh's geometric approximation of the problem domain, including re-inclusion of the small geometric model features when the adaptive mesh refinement procedures refine the mesh to the level of the small geometric model feature.

The second approach is to modify the mesh locally in the vicinity of the small geometric model features such that the resulting mesh does not have poor quality entities in it. A simple illustration of this idea is presented in Figure 8. The mesh with poor shape metric (Figure 8(b))



(a) (b) (c)

Figure 8. Local mesh modification approach: (a) geometric model with small model features, (b) initial mesh and (c) locally modified mesh.



(a) (b) (c)

Figure 9. (a) Original geometric model Ω_G, (b) initial Ω_M with feature eliminated and (c) adaptively refined Ω_M with feature re-included.

can be modified locally to obtain the desired mesh (Figure 8(c)). The modifications must be done such that the modified mesh is still tied to the original geometric model. This will enable adaptive refinements as predicted by *a-posteriori* error analysis to effectively improve the mesh's geometric approximation of the original geometric model (Figure 9). There are two complexities of this approach that must be overcome if it is to be used. First, it is not obvious how to maintain a proper set of relationships of the mesh entities of the modified mesh to the original geometric model, such that fundamental validity requirements of an automatically generated mesh as defined in references [73, 74, 81] are satisfied, and mesh refinements are able to properly recover the original model geometry as depicted in the example in Figure 9(c). The second issue is to ensure that local mesh modifications do not result in local dimensional reduction in the mesh model that invalidate or violate assumptions made by the analysis process.

This thesis presents a procedure based on the second approach of local mesh modification. The validity of this approach hinges on first solving the complexities introduced by this approach. This is done in the next section. With those issues resolved, the development and implementation of the procedure is reasonably straightforward.

3.1.2 Definition of a Valid Mesh Accounting for Small Geometric Model Feature Elimination by Local Mesh Modifications

In this section the definition of mesh validity is reviewed in the context of local mesh modifications needed to eliminate the adverse influence of small geometric model features on the mesh quality. The existing definition of mesh entity classification with respect to the geometric model is augmented to ensure validity of the modified mesh. The issue of dimensional reduction of the mesh model and its prevention is also addressed.

3.1.2.1 Review of the Definition of Mesh Validity and its Limitations in Eliminating Influence of Small Geometric Model Features A number of algorithmic approaches have been proposed for automatic three-dimensional mesh generation [2, 32, 82]. Many papers focus on the generation of meshes from a given surface triangulation. Ensuring the validity of the mesh generation process in these cases only requires maintenance of the given surface triangulation and ensuring that the elements created are interior to the model. In the case where the domain being meshed is a curved geometry defined within a geometric modeling system, there are no preexisting set of finite element entities separating the interior from the exterior of the domain. Therefore, a definition of a valid triangulation with respect to the geometric model, a geometric triangulation, is needed.

Definition: Geometric Triangulation — Given a set P of M unique points, each classified with respect to the geometry Ω_G, an n-dimensional geometric triangulation, T_g^n is a set of N nondegenerate elements $s_i^{d_i}$

$$T_g^n = \left\{ s_1^{d_1}, s_2^{d_2}, s_3^{d_3}, \ldots, s_N^{d_N} \right\} \tag{12}$$

with $0 \leq d_i \leq n$, satisfying the following properties:

i. *All vertices of each $s_i^{d_i} \in P$*
ii. *For each $i \neq j$, interior $\left(s_i^{d_i}\right) \cap$ interior $\left(s_j^{d_j}\right) = 0$*
iii. *T_g^n is topologically compatible with Ω_g*
iv. *T_g^n is geometrically similar to Ω_g*

In this case the creation of the mesh must be through interactions with the geometric model representation. Since the surface triangulation is only an approximation of the true boundary of the solid model (at least until the surface triangulation is properly identified and the mesh entities can be assigned the shape of that portion of the model), simple geometric measurements are not satisfactory for determination of validity [73, 74, 77]. This is why the definition of a geometric triangulation employs the concepts of *topological compatibility* and *geometrical similarity* of the mesh Ω_M with respect to the original geometric model Ω_G [73, 74, 77]. It can be shown that this definition leads to a general algorithm to convert a triangulation of a domain into a valid finite element mesh. A key component of this definition is the *classification* of each of the finite element mesh entities with respect to the entities defining the original geometric model, including the small geometric features.

Definition: Classification — The unique association of a topological mesh entity of dimension d_i, $M_i^{d_i}$, to a topological model entity of dimension d_j, $G_j^{d_j}$, where $d_i \leq d_j$, is termed classification and is denoted

$$M_i^d \sqsubset G_i^d \tag{13}$$

where the classification symbol, $\sqsubset$, indicates that the left hand entity or set is classified on the right hand entity.

Multiple M_i^d can be classified on a G_i^d, but each M_i^d can be classified on only one G_i^d.

Topological compatibility of the mesh with respect to the original geometric model requires that each G_i^d in Ω_G is completely covered by the set $\left\{M_j^d | M_j^d \sqsubset G_i^d\right\}$. It also requires that each classified mesh entity is used properly (both in number and orientation) by other classified mesh entities. From a simplistic viewpoint, geometric similarity requires that in the limit of mesh refinement the Ω_M replicates the exact geometry of Ω_G.

The local mesh modifications required to eliminate the adverse influence of small geometric model features leads to a mesh that violates the topological compatibility requirements as just stated. To illustrate this point reconsider the mesh entity deletion in the example of Figure 8. In the modified mesh (Figure 8(c)), no set of mesh entities is classified as being strictly on the geometric model face G_1^2 and the model edge G_2^1. As is evident from the mesh in Figure 8(c), these model entities are "covered" by the two remaining mesh entities, with the mesh being, in some sense, a valid one with respect to the entire domain. More specifically, the elimination of the adverse effects of the small geometric model features from the mesh model yields a mesh with entities that span more than one model entity. It is by maintaining knowledge of this information that, in the case of properly performed eliminations, one can proceed. However, the stated definition of *classification* insists that each mesh entity must be uniquely associated with one, and only one, model entity. To allow the elimination of the adverse influence of the small geometric model features, the definition of *classification* must be extended to enable mesh entities to be classified against multiple model entities.

3.1.2.2 Extended Definition of Mesh Classification: *Multiple Classification* If the elimination of the adverse influence of a small geometric model feature causes the resulting mesh entities to span the small geometric model feature and some portion of another model entity, the classification of those mesh entities is extended to indicate that they span portions of multiple model entities. Since the influence of the small geometric model features is eliminated by the modification of a valid mesh with respect to the original geometric model, the mesh entities with multiple classifications maintain the *primary* classification with respect to the model entity whose discretization caused the mesh entity to be created during the original mesh generation process. The added *auxiliary* classifications represent the additional model entities that the mesh entity ends up spanning as a result of the deletion of the *small* mesh edges. The extension of classification information to include multiple classifications adds no more complexity in the mesh database [13] since it already has the mechanisms setup to relate each and every mesh entity to the geometric model entities.

Geometric similarity requires that each mesh entity must be able to represent the shape of the portion of the model entities that it spans without intersecting other mesh entities. In the case of mesh entities classified on model edges and faces, the minimal set of geometric information that is needed to represent the shape of each mesh entity consists of the location of each mesh vertex in the parametric space of model entities that it is classified on [13] including the *primary* and the *auxiliary* ones. The coordinates of all mesh vertices in euclidean three space are also stored. For any $M_i^d, d > 0$, exact pointwise geometric information required during the mesh generation or the subsequent analysis process, including position and derivatives, can be evaluated using the location of the bounding mesh vertices in the parametric space of the model entities on which it is classified [26].

If as a result of a local mesh modification two mesh entities $M_i^d \sqsubset G_k^{d_k}$ and $M_j^d \sqsubset G_l^{d_l}$ become coincident then the mesh entity classified on the higher order model entity is replaced by the other mesh entity. If both mesh entities are classified on equal order model entities then the choice of the mesh entity to replace is arbitrary. In other words, M_i^d replaces M_j^d if $d_k < d_l$ whereas M_j^d replaces M_i^d if $d_l < d_k$. If $d_k = d_l$ then either M_i^d or M_j^d can be replaced by the other.

The general procedure to specify multiple classification and parametric coordinate information following a local mesh modification is based on the following rules:

1. For any mesh entity, $M_i^d \sqsubset G_k^{d_k}$, replaced by another mesh entity, $M_j^d \sqsubset G_l^{d_l}$, with $G_k^{d_k} \neq G_l^{d_l}$ and $G_l^{d_l} \notin \overline{G_k^{d_k}}$, attach $G_k^{d_k}$ as an auxiliary classification with M_j^d.
2. For any mesh vertex assigned the classification of a model edge or face, record the coordinates of the mesh vertex in the parametric space of the model entity.

These rules lead to unique and complete classification information of the modified mesh with respect to the original geometric model. The starting mesh is topologically compatible with the original geometric model and the local mesh modification(s) do not create any new gaps or holes in the mesh. Therefore, the locally modified mesh maintains topological compatibility with respect to the original geometric model.

The two-dimensional example in Figure 10 depicts the classification update procedure through two mesh edge deletions. The geometric model and the initial mesh is shown in Figure 10(a,b) respectively. Figure 10(c) shows the mesh entity classifications following the deletion of mesh edge M_5^1. Deletion of mesh edge M_5^1 results in mesh vertices $M_5^0 \sqsubset G_5^0$ and $M_3^0 \sqsubset G_3^0$ and mesh edges $M_7^1 \sqsubset G_7^1$ and $M_9^1 \sqsubset G_2^2$ to become coincident. Since M_3^0 and M_5^0 are both classified on model vertices, M_5^0 is chosen to be replaced and G_5^0 is attached as auxiliary classification of M_3^0 because $G_3^0 \neq G_5^0$ and $G_3^0 \notin \overline{G_5^0}$. On the other hand, M_9^1 is replaced by M_7^1 since G_2^2 is higher order than G_7^1. In addition, since $G_7^1 \in \overline{G_2^2}$, G_2^2 is not attached as auxiliary classification with M_7^1. Deletion of mesh edge M_6^1 leads to the mesh in Figure 10(d). In this case mesh vertices $M_4^0 \sqsubset G_4^0$ and $M_6^0 \sqsubset G_6^0$ and mesh edges $M_7^1 \sqsubset G_7^1$ and $M_2^1 \sqsubset G_2^1$ become coincident. Mesh vertex M_6^0 is replaced by M_4^0 and G_6^0 is attached as auxiliary classification with M_4^0 since $G_4^0 \neq G_6^0$ and $G_4^0 \notin \overline{G_6^0}$. Similarly, mesh edge M_7^1 is replaced by M_2^1 and G_7^1 is attached

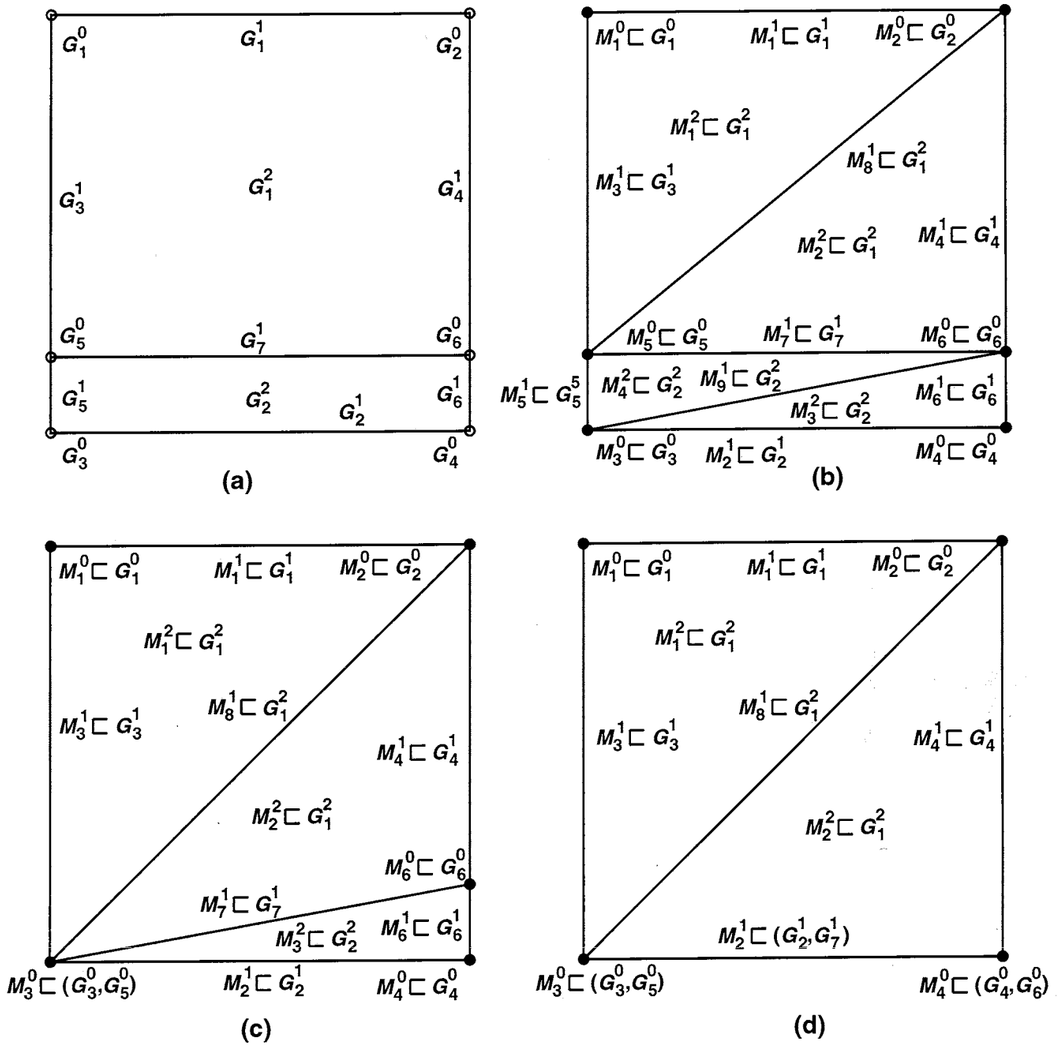


Figure 10. Multiple classification update: (a) geometric model, (b) initial mesh, (c) after deleting M_5^1 and (d) after deleting M_6^1.

as auxiliary classification with M_2^1 since $G_2^1 \neq G_7^1$ and $G_2^1 \notin \overline{G_7^1}$. No additional parametric information needs to be recorded with vertices M_3^0 and M_4^0 because both were assigned auxiliary classifications of model vertices only.

Splitting mesh entities with multiple classifications results in new mesh entities, some of which span multiple model entities, while others span a single model entity. To maintain consis-

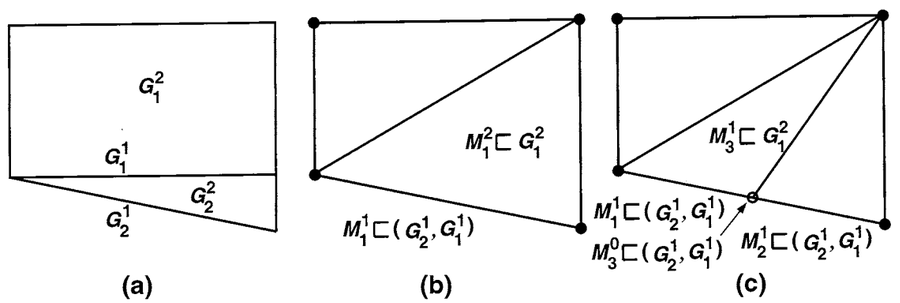
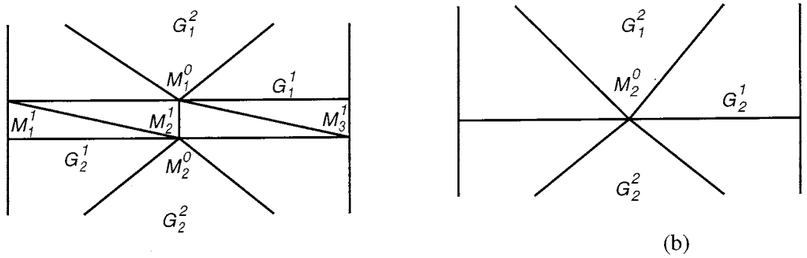


Figure 11. Splitting mesh entities with multiple classification: (a) geometric model, (b) starting mesh and (c) final mesh.

tency of multiple classifications, newly created mesh entities are assigned multiple classification only if they span multiple model entities. This is achieved by attaching multiple classification to only those new mesh entities that are used to split existing mesh entities with multiple classification. The example in Figure 11 illustrates this process. Consider splitting multiply classified mesh edge M_1^1 for the mesh in Figure 11(b) to obtain the mesh shown in Figure 11(c). Only new mesh vertex M_3^0 and mesh edges M_1^1 and M_2^1 are assigned multiple classification because they are used to split mesh edge M_1^1. On the other hand newly created mesh edge M_3^1 used to split mesh face $M_1^2 \sqsubset G_1^2$ is not assigned multiple classification because $M_1^2 \sqsubset G_1^2$.

Procedures using classification information to derive pointwise geometric information for mesh entities with multiple classification must use the correct classification. When dealing with a given model entity, the correct classification to use is the one that belongs to the closure of the model entity of interest. For example, "smoothing" of mesh vertices classified on the model boundary is often done by iterative relocation in the parametric space of the model entity [22]. The example in Figure 12 explains how correct and consistent pointwise geometric information is derived for mesh entities with multiple classification. Elimination of the influence of the thin model face results in mesh vertex M_2^0 being classified on model edges G_1^1 and G_2^1. When smoothing in the parametric space of model face G_1^2, geometric information about M_2^0 is derived based on $M_2^0 \sqsubset G_1^1$ and parametric coordinates with respect to G_1^1 because $G_1^1 \in \partial(G_1^2)$. On the other hand, when smoothing in the parametric space of G_2^2, $M_2^0 \sqsubset G_2^1$ and parametric coordinates with respect to G_2^1 are used because $G_2^1 \in \partial(G_2^2)$.

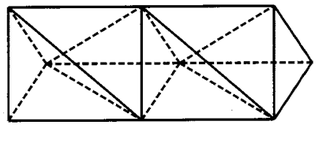
3.1.2.3 Ensuring no Dimensional Reductions in Locally Modified Mesh In general, the elimination of the adverse influence of a small geometric model feature by local mesh modification can lead to local, or even global, dimensional reductions which invalidate the discretization's ability to represent basic model behavior. A simple explanation of a dimensional reduction is when a portion of Ω_G of dimension d is reduced to dimension d', $d' < d$, in a local region of the mesh model, Ω_M. Figure 13 shows instances of local dimensional reductions introduced when a set of mesh entities are collapsed for a mesh of a three-dimensional geometric domain. For example, a portion of the solid region in Figure 13(a) is reduced to a mesh vertex,



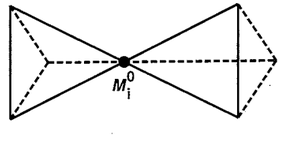
(a)

(b)

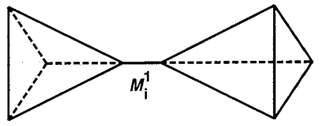
Figure 12. Recording geometric information.



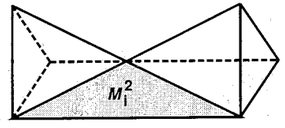
(a)



(b)



(c)



(d)

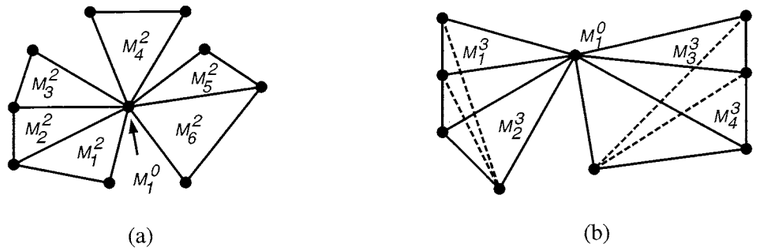
Figure 13. Dimensional reduction (a) valid mesh, (b,c,d) dimensionally reduced mesh models.

a mesh edge and a mesh face in Figures 13(b), 13(c) and 13(d) respectively. Without the introduction of appropriate analysis idealizations associated with each such dimensional reduction, the assumptions of the analysis procedures are invalidated. Since automated procedures to introduce such idealizations are not available in general, and even if they were, it is currently not clear how to control their influence on the quality of the results obtained. Therefore, a requirement of a valid mesh when eliminating the effects of small geometric model features is that there must not be any dimensional reductions introduced.

Definition: Dimensional Reduction Constraint – Local mesh modification(s) of a valid mesh is (are) valid only if

i. *for any mesh entity* $M_i^d, d = 0, 1$, *the total number of connected* ***mesh entity cycle(s)*** *of dimension* $d_j, (d+2) \leq d_j \leq 3$, *does not increase, and,*

ii. *the total number of* ***topological use(s)*** *of any mesh entity* $M_i^d, d = 0, 1, 2$, *by mesh entities* M_j^{d+1}, *is not reduced to zero.*



(a) (b)

Figure 14. Topological cycles of (a) mesh faces (b) mesh regions.

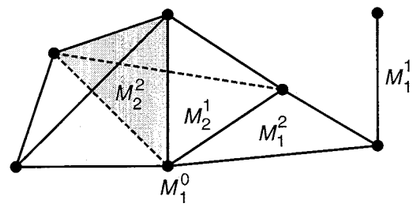


Figure 15. Topological use counts.

Otherwise, the modification leads to a dimensionally reduced (invalid) mesh.

Definition: Mesh Entity Cycle – A cycle of mesh topological entities of dimension d (d>0) is defined by the ordered set, $\lfloor M^d \rfloor$*, where any two adjacent elements,* $\lfloor M^d \rfloor_i$ *and* $\lfloor M^d \rfloor_{i+1}$*, share a d-1 dimension mesh entity* M_j^{d-1}*, that is,*

$$\exists\, M_j^{d-1} = \overline{\lfloor M^d \rfloor}_i \cap \overline{\lfloor M^d \rfloor}_{i+1}. \tag{14}$$

Figure 14(a) depicts three cycles of mesh faces, $\lfloor M_1^2, M_2^2, M_3^2 \rfloor$, $\lfloor M_4^2 \rfloor$ and $\lfloor M_5^2, M_6^2 \rfloor$ around mesh vertex M_1^0. Similarly, Figure 14(b) depicts two cycles of mesh regions, $\lfloor M_1^3, M_2^3 \rfloor$ and $\lfloor M_3^3, M_4^3 \rfloor$ around mesh vertex M_1^0.

Definition: Topological Use Count – Total number of ***topological use(s)*** *of any mesh entity* $M_i^d, d = 0, 1, 2$*, is defined by the number of elements in the set* $\left\{M^{d+1}\right\}$ *where*

$$M_i^d \in \partial\left(\left\{M^{d+1}\right\}_j\right) \forall j. \tag{15}$$

As an illustration of how topological uses are counted refer to the mesh in Figure 15. The topological use count for vertex M_1^0 is 5 since it is used by five mesh edges. Similarly, use count for M_2^1 and M_2^2 are 3 and 2 respectively. Use count for M_1^1 and M_1^2 are zero since they do not bound any higher order mesh entity.

The implementation of the dimensional reduction constraint for the local mesh modification tool(s) is described in Section 3.1.3.2.

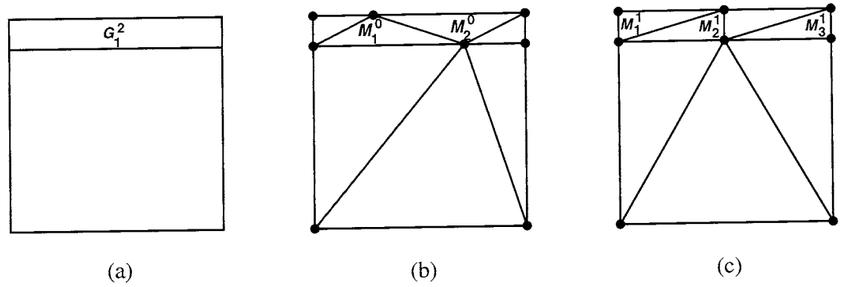


Figure 16. Large angle prevention for models with small geometric model features: (a) geometric model, (b) large angles due to misaligned vertices and (c) large angles removed.

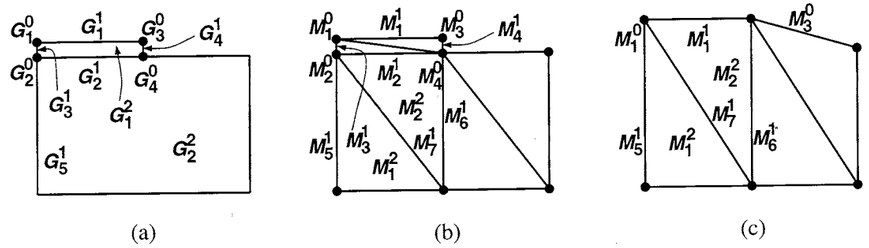
3.1.3 Procedure to Eliminate Adverse Effects of Small Geometric Model Features by Local Mesh Modifications

The steps in the procedure to eliminate the adverse influence of small geometric model features are:

1. Identify and flag mesh entities that need to be deleted in order to eliminate the mesh entities with poor quality metrics.
2. Check that the deletion of the flagged mesh entities result in a valid mesh by ensuring that:
   a. all mesh entities remain geometrically valid (positive volume/area), and
   b. no local dimensional reductions in the mesh are introduced.
3. Apply appropriate mesh modification operator(s) to delete the desired mesh entity and update classification and pointwise geometric information as required to ensure mesh validity.

3.1.3.1 Identifying Mesh Entities to be Eliminated The adverse effect of the small model features is reflected in a mixed dimension mesh by the presence of mesh edges, M_i^1, classified on the small geometric model feature which are smaller than locally prescribed mesh size, $h_{min}(\mathbf{x})$. The *small* mesh edges create mesh faces and regions with unacceptable small angle and large aspect ratio measures. The elimination of these unacceptable mesh faces and regions require the small mesh edge to be deleted. If the meshing algorithm does not explicitly control large angles, the small geometric model features can also lead to the creation of large angles due to misaligned mesh vertices. As an illustration, consider the example in Figure 16 where the geometric model has a "thin" model face, G_1^2. Large angles result due to misalignment of vertices M_1^0 and M_2^0 as shown in Figure 16(b). These large angles can be avoided by ensuring that the mesh vertices are positioned to avoid large angles (Figure 16(c)). However, the mesh faces with unacceptable small angles and aspect ratios cannot be eliminated unless the small mesh edges M_1^1, M_2^1 and M_3^1 are deleted from the mesh.

Mesh regions (faces) with poor aspect ratio and small angle measures caused by the small geometric model feature are eliminated from the mesh by deleting the small mesh edge opposite



(a) (b) (c)

Figure 17. (a) Ω_G with small model features, (b) initial mesh and (c) modified mesh.

to the edge (vertex) with the unacceptable small dihedral (face) angle. The example in Figure 17 illustrates this concept in two dimensions. Model edges G_3^1 and G_4^1 in Figure 17(a) force the creation of *small* mesh edges M_3^1 and M_4^1 leading to mesh faces with poor aspect ratio and small angles at vertices M_1^0 and M_4^0 respectively, as shown in Figure 17(b). Deleting the *small* mesh edges eliminates the unacceptable mesh faces resulting in the final mesh with $M_1^0 \sqsubset (G_1^0, G_2^0)$, $M_3^0 \sqsubset (G_3^0, G_4^0)$, $M_1^1 \sqsubset (G_1^1, G_2^1)$, $M_5^1 \sqsubset G_5^1$, $M_7^1 \sqsubset G_2^2$, $M_6^1 \sqsubset G_1^2$, $M_1^2 \sqsubset G_2^2$ and $M_2^2 \sqsubset G_2^2$ as shown in Figure 17(c). The example in Figure 18 shows how mesh edge deletion is used to eliminate unacceptable mesh regions. The small geometric model edge G_4^1 in Figure 18(a) forces the creation of mesh region M_3^1 with poor aspect ratio and dihedral angle along mesh edge M_3^1 (Figure 18(b)). Deletion of the *small* mesh edge $M_4^1 \sqsubset G_4^1$ eliminates the mesh region M_1^3 as shown in Figure 18(c). Mesh validity with respect to the original geometric model requires the following multiply classified mesh entities in the modified mesh: $M_2^0 \sqsubset (G_1^0, G_2^0)$, $M_1^1 \sqsubset (G_5^1, G_1^1)$, $M_2^1 \sqsubset (G_6^1, G_2^1)$, $M_5^1 \sqsubset G_7^1$, $M_1^2 \sqsubset G_1^2$ and $M_2^2 \sqsubset G_3^2$.

The adverse influences of small geometric model features can be eliminated using the correct combination of local mesh modification tool(s) capable of deleting a mesh edge. However, it is not always easy to prevent large angles in the vicinity of small geometric model features in automatically generated meshes. As a result, there may be some mesh faces (regions) with unacceptable large angles which need to be checked for and eliminated separately. An example of such a situation is illustrated by mesh face M_1^2 in Figure 19(a). One simple solution is to delete face M_1^2 from the mesh followed by reclassification of the interior edges M_2^1 and M_3^1 and interior vertex M_1^0 on the boundary as depicted in Figure 19(b).

Definition: Small Mesh Edge – A mesh edge, M_i^1, is classified as ***small*** *if*

$$l < \alpha h_{min}(\mathbf{x}); \; 0 < \alpha \leq 1 \qquad (16)$$

and

$$Q^k_{M_j^{d_j}} < Q^k_{d_j,min} \qquad (17)$$

for any k, where, l represents the length of the mesh edge, $h_{min}(\mathbf{x})$ represents the local mesh size expressed as the optimal edge length, $Q^k_{M_j^{d_j}}$ is the k-th quality metric for any mesh face or region

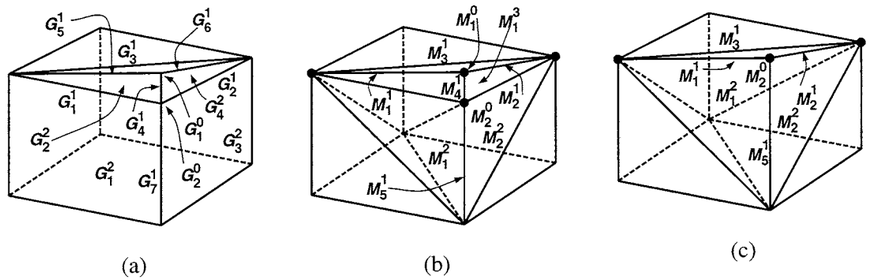


Figure 18. (a) Ω_G with small model edge G_4^1, (b) mesh with unacceptable small angle and aspect ratio, (c) locally modified mesh.

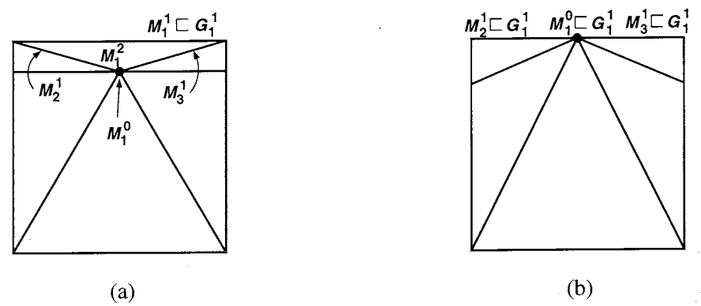


Figure 19. Eliminating unacceptable large angles.

bounded by the mesh edge, $\left\{M_j^{d_j} | d_j > 1, M_i^1 \in \partial\left(M_j^{d_j}\right)\right\}$ *and* $Q_{d_j,min}^k$ *represents prescribed minimum entity quality metric.*

In the current implementation, small mesh edges are flagged based on $Q_{2,min}^1$. The choice of face based aspect ratio metric is due to the lower computation cost compared to explicit computation of angles, and also because for mixed dimension meshes, mesh faces of interest may not have any regions connected to them.

3.1.3.2 Ensuring Validity of Locally Modified Mesh Any *small* mesh edge deletion that introduces dimensional reduction is disallowed.

Cycle Check: Deletion of a mesh edge, M_i^1*, is disallowed if*

i. *the surviving vertex,* $\left\{M_j^0 | M_j^0 \in \partial(M_i^1)\right\}$ *has more face cycles after deletion compared to before deletion, or*

ii. *any connected mesh face,* $\left\{M_j^2 | M_i^1 \in \partial\left(M_j^2\right)\right\}$*, is reduced to a mesh edge that has more region cycles after deletion than before deletion.*

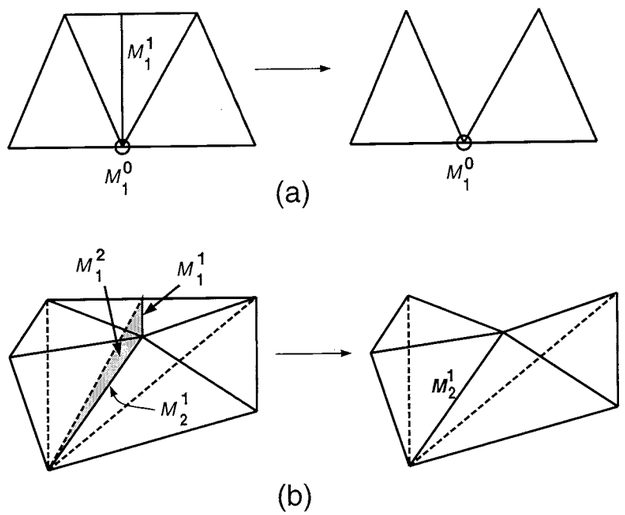


Figure 20. Deletion disallowed due to dimensional reduction: (a) multiple face cycles around a vertex and (b) multiple region cycles around an edge.

Figure 20(a) illustrates the first condition where the deletion of mesh edge M_1^1 is not allowed because it leaves mesh vertex M_1^0 with two face cycles after deletion when it had a single face cycle before deletion. Figure 20(b) shows an example of the second condition where deletion of mesh edge M_1^1 is unacceptable because it reduces mesh face M_1^2 to mesh edge M_2^1 which has two region cycles connected to it after deletion compared to one connected region cycle before deletion.

Topological Use check: Deletion of mesh edge, M_i^1, is disallowed if for all deleted mesh entities, $\left\{M_j^{d_j} | M_i^1 \in \partial\left(M_j^{d_j}\right), d_j = 2,3\right\}$, the topological use count for the surviving mesh entity, $\left\{M_k^{d_j-1} | M_k^{d_j-1} \in \partial\left(M_j^{d_j}\right)\right\}$, is reduced to zero.

Meshes in Figure 21 illustrate the detection and prevention of dimensional reduction by *use check*. Deletion of mesh edge M_1^1 in Figure 21(a) is disallowed because it leads to the mesh in Figure 21(b) where the topological use count of mesh edge M_2^1 is reduced to zero. Similarly, deletion of mesh edge M_1^1 in Figure 21(c) is disallowed because it leads to the mesh in Figure 21(d) where the topological use count of mesh face M_1^2 is reduced to zero

3.1.3.3 Applying Local Mesh Modifications and Recording *Multiple Classification* and Geometric Information *Small* mesh edges are deleted using an edge collapse operator. In many situations, the application of the edge collapse operator may first require one or more mesh entities to be swapped to ensure geometric validity (positive volume/area entities) of the final mesh. One such situation is depicted in Figure 22. *Small* mesh edge M_1^1 can be deleted by first swapping mesh edge M_2^1 followed by collapsing vertex M_1^0 to M_2^0. Alternatively, M_1^1 can be deleted by first swapping mesh edge M_3^1 followed by collapsing vertex M_2^0 to M_1^0. These

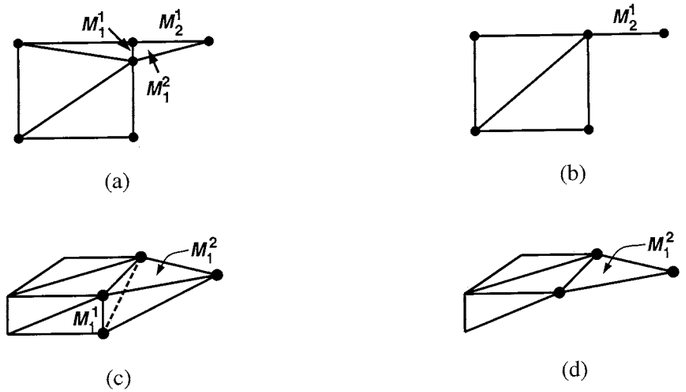


Figure 21. Unacceptable edge deletions detected by counting *topological uses*.

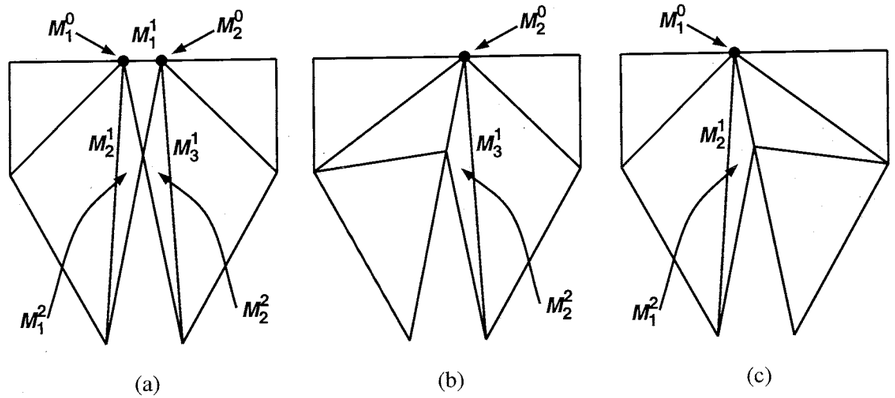


Figure 22. Swap-collapse example: (a) *small* edge M_1^1 cannot be deleted; deleted after swapping (b) M_1^1 or (c) M_2^1.

generic tools are based on the SCOREC mesh database [13] and a set of mesh adaptation and optimization procedures [21, 23, 78]. For more detailed information about properties of local mesh modification tools and constraints they must satisfy refer to [21, 23].

3.1.4 Results

The procedures described here have been implemented as part of a general mesh optimization procedure to improve mesh quality based on user prescribed parameters within the automatic mesh generator described in reference [80]. The detection and the elimination of the adverse influence of small geometric model features is based on two prescribed parameters:

1. The minimum size of the geometric model features to be retained, α, prescribed as a fraction of the largest dimension of the bounding box for the entire geometric model, L.
2. The minimum value for aspect ratio metric $Q^1_{M_i^2}$, in the resulting mesh: β.

The algorithm eliminates mesh edges, M_i^1, that are smaller than the prescribed minimum size

$$l < \alpha L \tag{18}$$

and connected to a mesh face, M_j^2, with aspect ratio metric less than the prescribed limit, β,

$$\exists M_j^2 | M_i^1 \in \partial(M_j^2), Q^1_{M_j^2} < \beta. \tag{19}$$

The results presented highlight the efficacy of the presented procedures in eliminating the adverse influence of small geometric features on the mesh shape quality without the need to resort to refinement in the vicinity of the small geometric model features. Comparison of mesh quality is done based on the global worst value of $Q^0_{M_i^3}$, $Q^1_{M_i^3}$ and $Q^3_{M_i^3}$ quality metrics. Histograms depicting the distribution of mesh regions with respect to the given quality metrics are also presented. Timing statistics are provided to determine the additional overhead of including the elimination of the adversely affected mesh entities in the existing mesh generation process. N_r is the total number of mesh regions. T_s and T represent the time spent in removing the adverse influence of the small geometric features and the total meshing time respectively. All the small features in the example models were created either as a design feature or as a result of "mending" operations by geometric modelling system to ensure closure of the model geometry. None of these features are artificially introduced.

3.1.4.1 Example 1 The geometric model consists of a simple block with very small radius blends on three of the edges and a vertex as shown in Figure 23. The meshes obtained for this model without and with the elimination of the influence of the small geometric model features is shown in Figure 24(a) and Figure 24(c), respectively. Comparison of the worst quality metric for both the meshes is given in Table 3. Histograms of the element quality distribution of the two meshes is shown in Figure 25. Mesh in Figure 24(a) has the worst $Q^0_{M_i^3}$, $Q^1_{M_i^3}$ and $Q^2_{M_i^3}$ values of 0.04, 0.02, 4.1e-4 (1.34 degree) respectively. Mesh in Figure 24(c) has the worst $Q^0_{M_i^3}$, $Q^1_{M_i^3}$ and $Q^2_{M_i^3}$ values of 0.51, 0.34, 0.13 (24.4 degree) respectively. Explicit identification and elimination of *small* mesh edges results in an order of magnitude improvement in the worst element quality with about 30% decrease in the total number of mesh regions. Since the effect of the small geometric model features was not localized, the histograms of element quality metric also show a significant shift towards the ideal value of each metric.

3.1.4.2 Example 2 The next example consists of the industrial part example from Figure 26(a). The geometric model containing the small feature was provided by one of the industrial users of the mesh generator [80]. Figure 26(b) shows the final mesh obtained with the adverse influence of the small feature eliminated. Figure 26(b) and Figure 26(c) illustrate the meshes

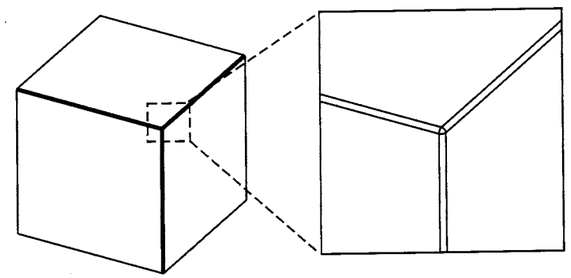
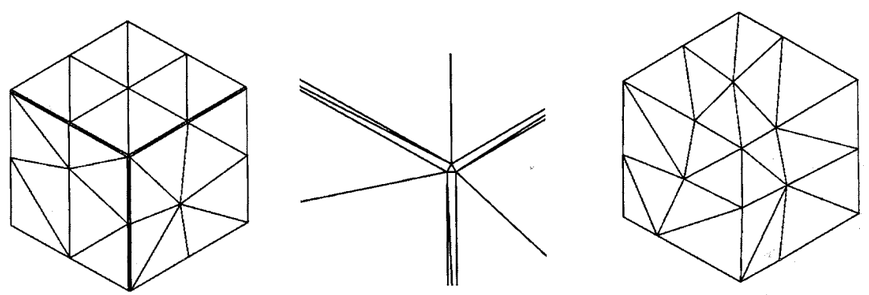


Figure 23. Geometric model with small manufacturing features.



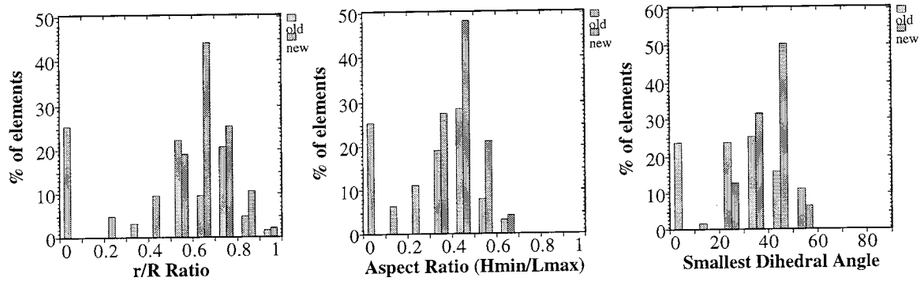
(a) (b) (c)

Figure 24. Example 1 images: (a) initial mesh, (b) zoom-in of initial mesh and (c) final mesh.

	Mesh -1	Mesh -2	% change
$Q^0_{M_i^3}$	0.04	0.51	+1175
$Q^1_{M_i^3}$	0.02	0.34	+1600
$Q^2_{M_i^3}$	4.1e-4 (1.3 degree)	0.13 (24.4 degree)	+31600
N_r	64	48	-25
T_s	-	0.06	-
T(seconds)	0.88	0.99	+12.5

Table 3. Mesh statistics for example 1. Mesh-1 and Mesh-2 refer to meshes obtained without and with the application of the current procedure.

in the vicinity of the small model edge obtained without and with the application of the presented procedures, respectively. The comparison of the mesh statistics is presented in Table



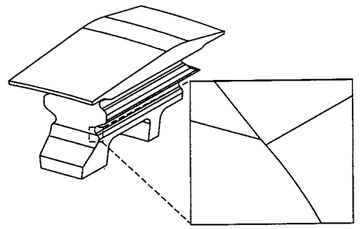
(a) (b) (c)

Figure 25. Histogram of element quality metric for example 1, "old" and "new" refer to meshes with and without the application of current procedures, respectively. Aspect ratio metrics are normalized.

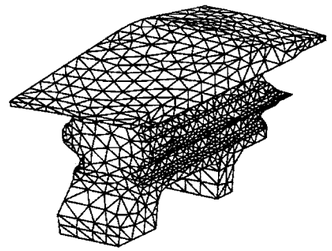
4. Application of the current procedures results in orders of magnitude improvement in the worst $Q^0_{M_i^3}$, $Q^1_{M_i^3}$ and $Q^2_{M_i^3}$ metric from 0.01, 0.006 and 4.6e-5 (0.45 degree) in Figure 26(a) to 0.10, 0.07 and 0.02 (8.32 degree) in Figure 26(b), respectively. The large improvement is due to the fact that the mesh entities with the worst metric were a direct consequence of the small model feature and downstream mesh quality optimization and "smoothing" procedures were unable to eliminate them due to topological constraints. Due to the extreme localization of the effects of the small model edge, the changes in the element distribution with respect to the quality metric is marginal.

3.1.4.3 Example 3 The geometric model for the third example is shown in Figure 27(a). The geometric model containing the small feature was provided by one of the industrial users of the mesh generator [80]. The small geometric model feature in this case is a sliver model face $\frac{1}{5000}$th the size of the largest model dimension. The severity of the disparity in the sizes of the mesh entities is highlighted by the fact that one has to zoom in extremely close to be able to see the adverse influence of the small model face on the mesh (Figure 27(c)). The presented procedures are able to remove all of the extremely small mesh edges and faces (Figure 27(d)) improving the worst $Q^0_{M_i^3}$, $Q^1_{M_i^3}$ and $Q^2_{M_i^3}$ metric from 0.07, 0.04 and 3.1e-3 (3.66 degree) to 0.11, 0.10 and 7.3e-3 (5.65 degree), respectively. The corresponding mesh statistics are shown in Table 5.

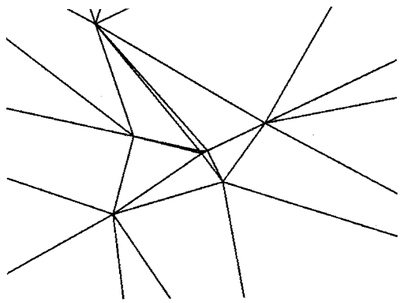
3.1.4.4 Example 4 Figure 28(a) depicts the geometric model for the gating arrangements for simulation of the casting process of a mechanical component. The geometric model containing numerous small features due to "mending" operations by the geometric modeler, was provided by one of the industrial users of the mesh generator [80]. The final mesh obtained after the elimination of the influence of all the small model features is shown in Figure 28(b). The geometric model has numerous small geometric features as shown in Figure 28(c, d). Mesh in the vicinity of the small geometric model features, without and with the elimination of influence



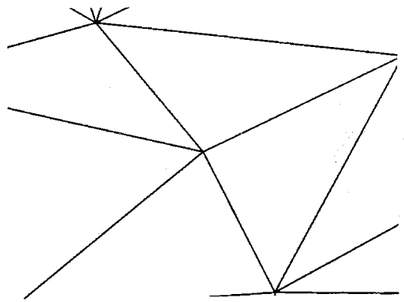
(a)



(b)



(c)



(d)

Figure 26. Example 2 images: (a) geometric model, (b) final mesh, (c) and (d) mesh without and with the elimination of adverse influence, respectively.

of the small features, is depicted in Figures 29 and 30 and the corresponding mesh statistics are shown in Table 6. Application of the current procedures has eliminated all the sliver mesh entities that are created due to the small geometric features resulting in an order of magnitude improvement in the worst $Q^0_{M_i^3}$, $Q^1_{M_i^3}$ and $Q^2_{M_i^3}$ metric from 0.01, 0.007 and 3.7e-5 (0.4 degree) to 0.08, 0.10 and 5.3e-3 (4.8 degree), respectively. In addition, the overall meshing time is reduced by 7.5% because direct elimination of the sliver mesh entities results in less work for downstream mesh optimization and "smoothing" procedures in trying to obtain an acceptable quality mesh.

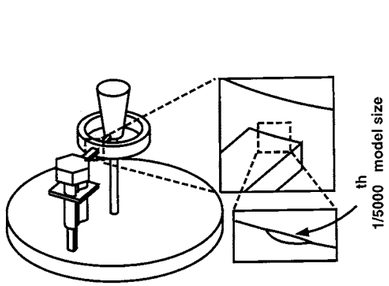
	Mesh -1	Mesh - 2	% change
$Q^0_{M_i^3}$	0.01	0.10	+900
$Q^1_{M_i^3}$	0.006	0.07	+1067
$Q^2_{M_i^3}$	4.6e-5 (0.45 degree)	0.02 (8.32 degree)	+43300
N_r	10817	10806	-0.1
T_s	-	0.58	-
T(seconds)	309.12	299.51	-3

Table 4. Mesh statistics for example 2. Mesh-1 and Mesh-2 refer to meshes obtained without and with the application of the current procedure.

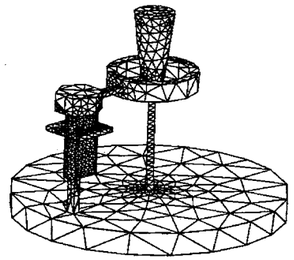
3.2 Local Mesh Modifications to Generate Curvilinear Finite Elements

The most straightforward approach [78] to generate a curvilinear mesh consists of first generating a mesh with straight edges and planar faces. Using classification information, each of the appropriate mesh entities can then be assigned appropriate curvilinear geometry based on a variety of techniques including *isoparametric* interpolation, blending, and curve/surface fitting. The details of construction of the curvilinear mesh geometry, discussed is Section 6, is not central at this point. *A-posteriori* assignment of curvilinear geometry to mesh entities classified on curved model boundaries can lead to two potential problems. First, requirements of a valid geometric triangulation can be violated if curved mesh entities intersect with neighboring mesh entities. The second problem can arise from excessive distortion in the shape of the elements resulting in numerical stiffening due to large Jacobian variations within the element domain. In either case the elements are deemed unacceptable and corrective actions must be carried out. Assuming that leaving the edges of the problem finite elements straight is not acceptable, corrective measures which modify the finite element mesh are required. The key to determining the type of corrective action required is the determination of the cause of the unacceptability of the element(s).

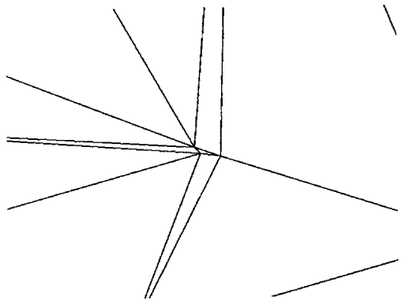
The effect of curving mesh edges is often not limited to one element since it is typically shared by other elements in the neighborhood. Thus an unacceptable element cannot always be corrected without looking into the neighboring mesh entities which are affected due to the curving. As one or more of the edges and/or faces of an element is curved, it can cause two situations leading to the element becoming unacceptable. First, unacceptably curved elements created as a result of other mesh entities coming in proximity of the curved entities of the element are depicted in Figures 31(a) through 31(d). Second, unacceptably curved elements can arise when entities are too far from their linear approximation as depicted in Figures 31(e) and 31(f). For the 2D case depicted in figure 31a curving of edge M_1^1 makes element M_1^2 unacceptable. The problem is not with the curved edge M_1^1, but with the fact that it intersects the edges M_2^1 and M_3^1 at points other than at the common vertices. In other words, the curved entities of element M_1^2 penetrate into the neighboring elements making it unacceptable. In 3D, bounding mesh edges, as well as faces, of an element will curve. If as a result of these curvings the edges or faces intersect other mesh entities then the element will become unacceptable. Figure 31(b) shows a case where



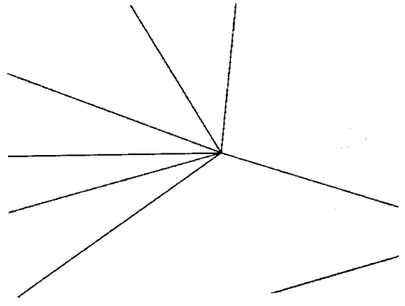
(a)



(b)



(c)



(d)

Figure 27. Example 3 images: (a) geometric model with sliver face, (b) final modified mesh, (c) and (d) meshes without and with the elimination of adverse influence of small feature respectively.

a curved face M_1^2 intersects edge M_1^1 at points other than their common boundary implying that the curved face M_1^2 will penetrate one or more neighboring elements which share edge M_1^1.

In the cases where mesh entities intersect or are in close proximity to the curved finite element entities the goal of any mesh modification operation is to properly modify or eliminate the entity that intersects or is too close to the curved entity. This point is clear in the case where the curved finite element entities exactly match the curved geometry of the model. Unless the entity that is intersecting or is too close to the curved entity is modified or eliminated, they will continue to be too close and always lead to unacceptable elements.

The steps involved with the development of a procedure for producing acceptable curved

	Mesh - 1	Mesh - 2	% change
$Q^0_{M^3_i}$	0.07	0.11	+57
$Q^1_{M^3_i}$	0.04	0.10	+150
$Q^2_{M^3_i}$	3.1e-3 (3.66 degree)	7.3e-3 (5.65 degree)	+136
N_r	7499	7499	0
T_s	-	0.45	-
T(seconds)	314.4	312.72	-0.6

Table 5. Mesh statistics for example 3. Mesh-1 and Mesh-2 refer to meshes obtained without and with the application of the current procedure.

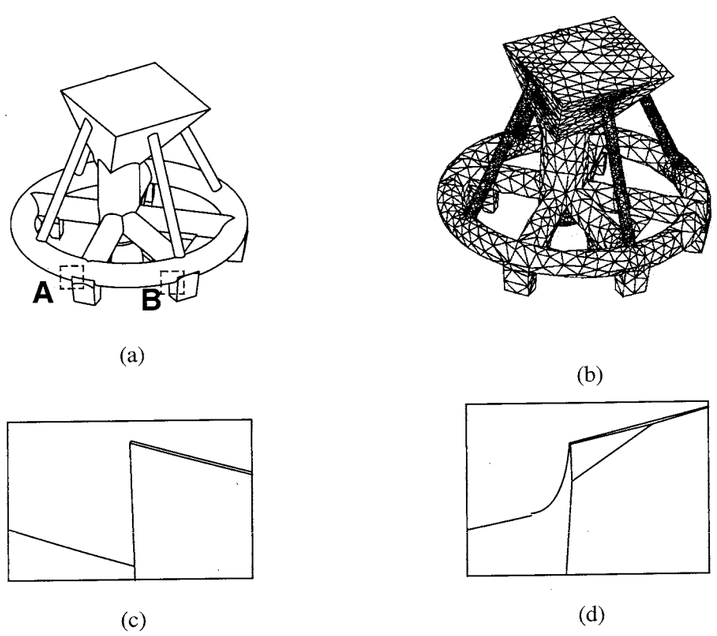
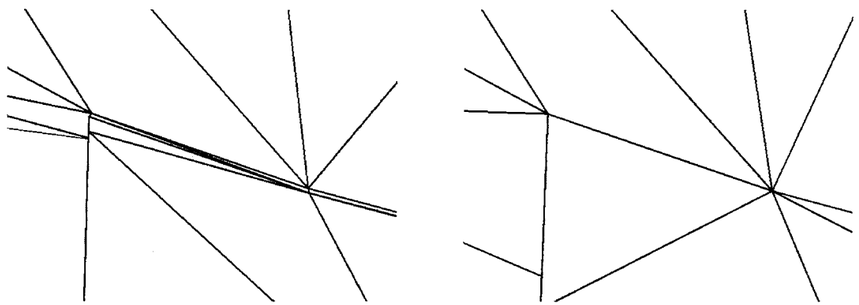


Figure 28. Gating arrangement for casting simulation: (a) geometric model, (b) final mesh, (c) zoom-in of A and (d) zoom-in of B.

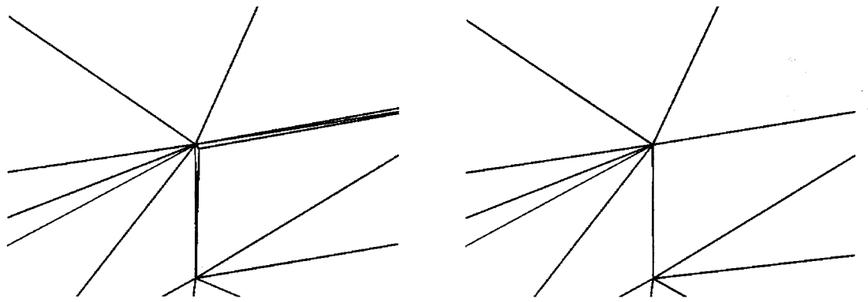
elements consists of the following:

1. Identification of the unacceptable elements.
2. Identification of the mesh entities that must be modified or eliminated.



(a) (b)

Figure 29. Zoom-in of example 4 mesh around A: (a) without elimination of adverse influence (b) with elimination of adverse influence.



(a) (b)

Figure 30. Zoom-in of example 4 mesh around B: (a) without elimination of adverse influence (b) with elimination of adverse influence.

3. Determination and execution of corrective actions necessary to modify or eliminate the problem mesh entities through local mesh modifications.

The next subsection discusses the metrics used to determine unacceptable elements and the mesh entities causing the elements to be unacceptable. The following subsection presents an incremental approach for eliminating the problem entities through a hierarchy of local mesh modifications.

	Mesh - 1	Mesh - 1	% change
$Q^0_{M^3_1}$	0.01	0.08	+700
$Q^1_{M^3_1}$	0.007	0.10	+1328
$Q^2_{M^3_1}$	3.7e-5 (0.4 degree)	5.3e-3 (4.8 degree)	+14200
N_r	15095	15066	-0.2
T_s	-	1.2	-
T(seconds)	905	837	-7.5

Table 6. Mesh statistics for example 4. Mesh-1 and Mesh-2 refer to meshes obtained without and with the application of the current procedure.

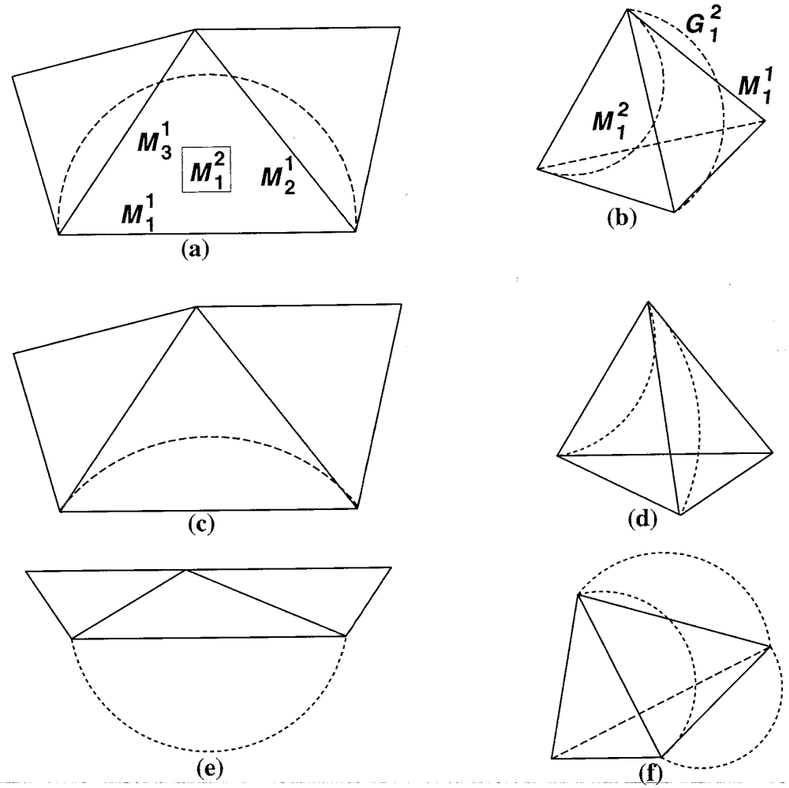


Figure 31. Unacceptable curvings in 2D and 3D: (a) and (b) penetration into neighboring entities, (c) and (d) have entities "too close" to neighboring entities, and in (e) and (f) entities curve too far from their linear representation.

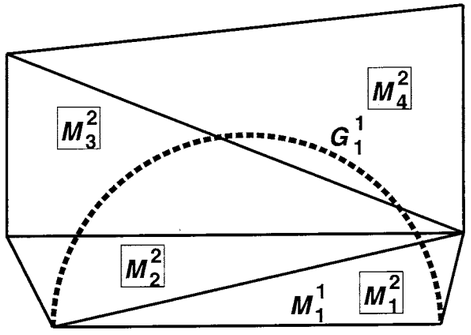


Figure 32. Interior elements affected due to curving of boundary entity

The last two subsections provide some specifics of the implementation of the mesh modification procedures and present the results obtained with those procedures implemented to date.

3.2.1 Determination of Unacceptable Elements and the Mesh Entities to be Modified or Eliminated

As indicated previously, elements become unacceptable when curved either because they self intersect, causing the triangulation of the mesh to violate the definition of a geometric triangulation, or the element shape becomes so poor that it will lead to numerical stiffening due to large variations in the determinant of the Jacobian. The two causes of unacceptable elements immediately lead to consideration of two metrics to determine unacceptable elements, intersection calculations and variations in the determinant of the Jacobian.

The use of intersection calculations has two advantages. The first is that determination of the intersection directly indicates which mesh entities of the current element must be modified or eliminated. The second advantage is its ability to also determine problem mesh entities of other elements in the neighborhood. To see this, consider the 2D case shown in figure 32 where the curved edge of element M_1^2 intersects mesh entities in elements M_1^2 through M_4^2. Clearly the mesh modifications will need to propagate into the mesh far enough to modify or eliminate all the intersected mesh edges. There are, however, two disadvantages to intersection calculations. The most critical is that intersections only identify the invalid element situations, they do not identify the elements that are valid, but so poorly shaped as to be unacceptable. The second disadvantage is the computational expense of doing intersections in three dimensions. It is possible to supplement the intersection test with a closest distance check to determine the situations where the mesh entities come too close to the curved entity. The addition of such calculations greatly increases the computational cost past the already computationally expensive intersection calculations.

The second metric of evaluating the variation and minimum value of the determinant of the Jacobian can identify both invalid and poorly shaped elements. Since it considers only the influence of the mesh entities of the element itself, this procedure will not identify additional problems with neighboring elements in the way a set of intersection checks can. Again consider Figure 32 where entities belonging to elements M_2^2 through M_4^2 intersect the curved edge of element M_1^2. Since all the edges of elements M_2^2 through M_4^2 are straight, and the elements have positive area, the Jacobian is constant through the element and positive. Other potential drawbacks of the determinant of the Jacobian variation evaluation is the ability to identify the problem entities in the element and the computational effort required to determine the location of the maximum and minimum determinants of the Jacobian. As discussed in the subsections that follow, the use of an incremental approach focused on elimination of the problem mesh entities is capable of incrementally determining all the entities in the neighborhood of the curved mesh entities for eventual elimination. The use of only a limited number of pointwise determinant of the Jacobian evaluations can greatly reduce the computational cost, but does introduce some level of approximation into the process. This approximation is typically acceptable since the common method used by analysis codes to determine unacceptable elements is to examine the positiveness of, and maximum difference between, the determinants of the Jacobian evaluated at the numerical integration points used in the calculation of the stiffness matrix [65, 41]. In addition, careful examination of pointwise determinant of the Jacobian evaluations should be able to identify the problem mesh entities which must be modified or eliminated.

A curved finite element is valid with respect to self intersection if the determinant of the Jacobian of the element mapping is positive at all points within the element. If $\mathbf{x}(\xi)$ defines the geometric mapping of the element where ξ represents the natural coordinates of the element then the Jacobian [44, 96] of the mapping is given by

$$\mathbf{J}(\xi) = \left| \frac{\partial \mathbf{x}}{\partial \xi} \right| \tag{20}$$

and a valid element has

$$\mathbf{J}(\xi) > 0 \quad \forall \quad \xi \tag{21}$$

Numerical stiffening due to large variations of the Jacobian is related to the numerical integration of the stiffness matrix over the element. Consider the most basic integral of evaluating the volume of a finite element. The exact volume, V_{exact}, of the element is given by

$$V_{exact} = \int_{\Omega^e} \mathbf{J}(\xi) \mathrm{d}\xi \tag{22}$$

When numerical integration is used as an approximate volume, V, is calculated as

$$V_{exact} \simeq V = \sum_{i=1}^{N_{int}} \mathbf{J}\left(\xi^i\right) w^i \tag{23}$$

where N_{int} is the number of integration points, ξ^i and w^i are the integration point coordinates and the corresponding weights respectively. Based on this, a measure of element shape distortion can be given by [65]

$$I = \frac{N_{int} \min_i \left(\mathbf{J}(\xi^i) w^i\right)}{V} \tag{24}$$

An element is considered acceptable when $\min_i \left(\mathbf{J}(\xi^i) w^i\right) > 0$ and $I > I_{min} > 0$, where I_{min} is the distortion limit prescribed based on the specific requirements of the analysis method.

Since other expressions can equally be used as a metric of the element acceptability, care is taken in the implementation to allow the flexibility of changing the distortion metric.

3.2.2 Incremental Approach to Eliminate Problem Entities

Given a list of unacceptable finite elements, and an indication for each such element which mesh entities must be modified or eliminated to potentially produce acceptable elements, a variety of approaches to the development of the mesh improvement procedures are possible. Since the number of unacceptable elements is typically small, it is appropriate to focus attention on local mesh modifications to produce the acceptable elements. This has the advantage that the individual components of the procedure are based on well qualified operations.

Because of the number and complexity of the local mesh modifications possible and required to make the elements in the neighborhood of an unacceptable element acceptable, an incremental approach is used which focuses on the specific mesh entities of an unacceptable element that must be modified or eliminated. The basic justification for this is that, under the assumption that an acceptable set of elements can be created through mesh modifications in the local neighborhood, any final mesh modification can be obtained through a series of basic mesh modification operations. It is also assumed that it is possible to determine the final modifications needed by the incremental application of a set of operations.

The classes of local mesh modification operators that can be used in the process of producing a mesh of acceptable elements includes:

1. Altering the shape of mesh edges and mesh faces classified interior to the domain.
2. Repositioning of mesh vertices without altering their classification.
3. Performing local retriangulation, where retriangulation is defined as an operation where the mesh topology is modified without changing the number of mesh vertices and without changing their position.
4. Deletion of elements with reclassification of interior mesh entities to the boundary. This operation includes the repositioning of reclassified mesh vertices onto the boundary entity they are reclassified onto. As with the previous operations, this one does not change the number of mesh vertices.
5. Performing local remeshing, where remeshing is defined as an operation that adds or deletes one or more mesh vertices.

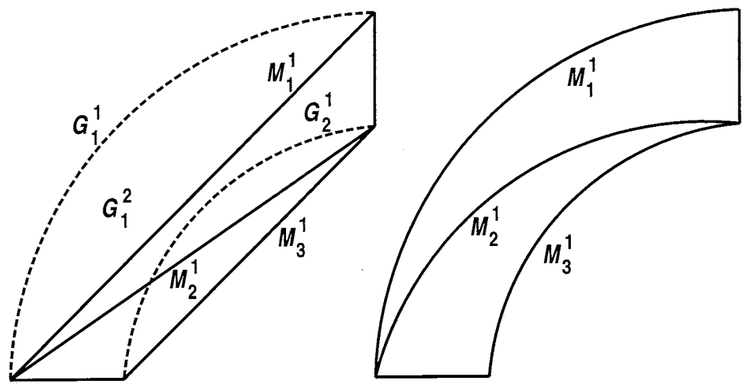


Figure 33. Curving interior mesh entities to prevent intersections.

Although the altering of the shape of mesh edges and faces classified interior to the domain can often yield acceptably shaped elements with little effort, it is considered the least desirable operation since it increases the number of curved finite elements which require expensive higher order numerical integration rules to ensure the convergence rate. However, if local refinement of the mesh is not allowed, then assigning curved geometries to interior mesh entities is necessary to ensure validity of the curvilinear mesh. Consider for example the two-dimensional example in Figure 33. Curving mesh edge $M_3^1 \sqsubset G_2^1$ leads to an invalid mesh due to intersection with interior mesh edge $M_2^1 \sqsubset G_1^2$. The situation can be corrected by curving mesh edge M_2^1 such that it does not intersect M_1^1 and M_3^1 when they are curved based on G_1^1 and G_2^1, respectively.

Mesh vertex reposition is a simple operation which is useful in some cases. However, it is of limited applicability as is easily seen in the example in Figure 32 where any reasonable repositioning of mesh vertices M_3^0 and M_4^0 under the constraint that the other mesh vertices are constrained will not yield acceptable elements.

Under the right conditions the application of a retriangulation operation can effectively yield an acceptable mesh. Consider the example shown on the left side of figure 34 where the element M_1^2 becomes unacceptable when M_1^1 is curved to the model edge G_1^1 since it intersects mesh edge M_2^1. However, if the same set of mesh vertices is retriangulated as shown on the right of figure 34, element M_1^2 is fully acceptable when M_1^1 is curved to the model edge G_1^1. Retriangulation has the basic advantage of closely maintaining the original mesh gradation since it used the original set of mesh vertices in their original positions. Retriangulation can not, however, yield acceptable elements in all situations. Again consider Figure 32 where no possible retriangulation will yield an acceptable set of elements.

Deletion of elements with reclassification of mesh entities is a useful local mesh modification operation for eliminating unacceptable elements. Figure 35 shows a simple 3D situation where all the mesh vertices and all but one mesh edge of the element M_1^3 are classified on G_1^2. The local modification in this case consists of deleting M_1^3, M_1^1, and the two mesh faces that M_1^1 bounds followed by reclassifying the mesh edge M_2^1 and the two mesh faces of the original

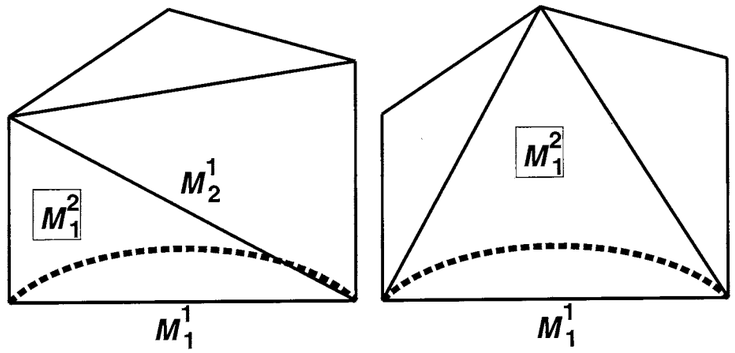


Figure 34. Example of creation of acceptable elements through retriangulation

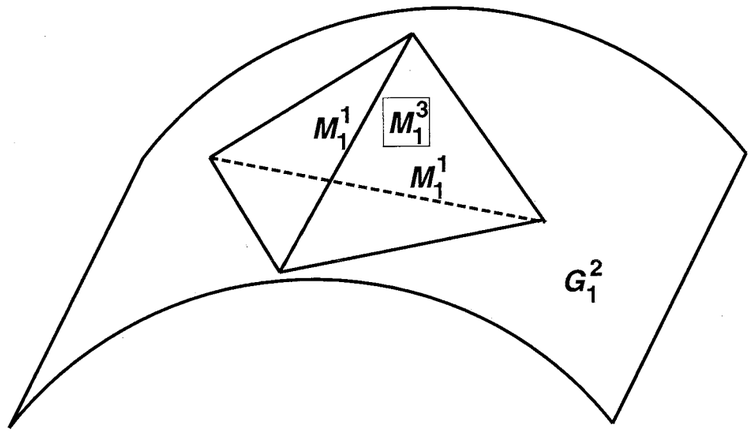


Figure 35. Deletion of a 3D element with reclassification of mesh entities

element M_1^3 that it bounds from the model region to the model face G_1^2. The most useful version of the element deletion operation is one which also allows the reclassification of mesh vertices from a model region to one of its boundary entities. Such reclassification of mesh vertices is only valid if the coordinates of the mesh vertex are modified so it is positioned on the model boundary entity upon which it is classified. Figure 36 shows a 2D example of this operation in which the element M_1^2 is deleted. In this case the mesh vertex M_1^0, and mesh edges M_1^1 and M_1^1 are reclassified on the model edge G_1^1. In the process the mesh vertex M_1^0 was repositioned to lie on the model edge G_1^1.

The ability of remeshing procedures to add and delete mesh vertices makes it a flexible tool in the process of obtaining an acceptably shaped curvilinear mesh at the cost of at least some variation in the local mesh gradation. Figure 37 shows two possible retriangulations starting from the mesh containing unacceptable elements shown in figure 32. The retriangulation on the left maintained all the original mesh vertices and created three new ones on the model edge,

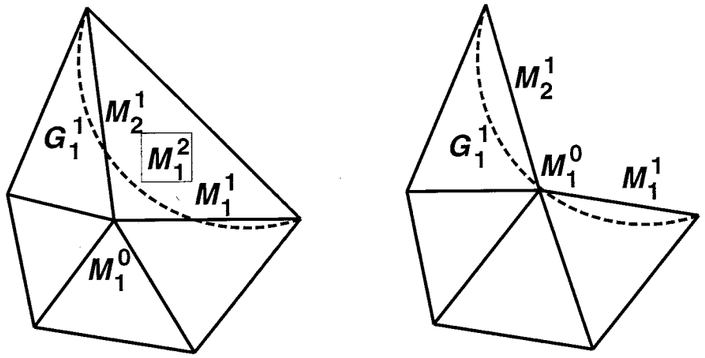


Figure 36. Deletion of a 2D element with reclassification of mesh entities and repositioning of a mesh vertex

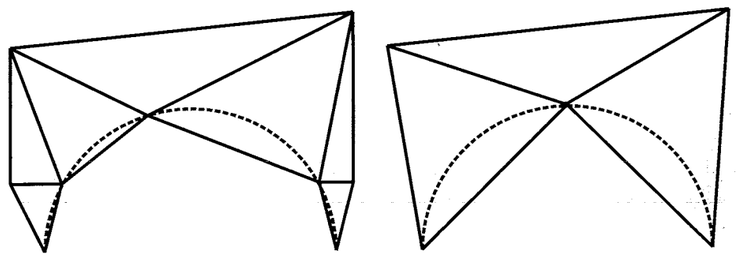


Figure 37. Two retriangulations of the mesh with unacceptable elements shown in figure 32

while the retriangulation on the right eliminated two interior mesh vertices and added one on the model edge.

The current implementation of the mesh modification algorithm uses retriangulation tools consisting of edge swaps and multi-face swaps [78] and the entity deletion and reclassification tools. A more general procedure should include local remeshing as described in Figure 37, mesh vertex motion, and the curving of mesh entities classified interior to the domain. Owing to the large number of combinations of local mesh modifications possible, a heuristic approach is adopted in the determination of the sequence of local mesh modifications to be applied to correct a specific situation.

Given a valid mesh with straight edges and planar faces along with a link to the geometric model on which the mesh is classified, the algorithm begins by creating a list of unacceptable shaped elements and an indication of the mesh entities which must be modified or eliminated. The steps in the incremental procedure are:

1. Select the next unacceptable element from the list. Terminate when the list is empty.

2. Considering the mesh entities that must be modified or eliminated, evaluate the best, if any, retriangulation operation which either yields an acceptable set of elements, or improves the situation by reducing the number of mesh entities to be modified or eliminated. If the best retriangulation operation yields an acceptable set of elements, it is performed, the list of unacceptable elements updated, and control returned to Step 1.
3. Considering the mesh entities that must be modified or eliminated, evaluate the best, if any, deletion operation which either yields an acceptable set of elements, or improves the situation by reducing the number of mesh entities to be modified or eliminated. If the best deletion, or combined deletion and retriangulation operation, that yields an acceptable set of elements is performed, the list of unacceptable elements updated, and control returned to Step 1. If there is a deletion, or combined deletion and retriangulation operation, that reduces the number of problem entities, perform the operation, update the list of unacceptable elements, and continue.

One reason retriangulation and deletion are attempted before more general mesh modifications is that they generally maintain the mesh gradation closer to that of the original mesh because they do not change the number of mesh vertices. An open issue however is the selection and application of appropriate retriangulation or deletion operations when they appear to improve the situation, but must still be followed by remeshing operations to yield acceptable elements locally. Since the remeshing operations alter the triangulation as well as introduce new mesh entities, it is not critical to exhaustively evaluate all the retriangulation and deletion procedures that may lead only to an improved situation before proceeding to attempting a remeshing. The number of options evaluated is driven more by the desire to create acceptable elements without the need to remesh local portions of the mesh. On the other hand, it is critical when a remeshing operation is required to determine the appropriate sequence of operation(s). Experience indicates that the constraints on the type of retriangulation and remeshing operations allowed, and the basic properties of the various specific retriangulation and remeshing operations, usually make it clear which mesh modifications should be performed. However, the lack of a proof of a convergent sequence of local operations for all situations indicates additional effort is required to ensure the overall reliability of the procedure.

The operations required when the curved mesh entities are too far from their linear approximation (figure 31e and f) are fairly obvious. In these cases remeshing operations which add mesh vertices classified on the geometric entities of concern are needed. In those cases where the curved mesh entities intersect or come too close to other mesh entities (figure 31a through d) it is not obvious which mesh modification is the most appropriate. Retriangulation and deletion operations are considered first followed by consideration of curving interior mesh entities and remeshing with addition and/or deletion of mesh vertices.

Unlike the 2D case where the basic retriangulation operation is a diagonal swap, there are a large number of retriangulation operations in 3D. Two specific options that have been qualified into useful tools are edge swapping [21, 33, 23] and multi-face removal [21]. Edge swapping considers a mesh edge that is to be eliminated from the triangulation. Deleting all the elements that are bounded by that edge creates a polyhedral domain that can be remeshed by creating

edges between selected mesh vertices and defining the appropriate mesh faces and mesh regions. The number of possible retriangulations grows as the square of the number of mesh vertices on the bounding polygon [33, 23], therefore, an effective implementation must quickly determine a limited number of possibilities to be evaluated [21, 33, 23]. Multi-face removal operator is the reverse of the edge swap operator.

Retriangulation options are evaluated based on the entities which must be modified or eliminated. If the undesirable entity is a mesh edge bounding the unacceptable element, the edge swap configurations based on that edge are considered. In addition, if the unacceptable edge is classified interior to the model then multi-face swaps which eliminate that edge are also considered. If the undesirable entity is an interior face then a multi-face swap is considered along with edge swap based on the bounding edges of the undesirable face. If a retriangulation exists that results in all affected elements being acceptable, it is applied. If no retriangulation could be found that makes all the elements acceptable in the resulting mesh, then the deletion options are considered. If a deletion option is found that results in a mesh that yields no unacceptable elements it is applied. If this does not yield all acceptable elements, the best possible retriangulation and deletion options are compared based on the number of undesirable mesh entities in the affected mesh and the one which results in fewer undesirable mesh entities is selected to be applied and the process repeated with the next unacceptable element.

If unacceptable elements remain after the application of retriangulation and deletion, remeshing procedures are then applied. Again, there are a wide variety of possible tools available for this process. In the current implementation the primary remeshing tool for creating additional mesh vertices is spliting of mesh entities, while the primary tool for reducing the number of mesh vertices is collapsing. The focus of the application of these two operations are mesh edges [21] since their splitting and collapsing operations typically produce the best results.

The edge collapse operation is typically applied to edges of unacceptable elements which have one mesh vertex classified on the boundary and one classified interior to the domain The collapse of such an edge typically eliminates the problem mesh entities by collasping them onto mesh entities classified on the boundary. Figure 38 demonstrates this process for a simple 2D example. The original mesh (figure 38a) is modified by collapsing M_4^1 which pulls M_2^0 onto M_1^0 eliminating the mesh entities M_2^0, M_4^1, M_5^1 and M_2^2. This collapsing process yields the acceptable element M_1^2 shown in figure 38b.

When the curved mesh entities are too far from their linear approximation edge spliting is applied. In edge spliting a new mesh vertex is introduced along the mesh edge the appropriate mesh entities connecting to it and the entities it bounds are created. Since the new mesh vertex inherits the boundary classification of the mesh edge, it is positioned at an appropriate location on the model boundary entity. Edge splitting plus retriangulation can also be used to eliminate the unacceptable elements which intersect the curved boundary shown in figure 32. The actual implementation in this case is best performed by:

1. Deleting the unacceptable elements creating a empty polygon of mesh entities.
2. Splitting the mesh edges classified on the model boundary the appropriate number of times.

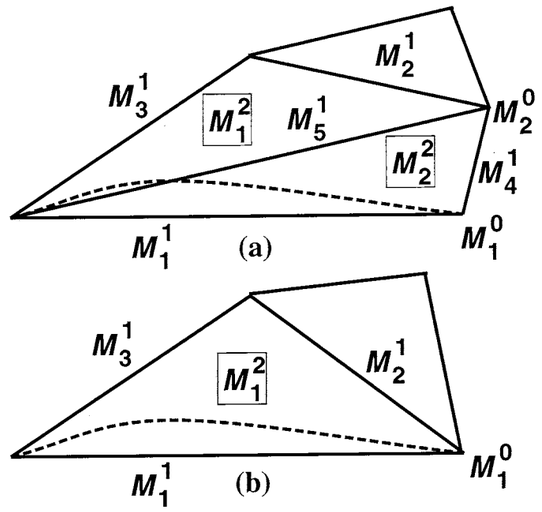


Figure 38. Example of a 2D edge collapse operation to eliminate an unacceptable element

3. Using the new mesh edges retriangulate (creation of surface triangulations) the appropriate portions of model faces, yielding an updated empty polygon.
4. Fill the updated empty polygon by tetrahedral element triangulation. This step may or may not introduce new mesh vertices into the polygon.

3.2.3 Examples of Local Mesh Modifications to Produce Curved Meshes of Acceptable Elements

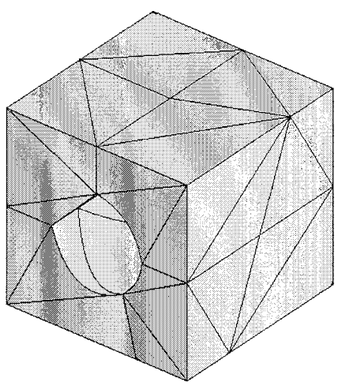
Two images are given for each of the examples discussed in this subsection. The first shows the original mesh in which the element edges for elements that would become unacceptable if curved are displayed as straight edges. The second image shows the mesh after performing the appropriate mesh modifications where all the mesh edges classified on curved boundary entities are curved.

The first curved mesh example is depicted in figure 39a and 39b. In the original mesh the mesh edge on the circular model edge that was left straight left straight (figure 39a) would cause an invalid element if curved. A simple retriangulation (an edge swap in this case) resulted in an acceptable curved mesh (figure 39b).

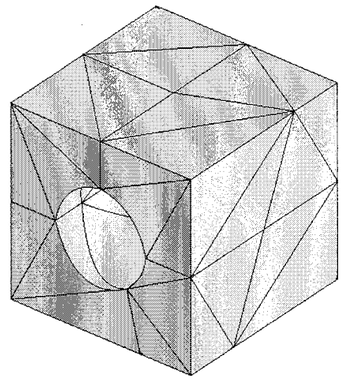
The second curved mesh example also had an invalid element as its worst case where one of the curved edges classified on a model face was intersecting a bounding mesh face classified in the interior of the model (figure 39c). Two local retriangulations (edge swaps of the bounding edges of the undesirable face) lead to an acceptable curved mesh (figure 39d).

The original mesh for the third example also had invalid elements resulting from the curving of some of the edges. The top mesh is figure 40 shows the mesh edges left straight in the original mesh. The corrective steps required to get to the acceptable curved mesh (bottom figure)

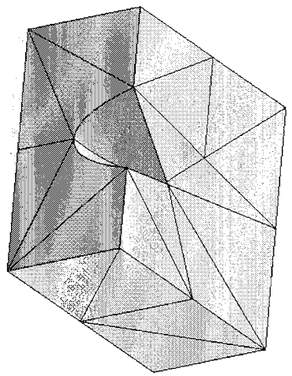
consisted of local retriangulation, element deletion and edge collapsing. The edge collapsing procedures lead to some coarsening of the final mesh.



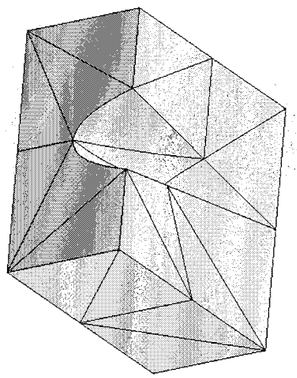
(a)



(b)



(c)



(d)

Figure 39. Mesh examples 1 and 2

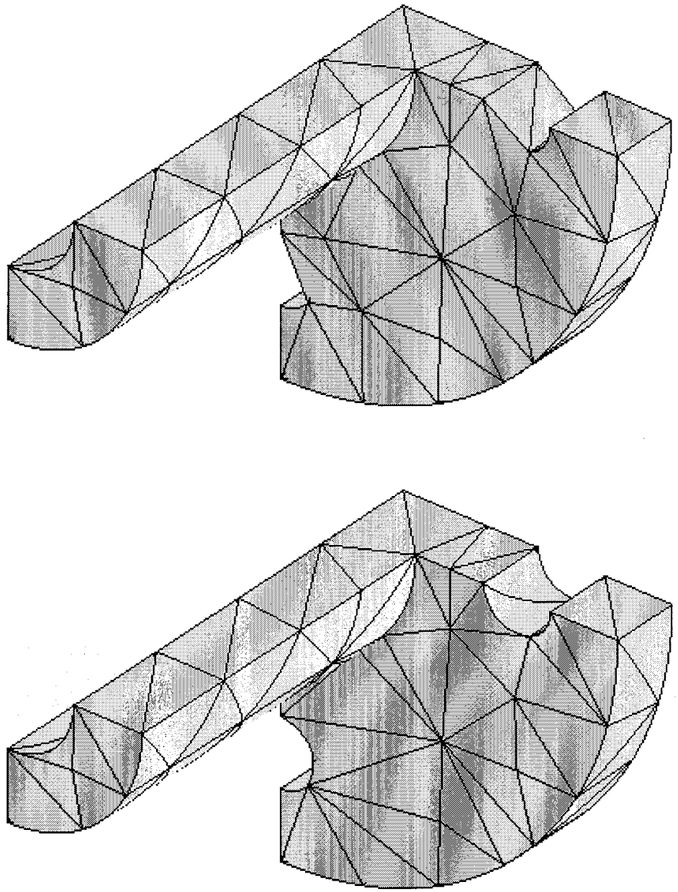


Figure 40. Curved mesh example 3

4. Topology Based Specification of Variable Order Finite Element Approximations

Variable order p-hierarchic shape function construction must consider that topological entities in $\overline{\Omega}_e$, can have independent polynomial order associated with them. Recalling equation (4), consider a shape function N associated with mesh entity $M_j^{d_j} \in \overline{\Omega}_e$ in the form

$$N = \psi\left(M_j^{d_j}, M_e^{d_e}\right)\phi\left(M_j^{d_j}\right) \tag{25}$$

where

- $\psi\left(M_j^{d_j}, M_e^{d_e}\right)$ is a blending function defined on $M_e^{d_e}$ specific to $M_j^{d_j}$, written in the parametric coordinate system ξ_i of $M_e^{d_e}$ and independent of the polynomial order of N.
- $\phi\left(M_j^{d_j}\right)$ is a function written in the parametric coordinate system $\hat{\xi}_j$ of $M_j^{d_j}$ that depends on the polynomial order of N, and is independent of $M_e^{d_e}$. Thus, it is the same for all elements (including those of different topologies and/or dimension) connected to $M_j^{d_j}$.

Figure 41 depicts one possibility for this decomposition for a cubic shape function on an edge between a triangular and quadrilateral face. In this example, the cubic edge shape function on the triangular face is defined as $N(M_1^1, M_1^2) = \psi(M_1^1, M_1^2)\phi(M_1^1)$ and on the quadrilateral face as $N(M_1^1, M_2^2) = \psi(M_1^1, M_2^2)\phi(M_1^1)$. Assuming that individual blending functions can be determined, this approach provides a convenient framework to construct shape functions for variable p-order approximations. It automatically satisfies the C^0 interelement continuity requirements. Most constructions write shape functions such that odd-order shape functions on mesh entities adjacent to two elements require a change of sign on one of the elements because of the opposite use of that entity [16, 84, 89]. With the decomposition (25), there is a single function on the mesh entity, $\phi\left(M_j^{d_j}\right)$, and $\psi\left(M_j^{d_j}, M_e^{d_e}\right)$ on each element ensures its correct behavior on the element. To be computationally efficient, the specific choice of blending functions in equation (25) must be such that the evaluation of the resulting decomposed shape function is competitive with respect to the shape functions written in explicit form.

The flexibility of a topologically-based p-order mesh specification allows consideration of a variety of approaches to construct the shape functions and to perform the operations required to calculate contributions to the element matrices. In any of them, a common feature is identification of the entities over which integrations involving shape functions must be performed to determine the coefficients of the stiffness matrix and load vector. For simplicity and consistency with standard finite element construction methods, these entities are referred to as elements and our focus is the definition of the proper set of shape functions over each element which:

1. Are of the polynomial order specified on each of the mesh entities included in $\overline{\Omega}_e$.
2. Satisfy the C^0 interelement continuity requirement.
3. Require a minimum of numerical operations in the evaluation of the shape functions and subsequent element integrations.

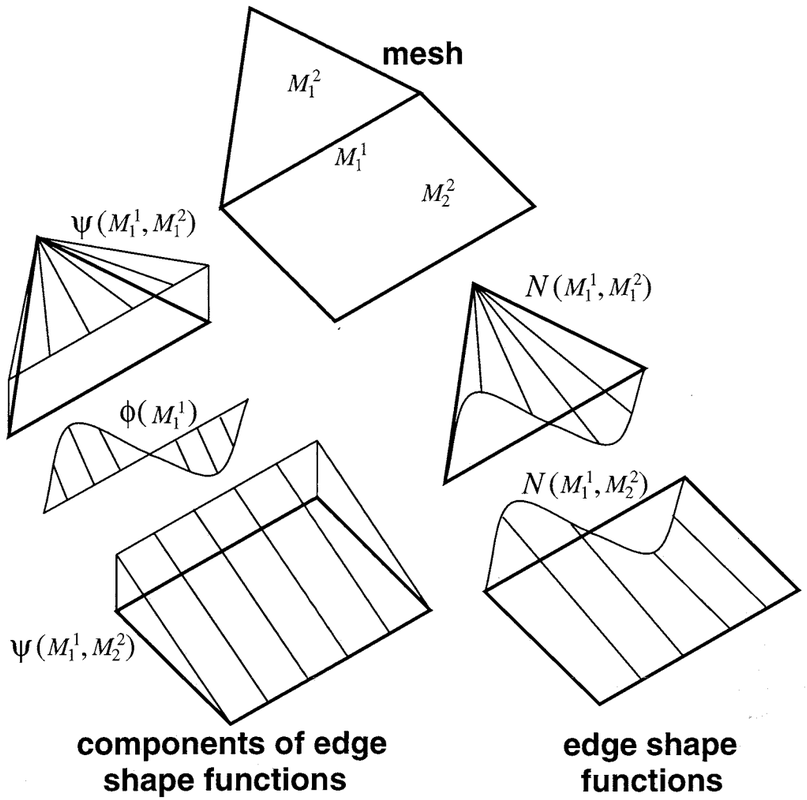


Figure 41. Construction of edge shape functions using a product of blending functions and an edge mode.

The first two requirements are easily met by the shape function decomposition based on topological entities. Although this decomposition is quite natural for some element topologies, specific care is required to ensure that it leads to efficient shape function evaluation over mixed meshes. The two most important issues here are the natural coordinate systems used over the various mesh entities and how the shape functions are decomposed.

Consideration of the available tools used to construct elements, and the importance of exercising specific options, makes it advantageous (but not necessary) to assume some specific limitations on the element topologies and polynomial orders specified on the mesh entities. In what follows a mesh face, M_i^2, is assumed to be either triangular (bounded by three M_j^1), or quadrilateral (bounded by four M_j^1). The motivation for this limitation is the complexity of defining parametric coordinates over faces with more edges. The topology of the region elements considered will be limited to tetrahedra, hexahedra, wedges and pyramids for the same reason. Although it is possible to assign shape functions of any order to any topological entity, it is not

desirable to associate higher-order polynomials with mesh vertices when only C^0 continuity is required. Therefore, the mesh vertices will be limited to have linear polynomial shape functions.

With these assumptions a mesh can have polynomials of any order associated with each mesh edge, face and region. For example, Figure 42 shows a variable p-order curvilinear hp-mesh in two dimensions.

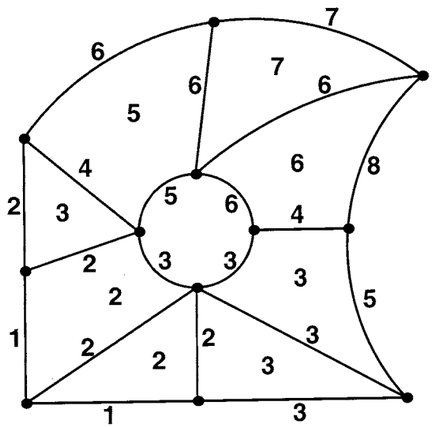


Figure 42. Example of a two-dimensional variable p-order mesh.

In addition, it is possible for the face and region entities to have different p-orders specified in each of the parametric directions of the face or region. Consideration of basic convergence rate expressions would indicate no obvious advantage of different polynomial orders in each direction. However, it is clear that in many problems the gradients are stronger in preferred directions. Therefore, computational efficiency can be improved by employing higher order polynomials in that direction. In general, to provide effective anisotropic p-refinement, the ability to control p-order in selected parametric directions for faces and regions is required in addition to the ability to vary the polynomial order on the bounding mesh entities. As an example, consider a rectangular domain modeled by one quadrilateral element. A linear combination of two cubic edge modes (Figure 43(a,b)) is sufficient to produce a cubic variation on the element in the direction parallel to these edges (Figure 43(c)). However, obtaining a mode which has cubic variation in one parametric direction and a quadratic variation in the other direction, and which is zero on the bounding edges, requires a face mode which has a cubic and quadratic variation in the desired directions (Figure 43(d)). Section 4.3 presents simple geometric constructs to interpret and control the directional behavior of face and region shape functions over the face and region topologies considered here.

4.1 Parametric Coordinate Systems

Efficient evaluation of mesh-entity-based shape functions depends on the parametric coordi-

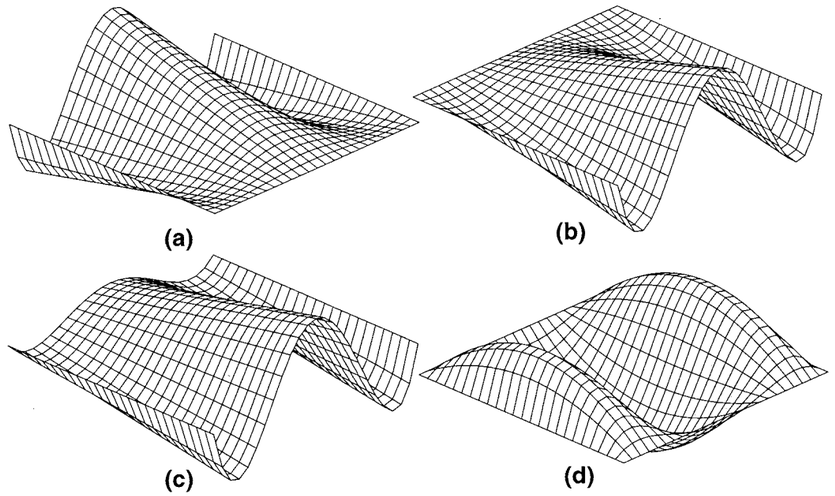


Figure 43. Directionally biased solution fields.

nates used on each of the mesh entities on $\overline{\Omega}_e$. Since a mesh entity may bound portions of different elements with different topologies, care must be taken to use the same parametric coordinates for the entity to ensure the ability to decompose the shape functions as described in Section 4.

The primary complications relate to the selection of the parametric coordinate systems for simplices (triangles and tetrahedra) and pyramids, and the matching of those coordinate systems with those of bounding lower order entities having different topologies. The choice of parametric coordinates for the mesh entities is based on the following two considerations:

1. Simplicity and efficiency of the blending functions, ψ.
2. Compatibility and efficiency of use with numerical integration scheme(s).

Parameterization of simplices by barycentric coordinates leads to symmetry of the shape functions with respect to the parametric coordinate components because bounding edges (faces) have equal length (area); simple polynomial forms for the blending functions; and compatibility with most existing integration schemes over triangles and tetrahedra which are defined in terms of area and volume coordinates, respectively [44, 45].

The parametric coordinates for various mesh entities used to obtain the shape function decomposition are shown in Figures 44 and 45. For reasons to be discussed, two edge parameterizations (*I* and *II*) are considered. The number in the circles indicates the identifiers of the bounding mesh vertices in the mesh database and are used in orienting the entity parametric coordinates. The domain of the parametric coordinate component(s) for each entity is listed in Table 7.

Since evaluation of shape functions (and their derivatives) at a point in the element parametric domain involves evaluating functions defined in terms of parametric coordinates of bounding mesh

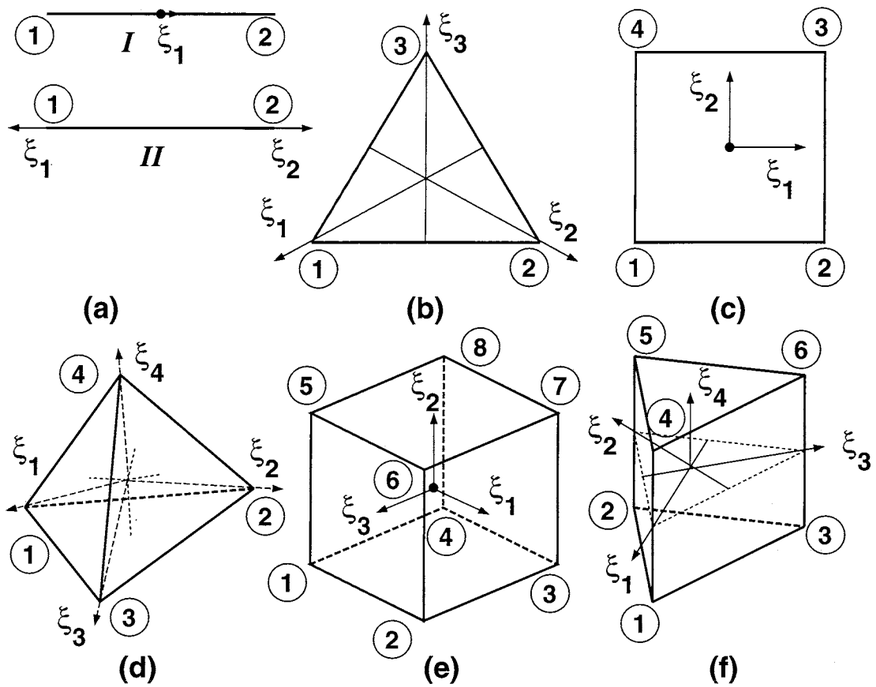


Figure 44. Entity parametric coordinates: (a) edge, (b) triangle, (c) quadrilateral, (d) tetrahedron, (e) hexahedron and (f) wedge form of a pentahedron.

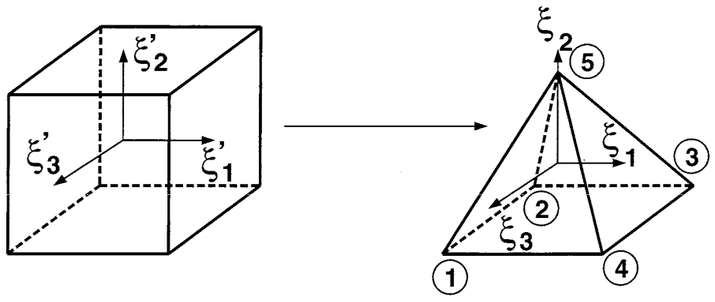


Figure 45. Parametric domain of a pyramid region obtained as a degeneration and scaling of the hexahedral parametric domain.

entities, a mapping transforming the element parametric coordinates, ξ_i, to the coordinates of the bounding lower order entities, $\hat{\xi}_j$, is required. The "^" is used to distinguish the parametric coordinates of the bounding lower order entities from the parametric coordinates of the element itself. When a mesh contains both simplex and non-simplex elements, the efficiency of mapping of element parameters, ξ_i, to the corresponding parameters on a bounding edge, $\hat{\xi}_j$, is dependent

Topology		Parametric Domain
Edge	*I*	$\xi_1 \in [-1,1]$
	II	$\xi_1, \xi_2 \in [0,1],\ \xi_1 + \xi_2 = 1$
Triangle		$\xi_1, \xi_2, \xi_3 \in [0,1],\ \xi_1 + \xi_2 + \xi_3 = 1$
Quadrilateral		$\xi_1, \xi_2 \in [-1,1]$
Tetrahedron		$\xi_1, \xi_2, \xi_3, \xi_4 \in [0,1],\ \xi_1 + \xi_2 + \xi_3 + \xi_4 = 1$
Hexahedron		$\xi_1, \xi_2, \xi_3 \in [-1,1]$
Pentahedron		$\xi_1, \xi_2, \xi_3 \in [0,1],\ \xi_1 + \xi_2 + \xi_3 = 1,\ \xi_4 \in [-1,1]$
Pyramid		$\xi_2 \in [-1,1],\ \frac{-(1-\xi_2)}{2} \leq \xi_1, \xi_3 \leq \frac{(1-\xi_2)}{2}$

Table 7. Entity parametric domains.

upon the choice of edge parameterization from Table 7.

Choosing edge parameterization *I* leads to a trivial mapping for quadrilateral and hexahedral elements, $\hat{\xi}_1 \equiv \xi_j$, based on the local edge index in the element topological hierarchy. However, for a simplex, the mapping for an edge between vertices with index i and j (Figures 44(b,d)) is nontrivial and is given by

$$\hat{\xi}_1 = \xi_j - \xi_i. \tag{26}$$

On the other hand, choice of edge parameterization *II* leads to trivial mapping for simplices given the local edge index, $\hat{\xi}_i \equiv \xi_j$. However, for a quadrilateral or a hexahedral element (Figure 46(a,b)) the mapping is now nontrivial and is given by

$$\begin{aligned} \hat{\xi}_1 &= \frac{1}{2}(1 - \xi_i) \\ \hat{\xi}_2 &= \frac{1}{2}(1 + \xi_i) \end{aligned} \tag{27}$$

for an edge directed along ξ_i parameter of the element. The important point here is that it is not possible to avoid a nontrivial mapping for both simplices and non-simplices when a mesh of mixed element topologies is used. However, for a mesh with all quadrilateral (hexahedral) elements, edge parameterization *I* is more efficient, while for a mesh with all simplex elements, edge parameterization *II* is more efficient.

For a quadrilateral face of a wedge pentahedron as shown in Figure 46(c), the mapping between the element parametric coordinates, ξ, to the face parametric coordinates, $\hat{\xi}$, is given by

$$\begin{aligned} \hat{\xi}_1 &= \xi_j - \xi_i \\ \hat{\xi}_2 &= \xi_4. \end{aligned} \tag{28}$$

A mesh region with the topology of a pyramid lacks a "natural" parameterization that fits its shape. It is parameterized as a degeneration of a hexahedron, with scaling as shown in Figure 45

using a procedure similar to that used in [83] to parameterize a tetrahedron. The mapping from the hexahedron parameters to the pyramid parameters is given by

$$\begin{aligned} \xi_1 &= \xi_1^{'}\left(1-\xi_2^{'}\right)/2 \\ \xi_2 &= \xi_2^{'} \\ \xi_3 &= \xi_3^{'}\left(1-\xi_2^{'}\right)/2 \end{aligned} \tag{29}$$

where $\xi_i^{'}$ define the hexadedral parametric space. The inverse mapping is given by

$$\begin{aligned} \xi_1^{'} &= \frac{2\xi_1}{(1-\xi_2)} \\ \xi_2^{'} &= \xi_2 \\ \xi_3^{'} &= \frac{2\xi_3}{(1-\xi_2)} \end{aligned} \tag{30}$$

The mapping is degenerate because one face in the parametric domain of the hexahedron maps to a point in the parametric domain of the pyramid. However, the singularity in the parametric domain does not imply that functions written in those coordinates are necessarily singular. The scaling of the parametric bounds for ξ_1 and ξ_3 by $\frac{(1-\xi_2)}{2}$ ensures that after the degeneration, the mapping between ξ_i and the Cartesian coordinates x_j remains one-to-one.

The mapping between the parametric coordinates of a pyramid and its bounding edges that also bound the quadrilateral face is given by $\hat{\xi}_1 \equiv \xi_i$ for edge directed along ξ_i with edge parameterization *I*. Using edge parameterization *II* leads to the mapping given by equation (27). For bounding edges that do not bound the quadrilateral face, the mapping is given by $\hat{\xi}_1 \equiv \xi_3$. The mapping for the parametric coordinates of a bounding triangular face is nontrivial. For example, for the face bounded by local vertices 1, 4 and 5 (Figure 45), the mapping is given by

$$\begin{aligned} \hat{\xi}_1 &= \frac{1}{4}(1-\xi_2-2\xi_1) \\ \hat{\xi}_2 &= \frac{1}{4}(1-\xi_2+2\xi_1) \\ \hat{\xi}_3 &= \frac{1}{2}(1+\xi_2) \end{aligned} \tag{31}$$

The mapping for the rest of the triangular faces can be obtained by permuting the parametric indices in equation (31) such that ξ_1 is replaced by the parameter along the edge of the triangle that is also shared by the quadrilateral face of the pyramid.

4.2 Construction of the Blending Functions ψ

Figure 41 depicted one form of blending function $\psi\left(M_j^{d_j}, M_e^{d_e}\right)$ in which the blending function is a constant on the mesh entity $M_j^{d_j}$ and blends to zero on the "opposite boundary" of the element $M_e^{d_e}$. Although this form is an obvious choice for edges of a quadrilateral and faces of hexahedron, its construction on edges of triangular faces leads to unnecessary numerical

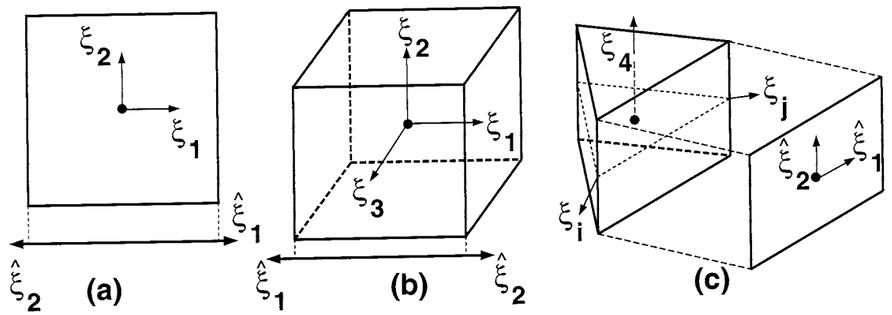


Figure 46. Parametric transformations with edge parameterization *II*.

operations since the edge blending function $\psi\left(M_j^1, M_e^{d_e}\right)$ in those cases must be written as a rational function.

Since the minimum polynomial order of any edge, face or region function is at least quadratic, it is not necessary that the decomposition of the shape functions be such that the blending function is a constant over the topological entity for which it blends. Considering again the simple case of an edge shape function for a two-dimensional element, it is possible to decompose each shape function into a product of a quadratic blending function and edge based functions which increase in polynomial order as the polynomial order of the element increases. In this case, the first edge function, $\phi(M_1^1)$, would be a constant, yielding the desired quadratic shape function, the second would be a linear, yielding the desired cubic shape function, etc. Figure 47 reconsiders the construction of cubic edge shape functions in the case where the blending function is quadratic. In this case a product of a linear edge function with the quadratic blends produces the desired cubic functions. The importance of employing the most effective blending functions can be demonstrated by the calculation of the shape function for the two cubic edge mode decompositions of Figures 41 and 47. The number of operations required for the two forms of decomposition for the quadrilateral element is the same. This is not the situation for the triangular element. In the case of producing a constant value over the edge, the blending term is a rational function, $\psi(M_1^1, M_1^2) = \frac{4\xi_i\xi_j}{1-(\xi_j-\xi_i)^2}$, and the edge function is based on the one-dimensional cubic shape function defined over the edge [89], $\phi(M_1^1) = -\frac{\sqrt{10}}{4}\left[1-(\xi_j-\xi_i)^2\right](\xi_j-\xi_i)$. The total cost[5] of evaluating the shape function in this case is 2 additions and 7 multiplications. On the other hand, the quadratic blending function is a simple polynomial, $\psi(M_1^1, M_1^2) = -2\xi_i\xi_j$, and the edge function, $\phi(M_1^1) = (\xi_j - \xi_i)$, is a linear polynomial giving a total cost of shape function evaluation as 1 addition and 3 multiplications which is the most efficient way to evaluate the cubic edge shape function $N(M_1^1, M_1^2) = -2\xi_i\xi_j(\xi_j - \xi_i)$. Clearly the decomposition based on the quadratic edge blend is more efficient. The rational blending term adds even more overhead in computing the derivatives of the edge shape functions required for stiffness computation.

---

5 The cost is in terms of counting total number of addition(s) and multiplication(s) required. A division is deemed equivalent to a multiplication and a subtraction equivalent to an addition.

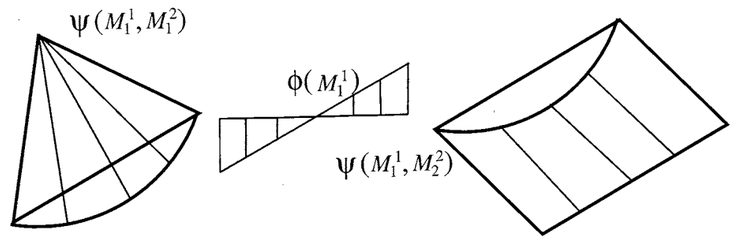


Figure 47. Alternative decomposition of a cubic edge shape function from Figure 41 using quadratic blending functions.

The vertex shape functions typically used, and used here, always contribute only to the constant and the linear terms [16, 89, 90]. Since the vertex blending function, $\psi\left(M_j^0, M_i^{d_e}\right)$, $M_j^0 \in \overline{M_i^{d_e}}$, must at least be a linear function, it is the shape function and hence there is no need to perform an explicit product decomposition for vertex modes. Also, if the highest dimension of elements is three, then mesh entities M_i^3 are not shared by multiple element topologies; hence, there is no need to decompose the shape functions contributed by mesh regions.

The blending functions, $\psi\left(M_i^{d_e}, M_j^{d_j}\right)$, and the entity functions, $\phi\left(M_j^{d_j}\right)$, must be constructed such that the resulting shape functions, $N\left(M_j^{d_j}, M_i^{d_e}\right) = \psi\phi$, vanish over all lower order bounding entities except $M_j^{d_j}$ [16]. One possibility is to enforce this endpoint property on the blending functions, in which case, the blending functions, $\psi\left(M_j^{d_j}, M_i^{d_e}\right)$, $M_j^{d_j} \in \overline{M_i^{d_e}}$, satisfy:

1. $\psi\left(M_j^{d_j}, M_i^{d_e}\right)$ is nonzero in $M_i^{d_e}$

$$\psi\left(M_j^{d_j}, M_i^{d_e}\right) \neq 0, \quad \forall \xi \in M_i^{d_e} \tag{32}$$

2. $\psi\left(M_j^{d_j}, M_i^{d_e}\right)$ is zero on $\partial\left(M_i^{d_e}\right)$ except on $M_j^{d_j}$

$$\psi\left(M_j^{d_j}, M_i^{d_e}\right) = 0, \quad \forall \xi \in \left[\partial\left(M_i^{d_e}\right) - M_j^{d_j}\right] \tag{33}$$

The higher order edge and face blending functions to follow lead to the most efficient evaluation of decomposed shape functions and satisfy these requirements. The expressions for blending functions are based on the topology of the element and the boundary mesh entity, and the parametric coordinate system of the element. The normalization coefficients used here are based on the orthogonalization described in [16] to improve numerical conditioning of resulting element level matrices.

1. Line Element:

   a. Edge blend: Use of parameterization *I* leads to the following expression for the blending function

$$\psi\left(M_i^1, M_i^1\right) = \frac{\left(\xi_1^2 - 1\right)}{2} \tag{34}$$

whereas use of parameterization *II* results in an equivalent blend given by

$$\psi(M_i^1, M_i^1) = -2\xi_1\xi_2. \tag{35}$$

2. Triangle Element:

   a. Edge blend:

$$\psi(M_i^1, M_j^2) = -2\xi_k\xi_l \tag{36}$$

   where edge M_i^1 is directed from the *k*-th to the *l*-th vertex of M_j^2. For example, for the edge between vertices 1 and 2 in Figure 44(b), *k*=1 and *l*=2.

   b. Face blend:

$$\psi(M_i^2, M_i^2) = \xi_1\xi_2\xi_3. \tag{37}$$

3. Quadrilateral Element:

   a. Edge blend:

$$\psi(M_j^1, M_i^2) = \frac{1}{4}[\xi_k^2 - 1](1 \pm \xi_l) \tag{38}$$

   for edges along ξ_k with $k = 1, 2;\ l = 2, 1$. For example, for the edge between vertices 1 and 2 in Figure 44(c), *k*=1 and *l*=2.

   b. Face blend:

$$\psi(M_i^2, M_i^2) = \frac{1}{4}[\xi_1^2 - 1]\,[\xi_2^2 - 1]. \tag{39}$$

4. Tetrahedral Element:

   a. Edge blend:

$$\psi(M_i^1, M_j^3) = -2\xi_k\xi_l \tag{40}$$

   where edge M_i^1 is directed from the *k*-th to the *l*-th vertex of M_j^3. For example, for the edge between vertices 1 and 2 in Figure 44(d), *k*=1 and *l*=2.

   b. Face blend:

$$\psi(M_i^2, M_j^3) = \xi_k\xi_l\xi_m \tag{41}$$

   where face M_i^2 is bounded by vertices of M_j^3 with local index *k*, *l* and *m*. For example, for the face bounded by vertices 1, 2 and 3 in Figure 44(d), *k*=1, *l*=2 and *m*=3.

5. Hexahedral Element:

   a. Edge blend:

$$\psi(M_i^1, M_j^3) = \frac{1}{8}(\xi_k^2 - 1)(1 \pm \xi_l)(1 \pm \xi_m) \tag{42}$$

   for edges along ξ_k with $k = 1,2,3$; $l = 2,3,1$; $m = 3,1,2$. For example, for the edge between vertices 1 and 2 in Figure 44(e), k=1, l=2 and m=3.

   b. Face blend:

$$\psi(M_i^2, M_j^3) = \frac{1}{8}(\xi_l^2 - 1)(\xi_m^2 - 1)(1 \pm \xi_k) \tag{43}$$

   for faces perpendicular to ξ_k with $k = 1,2,3$; $l = 2,3,1$; $m = 3,1,2$. For example, for the face bounded by vertices 1, 2, 3 and 4 in Figure 44(e), k=2, l=1 and m=3.

6. Wedge Element:

   a. Edge blend:

$$\psi(M_i^1, M_j^3) = -\xi_k \xi_l (1 \pm \xi_4) \tag{44}$$

   for edges lying in the plane perpendicular to the ξ_4 axis and defined between vertices at which $\xi_k = 1$ and $\xi_l = 1$, respectively. For example, for the edge between vertices 1 and 2 in Figure 44(f), k=1 and l=2. For edges parallel to the ξ_4 axis

$$\psi(M_i^1, M_j^3) = \frac{1}{2}\xi_k(\xi_4^2 - 1). \tag{45}$$

   For example, for the edge between vertices 1 and 4 in Figure 44(f), k=1.

   b. Face blend:

$$\psi(M_i^2, M_j^3) = \frac{1}{2}\xi_1 \xi_2 \xi_3 (1 \pm \xi_4) \tag{46}$$

   for the two triangular faces in Figure 44(f).

$$\psi(M_i^2, M_j^3) = -\xi_k \xi_l (\xi_4^2 - 1) \tag{47}$$

   for quadrilateral faces bounded by pairs of vertices at which $\xi_k = 1$ and $\xi_l = 1$. For example, for the face bounded by vertices 1, 3, 6, and 4, k=1 and l=3.

7. Pyramid element:

   a. Edge blend: For edges along ξ_k the blend is

$$\psi(M_i^1, M_j^3) = \frac{1}{8}\left(\left(\frac{2\xi_k}{1 - \xi_2}\right)^2 - 1\right)\left(1 \pm \frac{2\xi_l}{1 - \xi_2}\right)(1 - \xi_2) \tag{48}$$

   with k=1,3 and l=3,1. For example, for the edge between vertices 1 and 4 in Figure 45, k=1 and l=3. For edges along ξ_2 the blend is

$$\psi(M_i^1, M_j^3) = \frac{1}{8}(\xi_2^2 - 1)\left(1 \pm \frac{2\xi_k}{1 - \xi_2}\right)\left(1 \pm \frac{2\xi_l}{1 - \xi_2}\right) \tag{49}$$

   with k=1,3 and l=3,1. These are the edges bounded by vertex 5 in Figure 45.

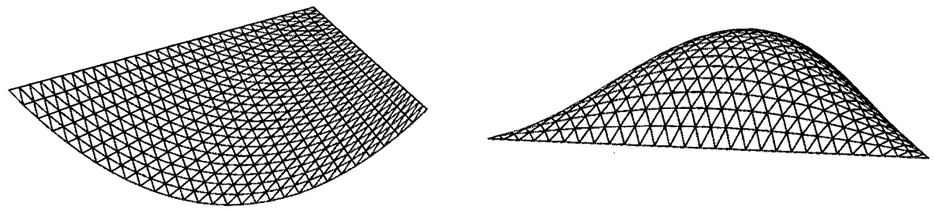


Figure 48. Triangular edge and face blending functions.

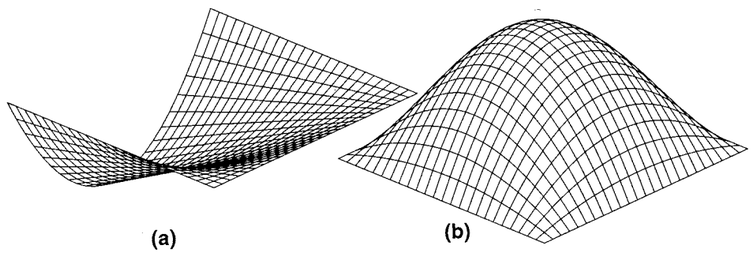


Figure 49. Quadrilateral edge and face blending functions.

b. Face blend:

$$\psi(M_i^2, M_j^3) = \frac{1}{8}\left(1 - \left(\frac{2\xi_1}{1-\xi_2}\right)^2\right)\left(1 - \left(\frac{2\xi_3}{1-\xi_2}\right)^2\right)(1-\xi_2) \tag{50}$$

for the quadrilateral face in Figure 45 and

$$\psi(M_i^2, M_j^3) = \frac{1}{8}\left(1 \pm \frac{2\xi_k}{(1-\xi_2)}\right)\left(1 - \left(\frac{2\xi_l}{1-\xi_2}\right)^2\right)(1-\xi_2^2) \tag{51}$$

for the triangular faces with k=1,3 and l=3,1.

The edge and face blends for triangular and quadrilateral elements are shown in Figures 48 and 49.

4.3 Construction of the Mesh Entity Level Functions ϕ

The polynomial order of ϕ for an entity is given by $(p-q)$ where p is the polynomial order of the shape function and q is the polynomial order of the blending function used for that mesh entity. For example, $\phi(M_i^1)$ are of the order $(p-2)$ since $\psi\left(M_i^1, M_j^{d_e}\right)$ is quadratic. The definition of ϕ can be based on any hierarchical polynomial basis. However, practical choices are governed by the numerical conditioning of the resulting element level matrices [5, 16, 89, 90].

4.3.1 Mesh Edge

There is only one edge shape function of a given polynomial order p. The exact expression for the edge functions depends on the choice of the basis and the parameterization used for mesh

p	ϕ	
	Equation (52)	Equation (53)
2	$\sqrt{\frac{3}{2}}$	1
3	$\sqrt{\frac{5}{2}}(\xi_2 - \xi_1)$	$\xi_2 - \xi_1$
4	$\sqrt{\frac{7}{2}}\left(\frac{5(\xi_2-\xi_2)^2-1}{4}\right)$	$\xi_1^2 - \xi_1\xi_2 + \xi_2^2$
5	$\sqrt{\frac{9}{2}}\left[\left(\frac{\xi_2-\xi_1}{4}\right)\left(7(\xi_2 - \xi_1)^2 - 3\right)\right]$	$\xi_2^2 - 6\xi_1\xi_2^2 + 6\xi_1^2\xi_2 - \xi_1^2$

Table 8. Edge functions.

edges. One possibility is to use the basis defined by the integration of the *Legendre* polynomials [1], $P_n(\zeta)$, $\zeta \in [-1, 1]$, as follows

$$\left[\frac{\zeta^2 - 1}{2}\right]\phi(M_1^1) = \sqrt{\frac{2p-1}{2}}\int_{-1}^{\zeta} P_{p-1}(t)dt,\ p \geq 2 \tag{52}$$

where, $\zeta = \xi_1$ and $\zeta = \xi_2 - \xi_1$ for edge parameterization *I* and *II*, respectively. The term $\frac{\zeta^2-1}{2}$ is the edge blend function and can always be factored from the right side of equation (52).

However, when simplices are used, then the following edge functions with parameterization *II* yields a better conditioning of element level matrices [16]

$$\phi(M_1^1) = \sum_{k=0}^{p-2}(-1)^k \frac{1}{k+1}\binom{p-2}{k}\binom{p-1}{k}\xi_1^k\xi_2^{p-2-k},\ p \geq 2. \tag{53}$$

The expressions for $\phi_{p-2}(M_1^1)$ using edge parameterization *II* for *p=2,3,4,5* based on equations (52) and (53) are given in Table 8. Graphs of these edge functions are given in Figure 50. The shape functions for a triangular and a quadrilateral element obtained from equation (53) are shown in Figures 51 and 52, respectively.

4.3.2 Triangular Mesh Face

There are a total of $(p-2)$ independent face functions for a triangular face that contribute to shape functions of order *p*, *p≥3*. For shape functions based on *Legendre* polynomials [89], one possible set of triangular face functions is given by

$$\phi(M_i^2) = P_{\alpha_1-1}(\xi_2 - \xi_1)P_{\alpha_2-1}(2\xi_3 - 1) \tag{54}$$

with

$$\begin{aligned} &\alpha_1, \alpha_2 = 1, .., p-2 \\ &\alpha_1 + \alpha_2 + \alpha_3 = p. \end{aligned} \tag{55}$$

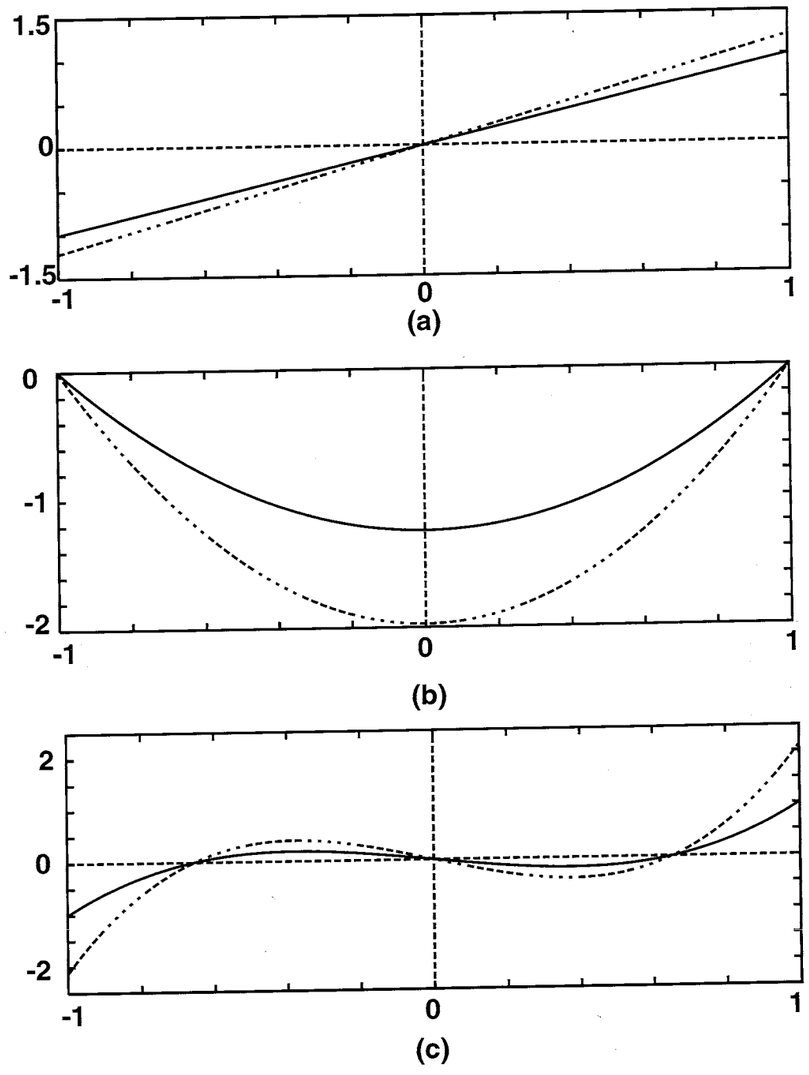


Figure 50. Edge functions $\phi\left(M_1^1\right)$ for (a) *p=3*, (b) *p=4* and (c) *p=5*. Solid lines correspond to equation (52) and dashed lines to equation (53).

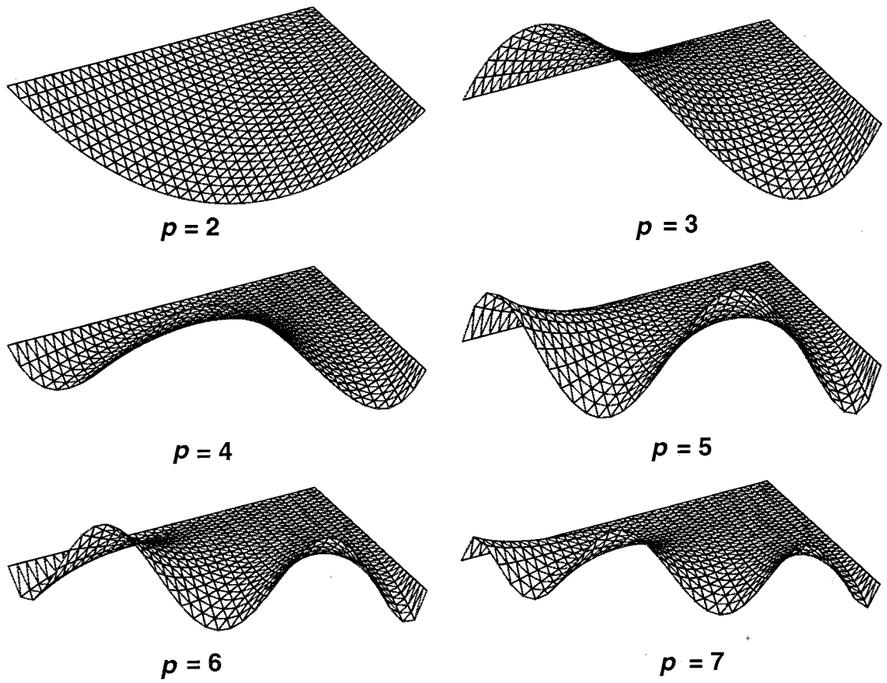


Figure 51. Triangular edge shape functions. The shape function for p=2 is scaled by 0.5 and those for p=5,6,7 are scaled by 2.0 for clarity.

An alternative form which yields better conditioned element matrices [16] is given by

$$\phi(M_1^2) = \sum_{i=0}^{\alpha_2-1}\sum_{j=0}^{\alpha_1-1}\left(\frac{-1}{2}\right)^{i+j} i!j!(i+j)!\binom{\alpha_1-1}{j}\binom{\alpha_1}{j}\binom{\alpha_2-1}{i}\binom{\alpha_2}{i} \times \frac{1}{\prod_{k=1}^{i+j}[k(\alpha_1+\alpha_2)-k(k-1)/2]}(\xi_1)^{\alpha_1-1-j}(\xi_2)^{\alpha_2-1-i} \tag{56}$$

where $\alpha_i,\ i = 1,2,3$, satisfy equation (55).

The symmetry and directional bias of the triangular face modes with respect to the barycentric coordinates can be explained by associating the triplet, $(\alpha_1, \alpha_2, \alpha_3)$, with each face function where α_i defines the highest power of ξ_i in the resulting face shape function. The sum of the triplet indices equals the polynomial degree of the shape function. Since only two of the barycentric coordinate components are independent, completeness of the basis can be ensured by varying two of the indices independently while holding the third one fixed. For example, for $\alpha_i = 1,\ i = 1,2,3$, the sets of triplets that define the face function combinations based on the three possible choices of the pair of independent indices can be generated using the geometric

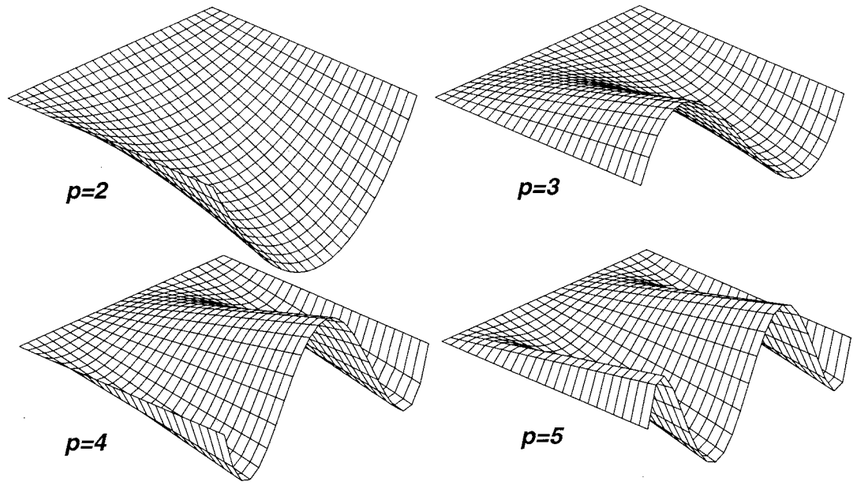


Figure 52. Quadrilateral edge shape functions.

construct shown in Figure 53. Triplets in each of the triangles ABO, ACO and BCO define the set of linearly independent face functions needed to complete the basis of a given order. Combinations in triangle ABO, BCO and ACO correspond to choosing (ξ_1, ξ_2), (ξ_2, ξ_3) and (ξ_1, ξ_3), respectively, as the independent pair of parameter components.

Face functions generated by equation (56) correspond to the triplets in triangle ABO and are listed in Table 9. The shape functions resulting from the face functions from triangle ABO are plotted in Figure 54. Note that the missing symmetric counterpart of (2,1,1) and (1,2,1) for p=4, given by (1,1,2), can be obtained as a linear combination of (2,1,1) and (1,2,1) modes as follows

$$\left(\xi_3 - \frac{1}{3}\right) = \left(\frac{2}{3} - \xi_1 - \xi_2\right) = -\left[\left(\xi_1 - \frac{1}{3}\right) + \left(\xi_2 - \frac{1}{3}\right)\right] = -[(2,1,1) + (1,2,1)]. \quad (57)$$

In general, the set of face functions defined by any of the triangles ABO, ACO and BCO can be expressed as a linear combination of the set defined by any one of the other triangles. However, in those cases where the error analysis indicates the need for anisotropic p-enrichment by picking only the face shape function defined by (1,1,2), its construction by the linear combination of the (1,2,1) and (2,1,1) functions is not the most effective means to construct it; instead it can be directly obtained by choosing ξ_3 as one of the independent parameters. This is equivalent to switching from triangle ABO to triangle BCO or to triangle ACO.

4.3.3 Quadrilateral Mesh Face

For a quadrilateral face, there are a total of $(p-3)$ face functions that contribute to shape functions of polynomial order p. The face functions are obtained as tensor product polynomials

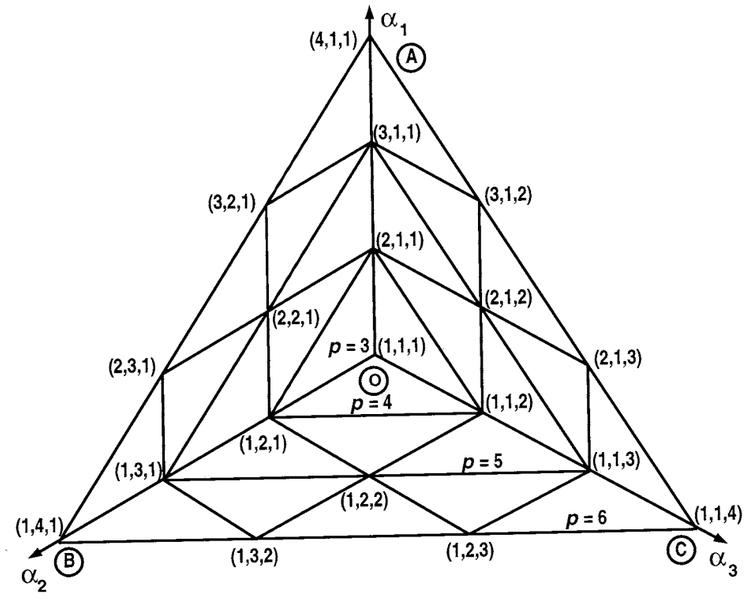


Figure 53. Face function triplet combinations.

p	$(\alpha_1, \alpha_2, \alpha_3)$	ϕ	
		Equation (54)	Equation (56)
3	1,1,1	1	1
4	2,1,1	$\xi_2 - \xi_1$	$\xi_1 - \frac{1}{3}$
	1,2,1	$2\xi_3 - 1$	$\xi_2 - \frac{1}{3}$
5	3,1,1	$\frac{1}{2}\left[3(\xi_2 - \xi_2)^2 - 1\right]$	$\xi_1^2 - \frac{3}{4}\xi_1 + \frac{3}{28}$
	2,2,1	$(\xi_2 - \xi_1)(2\xi_3 - 1)$	$\xi_1\xi_2 - \frac{1}{4}(\xi_1 + \xi_2) + \frac{1}{14}$
	1,3,1	$\frac{1}{2}\left[3(2\xi_3 - 1)^2 - 1\right]$	$\xi_2^2 - \frac{3}{4}\xi_2 + \frac{3}{28}$

Table 9. Triangular face functions.

of one-dimensional edge functions [89, 90]. The face functions are derived from

$$\frac{1}{4}[\xi_1^2 - 1][\xi_2^2 - 1]\phi(M_i^2) = \sqrt{\frac{2\alpha_1 - 1}{2}} \int_{-1}^{\xi_1} P_{\alpha_1 - 1}(t)dt \sqrt{\frac{2\alpha_2 - 1}{2}} \int_{-1}^{\xi_2} P_{\alpha_2 - 1}(t)dt \qquad (58)$$

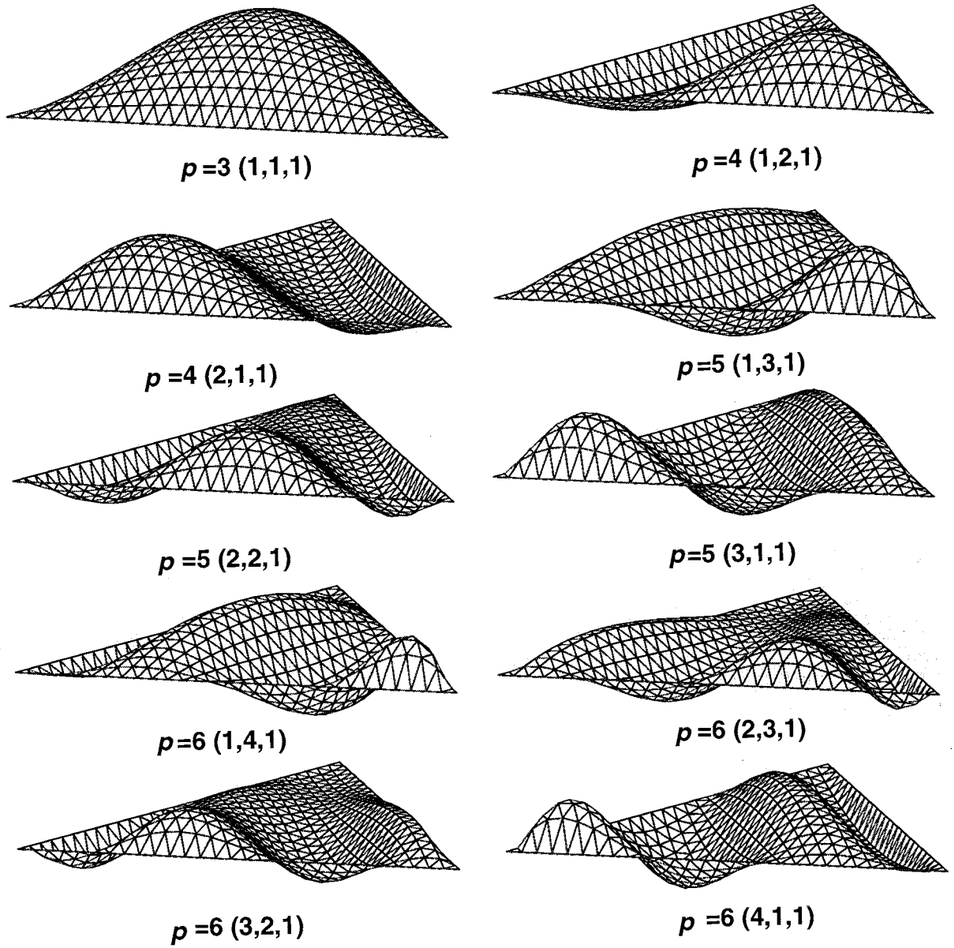


Figure 54. Triangular face shape functions. For clarity the shapes have been scaled by 5, 25, 100 and 200 for p=3, 4, 5 and 6 respectively.

with

$$\begin{aligned} \alpha_1, \alpha_2 &\geq 2 \\ \alpha_1 + \alpha_2 &= p \end{aligned} \tag{59}$$

and $p \geq 4$. The term $\frac{1}{4}\left(\xi_1^2 - 1\right)\left(\xi_2^2 - 1\right)$ is the quadrilateral face blend and can always be factored from the right hand side of equation (58). The pair (α_1, α_2) can be used to define the directional behavior of the quadrilateral face functions where α_i is the power of ξ_i in the resulting shape function. The geometrical construct used to generate the possible face shape functions in this case is analogous to the interior of Pascal's triangle [44, 96] and is shown in Figure 55. The

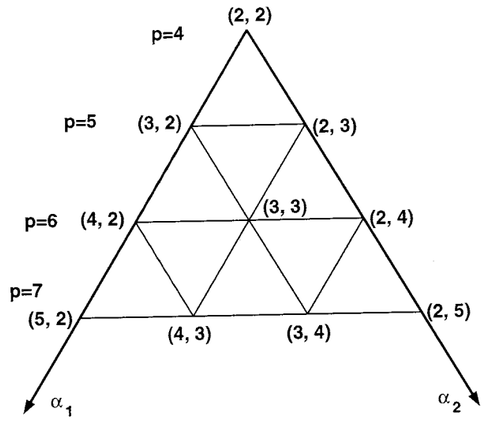


Figure 55. Construct to generate quadrilateral face modes.

p	(α_1, α_2)	ϕ
4	2, 2	1
5	2, 3	$\frac{\sqrt{12}}{4}\xi_2$
	3, 2	$\frac{\sqrt{12}}{4}\xi_1$
6	2, 4	$\frac{\sqrt{5}}{8}[5\xi_2^2 - 1]$
	3, 3	$\frac{3}{2}\xi_1\xi_2$
	4, 2	$\frac{\sqrt{5}}{8}[5\xi_1^2 - 1]$

Table 10. Quadrilateral face functions.

expressions for ϕ for *p=4,5,6* are given in Table 10. The corresponding face shape functions for a quadrilateral element are shown in Figure 56.

4.3.4 Tetrahedral Mesh Region

For a tetrahedral region, there are a total of $\frac{(p-2)(p-3)}{2}$ region functions that contribute to shape functions of polynomial order *p*, *p≥4*. Recall that for efficiency reasons $\phi(M_i^3)$ is the shape function. One possible set of region functions based on *Legendre* polynomials are given by [90]

$$\phi(M_i^3) = \xi_1\xi_2\xi_3\xi_4 P_{\alpha_1-1}(\xi_2 - \xi_1)P_{\alpha_2-1}(2\xi_3 - 1)P_{\alpha_3-1}(2\xi_4 - 1) \tag{60}$$

with

$$\begin{aligned} &\alpha_1, \alpha_2, \alpha_3 = 1, .., p-3 \\ &\alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 = p \end{aligned} \tag{61}$$

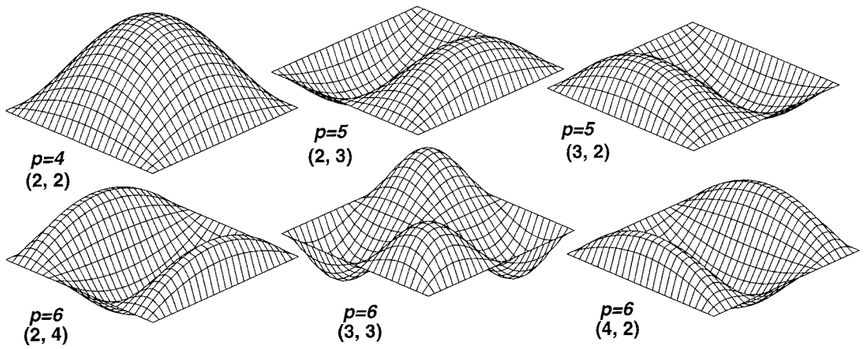


Figure 56. Quadrilateral face shape functions.

An alternate basis that results in better conditioning of element level matrices [16] is given by

$$\begin{aligned}
&\phi(M_i^3) = \xi_1\xi_2\xi_3\xi_4(A \times B \times C)\\
&A = \sum_{i=0}^{\alpha_1-1} (-1)^i i! \binom{\alpha_1 - 1}{i} \binom{\alpha_1}{i} \frac{(2m+5-i)!}{(2m+5)} (\xi_1)^{\alpha_1-1-i}\\
&B = \sum_{i=0}^{\alpha_2-1} i! \binom{\alpha_2 - 1}{i} \binom{\alpha_2}{i} \frac{(2n+3-i)!}{(2n+3)} (\xi_2)^{\alpha_2-1-i} (\xi_1 - 1)^i \qquad (62)\\
&C = \sum_{i=0}^{\alpha_3-1} i! \binom{\alpha_3 - 1}{i} \binom{\alpha_3}{i} \frac{(2\alpha_3-i)!}{2\alpha_3} (\xi_3)^{\alpha_3-1-i} (\xi_1 + \xi_2 - 1)^i
\end{aligned}$$

with α_i satisfying equation (61) and where $m = \alpha_1 + \alpha_2 + \alpha_3 - 3$, $n = \alpha_2 + \alpha_3 - 2$. Analogous to the triangular face functions, the directional behavior of the tetrahedral region functions can be interpreted by associating a quadruplet, $(\alpha_1, \alpha_2, \alpha_3, \alpha_4)$, with each region function where, α_i represents the power of ξ_i in the resulting shape function. Since only three of the four barycentric coordinates can be varied independently, the possible region functions are defined by the variation of any three of α_i based on equation (61). Equation (62) uses ξ_1, ξ_2, ξ_3 as the independent coordinates with $\alpha_4 = 1$. The corresponding geometric construct to generate the possible region functions is a set of four tetrahedra each of which allow for the independent variation of three of the four indices in the quadruplet. Figure 57 shows one of the tetrahedra that varies $\alpha_1, \alpha_2, \alpha_3$. A similar construction can be used to define the three tetrahedra that define the variation of $\alpha_1, \alpha_2, \alpha_4$ or $\alpha_1, \alpha_3, \alpha_4$ or $\alpha_2, \alpha_3, \alpha_4$. The expressions for the tetrahedral region functions for *p=4,5,6* based on equations (60) and (62) are listed in Table 11. Similar expressions are also given in [16, 90].

4.3.5 Hexahedral Mesh Region

For a hexahedral region, there are a total of $\frac{(p-4)(p-5)}{2}$ shape functions of polynomial order *p*, *p≥6*. The entity function for the regions are derived based on the tensor product polynomial

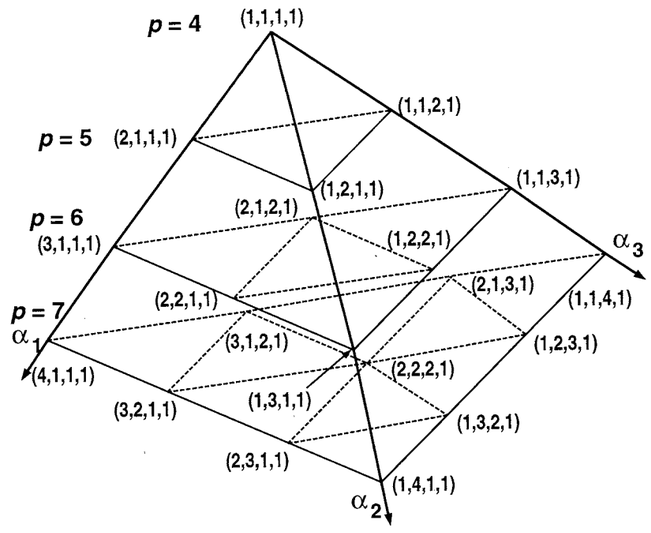


Figure 57. Geometric construct to generate tetrahedral region functions.

p	$(\alpha_1, \alpha_2, \alpha_3, \alpha_4)$	ϕ	
		Equation (60)	Equation (62)
4	(1,1,1,1)	1	1
5	(2,1,1,1)	$\xi_2 - \xi_1$	$\xi_1 - \frac{2}{7}$
	(1,2,1,1)	$2\xi_3 - 1$	$\xi_2 + \frac{2}{5}(\xi_1 - 1)$
	(1,1,2,1)	$1 - 2(\xi_1 + \xi_2 + \xi_3)$	$\xi_3 + \frac{2}{3}(\xi_1 + \xi_2 - 1)$
6	(3,1,1,1)	$\frac{3}{2}(\xi_2 - \xi_1)^2 - \frac{1}{2}$	$\xi_1^2 - \frac{2}{3}\xi_1 + \frac{1}{12}$
	(2,2,1,1)	$(\xi_2 - \xi_1)(2\xi_3 - 1)$	$\frac{-1}{45}(9\xi_1 - 2)(2\xi_1 + 5\xi_2 - 2)$
	(2,1,2,1)	$(\xi_2 - \xi_1)(1 - 2(\xi_1 + \xi_2 + \xi_3))$	$\frac{-1}{27}(9\xi_1 - 2)(2(\xi_1 + \xi_2 - 1) + 3\xi_3)$
	(1,3,1,1)	$\frac{3}{2}(2\xi_3 - 1)^2 - \frac{1}{2}$	$\xi_2^2 + \frac{6}{7}\xi_2(\xi_1 - 1) + \frac{1}{7}(\xi_1 - 1)^2$
	(1,2,2,1)	$(2\xi_3 - 1)(1 - 2(\xi_1 + \xi_2 + \xi_3))$	$\frac{-1}{27}(2\xi_1 + 7\xi_2 - 2)(2(\xi_1 + \xi_2 - 1) + 3\xi_3)$
	(1,1,3,1)	$\frac{3}{2}(1 - 2(\xi_1 + \xi_2 + \xi_3))^2 - \frac{1}{2}$	$\xi_3^2 + \frac{6}{5}\xi_3(\xi_1 + \xi_2 - 1) + \frac{3}{10}(\xi_1 + \xi_2 - 1)^2$

Table 11. Tetrahedral region functions.

p	$(\alpha_1, \alpha_2, \alpha_3)$	ϕ
6	2, 2, 2	$\frac{3\sqrt{6}}{8}\left[\left(3\xi_1^2-1\right)\left(3\xi_2^2-1\right)\left(3\xi_3^2-1\right)\right]$
7	3, 2, 2	$\frac{3\sqrt{10}}{8}\left[\left(5\xi_1^3-3\xi_1\right)\left(3\xi_2^2-1\right)\left(3\xi_3^2-1\right)\right]$
	2, 3, 2	$\frac{3\sqrt{10}}{8}\left[\left(3\xi_1^2-1\right)\left(5\xi_2^3-3\xi_2\right)\left(3\xi_3^2-1\right)\right]$
	2, 2, 3	$\frac{3\sqrt{10}}{8}\left[\left(3\xi_1^2-1\right)\left(3\xi_2^2-1\right)\left(5\xi_3^3-3\xi_3\right)\right]$

Table 12. Hexahedral region functions.

basis using the *Legendre* polynomials [89] and are given by

$$\begin{aligned}
\phi\left(M_i^3\right) &= A(\xi_1) \times B(\xi_2) \times C(\xi_3) \\
A(\xi_1) &= \sqrt{\frac{2\alpha_1 - 1}{2}} \int_{-1}^{\xi_1} P_{\alpha_1 - 1}(t)dt \\
B(\xi_2) &= \sqrt{\frac{2\alpha_2 - 1}{2}} \int_{-1}^{\xi_2} P_{\alpha_2 - 1}(t)dt \\
C(\xi_3) &= \sqrt{\frac{2\alpha_3 - 1}{2}} \int_{-1}^{\xi_2} P_{\alpha_3 - 1}(t)dt
\end{aligned} \tag{63}$$

with

$$\begin{aligned}
&\alpha_1, \alpha_2, \alpha_3 \geq 2 \\
&\alpha_1 + \alpha_2 + \alpha_3 = p.
\end{aligned} \tag{64}$$

Analogous to quadrilateral face functions, the directional behavior of the hexahedral region functions can be understood by associating a triplet, $(\alpha_1, \alpha_2, \alpha_3)$, with each region function where α_i is the power of ξ_i in the resulting shape function. Similar to the quadrilateral face, the geometric construct to generate the possible triplet combinations in this case is analogous to Pascal's tetrahedron used with equation (64). For example, at the apex (p=6) would be (2,2,2), followed by (3,2,2), (2,3,2) and (2,2,3) in the next layer (p=7). The expressions for hexahedral region functions generated from equation (63) are listed in Table 12 for p=6,7.

4.3.6 Wedge Mesh Region

There are a total of $\frac{(p-3)(p-4)}{2}$ wedge region functions with polynomial order p, $p \geq 5$. The region modes of polynomial order p are products of the triangular face functions and the one-dimensional edge function along ξ_4 given by

$$\begin{aligned}
\phi\left(M_i^3\right) &= A(\xi_1, \xi_2, \xi_3) \times B(\xi_4) \\
A(\xi_1, \xi_2, \xi_3) &= \xi_1 \xi_2 \xi_3 \phi\left(M_j^2\right) \\
B\left(\xi_4^3\right) &= -\left(\frac{1 - \xi_4^2}{2}\right) \phi\left(M_k^1\right)
\end{aligned} \tag{65}$$

p	$(\alpha_1\alpha_2,\alpha_3,\alpha_4)$	ϕ
5	(1,1,1,2)	$\phi(M_i^3) = -\frac{1}{2}\xi_1\xi_2\xi_3(1-\xi_4^2)$
6	(1,1,1,3)	$\phi(M_i^3) = -\frac{1}{2}\xi_1\xi_2\xi_3\xi_4(1-\xi_4^2)$
	(1,2,1,2)	$\phi(M_i^3) = -\frac{1}{2}\xi_1\xi_2\xi_3(\xi_2-\frac{1}{3})(1-\xi_4^2)$
	(2,1,1,2)	$\phi(M_i^3) = -\frac{1}{2}\xi_1\xi_2\xi_3(\xi_1-\frac{1}{3})(1-\xi_4^2)$

Table 13. Region functions for a wedge.

where $\phi\left(M_j^2\right)$ is given by equation (56) with $\hat{\xi}_1 = \xi_1$, $\hat{\xi}_2 = \xi_2$ and $\hat{\xi}_3 = \xi_3$. $\phi(M_k^1)$ is given by equation (53) with $\hat{\xi}_1 = \xi_4$ for parameterization *I* or $\hat{\xi}_1 = \frac{1-\xi_4}{2}$ and $\hat{\xi}_2 = \frac{1+\xi_4}{2}$ for parameterization *II*, and $p = \alpha_4$. The directional properties of the region functions can be interpreted by associating a quadruplet, $(\alpha_1, \alpha_2, \alpha_3, \alpha_4)$ with each mode. As in the case of triangles, since only two of ξ_1, ξ_2, ξ_3 can be varied independently. Therefore, two of $\alpha_1, \alpha_2, \alpha_3$, and α_4 are used in the generation of a possible set of region functions. For example, if ξ_1, ξ_2, ξ_4 are chosen as the independent coordinate parameters, the following rules generate the set of possible region functions,

$$\begin{aligned} &\alpha_1, \alpha_2 \geq 1 \\ &\alpha_4 \geq 2 \\ &\alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 = p. \end{aligned} \tag{66}$$

The region functions for *p=5,6* are listed in Table 13.

4.3.7 Pyramid Mesh Region

One possible way to construct region shape functions, similar to the scheme for hexahedral elements, is given by the following expression

$$\phi(M_i^3) = P_{\alpha_1}\left(\frac{2\xi_1}{1-\xi_2}\right) P_{\alpha_2}(\xi_2) P_{\alpha_3}\left(\frac{2\xi_3}{1-\xi_2}\right) \tag{67}$$

where the parametric coordinates are scaled based on equation (30) resulting from the degeneration described in Figure 45 with

$$\begin{aligned} &\alpha_1, \alpha_2, \alpha_3 \geq 2 \\ &\alpha_1 + \alpha_2 + \alpha_3 = p. \end{aligned} \tag{68}$$

This results in a total of $\frac{(p-4)(p-5)}{2}$ shape functions of polynomials order *p*, *p≥6*. The region shape functions from equation (67) can be generated by using the same triplet scheme as that for the hexahedron entity using Pascal's tetrahedron as the geometric construct. The region functions for a pyramid using equation (67) and *p*=6,7 are listed in Table 14.

p ($\alpha_1, \alpha_2, \alpha_3$)		ϕ
6	2, 2, 2	$\left(\frac{6\xi_1^2}{(1-\xi_2)^2}-\frac{1}{2}\right)\left(\frac{3}{2}\xi_2^2-\frac{1}{2}\right)\left(\frac{6\xi_3^2}{(1-\xi_2)^2}-\frac{1}{2}\right)$
7	3, 2, 2	$\left(\frac{20\xi_1^3}{(1-\xi_2)^3}-\frac{3\xi_1}{1-\xi_2}\right)\left(\frac{3}{2}\xi_2^2-\frac{1}{2}\right)\left(\frac{6\xi_3^2}{(1-\xi_2)^2}-\frac{1}{2}\right)$
	2, 3, 2	$\left(\frac{6\xi_1^2}{(1-\xi_2)^2}-\frac{1}{2}\right)\left(\frac{5}{2}\xi_2^3-\frac{3}{2}\xi_2\right)\left(\frac{6\xi_3^2}{(1-\xi_2)^2}-\frac{1}{2}\right)$
	2, 2, 3	$\left(\frac{6\xi_1^2}{(1-\xi_2)^2}-\frac{1}{2}\right)\left(\frac{3}{2}\xi_2^2-\frac{1}{2}\right)\left(\frac{20\xi_3^3}{(1-\xi_2)^3}-\frac{3\xi_3}{1-\xi_2}\right)$

Table 14. Pyramid region functions.

4.4 Computational Cost of Constructing Variable *p*-Order Meshes Using Decomposed Shape Functions and their Application in 3D

Comparisons of computational cost of shape-function evaluation using the decomposed and explicit uniform p forms are presented in terms of the operation counts required to evaluate a complete set of shape functions for a given p and the CPU times to evaluate element "stiffness" matrix

$$k_{ij} = \int_{\Omega^e} \frac{\partial N_i}{\partial \mathbf{x}} \left(\frac{\partial N_j}{\partial \mathbf{x}}\right)^T d\Omega. \tag{69}$$

This comparison is based on two-dimensional simplex elements. Counts for other element topologies can be performed in a similar manner. A third comparison illustrates the application of the methodology to the solution of *Poisson*'s equation in three dimensions showing the computational advantage of using the structures of the variable p environment.

The additional overhead of entity based p specification is associated with the queries of each entity for its polynomial order which cannot be avoided in a variable p-refinement framework. The decomposition of the shape function requires that the index of the bounding lower-order mesh entity in the element topological hierarchy be determined. This requires a constant time search of the list of the bounding faces, edges and vertices. For example, the determination of the local index of a mesh vertex when dealing with a tetrahedral element requires a traversal through the list of four vertices that belong to the closure of that element. The overhead in CPU time of this process compared to the explicit evaluation of the shape functions as a percent of the total CPU time required to solve a given problem is shown to be minimal in numerical example that follows.

Consider the set of all shape functions for a triangular element with uniform polynomial order 6 on all entities of its closure as shown in Table 15. With permutation of indices i and j, there are a total of 28 shape functions (3 from the vertices, 15 from the edges and 10 from the face). The number of multiplication and addition operations required to evaluate each of the shape functions is listed in the columns marked with "*" and "+", respectively, in Table 16. The total cost of evaluating all of the shape functions explicitly is 202 multiplications and 35 additions. The total cost of evaluating all of the shape functions in the decomposed form is 212 multiplications and 35 additions. The additional 10 multiplications in the decomposed form result from the redundant

Entity	p	Explicit	Decomposed	
			ψ	ϕ
vertex	0-1	ξ_i	ξ_i	1
edge	2	$-2\xi_i\xi_j$	$-2\xi_i\xi_j$	1
	3	$-2\xi_i\xi_j(\xi_j-\xi_i)$	$-2\xi_i\xi_j$	$\xi_j-\xi_i$
	4	$-2\xi_i\xi_j\left(\xi_j^2-3\xi_i\xi_j+\xi_i^2\right)$	$-2\xi_i\xi_j$	$\xi_j^2-3\xi_i\xi_j+\xi_i^2$
	5	$-2\xi_i\xi_j\left(\xi_j^3-6\xi_i\xi_j^2+6\xi_i^2\xi_j-\xi_i^3\right)$	$-2\xi_i\xi_j$	$\xi_j^3-6\xi_i\xi_j^2+6\xi_i^2\xi_j-\xi_i^3$
	6	$-2\xi_i\xi_j\left(\xi_j^4-10\left(\xi_i\xi_j^3-2\xi_i^2\xi_j^2+\xi_i^3\xi_j\right)+\xi_i^4\right)$	$-2\xi_i\xi_j$	$\xi_j^4-10\left(\xi_i\xi_j^3-2\xi_i^2\xi_j^2+\xi_i^3\xi_j\right)+\xi_i^4$
face	3	$\xi_1\xi_2\xi_3$	$\xi_1\xi_2\xi_3$	1
	4	$\xi_1\xi_2\xi_3(\xi_1-\frac{1}{3})$	$\xi_1\xi_2\xi_3$	$\xi_1-\frac{1}{3}$
		$\xi_1\xi_2\xi_3(\xi_2-\frac{1}{3})$	$\xi_1\xi_2\xi_3$	$\xi_2-\frac{1}{3}$
	5	$\xi_1\xi_2\xi_3(\xi_1^2-\frac{3}{4}\xi_1+\frac{3}{28})$	$\xi_1\xi_2\xi_3$	$\xi_1^2-\frac{3}{4}\xi_1+\frac{3}{28}$
		$\xi_1\xi_2\xi_3(\xi_1\xi_2-\frac{1}{4}(\xi_1+\xi_2)+\frac{1}{14})$	$\xi_1\xi_2\xi_3$	$\xi_1\xi_2-\frac{1}{4}(\xi_1+\xi_2)+\frac{1}{14}$
		$\xi_1\xi_2\xi_3(\xi_2^2-\frac{3}{4}\xi_2+\frac{3}{28})$	$\xi_1\xi_2\xi_3$	$\xi_2^2-\frac{3}{4}\xi_2+\frac{3}{28}$
	6	$\xi_1\xi_2\xi_3(\xi_2^3-\frac{1}{5}(6\xi_2^2-2\xi_2+\frac{1}{6}))$	$\xi_1\xi_2\xi_3$	$\xi_2^3-\frac{1}{5}(6\xi_2^2-2\xi_2+\frac{1}{6})$
		$\xi_1\xi_2\xi_3\left(\xi_1\xi_2(\xi_2-\frac{3}{5})+\frac{1}{5}\left(\frac{\xi_1}{3}-\xi_2^2+\frac{2\xi_2}{3}-\frac{1}{12}\right)\right)$	$\xi_1\xi_2\xi_3$	$\xi_1\xi_2(\xi_2-\frac{3}{5})+\frac{1}{5}\left(\frac{\xi_1}{3}-\xi_2^2+\frac{2\xi_2}{3}-\frac{1}{12}\right)$
		$\xi_1\xi_2\xi_3\left(\xi_1\xi_2(\xi_1-\frac{3}{5})+\frac{1}{5}\left(\frac{\xi_2}{3}-\xi_1^2+\frac{2\xi_1}{3}-\frac{1}{12}\right)\right)$	$\xi_1\xi_2\xi_3$	$\xi_1\xi_2(\xi_1-\frac{3}{5})+\frac{1}{5}\left(\frac{\xi_2}{3}-\xi_1^2+\frac{2\xi_1}{3}-\frac{1}{12}\right)$
		$\xi_1\xi_2\xi_3(\xi_1^3-\frac{1}{5}(6\xi_1^2-2\xi_1+\frac{1}{6}))$	$\xi_1\xi_2\xi_3$	$\xi_1^3-\frac{1}{5}(6\xi_1^2-2\xi_1+\frac{1}{6})$

Table 15. Shape functions for a triangle with uniform p=6.

multiplication $\psi * \phi$ when $\phi = 1$. This can be eliminated at the cost of checking for the p values, and is never performed at the three vertices since they are known *a-priori* to always have $\phi = 1$.

A variable p-refinement scheme solver for *Poisson*'s equation was implemented to compare the cost of decomposed shape-function evaluations versus the explicit uniform-p evaluation and its impact on the total solution time. To compute the element level matrices for a variable p mesh the following functionalities are required

1. Access to the topological hierarchy of mesh entities underlying an element. This functionality is available in the SCOREC mesh database [13].
2. Ability to query (and set) the polynomial order of interpolation for a given field variable over a given mesh entity.
3. Ability to query which of the specific shape functions of a given polynomial order are desired for a mesh entity.

Entity	p	Explicit		Decomposed					
				ψ		ϕ		$\psi * \phi$	
		*	+	*	+	*	+	*	+
vertex	0-1	0	0	0	0	0	0	1	0
edge	2	2	0	2	0	0	0	1	0
	3	3	1	2	0	0	1	1	0
	4	9	2	2	0	6	2	1	0
	5	13	3	2	0	10	3	1	0
	6	20	4	2	0	17	4	1	0
face	3	2	0	2	0	0	0	1	0
	4	3	1	2	0	0	1	1	0
		3	1	2	0	0	1	1	0
	5	5	2	2	0	2	2	1	0
		5	3	2	0	2	3	1	0
		5	2	2	0	2	2	1	0
	6	10	3	2	0	7	3	1	0
		9	5	2	0	6	5	1	0
		9	5	2	0	6	5	1	0
		10	3	2	0	7	3	1	0

Table 16. Operation count for shape functions for a triangle with uniform p=6.

4. Ability to evaluate shape function(s) of a given polynomial order from a mesh entity at a point in the domain of the element.
5. Ability to evaluate derivatives of shape function(s) of a given polynomial order from a mesh entity at a point in the domain of the element.

A set of generic operators provide these functionalities.

Consider the solution of *Laplace*'s equation on the unit square $[0,1] \times [0,1]$ with $u = 0$ on $x = 0$, $y = 0$, and $y = 1$ and $\frac{\partial u}{\partial x} = 1$ on $x = 1$. The exact solution of this problem is [52]

$$u(x,y) = \frac{4}{\pi^2} \sum_{n=1,3,5}^{\infty} \frac{\sinh(n\pi x)\sin(n\pi y)}{n^2 \cosh(n\pi)}. \tag{70}$$

The CPU time required to compute the element level matrices for a uniform 8 by 8 mesh of 128 triangular element mesh for p=1 to 8 with the explicit uniform p and the decomposed form of the shape functions are shown in Figure 58(a). Use of decomposed shape functions increases element matrix computation time by 10% relative to explicit evaluation when p=8. The increase is due to the local searching to determine the indices of lower-order mesh entities in the element

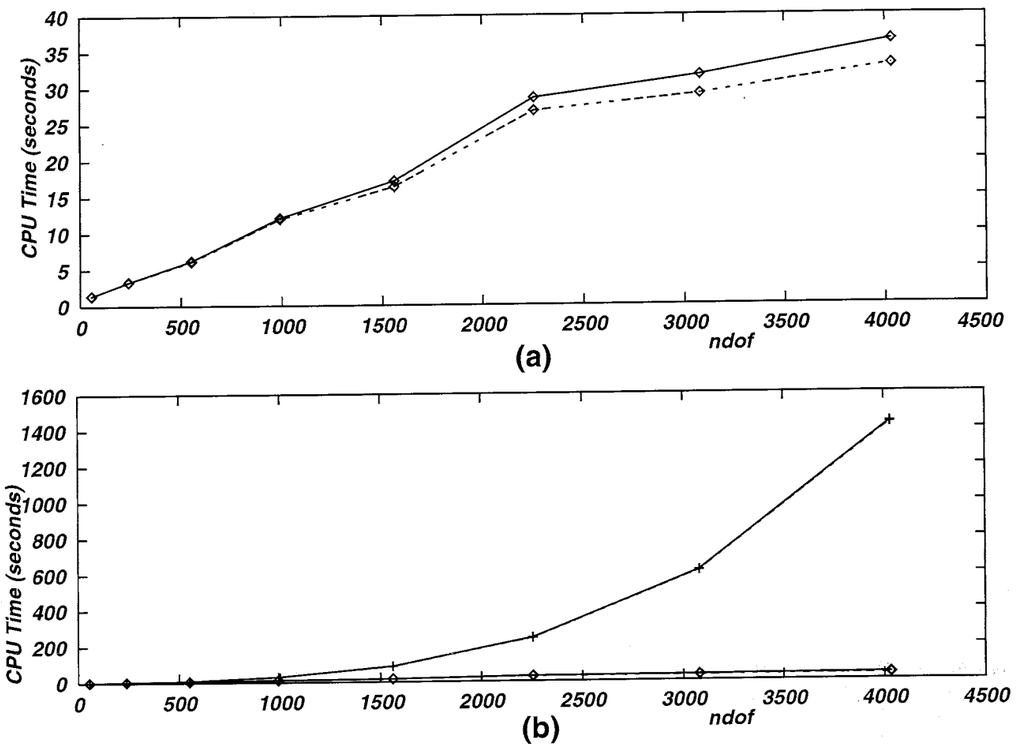


Figure 58. CPU time to compute (a) element level matrices and (b) total solution for 128 elements with uniform p from 1 to 8. Solid and broken lines represent decomposed and explicitly evaluated shape functions, respectively; ◇ and + represent times to compute element matrices and total solution time, respectively.

topological hierarchy. However, the increase in the CPU time required to solve the problem completely (Figure 58(b)), is minimal (0.3%).

Optimality of the *hp*-adaptive procedure lies in simultaneous variation of both the mesh size and the order of interpolation to attain solutions with prescribed accuracy using minimal computational resources [8, 63, 89]. To demonstrate the advantages of varying polynomial order over the domain of three-dimensional problems, consider the solution of *Poisson*'s equation

$$\begin{aligned} -\nabla^2 u(\mathbf{x}) &= f(\mathbf{x}),\ \mathbf{x} \in \overline{\Omega} \\ u(\mathbf{x}) &= 0,\ \mathbf{x} \in \partial(\Omega) \end{aligned} \tag{71}$$

where Ω is defined by a cube with side length 2 units and $f(x)$ is such that the exact solution is

$$u(\mathbf{x}) = x_1^n x_2^n x_3^n sin\Big(\frac{n\pi x_1}{2}\Big) sin\Big(\frac{n\pi x_2}{2}\Big) sin\Big(\frac{n\pi x_3}{2}\Big). \tag{72}$$

This solution is smooth within the domain but has highly localized gradients near the corners of Ω. Solution projections at $x_3 = \pm\frac{3}{4}$ for n=4 are shown in Figure 59.

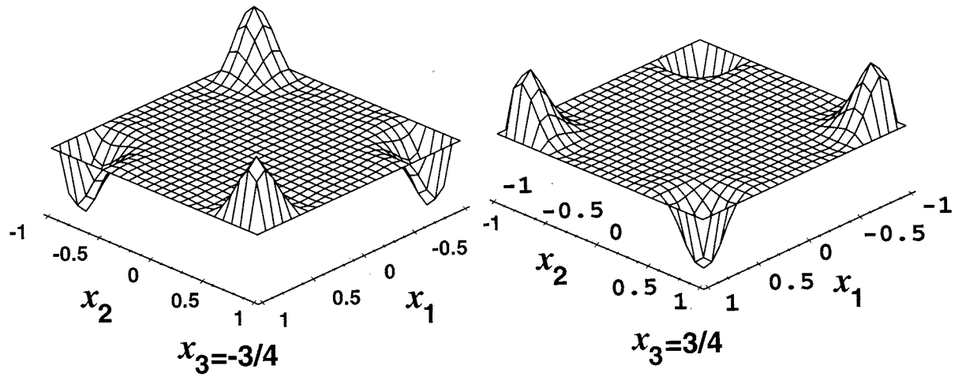


Figure 59. Solution profile at $x_3 = \pm\frac{3}{4}$ for n=4.

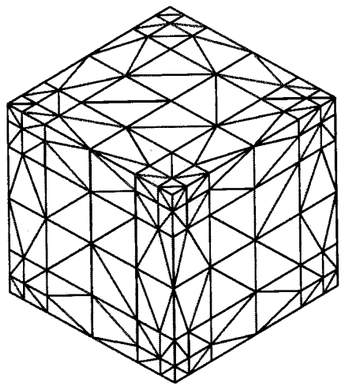


Figure 60. Mesh used for the three-dimensional example problem with 943 mesh regions, 2078 mesh faces, 1411 mesh edges and 277 mesh vertices.

Since the solution gradients are localized, a variable p-refinement strategy offers significant computational advantages over a uniform p-refinement when the mesh shown in Figure 60 is used to solve the problem. For example, if one considers the percentage global relative error, ρ, in the finite element solution, u^*, given by

$$\rho = 100\frac{\|u - u^*\|_2}{\|u\|_2} \tag{73}$$

then the use of a simple p-adaptation scheme based on equidistribution of regionwise error uses 25% fewer global *dof*'s compared to uniform p specification to achieve a numerical solution with 1% global error. The total CPU time needed by the adaptive procedure was 43% of that needed by the uniform p procedure.

5. Domain Geometry Representation Issues for *hp*-Adaptive Methods

Finite element methods use a *weak* form of the partial differential equation(s) and associated boundary condition(s) [44]. For a typical geometry based specification of a boundary value finite element problem (depicted in Figure 61), the need for a mapping, $\underline{x}(\underline{\xi})$, from the mesh entity parametric coordinate system, $\underline{\xi}$, to the coordinate system associated the partial differential equation, arises because of two specific operations involved in the finite element method:

1. Evaluation of the element level integrals associated with the *weak* form of the problem defined over the domain of the curvilinear element, Ω_i^e, or its boundary, Γ_i^e, and
2. Enforcement of essential boundary condition(s) specified on curved portion(s) of the boundary of Ω_G.

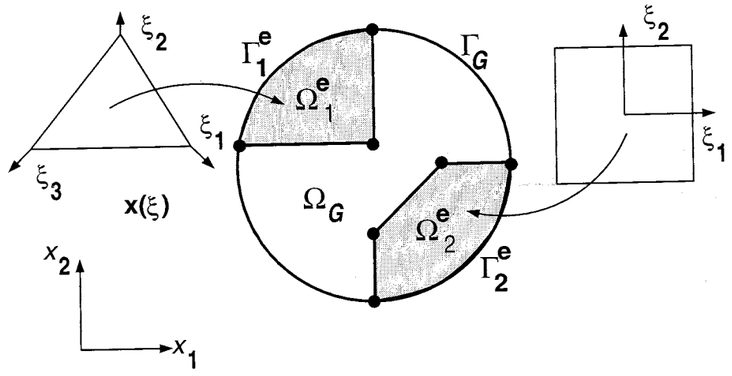


Figure 61. Mesh entity geometry mapping.

For n-dimensional Ω_G, components of the mapping are given by $x_i(\xi_j);\ i = 1,2,\cdots,n$ where the parametric coordinates ξ_j depend on the topology and the dimension of the mesh entity as described in Section 4.1. In the first operation, $\underline{x}(\underline{\xi})$ is used to transform the integrals to the parametric coordinate system where integration is performed. For example, the integral transformations for stiffness and mass type terms for the k-th element are given by [44, 87, 55, 62]

$$\int_{\Omega_k^e} \frac{\partial N_i}{\partial \underline{x}} \frac{\partial N_j}{\partial \underline{x}} d\underline{x} = \int_{\Omega_k^e} \frac{\partial N_i}{\partial \underline{\xi}} \left(\frac{\partial \underline{\xi}}{\partial \underline{x}}\right) \frac{\partial N_j}{\partial \underline{\xi}} \left(\frac{\partial \underline{\xi}}{\partial \underline{x}}\right) J(\underline{\xi}) d\underline{\xi} \tag{74}$$

and

$$\int_{\Omega_k^e} N_i(\underline{x}) N_j(\underline{x}) d\underline{x} = \int_{\Omega_k^e} N_i(\underline{x}(\underline{\xi})) N_j(\underline{x}(\underline{\xi})) J(\underline{\xi}) d\underline{\xi}, \tag{75}$$

respectively. Similarly, integrals resulting from the weak enforcement of natural boundary condition(s), $h_n(\underline{x})$, are given by [44, 87, 55, 62]

$$\int_{\Gamma_k^e} N_i(\underline{x}) h_n(\underline{x}) d\underline{x} = \int_{\Gamma_k^e} N_i\big(\underline{x}(\underline{\xi})\big) h_n\big(\underline{x}(\underline{\xi})\big) J\big(\underline{\xi}\big) d\underline{\xi} \tag{76}$$

where Γ_k^e defines the portion of the element boundary that has natural boundary condition specified, also require the geometric mapping, $\underline{x}(\underline{\xi})$, and the determinant of its jacobi matrix is defined as $J(\xi) = \left|\frac{\partial x_i}{\partial \xi_j}\right|$. The second operation is associated with the satisfaction of the essential boundary condition(s), $g(\underline{x})$, by the finite element solution, $u^{(h,p)}$

$$u^{(h,p)}(\underline{x}) = g(\underline{x}) \;\; \forall \; \underline{x} \in \Gamma_k^e \tag{77}$$

where Γ_k^e defines the portion of the element boundary over which $g(\underline{x})$ is prescribed. The standard *Galerkin weak* form requires explicit *a-priori* satisfaction of the essential boundary conditions [44]. The most straightforward method to achieve this is using an interpolatory approximation [42] of $g(\underline{x})$ over Γ_k^e using the shape functions used to construct the finite element solution as

$$\begin{aligned} g\big(\underline{x}(\underline{\xi})\big) &\approx g^*\big(\underline{x}(\underline{\xi})\big) = \sum_{l=1}^{n_p} a_l N_l\big(\underline{\xi}\big) \\ g^*\Big(\underline{x}\Big(\underline{\xi}^{(l)}\Big)\Big) &= g\Big(\underline{x}\Big(\underline{\xi}^{(l)}\Big)\Big);\; l = 1, \cdots n_p \end{aligned} \tag{78}$$

where $\left\{\underline{\xi}^{(l)}\right\}$ and $\{N_l\}$ define the sets of n_p interpolation points and basis functions for a p-th order interpolation of $g(\underline{x})$, respectively. Over the portion of Γ_k^e where $g(\underline{x})$ is prescribed, the coefficients of the interpolation, a_l, defined in equation (78) become the coefficients of interpolation of $u^{(h,p)}$

$$u^{(h,p)}\big(\underline{\xi}\big) = \sum_{l=1}^{n_p} a_l N_l\big(\underline{\xi}\big). \tag{79}$$

The solution of the interpolation problem to determine a_l requires the values of $g(\underline{x})$ at the interpolation points $\left\{\xi^{(l)}\right\}, l = 1 \cdots n_p$, which requires the pointwise evaluation of the mapping function $\underline{x}\Big(\underline{\xi}^{(l)}\Big)$.

The next key issue that must be addressed is the accuracy with which geometry must be accounted for during element level computations within a *hp*-adaptive environment. To this end, one needs to first examine the possible option(s) to account for geometry during the element level computations and then study the impact of any approximation(s) made on the convergence properties of *hp*-adaptive finite element approximations.

5.1 Approximate Element Level Computations

The solution obtained by the finite element method is approximate because it discretizes the *weak* form of the problem statement using finite-dimensional approximations of the function

spaces [44, 62, 55]. The error introduced by the discretization of the weak form is the *discretization error*. Even if all numerical computations were done exactly, the error in the finite element solution will decay only as fast the *discretization error*. Since the *discretization error* can only be eliminated in the limit $h \to 0$ and/or $p \to \infty$, the numerical computation of the element level integral(s) and essential boundary condition(s) enforcement can be done approximately as long as the errors introduced by such approximation(s), referred to as *variational crimes* [87] or *perturbation errors* [55], decay at a rate at least as fast as the rate of convergence of the *discretization error*.

If the element level integral(s) are represented abstractly as

$$\begin{aligned} I_{\Omega^e} &= \int_{\Omega^e} \kappa_{\Omega^e}\left(\underline{x}\left(\underline{\xi}\right)\right) d\underline{\xi} \\ I_{\Gamma^e} &= \int_{\Gamma^e} \kappa_{\Gamma^e}\left(\underline{x}\left(\underline{\xi}\right)\right) d\underline{\xi} \end{aligned} \tag{80}$$

where κ_{Γ^e} and κ_{Ω^e} represent the integrands associated with interior and the boundary of the element domain, respectively, then approximations can be introduced at one or more following basic functional levels:

1. Approximation of $\underline{x}\left(\underline{\xi}\right)$.
2. Approximation of κ_{Ω^e} and κ_{Γ^e}.
3. Approximate numerical integration method.

Useful operators to compute I_{Ω^e} and I_{Γ^e} denoted by $\overset{*_i}{\int\limits_{\Omega^e}}$ and $\overset{*_i}{\int\limits_{\Gamma^e}}$, respectively, that use combinations of the basic approximations possible can be stated as:

1. Exact geometry representation followed by approximate numerical integration yielding

$$\begin{aligned} \overset{*_1}{\int\limits_{\Omega^e}} \kappa_{\Omega^e} d\underline{\xi} &= \sum_{l=1}^{n_g} \left\{ \kappa_{\Omega^e}\left(\underline{x}\left(\underline{\xi}^{(l)}\right)\right) \right\} w^{(l)} \\ \overset{*_1}{\int\limits_{\Gamma^e}} \kappa_{\Gamma^e} d\underline{\xi} &= \sum_{l=1}^{n_g} \left\{ \kappa_{\Gamma^e}\left(\underline{x}\left(\underline{\xi}^{(l)}\right)\right) \right\} w^{(l)}. \end{aligned} \tag{81}$$

Using this approach, integral from equation (129) can be approximated as

$$\begin{aligned} \int_{\Omega^e} \frac{\partial N_i}{\partial \underline{x}} \frac{\partial N_j}{\partial \underline{x}} d\underline{x} &\approx \overset{*_1}{\int\limits_{\Omega^e}} \frac{\partial N_i}{\partial \underline{\xi}} \left(\frac{\partial \underline{\xi}}{\partial \underline{x}}\right) \frac{\partial N_j}{\partial \underline{\xi}} \left(\frac{\partial \underline{\xi}}{\partial \underline{x}}\right) J\left(\underline{\xi}\right) d\underline{\xi} \\ &= \sum_{l=1}^{n_g} \left\{ \left[\frac{\partial N_i}{\partial \underline{\xi}}\left(\underline{\xi}^{(l)}\right) \frac{\partial \underline{\xi}}{\partial \underline{\xi}}\left(\underline{\xi}^{(l)}\right) \right] \left[\frac{\partial N_j}{\partial \underline{\xi}}\left(\underline{\xi}^{(l)}\right) \frac{\partial \underline{\xi}}{\partial \underline{x}}\left(\underline{\xi}^{(l)}\right) \right] J\left(\underline{\xi}^{(l)}\right) \right\} w^{(l)} \end{aligned} \tag{82}$$

with $J\left(\underline{\xi}^{(l)}\right) = \left|\frac{\partial x_m}{\partial \xi_n}\right|\left(\underline{\xi}^{(l)}\right)$. Assuming that an exact representation of element geometry $\underline{x}\left(\underline{\xi}\right)$ is available, the errors with this approach are entirely the result of approximate numerical integration.

2. Approximate geometry representation, $\underline{x}^*(\underline{\xi}) \approx \underline{x}(\underline{\xi})$, followed by approximate numerical evaluation of the resulting integrand to yield

$$\begin{aligned} \int_{\Omega^e}^{*_2} \kappa_{\Omega^e} d\underline{\xi} &\approx \sum_{l=1}^{n_g} \left\{ \kappa_{\Omega^e} \left(\underline{x}^* \left(\underline{\xi}^{(l)} \right) \right) \right\} w^{(l)} \\ \int_{\Gamma^e}^{*_2} \kappa_{\Gamma^e} d\underline{\xi} &\approx \sum_{l=1}^{n_g} \left\{ \kappa_{\Gamma^e} \left(\underline{x}^* \left(\underline{\xi}^{(l)} \right) \right) \right\} w^{(l)} \end{aligned} \tag{83}$$

where $\left\{ \underline{\xi}^{(l)} \right\}$ and $\left\{ w^{(l)} \right\}$ define the sets of n_g coordinates and weights associated with the numerical integration scheme [1] chosen. For example, the approximate evaluation of the integral from equation (74) would be given by

$$\begin{aligned} \int_{\Omega^e} \frac{\partial N_i}{\partial \underline{x}} \frac{\partial N_j}{\partial \underline{x}} d\underline{x} &\approx \int_{\Omega^e}^{*_2} \frac{\partial N_i}{\partial \underline{\xi}} \left(\frac{\partial \underline{\xi}}{\partial \underline{x}^*} \right) \frac{\partial N_j}{\partial \underline{\xi}} \left(\frac{\partial \underline{\xi}}{\partial \underline{x}^*} \right) J(\underline{\xi}) d\underline{\xi} \\ &= \sum_{l=1}^{n_g} \left\{ \left[\frac{\partial N_i}{\partial \underline{\xi}} \left(\underline{\xi}^{(l)} \right) \frac{\partial \underline{\xi}}{\partial \underline{x}^*} \left(\underline{\xi}^{(l)} \right) \right] \left[\frac{\partial N_j}{\partial \underline{\xi}} \left(\underline{\xi}^{(l)} \right) \frac{\partial \underline{\xi}}{\partial \underline{x}^*} \left(\underline{\xi}^{(l)} \right) \right] J \left(\underline{\xi}^{(l)} \right) \right\} w^{(l)} \end{aligned} \tag{84}$$

with $J\left(\underline{\xi}^{(l)} \right) = \left| \frac{\partial x_m^*}{\partial \xi_n} \right| \left(\underline{\xi}^{(l)} \right)$. This is the commonly used technique in the *isoparametric h-version* finite element method [44, 96] where $\underline{x}^*(\underline{\xi})$ is constructed by interpolation identical to that used to construct the unknown solution u. There are two distinct sources of error in this process; first is the error due to approximation of the geometry and the second is due to the approximate numerical computation of the resulting integral.

3. Direct approximation of the integrand(s) followed by exact integration to yield

$$\begin{aligned} \int_{\Omega^e}^{*_3} \kappa_{\Omega^e} d\underline{\xi} &\approx \int_{\Omega^e} \kappa^*_{\Omega^e}(\underline{\xi}) d\underline{\xi} \\ \int_{\Gamma^e}^{*_3} \kappa_{\Gamma^e} d\underline{\xi} &\approx \int_{\Gamma^e} \kappa^*_{\Gamma^e}(\underline{\xi}) d\underline{\xi}. \end{aligned} \tag{85}$$

It is assumed that $\kappa^*_{\Omega^e}(\underline{\xi})$ and $\kappa^*_{\Gamma^e}(\underline{\xi})$ can be integrated exactly over their respective domains using either closed form formulae or numerical schemes [1]. One possible option that ensures this is the construction of $\kappa^*_{\Omega^e}$ and $\kappa^*_{\Gamma^e}$ as a linear combination of polynomials [42], $\phi_{\Omega^e}(\underline{\xi})$ and $\phi_{\Gamma^e}(\underline{\xi})$, defined over Ω^e and Γ^e, respectively, to obtain

$$\begin{aligned} \kappa^*_{\Omega^e}(\underline{\xi}) &= \sum_{l=1}^{n} a^{(l)}_{\Omega^e} \phi^{(l)}_{\Omega^e}(\underline{\xi}) \\ \kappa^*_{\Gamma^e}(\underline{\xi}) &= \sum_{l=1}^{m} a^{(l)}_{\Gamma^e} \phi^{(l)}_{\Gamma^e}(\underline{\xi}). \end{aligned} \tag{86}$$

The number of polynomials used, denoted by m and n, depend on the accuracy of the approximation desired and the topology of Ω^e and Γ^e. The errors associated with this approach result entirely from the approximation of $\kappa^*_{\Omega^e}$ and $\kappa^*_{\Gamma^e}$.

For computational efficiency of the *hp*-adaptive method, accuracy of the approximation(s) should only be as much as is required to maintain the rate of convergence of the *discretization error*. As is shown in section 6.2.1, the cost of approximations increases with increase in order of accuracy. Doing the approximation(s) more accurately than is required does not improve the rate of convergence of the *discretization error* and hence is not an efficient use of computational resources. To determine the required accuracy of approximation(s), one must relate the rate of convergence of the approximation error(s) to that of the finite element *discretization error*. The next section derives conditions that approximation(s) used in each approach must satisfy to preserve the rate of convergence of the *discretization error* for general second order elliptic boundary value problems.

5.2 Accuracy of Approximate Element Level Computations for *hp*-Adaptive Methods

Since finite element solutions are constructed using functions belonging to the *Sobolev Spaces*, $H^m(\Omega)$ [44, 62, 20], definitions of *Sobolev Spaces* and norms associated with it are required to understand the error analysis of the *hp* finite element approximations.

Definition: Sobolev Space of Functions [44, 62, 20]- $H^m(\Omega)$ defines the space of functions that have square integrable derivatives upto order m

$$H^m(\Omega) = \{v | v_{,i} \in L_2(\Omega), |i| \leq m\}$$
$$L_2(\Omega) = \left\{ w | \int_\Omega w^2 d\Omega < \infty \right\}. \tag{87}$$

Based on this definition the m-th *Sobolev norm* of a scalar function v is given by [44]

$$\|v\|_m = \left\{ \int_\Omega \left(vv + v_{,i}v_{,i} + \ldots + v_{,\underbrace{ijkl}_{m \text{ times}}} v_{,\underbrace{ijkl}_{m \text{ times}}} \right) \right\}^{\frac{1}{2}}. \tag{88}$$

For higher order tensors, the scalar products must be replaced by appropriate inner products.

If $u^{(h,p)}$ represents the *hp* finite element approximation of the exact solution u, then the *discretization error* can be represented in the following abstract form [7]

$$\left\| u - u^{(h,p)} \right\|_m \leq \frac{C h^{\alpha(p,m,r)} \|u\|_r}{p^{r-1}} \tag{89}$$

where, C is a constant independent of u, h, and p; h represents the maximum element diameter, and p represents the maximum degree of complete polynomials used in the shape functions. For second order elliptic boundary value problems, $\alpha = min\{p + 1 - m, r - m\}$ [44]. This implies

that if u is sufficiently smooth such that $r \geq p+1 \Rightarrow \alpha = p+1-m$, then the *discretization error* will converge exponentially with respect to p-refinement at a rate given by

$$\left\| u - u^{(h,p)} \right\|_m \leq \frac{C h^{p+1-m} \|u\|_r}{p^{r-1}}. \tag{90}$$

Recall that error(s) associated with approximate element level computation operators $\overset{*_i}{\underset{\Omega^e}{\int}}$ and $\overset{*_i}{\underset{\Gamma^e}{\int}}$ result from two sources:

1. Approximation of functions over Ω^e and Γ^e.
2. Approximate numerical integration over Ω^e and Γ^e.

Therefore, to control the errors associated with $\overset{*_i}{\underset{\Omega^e}{\int}}$ and $\overset{*_i}{\underset{\Gamma^e}{\int}}$, one must understand the errors in function approximations and numerical integration and relate them to the finite element *discretization error*.

5.2.1 Optimal Function Approximations

Recall that the specific functions approximations that must be dealt with are the approximation of the geometry mapping, $\underline{x}^*(\underline{\xi}) \approx \underline{x}(\underline{\xi})$ in $\overset{*_1}{\underset{\Omega^e}{\int}}$, $\overset{*_1}{\underset{\Gamma^e}{\int}}$, $\overset{*_3}{\underset{\Omega^e}{\int}}$, and $\overset{*_3}{\underset{\Gamma^e}{\int}}$, and the direct approximation of the integrand, $\kappa_{\Omega^e} \approx \kappa^*_{\Omega^e}$ and $\kappa_{\Gamma^e} \approx \kappa^*_{\Gamma^e}$ in $\overset{*_3}{\underset{\Omega^e}{\int}}$, and $\overset{*_3}{\underset{\Gamma^e}{\int}}$. The goal of this section is therefore to determine the correct order of approximation, q, in terms of p, such that the error in geometry approximation, $\underline{x}(\underline{\xi}) - \underline{x}^*(\underline{\xi})$, and integrand approximations, $\kappa_{\Omega^e} - \kappa^*_{\Omega^e}$ and $\kappa_{\Gamma^e} - \kappa^*_{\Gamma^e}$, measured in the appropriate norm, converge at least as fast as the finite element *discretization error* given by equation (90).

From basic approximation theory, the error in approximating a function $\theta(\underline{\xi})$ with $\theta^*(\underline{\xi})$, in any appropriate norm, is given by [67]

$$||\theta - \theta^*|| \leq (1+\lambda)\varphi \tag{91}$$

where φ is the *best approximation error* and λ is the Lebesgue constant of the approximation operator. Since finite element solutions are usually defined as linear combinations of polynomial functions in $H^m(\Omega)$ and their errors measured in appropriate *Sobolev* norms, the q-th order approximations of $\theta(\underline{\xi})$, $\underline{\xi} \in \overline{\Omega}^e$ will be restricted to those constructed by linear combinations of polynomial functions, $\left\{\phi_i(\underline{\xi}) | \phi_i \in H^{q+1}\right\}$, given by

$$\theta^{(q)}(\underline{\xi}) = \sum_{i=1}^{n_q} a_i \phi_i(\underline{\xi}) \tag{92}$$

where the number of polynomial basis functions used, n_q, depends on the topology and the dimension of Ω^e. The number of functions needed to construct a complete q-th degree polynomial basis for various element topologies is given in table 17.

Ω^e	n_q
Line	$q+1$
Triangle	$\frac{(q+1)(q+2)}{2}$
Quadrilateral	$(q+1)^2$
Tetrahedron	$\frac{(q+1)(q+2)(q+3)}{6}$
Hexahedron	$(q+1)^3$
Wedge	$\frac{(q+1)^2(q+2)}{2}$
Pyramid	$(q+1)^3$

Table 17. Number of basis functions needed for completeness.

For smooth functions ($\theta \in H^{q+1}$), φ is known to converge exponentially [67] and the overall convergence rate of $\theta - \theta^{(q)}$ is determined by the rate at which λ grows with respect to q [18, 19]. For interpolation sets where the interpolation points are equally spaced λ can grow exponentially causing lack of exponential convergence of $\left\|\theta - \theta^{(q)}\right\|$ even when θ is smooth. However, for the interpolation sets derived by [18, 19], λ grows approximately as $O(\log(q))$ which ensures exponential convergence of $\left\|\theta - \theta^{(q)}\right\|$.

Using the above arguments, the specific form of equation (91) for polynomial interpolation of θ (equation (92)) is given by[6] [62, 55]

$$\left\|\theta - \theta^{(q)}\right\|_m \leq C_1 h^{q+1-m} ||\theta||_r;\ r \geq q+1 \tag{93}$$

where C_1 is independent of h and θ and h defines the mesh size parameter. C_1 is a function of q and its growth is controlled based on the choice of appropriate interpolation set.

Since θ appears in the integrand, equation (93) is the error bound for the integrand approximation. The error bound for the integral approximation will be one order higher [62] given by

$$\left\| \int_{\Omega^e} \theta d\Omega - \int_{\Omega^e} \theta^{(q)} d\Omega \right\|_m \leq \tilde{C}_1 h^{q+2-m} ||\theta||_r;\ r \geq q+1. \tag{94}$$

To maintain the rate of convergence of *discretization error*, the rate of convergence in equation (94) must be greater than or equal to the rate given by equation (90). Assuming that θ is sufficiently smooth ($r \geq p+1$), this leads to the condition

$$q+2-m \geq p+1-m \tag{95}$$

yielding

$$q \geq p-1. \tag{96}$$

Note that if $r \geq p+1$, then equation (96) also defines the necessary and sufficient condition for approximation of κ_{Ω^e} and κ_{Γ^e} in the approach defined by $\overset{*_3}{\int}_{\Omega^e}$ and $\overset{*_3}{\int}_{\Gamma^e}$ to preserve the optimal convergence rate of the *discretization error* for a given elliptic boundary value problem.

[6] For details of derivation see [62], Chapter 6 and 8.

5.2.2 Optimal Numerical Integration Order

Section 5.2.1 presented the error analysis for direct polynomial interpolation of integrand assuming exact integration of the resulting polynomial integrand. This section reviews results based on the analysis of the error in approximate numerical integration process. Recall that numerical integration is required in options $\overset{*_1}{\underset{\Omega^e}{\int}}$, $\overset{*_1}{\underset{\Gamma^e}{\int}}$, $\overset{*_2}{\underset{\Omega^e}{\int}}$, and $\overset{*_2}{\underset{\Gamma^e}{\int}}$. Accuracy requirements for optimal numerical integration in the *h*- and *p*-version of the finite element method have been investigated in detail in [87, 20, 55, 46, 11]. It has been shown [20, 62] that optimal convergence of *discretization error* for second order elliptic boundary value problems can be ensured by picking a numerical integration scheme that accounts for the geometry related terms to order $p-1$. For example, for a given p, $\frac{\partial N_i}{\partial \xi}$ in equation (84) is of order $p-1$ and if the remaining geometry related terms from $\frac{\partial \xi_i}{\partial x_j}$ and $J\left(\underline{\xi}\right)$ are approximated by order $p-1$ then the order of the complete integrand is given by $2(p-1)+(p-1)=3(p-1)$ and the optimal number of integration points needed in n dimensions using a tensor product rule [1] is given by

$$n_g = \underbrace{\left(\left\lceil \frac{3p-3+\beta}{2} \right\rceil\right) \times \cdots \times \left(\left\lceil \frac{3p-3+\beta}{2} \right\rceil\right)}_{n\ times} \tag{97}$$

where β takes the value of 1 and 2 for *Gauss-Legendre* and *Gauss-Lobatto* schemes, respectively [1].

The next chapter outlines a technique to obtain finite element geometric mapping that is pointwise exact with respect to the shape of the domain boundary as defined within a geometric modelling system. Chapter 7 then discusses available options for numerical evaluation of element level integrals and details one that is computationally most efficient for *hp*-methods.

6. Mesh Geometry Mapping For Higher Order Methods

Functions related to geometry that appear in finite element computations can be represented in general by $\frac{\partial^n \underline{x}}{\partial \underline{\xi}^n}$, $n \geq 0$ with the case of $n = 0$ defining the position vector $\underline{x}(\underline{\xi})$. The highest order of derivative(s) required depends upon the order of the partial differential equation. For the case of second order elliptic equations only up to first order derivatives, $\frac{\partial \underline{x}}{\partial \underline{\xi}}$, are required. Since finite element computations only require them approximately and at specific locations $\underline{\xi}$, they can be constructed in a manner that allows for their evaluation at any $\underline{\xi} \in \overline{\Omega}^e$. For effective use with high order finite elements, three issues associated with the approximate position mapping $\underline{x}^*(\underline{\xi})$, and derivative, $\frac{\partial \underline{x}^*}{\partial \underline{\xi}}$ become important:

1. *Accuracy*: As shown in Section 5.2.1, $\underline{x}^*(\underline{\xi})$ must be accurate to order $p-1$ in order to preserve the optimal convergence rate of the discretization error.
2. *Smoothness*: The optimal rate of convergence of the finite element error given by equation (90) cannot be achieved in practice if $\underline{x}^*(\underline{\xi})$ is not smooth enough.
3. *Cost*: Since the required order of approximation and the number of pointwise evaluations both increase with increasing p, the construction and the evaluation of the mapping and its derivatives must be done efficiently.

The exact mathematical representation of the shape of Γ_G, $\underline{x}(\underline{\zeta})$, is housed within a geometric modelling system in terms of local parametric coordinate systems of model topological entities given by $\underline{\zeta}$; therefore, a technique to relate $\underline{\xi}$ to $\underline{\zeta}$, such that the resulting $\underline{x}(\underline{\xi})$ is exact with respect to Γ_G when $\underline{\xi} \in \Gamma^e \subset \Gamma_G$, is required. Pointwise evaluation of $\underline{x}(\underline{\xi})$ can then be used during finite element computations directly, or used in the construction of polynomial approximations of $\underline{x}^*(\underline{\xi})$ which have the required accuracy.

6.1 Blended Mapping Conforming Exactly to Γ_G

This section describes a technique for obtaining $\underline{x}(\underline{\xi})$ which is exact with respect to Γ_G when $\underline{\xi} \in \Gamma^e \sqsubset \Gamma_G$.

The geometry based mesh hierarchy provides an ideal means to construct the mapping $\underline{x}(\underline{\xi})$ based on the mathematical definition of the shape of the domain boundary entities as housed within the geometric modeler. For a mesh entity, M_i^d, classified on a model entity, G_j^d, the desired map can be obtained as

$$\underline{x} = \underline{x}(\zeta_i(\xi_j)) \tag{98}$$

where ζ_i represents the parametric coordinate system associated with G_j^d. The derivative(s) of the mapping can be evaluated by application of the chain-rule $\frac{\partial \underline{x}}{\partial \underline{\xi}} = \frac{\partial \underline{x}}{\partial \underline{\zeta}} \frac{\partial \underline{\zeta}}{\partial \underline{\xi}}$. Figure 62 depicts this mapping for a triangular and quadrilateral mesh face using the three coordinate systems $\underline{\xi}$, $\underline{\zeta}$, and $\underline{x}$. $\underline{x}(\underline{\zeta})$ and $\frac{\partial \underline{x}}{\partial \underline{\zeta}}$ are queried on a pointwise basis from the geometric modeler. The implicit assumption in this process, similar to the technique outlined in [14], is that the model entities have an underlying continuous, nondegenerate parametric space. The availability of

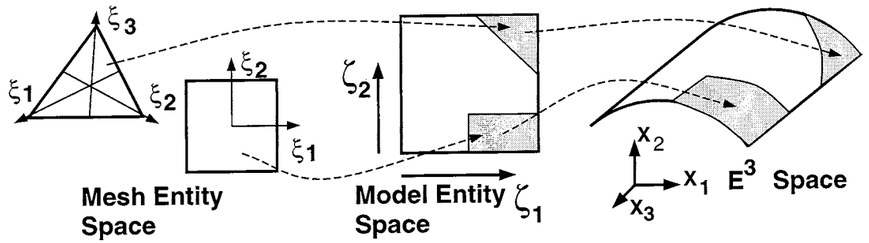


Figure 62. Mapping concept illustrated for a boundary face.

an underlying parameterization for specific G_i^d of Ω_G depends on the representation scheme used within the geometric modelling system. Most geometric modelers do not have an explicit parametric coordinate system for G_i^3; instead, they are represented implicitly as the portion of three-dimensional space that is bounded by a closed set of G_i^2 [94, 28, 86]. For example, a solid block is represented implicitly by the volume enclosed by the six faces that bound it. Therefore, for mesh entities $M_i^d \sqsubset G_j^3$, the shape of the mesh entity is constructed based on blending shapes of the model entities on which the bounding lower order mesh entities are classified. Since the blended shapes match the curved shape of Γ_G and they cover the interior of Ω_G, they introduce no geometric approximation.

The technique used to construct $\zeta_i(\xi_j)$ depends on the classification of a $M_k^{d_k}$ with respect to Ω_G. In addition, the construction of $\zeta_i(\xi_j)$ is also influenced by any *trimming* of the geometric model entities as used by the geometric modeler.

Definition: *A **trimmed** model entity is one which does not span the entire domain of the underlying parametric coordinate system.*

For example, Figure 63(a) depicts an *untrimmed* model face whereas, Figure 63(b) depicts a *trimmed* model face.

6.1.1 Mesh Edge Mapping

For a M_i^1 on a $G_i^{d_j}$; $d_j = 1, 2$, (Figure 64), in the absence of interference with any trimmed boundary entities and with acceptable resulting element shapes, $\zeta_i(\xi_j)$ can be given by a linear interpolation between ζ_i values at the vertices

$$\zeta_i(\xi) = \zeta_i(1,0)\xi_1 + \zeta_i(0,1)\xi_2, \;\; \xi_1 + \xi_2 = 1. \tag{99}$$

For a M_i^1 inside a G_j^3, the shape of the mesh edge can be represented as a straight line joining the vertices in the Cartesian coordinate system provided the element shapes are acceptable. Interference with trimming boundaries or excessive distortion may require a more general curvilinear representation of $\zeta_i(\xi_j)$ and the shape of the mesh edges classified in the interior of a model region. For example, the geometry of an interior mesh edge can be assigned a general quadratic form $x_i = a_i + b_i\xi_1 + c_i\xi_1^2$ or a cubic form $x_i = a_i + b_i\xi_1 + c_i\xi_1^2 + d_i\xi_1^3$.

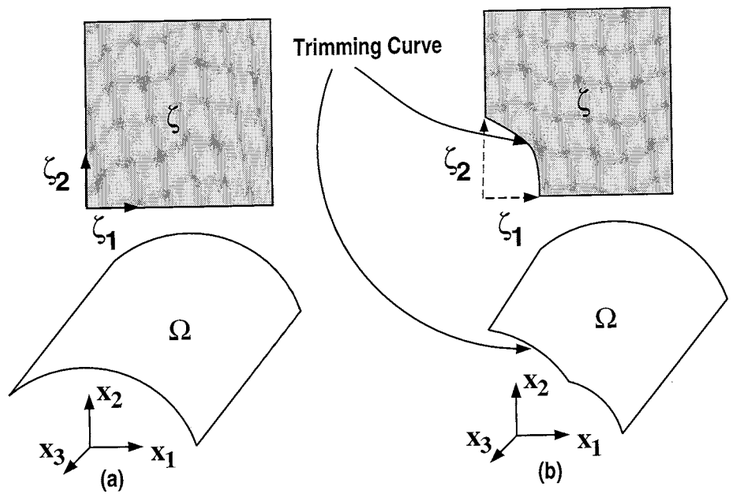


Figure 63. (a) Untrimmed and (b) trimmed model faces.

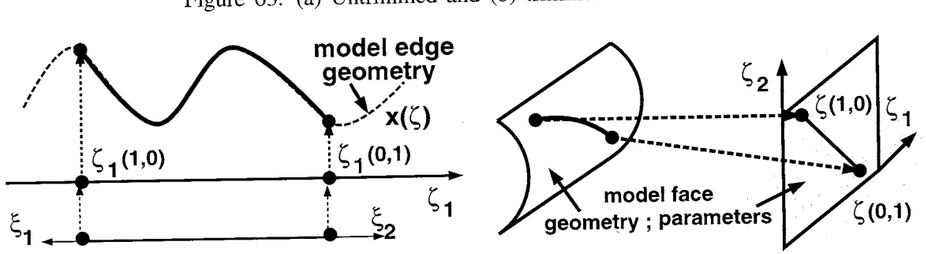


Figure 64. Construction of mesh edge mapping.

6.1.2 Mesh Face Mapping

The construction of $\zeta_i(\xi_j)$ for M_i^2 on G_j^2 needs to account for the possibility of trimmed model faces which do not span the entire parametric domain of the underlying surface as shown in Figure 63(b). If M_i^2 is classified on an untrimmed model face (Figure 65), then $\zeta_i(\xi_j)$ can be constructed as a linear interpolation of the ζ_i values at the vertices of the face

$$\zeta_i(\xi) = \zeta_i(1,0,0)\xi_1 + \zeta_i(0,1,0)\xi_2 + \zeta_i(0,0,1)\xi_3. \tag{100}$$

The linear interpolation of vertex values does not work if the G_j^2 is trimmed because the curves defining the mesh edges, in the parametric space of the G_j^2, may not be straight in that space leading to "spills" or "gaps" as shown in Figure 66. In addition, a higher order interpolation may be required if the resulting element shape is unacceptable.

In such cases the construction of $\zeta_i(\xi_j)$ must account for the curved shape of the boundary between B and C as shown in Figure 67. Techniques based on the Boolean sum interpolation theory can be used to blend the boundary curves [37, 40]. Using the scheme described in [40]

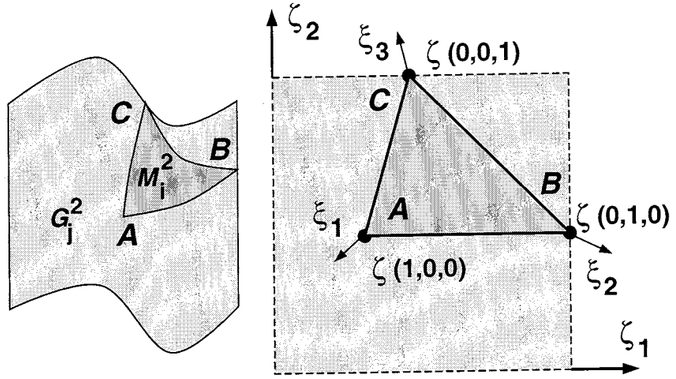


Figure 65. Construction of $\zeta_i(\xi)$ for untrimmed model faces.

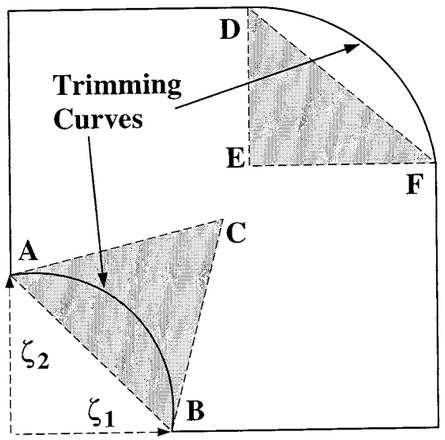


Figure 66. "Gaps" and "spills" with linear interpolation on trimmed model faces.

yields

$$\zeta_i(\xi_j) = \frac{1}{2}\{\left(\frac{\xi_1}{1-\xi_2}\right)p_i(\xi_2) + \left(\frac{\xi_2}{1-\xi_1}\right)p_i(1-\xi_1) + \left(\frac{\xi_3}{1-\xi_2}\right)q_i(1-\xi_2) + \left(\frac{\xi_2}{1-\xi_3}\right)q_i(\xi_3)$$
$$+ \left(\frac{\xi_1}{1-\xi_3}\right)r_i(1-\xi_3) + \left(\frac{\xi_3}{1-\xi_1}\right)r_i(\xi_1) - \xi_1\zeta_i(1,0,0) - \xi_2\zeta_i(0,1,0) - \xi_3\zeta_i(0,0,1)\} \quad (101)$$

where $p(t), q(t), r(t);\ t \in [0,1]$ define the shape of the curves *AB*, *BC* and *CD* respectively.

For M_i^2 inside G_j^3 with no underlying parametrization and one or more bounding edges classified on a curved model boundary, the positions of the bounding vertices and the shapes of

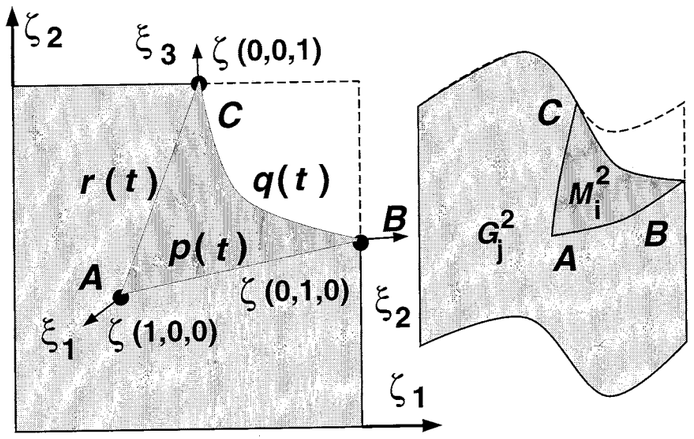


Figure 67. Construction of $\zeta_i(\xi)$ for trimmed model faces.

the bounding edges are blended to obtain $\mathbf{x}(\xi)$ as

$$x_i(\xi_j) = \frac{1}{2}\{\left(\frac{\xi_1}{1-\xi_2}\right)P_i(\xi_2) + \left(\frac{\xi_2}{1-\xi_1}\right)P_i(1-\xi_1) + \left(\frac{\xi_3}{1-\xi_2}\right)Q_i(1-\xi_2) + \left(\frac{\xi_2}{1-\xi_3}\right)Q_i(\xi_3)$$
$$+ \left(\frac{\xi_1}{1-\xi_3}\right)R_i(1-\xi_3) + \left(\frac{\xi_3}{1-\xi_1}\right)R_i(\xi_1) - \xi_1 x_i(1,0,0) - \xi_2 x_i(0,1,0) - \xi_3 x_i(0,0,1)\} \tag{102}$$

where P, Q, R define the position vectors for the shapes of the bounding edges. For mesh faces classified inside model regions with none of the bounding edges classified on a curved model boundary, $\mathbf{x}(\xi)$ can be given by linear interpolation of the vertex coordinates

$$x_i(\xi_j) = x_i(1,0,0)\xi_1 + x_i(0,1,0)\xi_2 + x_i(0,0,1)\xi_3. \tag{103}$$

However, interior mesh faces with no edges on a curved boundary may in general be curved to improve mesh quality in terms of distortion measures.

6.1.3 Mesh Region Mapping

The volume mapping for a tetrahedron is obtained by blending the boundary faces and edges [49] as

$$\begin{aligned} x_i(\xi_j) = {} & (1-\xi_1)G_i\left(\xi'\right) + (1-\xi_2)E_i\left(\xi'\right) + (1-\xi_3)F_i\left(\xi'\right) + (1-\xi_4)D_i\left(\xi'\right) \\ & - (1-\xi_1-\xi_2)W_i\left(\xi'\right) - (1-\xi_1-\xi_3)T_i\left(\xi'\right) - (1-\xi_1-\xi_4)Q_i\left(\xi'\right) \\ & - (1-\xi_2-\xi_3)S_i\left(\xi'\right) - (1-\xi_2-\xi_4)R_i\left(\xi'\right) - (1-\xi_3-\xi_4)P_i\left(\xi'\right) \\ & + \xi_1 x_i(1,0,0,0) + \xi_2 x_i(0,1,0,0) + \xi_3 x_i(0,0,1,0) + \xi_4 x_i(0,0,0,1) \end{aligned} \tag{104}$$

where P, Q, R, S, T, W and D, E, F, G represent the position vectors defining the shapes of the bounding edges and faces, respectively. ξ' defines the face (edge) parametric coordinates

normalized such that $\sum_{k=0}^{n_v} \xi_k^{'} = 1$ where n_v is the number of mesh vertices bounding the face (edge). For example, for face $\xi_4 = 0$, $\xi_1^{'} = \frac{\xi_1}{(\xi_1+\xi_3+\xi_3)}, \xi_2^{'} = \frac{\xi_2}{(\xi_1+\xi_3+\xi_3)}, \xi_3^{'} = \frac{\xi_3}{(\xi_1+\xi_3+\xi_3)}$ and for edge $\xi_2 = \xi_3 = 0$, $\xi_1^{'} = \frac{\xi_1}{(\xi_1+\xi_2)}, \xi_2^{'} = \frac{\xi_2}{(\xi_1+\xi_2)}$. For mesh regions with no bounding edges or faces classified on a curved model boundary, the mapping can be obtained by linear interpolation of vertex coordinates

$$x_i(\xi_j) = x_i(1,0,0,0)\xi_1 + x_i(0,1,0,0)\xi_2 + x_i(0,0,1,0)\xi_3 + x_i(0,0,0,1)\xi_4 \tag{105}$$

However, interference with other curved mesh entities in the neighborhood may require curving one or more boundary entities of an interior mesh region.

6.2 Options For Approximate Geometry Representation

Available approaches for the construction of $\underline{x}^*(\underline{\xi})$ for individual $\overline{\Omega}^e$ can be described as:

1. *Polynomial interpolation* [42].
2. *Approximate fitting* techniques used in curve and surface design [30].

6.2.1 Polynomial Interpolation

The basic idea underneath this technique is to construct $\underline{x}^*(\underline{\xi})$ as a linear combination of polynomial basis functions [67, 42] defined over $\overline{\Omega}^e$, $\phi_k(\underline{\xi})$, to give

$$x_i^*(\underline{\xi}) = \sum_{k=1}^{n_q} \phi_k(\underline{\xi}) b_i^{(k)} \tag{106}$$

where n_q, given by table 17, defines the number of functions in a basis complete up to order q. In the *isoparametric* form of equation (106), widely used for low order h-version finite elements [20, 87], $q = p$ and $\phi_k = N_k$, where N_k represents the shape functions used to construct the unknown solution u. The required derivatives $\frac{\partial \underline{x}^*}{\partial \underline{\xi}}$ are obtained by differentiating equation (106) with respect to $\underline{\xi}$

$$\frac{\partial x_i}{\partial \xi_j} = \sum_{k=1}^{n_q} b_i^{(k)} \frac{\partial \phi_k}{\partial \xi_j}. \tag{107}$$

The determination of the coefficients of interpolation, b_k, requires the solution of the following linear system of equations

$$\underbrace{\begin{bmatrix} a_{11} & \cdot & \cdot & a_{1n_q} \\ \cdot & \cdot & & \\ \cdot & & \cdot & \\ a_{n_q 1} & & & a_{n_q n_q} \end{bmatrix}}_{\mathbf{a}} \begin{Bmatrix} b_1 \\ \cdot \\ \cdot \\ b_{n_q} \end{Bmatrix} = \begin{Bmatrix} c_1 \\ \cdot \\ \cdot \\ c_{n_q} \end{Bmatrix} \tag{108}$$

where

$$\begin{aligned} a_{ij} &= \phi_j\left(\xi^{(i)}\right) \\ c_i &= \mathbf{x}\left(\xi^{(i)}\right) \end{aligned} \tag{109}$$

and $\{\xi^{(i)}\}, i = 1, \cdots n_q$, defines the set of interpolation points used, for example [18, 19]. For general ϕ_k, the cost of solving this linear system is [42, 36] $O(n_q^3)$ arithmetic operations. However, if the same interpolation sets are used then $\mathbf{a}^{-1}$ for sets corresponding to different orders can be precomputed and stored. In this case the coefficients can be computed using the matrix-vector product operation

$$b_i = \sum_{j=1}^{n_q} a_{ij}^{-1} c_j \tag{110}$$

costing $O(n_q)$ operations. Furthermore, when ϕ_k are Lagrangian polynomials then the property [67]

$$\phi_i\left(\underline{\xi}^{(j)}\right) = \delta_{ij} \tag{111}$$

where δ_{ij} defines the *kroneker delta*, results in $\mathbf{a}$ being an identity matrix and hence there is no explicit cost of equation solution involved. However, this is not true for hierarchic shape functions typically used in higher order finite element methods [89].

The accuracy requirements described in Section 5.2.1 still apply provided that there is no error in evaluating

$$\underline{x}\left(\underline{\xi}^{(i)}\right) \in \Gamma_G \ \forall \ \underline{\xi}^{(i)} \in \Gamma^e \sqsubset \Gamma_G. \tag{112}$$

to construct c_i. Use of the blended mapping from Section 6.1 meets this criteria.

6.2.2 Approximate Fitting Techniques

These methods, widely used in the geometric modelling community [15, 29], are based on constructing $\underline{x}^*(\underline{\xi})$ using *spline* or *Bezier* fitting using polynomial or rational forms of the *Bernstein* basis. Although very useful in geometric modelling applications because of their flexibility in approximating free form curves and surfaces [30], they are not suitable for element geometry approximation in a *p*-adaptive environment because:

1. The process of obtaining the *control vertices* that define the fitted shape involve the solution of multiple linear algebraic systems of size $O(n_q)$ [12] which need to be factored at runtime. This process is more expensive than polynomial interpolation where the inverse of the interpolation matrices can be precomputed for all needed orders.
2. Even though it costs more to construct than polynomial interpolation, it offers the same basic rate of error convergence with respect to q if Ω_G has piecewise smooth boundary [67].

6.3 Smoothness Of Blended Geometric Mapping On Simplex Elements

The linear blending functions for simplices are only C^0. To see this it suffices to examine the continuity of the rational terms from any one face and an edge. Consider for example the blending contribution of the face defined between vertices 2, 3 and 4 from equation (104) given by

$$x_i(\xi_j) = (1 - \xi_1) G_i\left(\xi_1', \xi_2'\right) \tag{113}$$

where

$$\xi_1^{'} = \frac{\xi_2}{\xi + \xi_3 + \xi_4} = \frac{\xi_2}{1 - \xi_1}$$
$$\xi_2^{'} = \frac{\xi_2}{\xi_2 + \xi_3 + \xi_4} = \frac{\xi_3}{1 - \xi_1}. \tag{114}$$

It can be shown via the geometrical construct in Figure 68(a) that $\xi_1^{'}$ and $\xi_2^{'}$ are not undefined at the vertex where $\xi_1 = 1$ but instead correspond to the projection to the centroid of the face implying $\xi_1^{'} = \xi_2^{'} = \frac{1}{3}$. Furthermore, from equation (113) it can be seen that at $\xi_1 = 1, G_i(\xi_1^{'}, \xi_2^{'})$ contributes nothing since the multiplying factor $1 - \xi_1$ is zero. Now consider the edge blend contribution from equation (104) for the edge between vertices 1 and 4 given by

$$x_i(\underline{\xi}) = (1 - \xi_2 - \xi_3) S_i\left(\underline{\xi}^{'}\right) \tag{115}$$

where the independent edge coordinate $\xi_1^{'} = \frac{\xi_1}{\xi_1 + \xi_2} = \frac{\xi_1}{1 - \xi_2 - \xi_3}$. Once again from Figure 68(b) it is clear that $\xi_2 = \xi_3 = \frac{1}{2}$ corresponds to the midpoint of the edge and hence $\xi_1^{'} = \frac{1}{2}$. Furthermore, at $\xi_2 = \xi_3 = \frac{1}{2}$ the multiplying term $1 - \xi_2 - \xi_3 = 0$, hence S_i contributes zero. This analysis does show that the linear blends are at least C^0. However, the analysis in section 5.2.1 is based on the assumption that $\mathbf{x} \in H^{p+1}$. This implies that for use in p-adaptive environments, continuity of higher order derivatives of the rational blend terms must also be examined.

To obtain $\frac{\partial x}{\partial \underline{\xi}}$ one must differentiate equation (113) to get

$$\frac{\partial x_i}{\partial \xi_j} = \frac{\partial (1 - \xi_1)}{\partial \xi_j} G_i + (1 - \xi_1) \frac{\partial G_i}{\partial \xi_k^{'}} \frac{\partial \xi_k^{'}}{\partial \xi_j}. \tag{116}$$

Expanding all the partial derivatives yields

$$\frac{\partial x_i}{\partial \xi_1} = -G_i + \left(\frac{1}{1 - \xi_1}\right) \left\{ \xi_2 \frac{\partial G_i}{\partial \xi_1^{'}} + \xi_3 \frac{\partial G_i}{\partial \xi_2^{'}} \right\}$$
$$\frac{\partial x_i}{\partial \xi_2} = \frac{\partial G_i}{\partial \xi_1^{'}} \tag{117}$$
$$\frac{\partial x_i}{\partial \xi_3} = \frac{\partial G_i}{\partial \xi_2^{'}}$$

where the component $\frac{\partial x_i}{\partial \xi_1}$ is unbounded when $\xi_1 = 1$. Similar analysis can be performed to show that the first derivative of the edge blend term from equation (115) is unbounded at $\xi_2 = \xi_3 = \frac{1}{2}$. This completes the proof that the linear rational blends are only C^0.

Blending schemes for simplex domains, analogous equation (104), but with higher order of smoothness can be constructed. One possible tetrahedral blend which is C^k is given by

$$\begin{aligned} x_i(\xi_j) = {} & (1 - \xi_1)^{k+1} G_i\left(\xi^{'}\right) + (1 - \xi_2)^{k+1} E_i\left(\xi^{'}\right) + (1 - \xi_3)^{k+1} F_i\left(\xi^{'}\right) + (1 - \xi_4)^{k+1} D_i\left(\xi^{'}\right) \\ & - (1 - \xi_1 - \xi_2)^{k+1} W_i\left(\xi^{'}\right) - (1 - \xi_1 - \xi_3)^{k+1} T_i\left(\xi^{'}\right) - (1 - \xi_1 - \xi_4)^{k+1} Q_i\left(\xi^{'}\right) \\ & - (1 - \xi_2 - \xi_3)^{k+1} S_i\left(\xi^{'}\right) - (1 - \xi_2 - \xi_4)^{k+1} R_i\left(\xi^{'}\right) - (1 - \xi_3 - \xi_4)^{k+1} P_i\left(\xi^{'}\right) \\ & + \xi_1^{k+1} x_i(1,0,0,0) + \xi_2^{k+1} x_i(0,1,0,0) + \xi_3^{k+1} x_i(0,0,1,0) + \xi_4^{k+1} x_i(,0,0,0,1). \end{aligned} \tag{118}$$

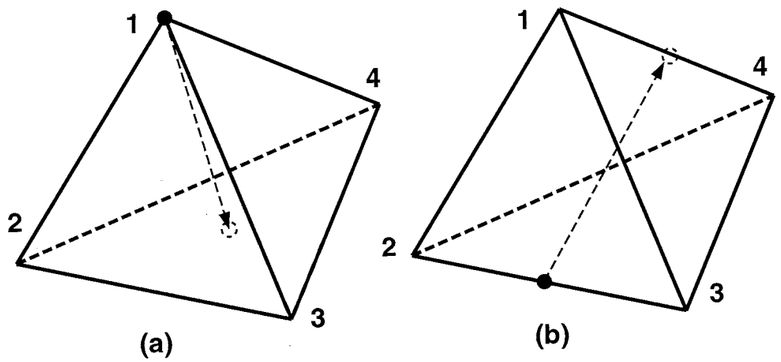


Figure 68. Coordinate projection for blending.

The face term from equation (113) now becomes

$$x_i(\xi) = (1-\xi_1)^{k+1} G_i\left(\xi'\right) \tag{119}$$

and the first partial derivatives are now given by

$$\frac{\partial x_i}{\partial \xi_j} = \frac{\partial (1-\xi_1)^{k+1}}{\partial \xi_j} G_i + (1-\xi_1)^{k+1} \frac{\partial G_i}{\partial \xi_k'} \frac{\partial \xi_k'}{\partial \xi_j}. \tag{120}$$

which on expanding yield

$$\begin{aligned}
\frac{\partial x_i}{\partial \xi_1} &= -(k+1)(1-\xi_1)^k G_i + (1-\xi_1)^{k+1} \left\{ \frac{\xi_2}{(1-\xi_1)^2} \frac{\partial G_i}{\partial \xi_1'} + \frac{\xi_3}{(1-\xi_1)^2} \frac{\partial G_i}{\partial \xi_2'} \right\} \\
\frac{\partial x_i}{\partial \xi_2} &= (1-\xi_1)^k \frac{\partial G_i}{\partial \xi_1'} \\
\frac{\partial x_i}{\partial \xi_3} &= (1-\xi_1)^k \frac{\partial G_i}{\partial \xi_2'}
\end{aligned} \tag{121}$$

where $\frac{\partial x_i}{\partial \xi_1}$ can be simplified to

$$\frac{\partial x_i}{\partial \xi_1} = -(k+1)(1-\xi_1)^k G_i + (1-\xi_1)^{k-1} \left\{ \xi_2 \frac{\partial G_i}{\partial \xi_1'} + \xi_3 \frac{\partial G_i}{\partial \xi_2'} \right\}. \tag{122}$$

$\frac{\partial x_i}{\partial \xi_1}$ is not unbounded at $\xi_1 = 1$ anymore. In fact all partial derivatives up to order k are now continuous because the term $\frac{1}{(1-\xi_1)^{k+1}}$ arising from the derivative of equation (114) is now cancelled by the blend term $(1-\xi_1)^{k+1}$. Similar analysis can be used to prove continuity of the edge blend terms up to k derivatives.

The fact that the blends are of limited smoothness implies that when used in p-adaptive finite element methods, the exponential convergence of the *discretization error* with increasing p will only hold up to some critical value of p even for problems that have smooth analytic solutions. The following numerical experiment is used to demonstrate two issues: (1) the loss of exponential

convergence with the use ofC^0 rational blends on tetrahedral meshes, and (2) a cost comparison of using geometric mapping described by equations (104) and (106) when the degenerated form of the$n \times n \times n$ tensor-product Gaussian integration is used to evaluate the element level integrals.

Consider the solution of *Poisson's* equation

$$\begin{aligned} -\bigtriangledown u(\underline{x}) &= f(\underline{x}) \;\; \forall \underline{x} \in \Omega_G \\ u(\underline{x}) &= 0 \;\; \forall \underline{x} \in \Gamma_G \end{aligned} \tag{123}$$

where Ω_G is a sphere of unit radius. $f(\underline{x})$ is specified such that the exact solution is analytic

$$\begin{aligned} u &= 1000(1-r)r^5 \\ r &= \left(x_1^2 + x_2^2 + x_3^2\right)^{\frac{1}{2}} \end{aligned} \tag{124}$$

with the analytic *energy norm* of the solution given by

$$\|u\|_e = \left\{ \int_{\Omega_G} \left(\frac{\partial u}{\partial \underline{x}}\right)\left(\frac{\partial u}{\partial \underline{x}}\right) d\Omega \right\}^{\frac{1}{2}} = \frac{2000}{143}\sqrt{858\pi}. \tag{125}$$

The problem was solved using a uniform p-enrichment scheme using p=1 to p=9. A coarse curvilinear mesh shown in Figure 69 consisting of 64 tetrahedral regions is used with $\underline{x}^*(\underline{\xi})$ constructed using the following two methods:

1. Blending scheme described in equation (104), and
2. Polynomial interpolation described in equation (106). The interpolation basis is the tetrahedral shape functions described in section 4 with the interpolation points given in [18, 19]. Although the analysis in section 5.2.1 requires interpolation order to be only p-1, a one time interpolation of order 8 was used to avoid having to repeat the solution of equation system (108) as p was increased from 1 to 9.

In both cases, the accuracy of numerical integration was based on the analysis in section 5.2.2. Figure 69 plots the relative error in the finite element solution in the energy norm given by [87]

$$\left\| u - u^{(h,p)} \right\|_e = \frac{\|u\|_e - \left\|u^{(h,p)}\right\|_e}{\|u\|_e} \tag{126}$$

versus p. Since the problem has smooth analytic solution, the logarithm of the error is expected to go down linearly with respect to p. It is observed that the solution error obtained using equation (104) fails to continue to converge exponentially past p=6. However, the solution error using equation (106) continues to converge exponentially past p=6. The continued convergence of the polynomial approximation is expected because the mapping defined by equation (106) isC^∞.

An explanation of why exponential convergence holds up to$p = 6$ when using C^0 rational blends with tetrahedral elements has not yet been determined. Issues related to the impact of using rational blends for elements with simplex entities in their closure on the convergence rate of the discretization error for a given problem does require further analysis. Issues related to

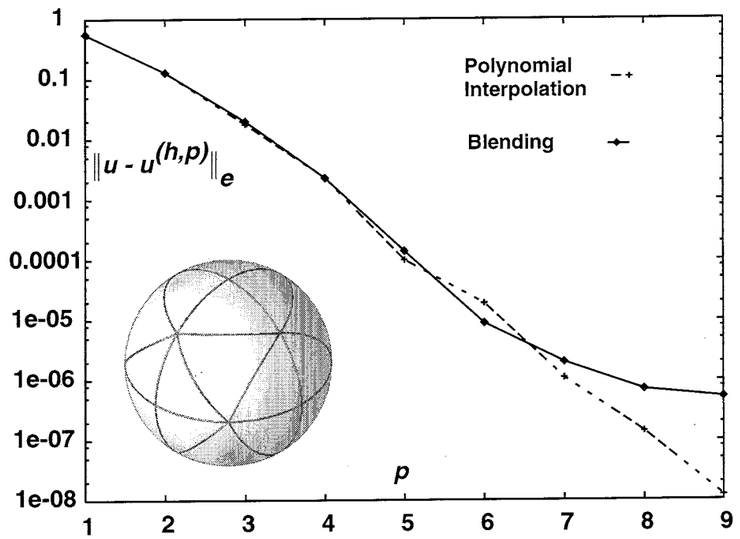


Figure 69. Relative error in energy norm for smooth problem.

use of higher order blends in higher order finite element computations which require further investigation are the effect of order of continuity of the higher order blends on the convergence rate of the *discretization error* for a given problem, and the cost of evaluating the higher order blends compared to doing a variable order polynomial interpolation for geometry as described by equation (106).

Figure 70 plots the time required to do the element level computations corresponding to the bilinear forms associated with the problem [44] $a(u,v)$ and (f,v). For degenerate tensor-product Gaussian integration, the element computation time grows at the same rate for geometry representation by direct blended mapping and polynomial interpolation of pointwise exact values. However, for $p \geq 4$, element level computations using polynomial approximation of geometry, as done in this experiment, were 1.25 times faster than those using blended mappings. The primary reason for this is that the approximation of the geometry was done once and the resulting coefficients b_k in equation (106) were stored for subsequent use. This means that pointwise queries of $\underline{x}(\underline{\zeta})$ from the geometric modeling system were only made once. In contrast, mapping using the blending scheme in equation (104) must query the modeler for pointwise $\underline{x}(\underline{\zeta})$ for each computation with a different p. For the present example, a profiling of the computation time for p=4 showed that modeler queries used about 9% of total computation time when polynomial approximation of geometry was used in comparison to 28% required using direct blended mapping. This example shows that the efficiency of the geometric modelling system in evaluating $\underline{x}(\underline{\zeta}), \frac{\partial \underline{x}}{\partial \underline{\zeta}}$ is an important factor in determining the efficiency of geometry representation during higher-order finite element computations. This example should not be used to infer that polynomial interpolation is more efficient than direct blended mapping because the

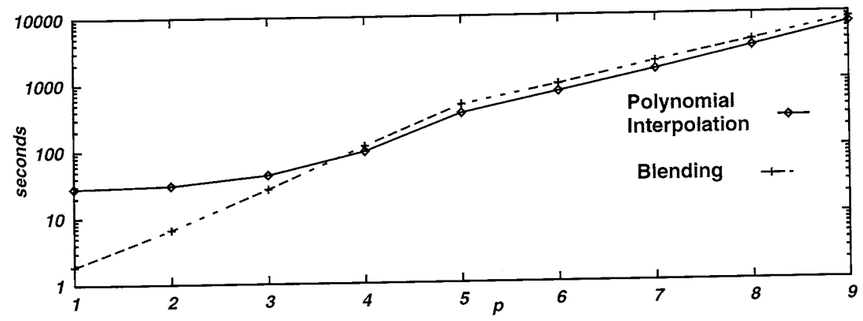


Figure 70. Elemental computation time using degenerate tensor-product Gaussian integration.

idea of storing the interpolation coefficient only works if there is no spatial adaptation of the mesh and furthermore, as is shown in a the next chapter, tensor-product Gaussian integration for simplex elements of general curved shapes is not efficient when compared to alternate schemes that directly approximate the integrand by polynomials followed by the use of precomputed integrals.

Having analyzed the accuracy requirements and available options for geometry representation in higher-order adaptive computations, the next chapter presents options for efficient element level computations incorporating accurate enough domain geometry information.

7. Effective and Efficient Element Level Integrations For Curved Domains

Element level computations involve integrating matrices that contribute to the left hand side and vectors that contribute to the right hand side of the global system of equations. When using polynomial shape functions, the integrals from equation (80) can be decomposed as

$$\begin{aligned} &\int_{\Omega^e} \Theta_{\Omega^e}(\xi)\Upsilon_{\Omega^e}(\xi)d\xi \\ &\int_{\Gamma^e} \Theta_{\Gamma^e}(\xi)\Upsilon_{\Gamma^e}(\xi)d\xi \end{aligned} \tag{127}$$

where, $\Theta_{\Omega^e}(\xi)$ and $\Theta_{\Gamma^e}(\xi)$ represent the *polynomial* parts of the integrand that can be easily exactly integrated, and $\Upsilon_{\Omega^e}(\xi)$ and $\Upsilon_{\Gamma^e}(\xi)$ represent the *non-polynomial* parts that either cannot be integrated exactly, or requires complex integration processes that are not efficient to implement in a computer code. In the discussion that follows, the *polynomial* and the *non-polynomial* parts will be denoted simply by $\Theta(\xi)$ and $\Upsilon(\xi)$, respectively, and the subscripts will be used only when necessary. The exact forms of $\Theta(\xi)$ and $\Upsilon(\xi)$ depend upon the partial differential equation(s) at hand and their associated coefficients, the choice of *weak* form used to arrive at the element level formulation [44], and the form of element geometry mapping used. As examples, decompositions of "stiffness" and "mass" type matrices and element level vectors contributing to the right hand side follow.

Stiffness Matrix: Consider the computation of a typical element level "stiffness" type term [44] given by (repeated indices imply summation)

$$K^e_{ij} = \int_{\Omega^e} \frac{\partial N_i}{\partial x_k} c(\mathbf{x}) \frac{\partial N_j}{\partial x_k} d\mathbf{x} \tag{128}$$

where, N_i and N_j represent the i-th and j-th shape functions, respectively, and $c(\mathbf{x})$ represents the coefficient associated with the constitutive equations governing the physics associated with the specific problem, for example, elastic modulus for elasticity problems and spatial conductivity in a heat transfer problem. Using the geometric mapping, $\mathbf{x}(\xi)$, described in Section 6, equation (128) can be rewritten as

$$K^e_{ij} = \int_{\Omega^e} \Theta(\xi)\Upsilon(\xi)d\xi \tag{129}$$

where,

$$\Theta(\xi) = \frac{\partial N_i}{\partial \xi_l} \frac{\partial N_j}{\partial \xi_m} \tag{130}$$

and

$$\Upsilon(\xi) = c(\mathbf{x}(\xi)) \frac{\partial \xi_l}{\partial x_k} \frac{\partial \xi_m}{\partial x_k} J(\xi) \tag{131}$$

with $\frac{\partial \xi_l}{\partial x_k}$ denoting the derivative of the inverse mapping $\xi_l(x_k)$. Since $\xi_l(x_k)$ is never explicitly constructed, $\frac{\partial \xi_l}{\partial x_k}$ is obtained as the inverse of the jacobi matrix of mapping given by $\frac{\partial \xi_l}{\partial x_k} = \left[\frac{\partial x_i}{\partial \xi_j}\right]^{-1}$. The Jacobian of the geometric mapping is given by $J(\xi) = \left|\frac{\partial x_i}{\partial \xi_j}\right|$. Since N_i and N_j are polynomial, $\Theta(\xi)$ is polynomial and hence easily exactly integrated. If $\mathbf{x}(\xi)$ is polynomial then $J(\xi)$ is polynomial. However, if $\mathbf{x}(\xi)$ uses rational blends then $J(\xi)$ is non-polynomial. Also, $\frac{\partial \xi_l}{\partial x_k}$ is not in general polynomial for $\mathbf{x}(\xi)$ of order greater than one. Furthermore, $c(\mathbf{x}(\xi))$ can also be non-polynomial. Therefore, for general problems using general curvilinear mappings, $\Upsilon(\xi)$ is not easily exactly integrable.

Mass Matrix: Consider a typical "mass" matrix type term [44] given by

$$M_{ij}^e = \int_{\Omega^e} N_i \rho(\mathbf{x}) N_j d\Omega \tag{132}$$

where $\rho(\mathbf{x})$ denotes relevant material data, for example, mass density of the domain. Using the mapping function, equation (132) can be rewritten as

$$M_{ij}^e = \int_{\Omega^e} \Theta(\xi)\Upsilon(\xi) d\xi \tag{133}$$

where

$$\Theta(\xi) = N_i N_j \tag{134}$$

and

$$\Upsilon(\xi) = \rho(\mathbf{x}(\xi))J(\xi). \tag{135}$$

$\Theta(\xi)$ is a polynomial and exactly integrable with polynomial shape functions. In this case Υ cannot be easily exactly integrated if either $J(\xi)$ or $\rho(\mathbf{x}(\xi))$ cannot be easily integrated.

Right Hand Side Vector from Ω^e: Consider a typical element level force vector term [44] given by

$$f_i^e = \int_{\Omega^e} N_i f(\mathbf{x}) d\Omega \tag{136}$$

Using the mapping function, equation (136) can be rewritten as

$$f_i^e = \int_{\Omega^e} \Theta(\xi)\Upsilon(\xi) d\xi \tag{137}$$

with

$$\Theta(\xi) = N_i \tag{138}$$

and

$$\Upsilon(\xi) = f(\mathbf{x}(\xi))J(\xi). \tag{139}$$

Υ is not easily exactly integrable if either $J(\xi)$ or $f(\mathbf{x}(\xi))$ is not easily integrated.

Right Hand Side Vector from Γ^e: For natural boundary conditions, $h_n(\mathbf{x})$, the vector contributing to the right hand side of the global system is given by [44]

$$f_i^e = \int_{\Gamma^e} N_i h_n(\mathbf{x}) d\Gamma \tag{140}$$

where Γ^e is the portion of the element boundary on which h_n is prescribed. Using $\mathbf{x}(\xi)$, equation (140) can be Γ^e rewritten as

$$f_i^e = \int_{\Gamma^e} \Theta(\xi) \Upsilon(\xi) d\xi \tag{141}$$

where

$$\Theta(\xi) = N_i \tag{142}$$

and

$$\Upsilon(\xi) = h_n(\mathbf{x}(\xi)) J(\xi). \tag{143}$$

Υ is not easily exactly integrable if either $J(\xi)$ or $h_n(\mathbf{x}(\xi))$ is not easily integrated.

Having described the various types of integrals that need to be computed at the element level, the next section outlines the possible options to numerically evaluate the integrals and identifies a general and efficient option for high p-order curvilinear elements. Since the focus of the thesis is on problems in three-dimensions, the discussion of specific element integration schemes is restricted to three-dimensional element topologies.

7.1 Evaluation of Elemental Integration Options

In three dimensions, the integral in equation (127) can be written in terms of the independent element parametric coordinate system as

$$I = \int_{\Omega^e} \Theta(\xi_i) \Upsilon(\xi_i) d\xi \tag{144}$$

The following observations, pertinent for a p-adaptive computation, will be used in the determination of applicable elemental integration schemes:

1. Recall from the discussion in section 5.2 that equation (144) can be computed approximately as long as the error in integration does not destroy the optimal convergence rate of the finite element *discretization error*.
2. If geometry is approximated, $x_i(\xi_j) \approx x_i^*(\xi_j)$, as defined in section 6.2.1 then for a p-adaptive computation $x_i^*(\xi_j)$ must also be adapted to order $q = p-1$. Analysis in section 6.2.1 shows that minimum cost of doing this operation is $O(n_q)$ where $n_q = O(p^3)$ from table 17.
3. If geometry is represented by blending the exact shape of Γ_G as described in section 6.1 then there is no need to reconstruct $x_i^*(\xi_j)$ every time p is changed. This implies that unlike explicit polynomial interpolation, the cost associated with blended mapping remains fixed

with respect to p. Recall however, that the blends must be sufficiently smooth and that the numerical experiment in Section 6.3 showed that the efficiency of pointwise evaluation of $\underline{x}(\zeta)$ and $\frac{\partial \underline{x}}{\partial \zeta}$ within the geometric modeling system plays a crucial role in the efficiency of the overall integration process.

Both direct blending or polynomial approximation, are viable options for geometry representation for use in element level integrations. When used with integration schemes that approximate whole or parts of the integrands, the direct use of the scheme described in Section 6.1 appears more suitable because it requires the solution of only one approximation problem at the element level.

Specific implementations of the basic integral approximation operators described in Section 5.1, using geometric mapping described in Section 6.1, which either approximate the entire integral, denoted by $\int\limits_{\Omega^e}^{*_1}$ in Section 5.1, or approximate the integrand into an easily exactly integrable form, denoted by $\int\limits_{\Omega^e}^{*_3}$ in Section 5.1, and are of interest for high order integrands arising in a p-adaptive computation can be described as:

1. Implementation of $\int\limits_{\Omega^e}^{*_1}$ using:
   a. *Gaussian integration* [1],
   b. *Sum factorization* [27, 83], or
   c. *Vector quadrature* [43].
2. Implementation of $\int\limits_{\Omega^e}^{*_3}$ using *precomputed integral* tables [72].

For a d-dimensional element with maximum polynomial degree of shape functions given by p, there are $O(p^d)$ total shape functions. This leads to $O(p^{2d})$ terms in elemental matrices K^e and M^e, and $O(p^d)$ terms in the elemental right hand side vectors f^e. The sparsity of elemental matrices and hence the number of total terms to be evaluated for each element level matrix and vector, depends upon the orthogonality of the chosen basis functions [5, 16]. Although the use of specially chosen points for numerical integration can change the sparsity pattern of numerically evaluated elemental matrices and vectors, the use of standard *Gaussian* integration rules [1] does not alter it significantly. Assuming that the same integration scheme is used for all terms in a matrix (vector), the computational cost of the integration schemes can be estimated by analyzing the cost of computing the term in the elemental matrices and vectors that has the highest polynomial order. To be equitable, the cost analysis will be based on approximating the number of function evaluations, n_f, of the integrand and the number of arithmetic operations (addition and multiplication), n_o, required with each integration scheme for K^e. Similar analysis can be carried out for terms resulting from M^e and f^e.

If the maximum degree of complete polynomial in the shape functions equals p, then Θ is a polynomial of degree $2(p-\mu)$ with $\mu = 0$ for M^e and $\mu = 1$ for K^e. In addition, when Υ is not a polynomial, it is assumed to be approximated as a degree $q = p - 1 = O(p)$ polynomial.

Of course, the specific method of approximation used will vary based on the choice of the approximate integration scheme selected.

7.1.1 Gaussian Integration

In three dimensions, integration over hexahedral element topologies can be done by using the tensor product form of the one-dimensional Gaussian integration schemes [1]; thus, the integral of the form in equation (144) can be approximated as

$$I \approx \sum_{i=1}^{n_g}\sum_{j=1}^{n_g}\sum_{k=1}^{n_g} \Theta\left(\xi_1^{(i)}, \xi_2^{(j)}, \xi_3^{(k)}\right) \Upsilon\left(\xi_1^{(i)}, \xi_2^{(j)}, \xi_3^{(k)}\right) w^{(i)} w^{(j)} w^{(k)} \tag{145}$$

where, $\{\xi^{(i)}\}$, $\{w^{(i)}\}$ define points and weights, respectively, for a one-dimensional Gaussian integration rule with n_g points. Based on equation (97), the optimal number of integration points for K_{ij}^e using *Gauss-Legendre* scheme ($\beta = 1$) is given by

$$n_g = \left\lceil \frac{3p-2}{2} \right\rceil \approx O(p). \tag{146}$$

For tetrahedral elements, symmetric integration rules using barycentric coordinates only exist for low to moderate order accuracies [45]. For high order integrands, the tensor product Gauss-Legendre rule can be applied by transforming the independent barycentric coordinates, $\{\xi_i : \xi_i \in [0,1], \xi_1 + \xi_2 + \xi_3 \leq 1\}$, into orthogonal parametric coordinate system, $\{\xi_i' : \xi_i' \in [-1,1]\}$ as shown in Figure (71). The transformation is defined by

$$\begin{aligned} \xi_1 &= \frac{1}{8}\left(1+\xi_1'\right)\left(1-\xi_2'\right)\left(1-\xi_3'\right) \\ \xi_2 &= \frac{1}{4}\left(1+\xi_2'\right)\left(1-\xi_3'\right) \\ \xi_3 &= \frac{1}{2}\left(1+\xi_3'\right) \end{aligned} \tag{147}$$

and the corresponding Jacobian is given by

$$\left|\frac{\partial \xi_i}{\partial \xi_j'}\right| = det \begin{bmatrix} \frac{1}{8}(1-\xi_2')(1-\xi_3') & 0 & 0 \\ -\frac{1}{8}(1+\xi_1')(1-\xi_3') & \frac{1}{4}(1-\xi_3') & 0 \\ -\frac{1}{8}(1+\xi_1')(1-\xi_2') & -\frac{1}{4}(1-\xi_2') & \frac{1}{2} \end{bmatrix} = \frac{1}{64}\left(1-\xi_2'\right)\left(1-\xi_3'\right)^2. \tag{148}$$

The element level integral written in barycentric coordinate system given by

$$I = \int_{\Omega^e} \Theta(\xi)\Upsilon(\xi)d\xi = \int_0^1 \int_0^{1-\xi_3} \int_0^{1-\xi_2-\xi_3} \Theta(\xi_1,\xi_2,\xi_3)\Upsilon(\xi_1,\xi_2,\xi_3)d\xi_1 d\xi_2 d\xi_3 \tag{149}$$

can be rewritten in terms of orthogonal parametric coordinates as

$$I = \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} I' d\xi_1' d\xi_2' d\xi_3' \tag{150}$$

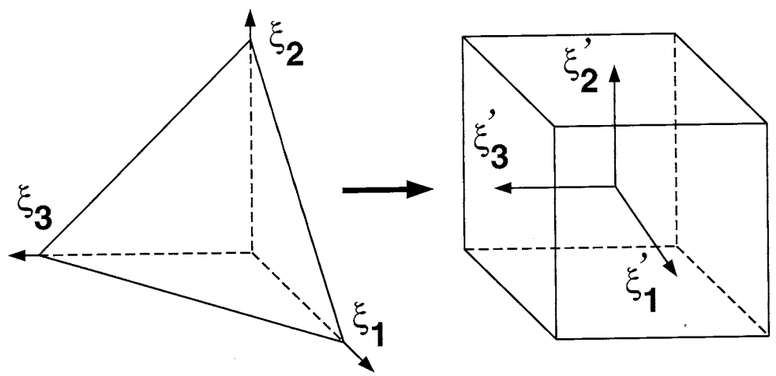


Figure 71. Coordinate transform for tetrahedron.

where, $I^{'} = \Theta(\xi_1, \xi_2, \xi_3)\Upsilon(\xi_1, \xi_2, \xi_3)\left|\frac{\partial \xi_i}{\partial \xi_j^{'}}\right|$ with ξ_i and $\left|\frac{\partial \xi_i}{\partial \xi_j^{'}}\right|$ given by equations (147) and (148), respectively. Equation (150) can now be evaluated with a tensor product rule similar to the hexahedral element. However, since $\left|\frac{\partial \xi_i}{\partial \xi_j^{'}}\right|$ is a *3rd* degree polynomial, the number of points needed for K_{ij}^{e} using *Gauss-Legendre* rule is now given by

$$n_g = \left\lceil \frac{3p+1}{2} \right\rceil \approx O(p). \tag{151}$$

7.1.1.1 Computational Complexity of Tensor Product *Gauss-Legendre* Scheme From a computational viewpoint, tensor product Gaussian integration in n dimensional domain requires $O\left(n_g^n\right)$ integrand evaluations. Based on equation (145), the number of function evaluations for a three-dimensional element using the tensor product Gaussian rule is given by

$$n_f = O\left(n_g^3\right). \tag{152}$$

Using equation (151) results in

$$n_f = O\left(p^3\right) \tag{153}$$

complexity. Equation (145) also implies that the number of arithmetic operations is given by

$$n_o = O\left(n_g^3\right) \tag{154}$$

which is $O\left(p^3\right)$ in complexity.

7.1.2 Precomputed Integrals

If the entire integrand in equation (144) can be reduced to a polynomial form then the necessary integrals can be precomputed exactly and stored. This idea has already been exploited for efficient finite element computations for specific problems. It has been used with hierarchic

finite elements for quadratic functional on two-dimensional planar domains [72]. Precomputed matrices for triangular finite elements for second order elliptic partial differential equations in planar domains are derived in [85]. Similar ideas have been proposed that use symbolic computation to efficiently solve specific problems [57, 56, 95]. In general curvilinear domains, the integrands are not polynomials and hence the idea of precomputed integral tables cannot be applied directly. This section generalizes the technique to general three-dimensional curvilinear finite elements based on ideas developed in reference [64].

Recall that the portion of the integrand in equation (144) that cannot be exactly integrated is represented by Υ. Therefore, if Υ is approximated by a order $q = p - 1$ polynomial basis as

$$\Upsilon \approx \sum_{i=1}^{n_q} a_i \phi_i \tag{155}$$

where, $n_q \approx O(p^n)$ is given by table 17 for a n-dimensional domain. The resulting integrand is given by

$$I \approx \int_{\Omega^e} \Theta \sum_{i=1}^{n_q} a_i \phi_i d\Omega = \sum_{i=1}^{n_q} a_i \underbrace{\int_{\Omega^e} \Theta \phi_i d\Omega}_{precomputed} \tag{156}$$

where the set of integrals over the braces can be exactly precomputed as

$$\varrho_i = \int_{\Omega^e} \Theta \phi_i d\Omega. \tag{157}$$

7.1.2.1 Computational Complexity of Precomputed Integrals Recalling that Υ is given by equations (131), (135), and (139), one can observe that terms resulting from Υ need to be interpolated only once over the domain of the integration. The interpolation problem is similar to that described in equation (108) with the right hand side vector now given by

$$c_i = \Upsilon\left(\xi^{(i)}\right), i = 1, \cdots, n_q \tag{158}$$

with n_q given by table 17 using $q = p - 1$. As explained in Section 106, the number of function evaluations of Υ in setting up the interpolation problem is $n_f = O(n_q) \approx O\left(p^3\right)$. In addition, the solution of the resulting system of $O(n_q)$ linear equations requires $O\left(n_q^3\right) \approx O\left(p^6\right)$ arithmetic operations [36, 42]. However, the cost of solving the linear system can be reduced to that of a matrix-vector multiplication if the inverse of the interpolation matrices for various interpolation orders are precomputed and stored. The reduced cost is given by $O(n_q) \approx O\left(p^3\right)$ arithmetic operations.

Unlike Gaussian integration, there are no direct function evaluations of Υ required during the computation of individual entries. The only cost of evaluating individual terms during element level computation comes from the cost of

$$n_o = O(n_q) \approx O\left(p^3\right) \tag{159}$$

arithmetic operations in summing the terms $a_i \varrho_i$ in equation (156).

7.1.3 Sum Factorization

If $\xi_i \in [-1,1], i = 1,2,3$ denote the independent parametric coordinate components, then higher dimensional integrals with integrands in tensor product form

$$I(\xi_1,\xi_2,\xi_3) = I_1(\xi_1)I_2(\xi_2)I_3(\xi_3) \tag{160}$$

can be evaluated as a product of one-dimensional integrals as follows [27]

$$\int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} I_1(\xi_1)I_2(\xi_2)I_3(\xi_3)d\xi_1 d\xi_2 d\xi_3 = \int_{-1}^{1} I_1 d\xi_1 \int_{-1}^{1} I_2 d\xi_2 \int_{-1}^{1} I_3 d\xi_3. \tag{161}$$

Since each of the one-dimensional integrals require $O(n)$ integrand evaluations using a n point rule, the entire integral can be computed using $O(n)$ integrand evaluations instead of $O(n^3)$ integrand evaluations required using the $n \times n \times n$ tensor product rule. This property is referred to as *sum factorization* and has been used to speed up computations in spectral methods [27].

Sum factorization can be used to evaluate element level integrals in finite element methods provided Θ and Υ are in tensor product form. If the shape functions are written in tensor product form

$$N_i(\xi_1,\xi_2,\xi_3) = N_i^{(1)}(\xi_1)N_i^{(2)}(\xi_2)N_i^{(3)}(\xi_3) \tag{162}$$

then Θ will naturally be in tensor product form given by

$$\Theta = \Theta^{(1)}(\xi_1)\Theta^{(2)}(\xi_2)\Theta^{(3)}(\xi_3). \tag{163}$$

For quadrilateral and brick and element topologies, the natural parametric system admits tensor product shape functions as described in Section 4 and literature [89]. Shape functions for pyramid element topology derived in Section 4 are also tensor product. However, for triangle, tetrahedral and wedge element topologies, the natural parametric coordinate system does not admit tensor product basis functions. Tensor product basis functions over triangular domains using a degenerate form of the parametric space of a unit square was developed in reference [27]. This technique was extended in reference [83, 84] to define hierarchic C^0 tensor product shape functions over tetrahedral domains. If Θ is not in tensor product form, then to use sum factorization, it needs to be approximated as an expansion in terms of tensor product basis functions as follows

$$\Theta \approx \sum_{i=1}^{n_p} a_i N_i^{(1)}(\xi_1)N_i^{(2)}(\xi_2)N_i^{(3)}(\xi_3) \tag{164}$$

where $n_p = O(p^3)$.

Even if the basis functions are written in tensor product form, Υ, in general, is not in tensor product form. This is because for curved domains, $\frac{\partial \xi_i}{\partial x_j}$ and $J(\xi)$ are not tensor product. In addition, terms defining material data for general problems are not in general tensor product.

Therefore, to be able to apply sum factorization to element level integrals for problems in curved domains, one must first approximate Υ in terms of a tensor product basis as

$$\Upsilon \approx \sum_{i=1}^{n_q} a_i N_i^{(1)}(\xi_1) N_i^{(2)}(\xi_2) N_i^{(3)}(\xi_3). \tag{165}$$

where $n_q = O(p^3)$. When Θ is naturally in tensor product form, the scheme can be realized by substituting equations (163) and (165) into equation (161) to obtain

$$\begin{aligned} I &\approx \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} \Theta^{(1)}(\xi_1)\Theta^{(2)}(\xi_2)\Theta^{(3)}(\xi_3) \sum_{i=1}^{n_q} a_i N_i^{(1)}(\xi_1) N_i^{(2)}(\xi_2) N_i^{(3)}(\xi_3) d\xi_1 d\xi_2 d\xi_3 \\ &\approx \sum_{i=1}^{n_q} a_i \int_{-1}^{1} \Theta^{(1)} N_i^{(1)} d\xi_1 \int_{-1}^{1} \Theta^{(2)} N_i^{(2)} d\xi_2 \int_{-1}^{1} \Theta^{(3)} N_i^{(3)} d\xi_3. \end{aligned} \tag{166}$$

7.1.3.1 Computational Complexity of Sum Factorization The cost of the *sum factorization* method can be broken down into two parts:

1. Cost of interpolating Υ in terms of tensor product basis functions once per element. The primary computational cost in this case is from the integrand function evaluations to form the interpolation problem.
2. Cost of doing the resulting one-dimensional integrals.

Assuming an interpolatory approximation, the number of function evaluations of the original integrand is $O(p^n)$ for a *n*-dimensional domain [67]. Therefore, it will require $O(p^3)$ function evaluations in three dimensions. In addition, the solution of the interpolation problem requires solution of a linear system of $O(p^3)$ equations costing $O(p^6)$ arithmetic operations if the interpolation matrix is factored at runtime [36, 42]. However, the arithmetic operation involved in solving the interpolation problem can be reduced to $O(p^3)$ if the inverse of the interpolation matrices is precomputed and stored for all the required interpolation orders.

Once the element level interpolation is done, the cost of evaluating individual terms in the elemental matrices is given by the cost of numerical evaluation of equation (166). Notice that the order of the integrands in the one-dimensional integrals is *p+q*, which requires $O(p + O(p)) \approx O(p)$ function evaluations and $O(p + O(p)) \approx O(p)$ arithmetic operations. However, since the integrands are different for each summation index, the function evaluations need to be repeated $O(n_q) \approx O(p^3)$ times if implemented in the form described by equation (166). Although not explored in [83, 84], the *sum factorization* can be restated in a form identical to equation (156) which is more efficient than that described by equation (166). If the product of the one-dimensional integrals in equation (166) are precomputed and stored as

$$\omega_i = \underbrace{\int_{-1}^{1} \Theta^{(1)} N_i^{(1)} d\xi_1 \int_{-1}^{1} \Theta^{(2)} N_i^{(2)} d\xi_2 \int_{-1}^{1} \Theta^{(3)} N_i^{(3)} d\xi_3}_{precompute} \tag{167}$$

then equation (166) is reduced to

$$I \approx \sum_{i=1}^{N_q} a_i \omega_i \tag{168}$$

which is identical to equation (156) and is more efficient than equation (166) because it eliminates the runtime cost of function evaluations associated with doing the three one-dimensional integrals. The cost of arithmetic operations in evaluating the modified form of equation (168) is now identical to that for equation (156) given by $n_o = O(p^3)$.

7.1.4 Vector Quadrature

The idea of *vector quadrature* was first proposed in [43] as an efficient alternative to tensor product Gauss Quadrature for p-version brick elements. Consider the expansion of Θ and Υ given by

$$\begin{aligned} \Theta &\approx \sum_{i=1}^{n_v} a_i \varphi_i \\ \Upsilon &\approx \sum_{i=1}^{n_v} b_i \varphi_i \end{aligned} \tag{169}$$

where the expansion basis $\{\varphi_i\}$ satisfy the following orthogonality relation

$$\int_{\Omega^e} \varphi_i \varphi_j d\Omega = \delta_{ij}. \tag{170}$$

The number of basis functions n_v is $O(p^3)$ in three-dimensions. Substituting equation (169) into equation (144) leads to

$$I \approx \int_{\Omega^e} \left\{ \sum_{i=1}^{n_v} a_i \varphi_i \right\} \left\{ \sum_{j=1}^{n_v} b_i \varphi_i \right\} d\Omega. \tag{171}$$

Extracting the summation outside the integration gives

$$I \approx \sum_{i=1}^{n_v} \sum_{j=0}^{n_v} a_i b_j \int_{\Omega^e} \varphi_i \varphi_j d\Omega \tag{172}$$

which can be further simplified using equation (170) to yield

$$I \approx \sum_{i=1}^{n_v} a_i b_i = \mathbf{a} \odot \mathbf{b} \tag{173}$$

Equation (173) represents a dot product of the coefficient vectors, $\mathbf{a}$ and $\mathbf{b}$ obtained from the approximation of Θ and Υ, respectively.

Since Θ and Υ can differ significantly in the polynomial order, efficient implementation requires that the integrand decomposition be symmetric in polynomial order. Reference [31] addresses these issues in describing a *symmetric dot product quadrature* scheme for p-version shell elements.

7.1.4.1 Computational Complexity of Vector Quadrature Recall that the highest polynomial degree of the element level integrands is $O(p + O(p)) \approx O(p)$. Assuming a symmetric decomposition of the integrand [31], order of each approximation will be $O\left(\frac{O(p)}{2}\right) \approx O(p)$. The number of function evaluations for each term will therefore be given by

$$n_f = O(n_v) \approx O\left(p^3\right). \tag{174}$$

Similarly, the number of arithmetic operations needed to carry out the dot product in equation (173) will also require

$$n_o = O(n_v) \approx O\left(p^3\right). \tag{175}$$

7.2 Elemental Integration Scheme For General Curvilinear Problems

Based on the foregoing analysis of each integration option, the following remarks can be made:

1. The schemes using *precomputed integrals* and *sum factorization* are identical in computational complexity in light of equations (156) and (168).
2. Asymptotically all the schemes described have the same cost $O\left(p^3\right)$.
3. *Precomputed integrals*, *sum factorization* and *vector quadrature* all require approximation of the entire or part of the integrand. However, for general problems on curvilinear domains of arbitrary shape, the scheme described by equation (156) offers a general and efficient solution because unlike *sum factorization* or *vector quadrature* it does not require the approximating basis to satisfy specific properties of orthogonality or decomposition into a tensor product form.

7.3 Derivation of Precomputed Integrals

This section derives the precomputed table entries defined by equation (157) that are required for K^e, M^e, and f^e using the mapping described in section 6.1. Terms from Υ that must be approximated over the domain of the element for each type of elemental matrix/vector computation are also derived.

7.3.1 Precomputed Integrals for K^e

Equation (128) can be expanded as

$$\begin{aligned} K_{ij}^e &= \int_{\Omega^e} c(\mathbf{x}) \left(\frac{\partial N_i}{\partial \mathbf{x}}\right)^T \left(\frac{\partial N_j}{\partial \mathbf{x}}\right) d\Omega \\ &= \int_{\Omega^e} c(\mathbf{x}(\xi)) \left(\frac{\partial N_i}{\partial \xi}\frac{\partial \xi}{\partial \mathbf{x}}\right)^T \left(\frac{\partial N_j}{\partial \xi}\frac{\partial \xi}{\partial \mathbf{x}}\right) J(\xi) d\xi \\ &= \int_{\Omega^e} \left\{ \frac{\partial N_i}{\partial \xi_1}\frac{\partial N_j}{\partial \xi_1} A_{11} + \frac{\partial N_i}{\partial \xi_1}\frac{\partial N_j}{\partial \xi_2} A_{12} + \ldots + \frac{\partial N_i}{\partial \xi_3}\frac{\partial N_j}{\partial \xi_3} A_{33} \right\} d\xi \end{aligned} \tag{176}$$

with

$$A_{lm}(\xi) = \sum_{k=1}^{3} \frac{\partial \xi_k}{\partial x_l} \frac{\partial \xi_k}{\partial x_m} c(\mathbf{x}(\xi)) J(\xi);\ l, m = 1 \ldots 3, \tag{177}$$

where, $\frac{\partial \xi_k}{\partial x_l}$ and $J(\xi)$ define the derivative of the inverse mapping $\xi_j(x_i)$ and the Jacobian of the geometric mapping $x_i(\xi_j)$, respectively. Equation (176), can be written more concisely using equation (177) as

$$K_{ij}^{e} = \int_{\Omega^e} \left\{ \sum_{l=1}^{3} \sum_{m=1}^{3} A_{lm} \frac{\partial N_i}{\partial \xi_l} \frac{\partial N_j}{\partial \xi_m} \right\} d\xi. \tag{178}$$

$A_{lm}(\xi)$ is not easily exactly integrable and needs to be interpolated using polynomial basis functions. Since finite element shape functions, N_i, are already defined, they provide a convenient basis for interpolating A_{lm}. Using interpolation basis of order $q = p - 1$ gives

$$A_{lm}(\xi) \approx \sum_{n=1}^{n_q} a_{lmn} N_n \tag{179}$$

with n_q given in table 17. As explained in section 5.2.1, the interpolation set used in reference [18, 19] ensures exponential convergence of the interpolation error when A_{lm} is smooth. Substituting equation (179) into equation (178) yields

$$\begin{aligned} K_{ij}^{e} &= \int_{\Omega^e} \sum_{n=1}^{n_q} \sum_{l=1}^{3} \sum_{m=1}^{3} a_{lmn} \frac{\partial N_i}{\partial \xi_l} \frac{\partial N_j}{\partial \xi_m} N_n d\xi \\ &= \sum_{n=1}^{n_q} \sum_{l=1}^{3} \sum_{m=1}^{3} a_{lmn} \underbrace{\int_{\Omega^e} \frac{\partial N_i}{\partial \xi_l} \frac{\partial N_j}{\partial \xi_m} N_n d\xi}_{\varrho_{ijlmn}^{K}} \end{aligned} \tag{180}$$

where ϱ_{ijlmn}^{K} define the entries in the precomputed integral table needed for computing terms of K^e at runtime as follows

$$K_{ij}^{e} = \sum_{n=1}^{N_q} \sum_{l=1}^{3} \sum_{m=1}^{3} \varrho_{ijlmn}^{K} a_{lmn}. \tag{181}$$

7.3.2 Precomputed Integrals for M^e

Equation (132) can be expanded as

$$M_{ij}^{e} = \int_{\Omega^e} \rho(\mathbf{x}) N_i N_j d\Omega = \int_{\Omega^e} B(\xi) N_i N_j d\xi \tag{182}$$

with

$$B(\xi) = \rho(\mathbf{x}(\xi)) J(\xi) \tag{183}$$

where, $J(\xi)$ is the Jacobian of the geometric mapping $\mathbf{x}(\xi)$. Using order $q = p - 1$ polynomial interpolation of B gives

$$B \approx \sum_{n=1}^{n_q} b_n N_n. \tag{184}$$

where n_q is given by table 17. Substituting equation (184) into equation (182) yields

$$M_{ij}^e = \int_{\Omega^e} \sum_{n=1}^{n_q} b_n N_i N_j N_n d\xi = \sum_{n=1}^{n_q} b_n \underbrace{\int_{\Omega^e} N_i N_j N_n d\xi}_{\varrho_{ijn}^M} \tag{185}$$

where ϱ_{ijn}^M define the entries in the precomputed integral table needed for computing terms of M^e at runtime as follows

$$M_{ij}^e = \sum_{n=1}^{n_q} \varrho_{ijn}^M b_n. \tag{186}$$

7.3.3 Precomputed Integrals for f^e from Ω^e

Equation (136) can be expanded as

$$f_i^e = \int_{\Omega^e} f(\mathbf{x}) N_i d\Omega = \int_{\Omega^e} f(\mathbf{x}(\xi)) N_i d\xi \tag{187}$$

with

$$D(\xi) = f(\mathbf{x}(\xi)) J(\xi) \tag{188}$$

where, $J(\xi)$ is the Jacobian of the geometric mapping $\mathbf{x}(\xi)$. Using order $q = p - 1$ polynomial interpolation of D gives

$$D \approx \sum_{n=1}^{n_q} d_n N_n. \tag{189}$$

where n_q is given by table 17. Substituting equation (189) into equation (187) yields

$$f_i^e = \int_{\Omega^e} \sum_{n=1}^{n_q} d_n N_i N_n d\xi = \sum_{n=1}^{n_q} d_n \underbrace{\int_{\Omega^e} N_i N_n d\xi}_{\varrho_{in}^{f\Omega}} \tag{190}$$

where $\varrho_{in}^{f\Omega}$ define the entries in the precomputed integral table needed for computing terms of f^e at runtime as follows

$$f_i^e = \sum_{n=1}^{n_q} \varrho_{in}^{f\Omega} d_n. \tag{191}$$

7.3.4 Precomputed Integrals for f^e from Γ^e

Equation (76) can be expanded as

$$f_i^e = \int_{\Gamma^e} h_n(\mathbf{x})(\mathbf{x}) N_i d\Omega = \int_{\Gamma^e} h_n(\mathbf{x}(\xi)) N_i d\xi \tag{192}$$

with

$$E(\xi) = h_n(\mathbf{x}(\xi)) J(\xi) \tag{193}$$

where, $J(\xi)$ is the Jacobian of the geometric mapping $\mathbf{x}(\xi)$. Using order $q = p - 1$ polynomial interpolation of E over Γ^e gives

$$E \approx \sum_{n=1}^{n_q} e_n N_n. \tag{194}$$

where n_q is given by table 17. Substituting equation (194) into equation (192) yields

$$f_i^e = \int_{\Gamma^e} \sum_{n=1}^{n_q} e_n N_i N_n d\xi = \sum_{n=1}^{n_q} e_n \underbrace{\int_{\Gamma^e} N_i N_n d\xi}_{\varrho_{in}^{f^\Gamma}} \tag{195}$$

where $\varrho_{in}^{f^\Gamma}$ define the entries in the precomputed integral table needed for computing terms of f^e at runtime as follows

$$f_i^e = \sum_{n=1}^{n_q} \varrho_{in}^{f^\Gamma} e_n. \tag{196}$$

7.4 Generation of Compact Precomputed Integral Tables

Based on equations (180), (185), and (190) it can be observed that the general form of precomputed integrals required during elemental computations is given by

$$\begin{gathered}
\int_{\Omega^e} \underbrace{\left(\frac{\partial^{n_i} N_i}{\partial \xi^{n_i}}\right)\left(\frac{\partial^{n_j} N_j}{\partial \xi^{n_j}}\right) \cdots \left(\frac{\partial^{n_k} N_k}{\partial \xi^{n_M}}\right)}_{M \text{ terms}} N_l d\xi \\
\int_{\Gamma^e} \underbrace{\left(\frac{\partial^{n_i} N_i}{\partial \xi^{n_i}}\right)\left(\frac{\partial^{n_j} N_j}{\partial \xi^{n_j}}\right) \cdots \left(\frac{\partial^{n_k} N_k}{\partial \xi^{n_M}}\right)}_{M \text{ terms}} N_l d\xi \\
n_\alpha \geq 0, \alpha = 1, \cdots, M
\end{gathered} \tag{197}$$

where the number and order of shape function derivatives involved depends upon the specific problem and *weak* form used to arrive at the finite element formulation. For example, $M = 2, n_1 = n_2 = 1$ defines terms needed in K^e, $M = 2, n_1 = n_2 = 0$ defines terms needed in M^e and $M = 1, n_1 = 0$ defines the terms needed in f^e. Note that the term N_l comes from the

p	K^e/M^e		f^e	
	Entries (Million)	Storage (MB)	Entries (Million)	Storage (MB)
4	0.2	2	4e-3	0.03
5	1.9	15	0.01	0.08
6	10	80	0.05	0.40
7	40	320	0.11	0.88
8	134	1072	0.26	2

Table 18. Asymptotic precomputed table sizes.

interpolation of Υ. If $i, j, \ldots k = 1 \ldots n_p$ and $l = 1 \ldots n_q$ with n_p and n_q defining the number of shape functions used to construct the finite element interpolation and the interpolation for Υ, respectively, then the total number of entries, n_ϱ, grows as

$$n_\varrho = \underbrace{n_p n_p \cdots n_p}_{M \text{ times}} n_q = O\left(n_p^M n_q\right). \tag{198}$$

Since $q = p - 1$, in three-dimensions $n_q \approx n_p = O\left(p^3\right)$ yielding

$$n_\varrho = O\left(n_p^{M+1}\right) = O\left(p^{3(M+1)}\right). \tag{199}$$

Observing that $M = 2$ for ϱ^K and ϱ^M entries and that $M = 1$ for ϱ^f entries, Table 18 lists the total number of possible entries in ϱ^K, ϱ^M, and ϱ^{f^Ω} along with the approximate storage required[7] to store each table.

The numbers in Table 18 apply if the elemental matrices and vectors are fully dense and there are no symmetries. In reality however, orthogonality of the shape functions lead to sparse elemental matrices. In addition, symmetries in the shape functions and elemental matrices further reduce the number of unique nonzero entries. The next section describes the use of such symmetries for generation of precomputed tables of reduced sizes for a tetrahedral elemental domain. Similar arguments can be used to reduce table sizes for other element topologies.

7.5 Table Generation For Tetrahedral Elemental Domains

Since the barycentric coordinates are symmetric, the symmetries in shape functions and their derivatives with respect to integration over the domain of the element can be explained by first defining the concept *cyclic* and *acyclic* permutation of the parametric coordinates for a tetrahedron.

Definition: ***Cyclic parametric coordinate permutation-$\oplus_i$:*** *The action of $\oplus_i$ on a given parametric coordinate system, denoted by $\oplus_i(\xi_j)$, is defined as the cyclic permutation of the sequence of ordered coordinate components $\lfloor \xi_j \rfloor$; $j \neq i$.*

Definition: ***Acyclic parametric coordinate permutation-$\ominus_i$:*** *The action of $\ominus_i$ on a given parametric coordinate system, denoted by $\ominus_i(\xi_j)$, is defined as the acyclic permutation of the sequence of ordered coordinate components $\lfloor \xi_j \rfloor$; $j \neq i$.*

[7] Assuming 8 bytes per entry.

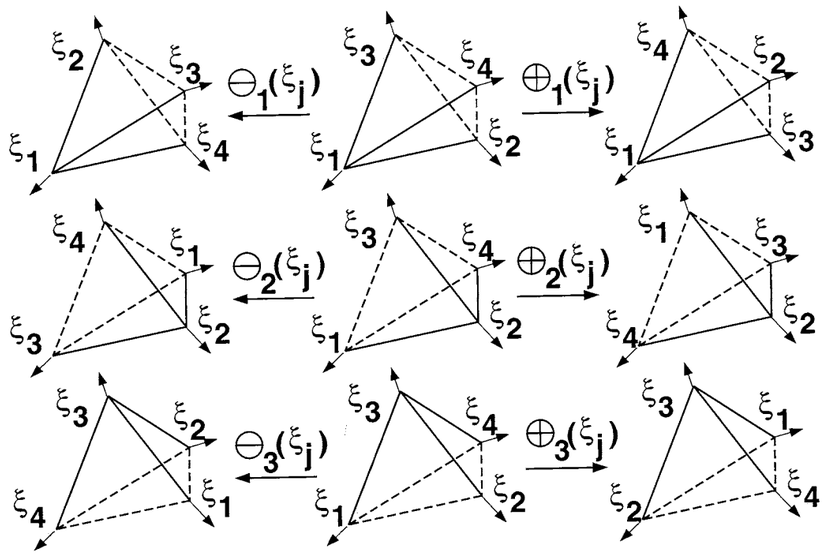


Figure 72. Clockwise and counter clockwise permutation of parameter indices.

i	$\oplus_1$	$\ominus_1$	$\oplus_2$	$\ominus_2$	$\oplus_3$	$\ominus_3$
1	1	1	4	3	2	4
2	3	4	2	2	4	1
3	4	2	1	4	3	3
4	2	3	3	1	1	2

Table 19. Clockwise and counter clockwise permutation of parameter indices.

Figure 72 depicts the action of $\oplus_i$ and $\ominus_i$ based on the three independent components ξ_1, ξ_2 and ξ_3 of the barycentric coordinate system of a tetrahedron. In this specific case the operators permute the coordinate component associated with the face opposite to ξ_i. The face involved in the permutation is drawn with broken lines. The permuted value of individual parameter indices i resulting from the action of $\oplus_i$ and $\ominus_i$ are given in Table 19.

The shape functions are permuted by permuting the coordinate components used in the shape function based on table 19. For example, the vertex shape function for a tetrahedron $N_v = \xi_1$

permutes as

$$\begin{aligned}
\oplus_1(N_v) &= \xi_1 \\
\ominus_1(N_v) &= \xi_1 \\
\oplus_2(N_v) &= \xi_4 \\
\ominus_2(N_v) &= \xi_3 \\
\oplus_3(N_v) &= \xi_2 \\
\ominus_3(N_v) &= \xi_4.
\end{aligned} \tag{200}$$

Similarly, the quadratic edge shape function $N_e = -2\xi_1\xi_2$ permits as

$$\begin{aligned}
\oplus_1(N_e) &= -2\xi_1\xi_3 \\
\ominus_1(N_e) &= -2\xi_1\xi_4 \\
\oplus_2(N_e) &= -2\xi_4\xi_2 \\
\ominus_2(N_e) &= -2\xi_3\xi_2 \\
\oplus_3(N_e) &= -2\xi_2\xi_4 \\
\ominus_3(N_e) &= -2\xi_4\xi_1.
\end{aligned} \tag{201}$$

The derivatives of the shape functions are permuted by permuting both the shape functions and the coordinate component with respect to which the derivative is taken

$$\begin{aligned}
\oplus_i\left(\frac{\partial N_a}{\partial \xi_j}\right) &= \frac{\partial[\oplus_i(N_a)]}{\partial \oplus_i(\xi_j)} \\
\ominus_i\left(\frac{\partial N_a}{\partial \xi_j}\right) &= \frac{\partial[\ominus_i(N_a)]}{\partial \ominus_i(\xi_j)}.
\end{aligned} \tag{202}$$

As an example, the permutation of $\frac{\partial N_e}{\partial \xi_1} = -2\xi_2$ with respect to $\oplus_1$ and $\ominus_1$ is given by

$$\begin{aligned}
\oplus_1\left(\frac{\partial N_e}{\partial \xi_1}\right) &= \frac{\partial(-2\xi_1\xi_3)}{\partial \xi_1} = -2\xi_3 \\
\ominus_1\left(\frac{\partial N_e}{\partial \xi_1}\right) &= \frac{\partial(-2\xi_1\xi_4)}{\partial \xi_1} = \frac{\partial(-2\xi_1(1-\xi_1-\xi_2-\xi_3))}{\partial \xi_1} = 2(2\xi_1+\xi_2+\xi_3-1).
\end{aligned} \tag{203}$$

7.5.1 Symmetries in Stiffness Matrix Entries

Recall that the stiffness matrix terms have the following symmetry [44]

$$K^e_{ij} = K^e_{ji}. \tag{204}$$

This implies that the entries in ϱ^K are defined by

$$\varrho^K_{ijlmn} = \int_{\Omega^e} \left(\frac{\partial N_i}{\partial \xi_l}\right)\left(\frac{\partial N_j}{\partial \xi_m}\right)(N_n)d\xi \tag{205}$$

are symmetric with respect to permutations of i, j

$$\varrho^K_{ijlmn} = \varrho^K_{jilmn}. \tag{206}$$

In addition, following general symmetry based on the permutation of the shape function N_i, N_j and N_n and derivative indices l and m can be exploited

$$\varrho^K_{ijlmn} = \varrho^K_{i^{\oplus_\kappa} j^{\oplus_\kappa} l^{\oplus_\kappa} m^{\oplus_\kappa} n^{\oplus_\kappa}} = \varrho^K_{i^{\ominus_\kappa} j^{\ominus_\kappa} l^{\ominus_\kappa} m^{\ominus_\kappa} n^{\ominus_\kappa}} \qquad \kappa = 1,2,3 \tag{207}$$

where the shape function permutations are given by

$$\begin{aligned} i^{\oplus_\kappa} &\equiv \oplus_\kappa(N_i) \\ i^{\ominus_\kappa} &\equiv \ominus_\kappa(N_i) \\ j^{\oplus_\kappa} &\equiv \oplus_\kappa(N_j) \\ j^{\ominus_\kappa} &\equiv \ominus_\kappa(N_j) \\ n^{\oplus_\kappa} &\equiv \oplus_\kappa(N_n) \\ n^{\ominus_\kappa} &\equiv \ominus_\kappa(N_n) \end{aligned} \tag{208}$$

and the permutation of the derivative indices are given by

$$\begin{aligned} l^{\oplus_\kappa} &\equiv \oplus_\kappa(\xi_l) \\ l^{\ominus_\kappa} &\equiv \ominus_\kappa(\xi_l) \\ m^{\oplus_\kappa} &\equiv \oplus_\kappa(\xi_m) \\ m^{\ominus_\kappa} &\equiv \ominus_\kappa(\xi_m). \end{aligned} \tag{209}$$

Recalling that for a tetrahedron $l, m = 1,2,3$, following specific symmetries are utilized in generating the precomputed tables describe in this thesis

$$\varrho^K_{ij13n} = \varrho^K_{i^{\ominus_3} j^{\ominus_3} 21 n^{\ominus_3}} \tag{210}$$

$$\varrho^K_{ij22n} = \varrho^K_{i^{\oplus_3} j^{\oplus_3} 11 n^{\oplus_3}} \tag{211}$$

$$\varrho^K_{ij23n} = \varrho^K_{i^{\oplus_3} j^{\ominus_3} 12 n^{\oplus_3}} \tag{212}$$

$$\varrho^K_{ij31n} = \varrho^K_{i^{\ominus_3} j^{\ominus_3} 12 n^{\ominus_3}} \tag{213}$$

$$\varrho^K_{ij32n} = \varrho^K_{i^{\ominus_3} j^{\ominus_3} 21 n^{\ominus_3}} \tag{214}$$

$$\varrho^K_{ij33n} = \varrho^K_{i^{\ominus_3} j^{\ominus_3} 11 n^{\ominus_3}} \tag{215}$$

Use of the above symmetries imply that only the nonzero entries from ϱ^K_{ij11n}, ϱ^K_{ij12n}, and ϱ^K_{ij21n} need to be computed and stored. Those for ϱ^K_{ij13n}, ϱ^K_{ij22n}, ϱ^K_{ij23n}, ϱ^K_{ij31n}, ϱ^K_{ij32n}, and ϱ^K_{ij33n} can be obtained using the permutations defined by equations (210) through (215).

	K^e		M^e		f^e	
	Asymp.	Compact	Asymp.	Compact	Asymp.	Compact
Entries	~1e7	45839	~1e7	3659	~5e4	451
Size(KB)	~8e4	367	~8e4	30	~4e2	4

Table 20. Unique nonzero precomputed entries for p=6 compared to asymptotic estimates.

Symmetry of the stiffness matrix, $K^e_{ij} = K^e_{ji}$, lead to additional symmetries defined by

$$\begin{aligned} \varrho^K_{ij11n} &= \varrho^K_{ji11n} \\ \varrho^K_{ij12n} &= \varrho^K_{ji12n} \\ \varrho^K_{ij21n} &= \varrho^K_{ji21n}. \end{aligned} \tag{216}$$

For p=6, and q=p-1=5, use of all the symmetries defined in this thesis lead to a total of 45839 unique nonzero entries in ϱ^K as shown in Table 20 instead of about 10 million possible entries from Table 18. This represents a more than 100 fold decrease in the number of unique nonzero entries.

7.5.2 Symmetries in Mass Matrix Entries

Symmetry of the mass matrix, $M^e_{ij} = M^e_{ji}$, leads to the following symmetry in ϱ^M

$$\varrho^M_{ijn} = \varrho^M_{jin}. \tag{217}$$

For p=6, and q=p-1=5, there are a total of 3659 unique nonzero entries in ϱ^M as shown in Table 20 instead of about 10 million possible entries from Table 18. This represents a more than 1000 fold decrease in the number of unique nonzero entries.

7.5.3 Symmetries in Force Vector Entries from Ω^e

The only symmetry to be exploited in this case is given by $\varrho^{f^\Omega}_{in} = \varrho^{f^\Omega}_{ni}$. For p=6, and q=p-1=5, there are a total of 451 unique nonzero entries in ϱ^{f^Ω} as shown in Table 20 instead of 0.26 million possible entries from Table 18. This represents a more than 100 fold decrease in the number of unique nonzero entries.

7.6 Numerical Examples

The *Poisson*'s equation will be used as a model problem to demonstrate the computational advantages of using precomputed integral tables to do the elemental integrations. The *strong* form of the model problem is defined by [62, 20]

$$\begin{aligned} -\bigtriangledown u(\mathbf{x}) &= f(\mathbf{x}),\ \mathbf{x} \in \Omega \\ u(\mathbf{x}) &= 0,\ \mathbf{x} \in \partial\Omega. \end{aligned} \tag{218}$$

The *weak* form of the problem is defined by [62, 20]

$$a(u, v) = (f, v) \tag{219}$$

with the following element level integrals

$$a(u,v) = \int_{\Omega^e} \left(\frac{\partial u}{\partial \mathbf{x}}\right)\left(\frac{\partial v}{\partial \mathbf{x}}\right) d\Omega \equiv K^e$$
$$(f,v) = \int_{\Omega^e} f v d\Omega \equiv f^e. \tag{220}$$

Two examples are chosen to evaluate specific aspects of the *precomputed integral* scheme described in section 7.1.2 compared to the *gauss integration* scheme described in section 7.1.1. In the first example, polygonal domain with polynomial exact solution is used to show two things. First, it tests the ability of the scheme to do element level computations up to machine precision when the entire integrand is polynomial for all the elemental matrices and vectors. Since the cost of doing the polynomial interpolation of Υ terms is zero in this case, it demonstrates the savings that can be realized in elemental computations for elements that lie in the interior of a curved domain and do not have any curvilinear boundary entities.

The second example solves a problem with non-polynomial but smooth analytic solution over a curved domain using the geometric mapping scheme described in Section 6.1. This example is used to numerically verify exponential convergence with approximate integral evaluation using the *precomputed integral* scheme and Υ interpolation order derived in equation (96). It also shows the computational savings realized by *precomputed integral* in comparison to *gauss integration* for curved domains.

The solution error is measured in the normalized *energy norm* as defined in equation (126). The equation system is solved using a direct solver based on skyline storage of the global matrix [44]. Time to compute elemental level $a(u,v)$ and (f,v) is represented by t_g^e and t_p^e for *Gaussian integration* and *precomputed integrals*, respectively. The accuracy of the integration points is up to 16 digits after the decimal place; however, the accuracy of the interpolation points [19] is only up to 10 digits after the decimal place. The total solution time (including time to solve the global system of equations) is represented by t_g^t and t_p^t for *Gaussian integration* and *precomputed integrals*, respectively.

7.6.1 Example 1

In this example, equation (218) is solved over a domain defined by a cube of side length 2 units. The source term $f(\mathbf{x})$ is specified such that the exact solution is given by

$$u(\mathbf{x}) = \left(1 - x_1^2\right)\left(1 - x_2^2\right)\left(1 - x_3^2\right) \tag{221}$$

with the *energy norm* of the exact solution given by

$$\|u\|_e = \left[\int_{\Omega_G} \left(\frac{\partial u}{\partial \mathbf{x}}\right)\left(\frac{\partial u}{\partial \mathbf{x}}\right) d\Omega\right]^{\frac{1}{2}} = \frac{32}{15}\sqrt{2}. \tag{222}$$

Figure 73 depicts the mesh used in the experiment with uniform p-enrichment along with the convergence of the finite element solution error with respect to p. As expected for a polynomial

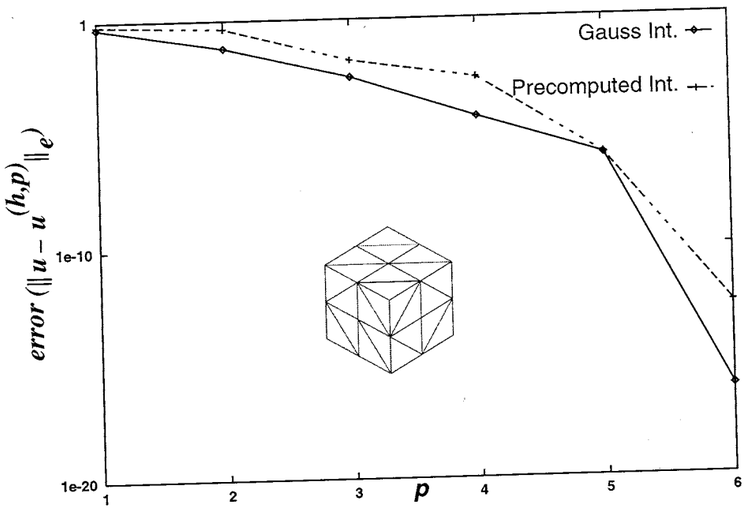


Figure 73. Solution error for example 1.

solution over a domain with planar boundary, the solution error, using the *gauss integration* scheme, drops to the accuracy limit of the integration points (1e-15). Similarly, the solution error using scheme $\int_{\Omega}^{*3}$ with precomputed entries in ϱ^K and ϱ^f drops to the precision limit of the interpolation points used (1e-10) [19] at the same rate as that produced using the *gauss integration* scheme.

Figure 74 plots the computation time versus p. Since the size of the global system is small (<1500 equations), the computation time is heavily dominated by the time to do element level computation. Figure 75 plots the $\frac{t_g^e}{t_p^e}$ and $\frac{t_g^t}{t_p^t}$ with respect to p to measure the relative efficiency of using the *precomputed integral* scheme over *Gaussian integration*. It is observed that for domains with planar boundaries the *precomputed integral* scheme is more than 10 times faster than *Gaussian integration* in doing the element level computations represented by $a(u,v)$ and (f,v).

7.6.2 Example 2

In this example Ω_G is defined by a sphere of unit radius. The source term f prescribed such that the exact solution in the spherical coordinates is given by

$$u = 100(1-r)r\sin(\theta)\cos(\phi) \tag{223}$$

with the *energy norm* of the exact solution given by

$$\|u\|_e = \left[\int_{\Omega_G} grad(u)grad(u)d\Omega\right]^{\frac{1}{2}} = 20\sqrt{5\pi} \tag{224}$$

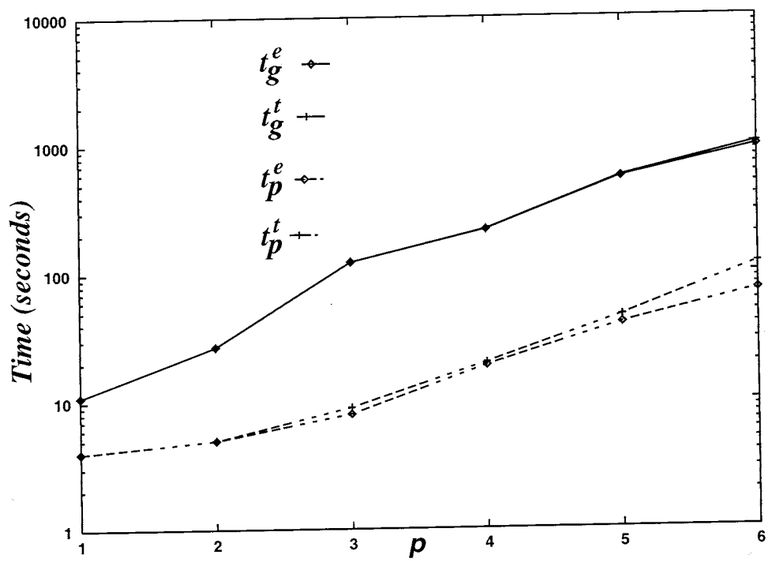


Figure 74. Computation time for example 1.

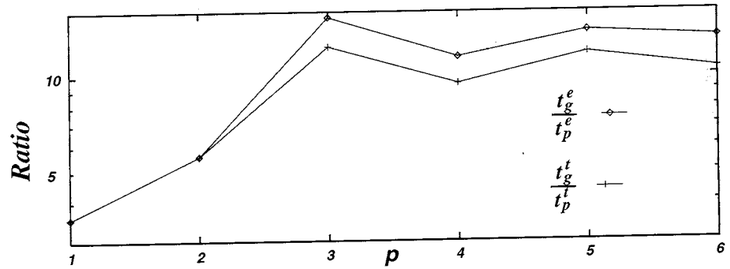


Figure 75. Ratio of computation time for example 1.

where the Cartesian coordinate components are related to the spherical coordinates by the following transformation [88]

$$\begin{aligned} x_1 &= r\sin(\theta)\cos(\phi) \\ x_2 &= r\sin(\theta)\sin(\phi) \\ x_3 &= r\cos(\phi). \end{aligned} \tag{225}$$

Figure 76 depicts the convergence of the finite element error in the normalized *energy norm* using *gauss integration* and *precomputed integrals* with Υ approximated by $p-1$ order polynomials. The mesh used in this experiment is shown in figure 69. At p=1 the errors obtained with both methods are identical because the only interpolation point for a 0-th order scheme is the centroid which happens to be identical to the location of the one point integration scheme

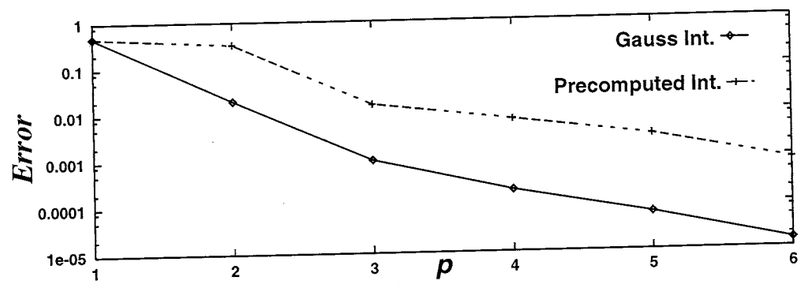


Figure 76. Solution error $\frac{||u-u^{(h,p)}||_e}{||u||_e}$ for example 2.

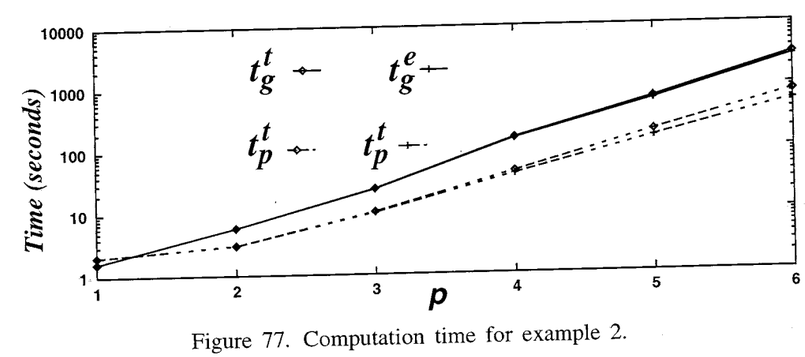


Figure 77. Computation time for example 2.

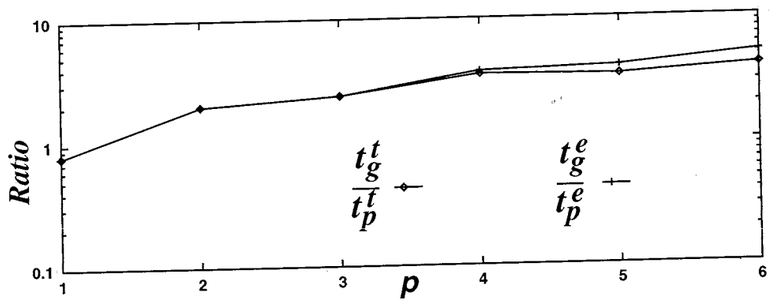


Figure 78. Ratio of computation time for example 2.

using *gauss quadrature*. As p is increased both schemes produce the same rate of exponential convergence of the finite element error.

Figure 77 plots the computation time versus p and Figure 78 plots the $\frac{t_g^e}{t_p^e}$ and $\frac{t_g^t}{t_p^t}$ with respect to p. In this case the *precomputed integral* scheme is approximately 5 times faster than *Gaussian integration* in doing the element level computations represented by $a(u,v)$ and (f,v).

8. Summary

8.1 Thesis Accomplishments

This thesis has addressed the following issues related to *hp* finite element computations on curved three-dimensional domains:

1. Automated generation of good quality curvilinear meshes suitable for use with *p*-adaptive approximations directly from the geometric description of the problem domain as defined within a geometric modelling system.
2. Construction of variable *p*-order finite element approximations for general unstructured conforming meshes.
3. Optimal domain geometry approximation.
4. Efficient element level computations.

Section 3.1 presented a methodology to eliminate the adverse effects of small geometric model features on the mesh quality based on local mesh modifications. The application of the procedure described resulted in dramatic improvement in the worst aspect ratio and smallest dihedral angle quality metrics. The method is extremely useful in generating coarse *p*-meshes because it obviates the need for mesh refinement to control mesh quality. Requirements of mesh validity with respect to the original geometric model were identified for local mesh modifications that eliminate poorly shaped elements. The thesis developed a new algorithm to detect and prevent local dimensional reductions in the mesh model using local mesh topological information.

The thesis presented an algorithm based on local mesh modifications to incrementally correct invalidity and excessive distortion in the mesh geometry resulting from curving of mesh entities classified on curved model boundaries. This presents a significant improvement over the previously used solution of uncurving selected boundary mesh entities because the analysis in Chapter 5 showed that a linear representation of curved boundary geometry will result in a loss in the optimal convergence rate of the *discretization error* for $p>2$.

The general mesh topology based decomposition of variable order *p*-hierarchic shape functions described in Chapter 4 yields a framework for construction of variable order *hp*-approximations for meshes with a combination of various element topologies in one- through three-dimensions. The thesis also developed hierarchic basis functions for a pyramid element topology. The use of pyramid element topology is necessary when constructing conforming meshes with tetrahedral and hexahedral elements. An example of a conforming mesh of a block that has tetrahedral and hexahedral elements is shown in Figure 79. The edges of the pyramid element are drawn with thicker lines.

The thesis presented the requirements of approximations related to domain geometry representation made during element level computations for second order elliptic boundary value problems. The order of direct approximation of the geometry and the non-polynomial parts of the integrands using polynomial basis functions must be $p-1$ to preserve the optimal rate of convergence of the *discretization error*. Chapter 6 presented a mesh geometry mapping scheme

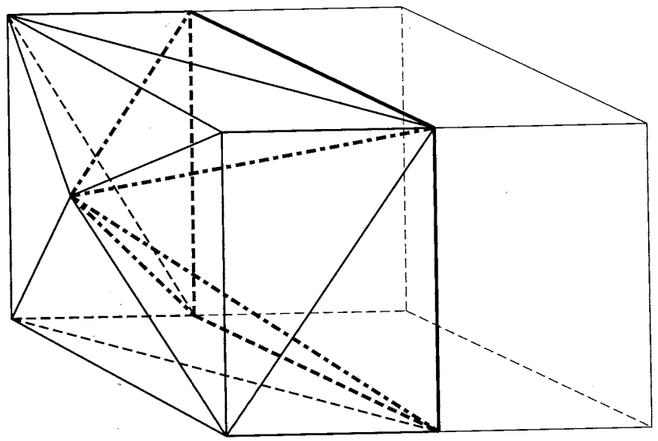


Figure 79. Conforming mesh with tetrahedra, hexahedra and pyramid elements.

based on the Boolean sum interpolation theory that conforms exactly to the shape of the domain boundary as defined in the geometric modelling system. On simplex domains, these blending schemes have limited smoothness and hence theoretically only guarantee optimal convergence of the *discretization error* up to a critical p. For the examples presented in this thesis with tetrahedral elements, exponential convergence of the *discretization error* for smooth analytic solutions was preserved up to p=6 which is a reasonable limit for problems in three-dimensions.

Chapter 7 presented a general technique for effective numerical computation of the element level integrals based on direct approximation of the integrand in terms of polynomials followed by use of precomputed values of the resulting polynomial integrand. It also showed that the method is better suited for use in general curved domains compared to the methods based on *Gaussian integration*, *sum factorization*, and *vector quadrature*.

8.2 Future Research

Areas considered in this thesis that would benefit from further research are listed below:

1. The understanding of the relationship of the mesh model to the geometric model during local mesh modifications, as presented in this thesis, require maintaining multiple classification information with selected mesh entities. A more general strategy would be to uniquely classify entities in the modified mesh against a modified geometric model representation which would maintain information necessary to relate the modified mesh to the original geometric model. Since each mesh entity is now uniquely classified with respect to a geometric model representation, subsequent local mesh modifications, to either improve the mesh quality or construct curvilinear meshes, could be done consistently because of the lack of ambiguity that results from multiple mesh entity classifications. Multiple levels of representation of the geometric model are also necessary in problems where different idealizations of the same domain are required by different processes.

2. A rigorous mathematical analysis of the effect on the rational blends on simplex domains on the convergence rate of the *discretization error* of finite element solution is required. First, the order of smoothness of the blend derivatives required to preserve the optimal convergence rate of the *discretization error* of finite element solution must be derived as a function of p for specific problems. The cost of using higher order rational blending schemes to represent geometry must be compared to direct geometry approximation using polynomials of appropriate order.
3. The current procedures to determine the acceptability of a curvilinear element do so by evaluating the determinant of the Jacobian at discrete locations within the element domain. This does not ensure invertibility of the geometric map everywhere within the closure of the element domain. Research is required to develop a more comprehensive check for element geometry distortion that is not tied down to specific discrete locations within the element domain. The computational cost of doing the comprehensive check must be competitive with those that use Jacobian evaluations at discrete locations.
4. The local mesh modification algorithm to correct invalid and unacceptably distorted elements must be augmented to include procedures that allow for curving mesh edges and faces classified interior to the domain of the geometric model. It must also consider relocation of mesh vertices as another option.
5. The number of the precomputed integral entries required to be stored for various element level integrals can be further reduced by investigating additional symmetries in the shape functions and their derivatives if they exist.
6. The efficiency of using polynomial integrand approximation followed by the use of pre-computed values for element level integral computation needs to be evaluated for three-dimensional element topologies besides tetrahedron.

Literature Cited

[1] M. Abramowitz and I. A. Stegun, editors. *Handbook of Mathematical Functions*. Dover Pulications Inc., New York, 1972.

[2] C. G. Armstrong, editor. *Advances in Engng. Software*, volume 13:5/6. Computational Mechanics Publ., Southhampton, England, 1991.

[3] I. Babuska. The theory and practice of the h, p and h-p versions of the finite element method for solving 3 dimensional problems of elasticity. *Third U.S. National Congress on Computational Mechanics*, 1995. Dallas, Texas, June 12-14.

[4] I. Babuska and A. K. Aziz. On the angle condition in the finite element method. *SIAM J. Numer. Anal.*, 13(12), 1976.

[5] I. Babuska, M. Griebel, and J. Pitkaranta. The problem of selecting the shape functions for p-type finite elements. *Int. J. Numer. Meth. Engng.*, 28:1891–1908, 1989.

[6] I. Babuska and B. Q. Guo. The h-p version of the finite element method for domains with curved boundaries. *SIAM J. Numer. Anal.*, 25(4):837–861, 1988.

[7] I. Babuska and B. Q. Guo. Approximation properties of the h-p version of the finite element method. *Comp. Meth. Appl. Mech. Engng.*, 133:319–346, 1996.

[8] I. Babuska and M. Suri. The p- and hp-versions of the finite element method: An overview. *Comp. Meth. Appl. Mech. Engng.*, pages 5–26, 1990.

[9] P. L. Baehmann. *Automated Finite Element Modeling and Simulation*. PhD thesis, Mechanical Eng., Aeronautical Eng., & Mechanics, Rensselaer Polytechnic Institute, Scientific Computation Research Center, RPI, Troy, NY 12180-3590, May 1989.

[10] T. J. Baker. Automatic mesh generation for complex three-dimensional regions using a constrained Delaunay triangulation. *Engineering with Computers*, 5:161–175, 1989.

[11] U. Banerjee and M. Suri. The effect of numerical quadrature in the p-version of the finite element method. *Math. of Comp.*, 59(199):1–20, 1992.

[12] B. A. Barsky and D. P. Greenberg. Determining a set of b-spline control vertices to generate an interpolating surface. *Computer Graphics and Image Processing*, 14:203–226, 1980.

[13] M. W. Beall and M. S. Shephard. A general topology-based mesh data structure. Technical Report 19-1996, Scientific Computation Research Center, Rensselaer Polytechnic Institute, Troy, NY 12180-3590, 1996. submitted to Int. J. Num. Meth. Engng.

[14] C. Bernard. Optimal finite-element interpolation on curved domains. *SIAM J. Numer. Anal.*, 26(5):1212–1240, 1989.

[15] W. Bohm, G. Farin, and K. J. A survey of curve and surface methods in cagd. *Computer Aided Geometric Design*, 1:1–60, 1984.

[16] P. Carnevali, R. B. Morris, Y. Tsuji, and G. Taylor. New basis functions and computational procedures for p-version finite element analysis. *Int. J. Numer. Meth. Engng.*, 36:3759–3779, 1993.

[17] J. C. Cavendish, D. A. Field, and W. H. Frey. An approach to automatic three-dimensional mesh generation. *Int. J. Numer. Meth. Engng.*, 21:329–347, 1985.

[18] Q. Chen and I. Babuska. Approximate optimal points for polynomial interpolation of real functions in an interval and in a triangle. *Comp. Meth. Appl. Mech. Engng.*, 128:405–417, 1995.

[19] Q. Chen and I. Babuska. Approximate optimal polynomial interpolation points in the tetrahedron. 1995. preprint.

[20] P. G. Ciarlet. *The finite element method for elliptic problems*. North-Holland Publishing Company, Amsterdam, Holland, 1978.

[21] H. L. de Cougny and M. S. Shephard. Refinement, derefinement and local retriangulations in three-dimensions. 1994. in preparation for submission.

[22] H. L. de Cougny, M. S. Shephard, and M. K. Georges. Explicit node point smoothing within the Finite Octree mesh generator. Technical Report SCOREC # 10-1990, Scientific Computation Research Center, Rensselaer Polytechnic Institute, Troy, NY 12180-3590, 1990.

[23] B. E. de l'Isle and P. L. George. Optimization of tetrahedral meshes. In I. Babuska, J. E. Flaherty, J. E. Hopcroft, W. D. Henshaw, J. E. Oliger, and T. Tezduyar, editors, *Modeling, Mesh Generation, and Adaptive Numerical Methods for Partial Differential Equations*, pages 97–128. Springer-Verlag, New York, 1995.

[24] L. Demkowicz, J. T. Oden, W. Rachowicz, and O. Hardy. Toward a universal h-p adaptive finite element strategy, part 1: Constrained approximation and data structure. *Computer Methods in Applied Mechanics and Engineering*, 77:79–112, 1989.

[25] K. D. Devine and J. E. Flaherty. Parallel adaptive hp-refinement techniques for conservation laws. *Applied Numer. Math.*, 20:367–386, 1996.

[26] S. Dey and M. S. Shephard. Mapping finite element entities to true model geometry. *Third U.S. National Congress on Computational Mechanics*, 1995. Dallas, Texas, June 12-14.

[27] M. Dubiner. Spectral methods on triangles and other domains. *J. of Sci. Comp.*, 6(4):345–390, 1991.

[28] Electronic Data Syatems Corporation, Parker's House, 46 Regent Street, Cambridge CB2 1DB England. *Parasolid Version 6 KI Programming Reference Manual*, October 1994.

[29] G. Farin. Triangular bernstien-bezier patches. *Computer Aided Geometric Design*, 3:83–127, 1986.

[30] G. Farin. *Curves and Surface for Computer Aided Geometric Design*. Academic Press Inc., 1990.

[31] J. Fish and R. Guttal. Recent advances in the p-version of the finite element method for shells. *Computing Systems in Engineering*, 6(3):195–211, 1995.

[32] P. L. George. *Automatic Mesh Generation*. John Wiley and Sons, Ltd, Chichester, 1991.

[33] P. L. George. Generation de maillages par une methode de type Voronoi, Partie 2: le cas tridimensionnel. Technical Report RR INRIA n 1664, INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 Le Chesnay Cedex, France, 1993.

[34] P. L. George, F. Hecht, and E. Saltel. Fully automatic mesh generator for 3D domains of any shape. *Impact of Com. in Sci. and Eng.*, 2:187–218, 1990.

[35] M. K. Georges. *A Two-Dimensional Automatic Modeling System for the h-p Version of the Finite Element Method*. PhD thesis, Civil Engineering, Rensselaer Polytechnic Institute, Scientific Computation Research Center, RPI, Troy, NY 12180-3590, December 1989.

[36] G. H. Golub and V. L. C. F. *Matrix Computations*. The John Hopkins University Press, Baltimore, MD 21218-4319, 1993.

[37] W. J. Gordon and C. A. Hall. Construction of curvilinear co-ordinate systems and applications to mesh generation. *Int. J. Numer. Meth. Engng.*, 7:461–477, 1973.

[38] W. Gui and I. Babuska. The h-, p- and hp-version of the finite element method in one dimension. part 1: The analysis of the p-version. part 2: The error analysis of the h- and hp-version. part 3: The adaptive hp-version. *Numerische Mathematik*, 49:577–612; 613–657; 659–683, 1986.

[39] E. L. Gursoz, Y. Choi, and F. B. Prinz. Vertex-based representation of non-manifold boundaries. In M. J. Wozny, J. U. Turner, and K. Priess, editors, *Geometric Modeling Product Engineering*, pages 107–130. North Holland, 1990.

[40] R. B. Haber, M. S. Shephard, J. F. Abel, R. H. Gallagher, and D. P. Greenberg. A generalized two-dimensional finite element preprocessor utilizing discrete transfinite mappings. *Int. J. Numer. Meth. Engng.*, 17:1015–1044, 1981.

[41] Hibbitt, Karlsson and Sorensen, Inc., 1080 Main Street, Pawtucket, RI. *ABAQUS/Standard User's Manual, Volume I Version 5.4*, 1994.

[42] F. B. Hildebrand. *Introduction to Numerical Analysis*. Dover Publications, Inc., New York, NY, 1987.

[43] H. E. Hinnat. A fast method of numerical quadrature for p-version finite element matrices. *Int. J. Numer. Meth. Engng.*, 37:3723–3750, 1994.

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

[45] Y. Jinyun. Symmetric gaussian quadrature formulae for tetrahedral regions. *Comp. Meth. Appl. Mech. Engng.*, 43(3):349–353, 1984.

[46] C. G. Kim and M. Suri. On the p version of the finite element method in the presence of numerical integration. *Num. Meth. for Partial Diff. Eqn.*, 9:593–629, 1993.

[47] P. M. Knup. On the invertibility of the isoparametric map. *Comp. Meth. Appl. Mech. Engg.*, 78:313–329, 1990.

[48] M. Krizek. On the maximal angle condition for linear tetrahedral elements. *SIAM J. Numer. Anal.*, 29:513–520, 1992.

[49] C. Lacombe and C. Bedard. Face-apex projectors for the interpolation function of a general tetrahedral mid-edge finite element. *Comp. Meth. Appl. Mech. Engng.*, 68:177–188, 1988.

[50] T. J. Liszka, C. W. Berry, and O. P. Hardy. Constrained hp-approximation in anisotropic hex-based finite element code. In *Third US National Congress on Computational Mechanics*, page 232, 1995.

[51] R. Löhner and P. Parilch. Three-dimensional grid generation by the advancing front method. *Int. J. Num. Meths. Fluids*, 8:1135–1149, 1988.

[52] C. R. MacCluer. *Boundary Value Problems and Orthogonal Expansions Physical Problems from a Sobolev Viewpoint.* IEEE Press, 1994.

[53] M. Mäntylä. *Introduction to Solid Modeling*. Computer Science Press, Rockville, Maryland, 1988.

[54] D. J. Mavriplis. An advancing front delaunay triangulation algorithm designed for robustness. Number AIAA-93-0671. AIAA, Washington, D.C., 1993. 31st Aerospace Sciences Meeting and Exhibit, Jan. 1993.

[55] A. R. Mitchell and R. Wait. *The finite element method in partial differential equations.* John Willey and Sons, New York, 1977.

[56] A. K. Noor and C. M. Andersen. Computerized symbolic manipulation in nonlinear finite element analysis. *Computers and Structures*, 13:379–403, 1979.

[57] A. K. Noor and C. M. Andersen. Computerized symbolic manipulation in structural mechanics-progress and potential. *Computers and Structures*, 10:95–118, 1979.

[58] J. T. Oden. Parallel adaptive hp-finite element methods for problems in fluid and solid mechanics. In T. J. R. Hughes, E. Onate, and O. C. Zienkiewicz, editors, *Recent developments in finite element analysis*, pages 29–35. International Center for Numerical Methods in Enginee ring, Barcelona, Spain, 1994.

[59] J. T. Oden. Control of the computational process: Modeling, accuracy, stability and efficiency. *Third U.S. National Congress on Computational Mechanics*, 1995. Dallas, Texas, June 12-14.

[60] 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.

[61] J. T. Oden and A. Patra. A parallel adaptive strategy for hp finite element computations. *Comp. Meth. Appl. Mech. Engng.*, 121:449–470, 1995.

[62] J. T. Oden and J. N. Reddy. *An introduction to the mathematical theory of finite elements.* John Willey and Sons, New York, 1976.

[63] J. T. Oden, W. Wu, and M. Ainsworth. Three-step h-p adaptive strategy for the incompressible Navier-Stokes equations. In I. Babuska, J. E. Flaherty, J. E. Hopcroft, W. D. Henshaw, J. E. Oliger, and T. Tezduyar, editors, *Modeling, Mesh Generation, and Adaptive Numerical Methods for Partial Differential Equations*, volume 75. Springer-Verlog, 1995. IMA Volumes in Mathematics and its Applications.

[64] T. K. Ohsumi, S. Dey, J. E. Flaherty, and M. S. Shephard. Table look-up: High order integration on tetrahedra region. 1996. in preparation.

[65] PDA Engineering, PATRAN Division, 2975 Redhill Avenue, Costa Mesa, CA 92626. *PATRAN PLUS USER MANUAL, Release 2.5*, October 1990.

[66] R. Perucchio, M. Saxena, and A. Kela. Automatic mesh generation from solid models based on recursive spatial decomposition. *Int. J. Numer. Meth. Eng.*, 28:2469–2502, 1989.

[67] M. J. D. Powell. *Approximation theory and methods*. Cambridge University Press, New York, 1981.

[68] Y. Pressburger and R. Perucchio. A self-adaptive fe system based on recursive spatial decomposition and multigrid analysis. *Int. J. Numer. Meth. Engng.*, 38:1399–1421, 1995.

[69] 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.

[70] A. A. G. Requicha and H. B. Voelcker. Solid modeling: Current status and research directions. *IEEE Computer Graphics and Applications*, 3(7):25–37, 1983.

[71] D. G. Roddeman and L. F. Jansen. An a-priori geometry check for a single isoparametric finite element. *Computers and Structures*, 47(1):69–72, 1993.

[72] M. P. Rossow and I. N. Katz. Hierrachical finite elements and precomputed arrays. *Int. J. Numer. Meth. Engng.*, 12(6):997–999, 1978.

[73] W. J. Schroeder. *Geometric Triangulations: with Application to Fully Automatic 3D Mesh Generation*. PhD thesis, Rensselaer Polytechnic Institute, Scientific Computation Research Center, RPI, Troy, NY 12180-3590, May 1991. SCOREC Report # 9-1991.

[74] W. J. Schroeder and M. S. Shephard. On rigorous conditions for automatically generated finite element meshes. In J. Turner, J. Pegna, and M. Wozny, editors, *Product Modeling for Computer-Aided Design and Manufacturing*, pages 267–281. North Holland, 1991.

[75] M. S. Shephard. Approaches to the automatic generation and control of finite element meshes. *Appl. Mech. Rev.*, 41:169–185, 1988.

[76] M. S. Shephard. The specification of physical attribute information for engineering analysis. *Engineering with Computers*, 4:145–155, 1988.

[77] M. S. Shephard. Automatic generation of finite element models. In M. Papadrakakis, editor, *Solving Large Scale Problems in Mechanics*, pages 431–460. John Wiley & Sons, Ltd, Chichester, 1993.

[78] M. S. Shephard, S. Dey, and M. K. Georges. Automatic meshing of curved three-dimensional domains: Curving finite elements and curvature-based mesh control. In I. Babuska, J. E. Flaherty, J. E. Hopcroft, W. D. Henshaw, J. E. Oliger, and T. Tezduyar, editors, *Modeling, Mesh Generation, and Adaptive Numerical Methods for Partial Differential Equations*, pages 67–98. Springer-Verlag, New York, 1995.

[79] M. S. Shephard and P. M. Finnigan. Toward automatic model generation. In A. K. Noor and J. T. Oden, editors, *State-of-the-Art Surveys on Computational Mechanics*, pages 335–366. ASME, 1989.

[80] M. S. Shephard and M. K. Georges. Automatic three-dimensional mesh generation by the Finite Octree technique. *Int. J. Numer. Meth. Engng.*, 32(4):709–749, 1991.

[81] M. S. Shephard and M. K. Georges. Reliability of automatic 3-D mesh generation. *Comp. Meth. Appl. Mech. Engng.*, 101:443–462, 1992.

[82] M. S. Shephard and N. P. Weatherill, editors. *Int. J. Numer. Meth. Engng.*, volume 32. Wiley-Interscience, Chichester, England, 1991.

[83] S. J. Sherwin and G. E. Karniadakis. A new triangular and tetrahedral basis for high-order (hp) finite element methods. *Int. J. Numer. Meth. Engng.*, 38:3775–3802, 1995.

[84] S. J. Sherwin and G. E. Karniadakis. Tetrahedral hp finite elements: algorithms and flow simulations. *J. of Comp. Phy.*, 124:14–45, 1996.

[85] P. Silvester. Construction of triangular finite element universal matrices. *Int. J. Numer. Meth. Engng.*, 12:237–244, 1978.

[86] Spatial Technology, Inc., 2425 25th St., Building A, Boulder, Colorado. *ACIS Interface Guide and ACIS API Guide*, December 1992.

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

[88] E. W. Swokowski. *Calculus with analytic geometry*. Prindle, Weber & Schmidt, Boston, MA, 1980.

[89] B. A. Szabo and I. Babuska. *Finite Element Analysis*. Wiley Interscience, New York, 1991.

[90] B. A. Szabo and A. G. Peano. Hierarchic finite elements. In H. Kardestuncer and D. H. Norrie, editors, *Finite Element Handbook*, pages 2.227–2.233. McGraw-Hill, New York, 1987.

[91] E. L. Wachspress. *A rational finite element basis*. Academic Press, New York, NY, 1975.

[92] N. P. Weatherill and O. Hassan. Efficient three-dimensional delaunay triangulation with automatic point creation and imposed boundary constraints. *Int. J. Numer. Meth. Engng.*, 37:2005–2039, 1994.

[93] K. J. Weiler. The radial-edge structure: A topological representation for non-manifold geometric boundary representations. In M. J. Wozny, H. W. McLaughlin, and J. L. Encarnacao, editors, *Geometric Modeling for CAD Applications*, pages 3–36. North Holland, 1988.

[94] XOX Corporation, Two Appletree Square, Suite 334, Minneapolis, Minnesota 55425. *SHAPES Reference Manual, Release 2.0.8*, July 20, 1993.

[95] G. Yagawa, G.-W. Ye, and S. Yoshimura. A numerical integration scheme for finite element method based on symbolic manipulation. *Int. J. Numer. Meth. Engng.*, 29:1539–1549, 1990.

[96] O. C. Zienkiewicz and R. L. Taylor. *The Finite Element Method - Volume 1*. McGraw-Hill Book Co., New York, 4th edition, 1987.