In compliance with the
Canadian Privacy Legislation
some supporting forms
may have been removed from
this dissertation.

While these forms may be included
in the document page count,
their removal does not represent
any loss of content from the dissertation.

A Posteriori Error Estimates of Finite Element Solutions

by

Buyuan Cheng

B.Eng., Mechanical, Chong Qing University, P.R. China, 1986

M.Eng., Mechanical, Chong Qing University, P.R. China, 1989

A thesis submitted to

the Faculty of Graduate Studies and Research

in partial fulfillment of the requirements for the degree of

Master of Applied Science

Department of Mechanical and Aerospace Engineering

Ottawa - Carleton Institute for Mechanical and Aerospace Engineering

Carleton University

Ottawa, Ontario

Canada

August 2003

© Copyright, 2003

Buyuan Cheng

National Library
of Canada

Bibliothèque nationale
du Canada

Acquisitions and
Bibliographic Services

Acquisitions et
services bibliographiques

395 Wellington Street
Ottawa ON K1A 0N4
Canada

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

ISBN: 0-612-88869-X

Our file Notre référence

ISBN: 0-612-88869-X

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

Canada

Abstract

The objective of this thesis is: a). to apply A.Mashaie's algorithm to compute the error estimator of the finite element solutions; b). to extend A.Mashaie's work from solid mechanics to energy calculation; c). to extend A.Mashaie's work from 8-node hexahedral element to 4-node and 10-node tetrahedral element, 6-node and 15-node prism element and 20-node hexahedral element.

The duality method in error analysis employing a kinematically and a statically admissible stress field and a kinematically and a statically admissible thermal flux field is used to compute an error bound.

The details of matrix equations being used to develop program are derived in this thesis. The code is implemented in an existing FEM program for solving the problems of stresses and energy analysis. To verify implemented code, several test problems were studied.

To my wife, Yin Jingmiao, and son, Cheng Yinda

Acknowledgments

I would like to express my sincerest appreciation and deep gratitude to my thesis supervisor Prof. John.A. Goldak for his knowledge, guidance and encouragement during my studies. Many thanks to the teachers at the Mechanical and Aerospace Department, Carleton University, for all the things that I have learned. Sincere gratitude is expressed to the colleagues for invaluable help and supporting, especially to ShaoDong Wang, Stanislav Tchernov, Jianguo Zhou and Dan Downey. The financial support of Carleton University, National Science and Engineering Research Council (CRD Grant: Optimization of High Pressure Die Casting) and Material and Manufacturing Ontario (enabling grant #PP60109) is gratefully acknowledged. Finally, I would like to express my deep gratitude to my wife, Yin Jingmiao, to my son, Cheng Yinda and to my parents for their understanding and continuous encouragement.

Nomenclature

x, y, z	Cartesian coordinate system.
ξ, η, ζ	Natural coordinate system.
Ω	Physical domain.
$\partial\Omega$	Boundary of the domain.
$\partial_1\Omega$	Part of boundary of domain with Dirichlet Boundary Condition.
$\partial_2\Omega$	Part of boundary of domain with Neumann Boundary Condition.
σ	Stress field, Stefan-Boltzmann constant.
σ_k	KAS Stress field.
σ_s	SAS Stress field.
$\sigma_{\mathbf{k}}$	KAS Stress vector.
$\sigma_{\mathbf{s}}$	SAS Stress vector.
$\bar{\sigma}$	SAS Stress vector without body force and/or inertial force.
$\delta\bar{\sigma}$	Increment SAS Stress vector due to body force and/or inertial force.
$\bar{\sigma}^8$	SAS Stress vector of nodal value of 8-node brick.
ε	Strain Field and Strain Vector.
u	Displacement field.
$\mathbf{u}$	Displacement vector.
n	Number of nodes in an element.
m	Number of nodes on a face.

f	Body force.
f_x, f_y, f_z	Components of body force in the orientation of x, y, z axes.
$\mathbf{F}_k^e$	The vector of KAS nodal force on an element.
$\mathbf{F}_k^f$	The vector of KAS nodal force on face.
$\mathbf{F}_s^f$	The vector of SAS nodal force on face.
$\bar{u}$	The Dirichlet displacement specified on $\partial_1\Omega$.
$\bar{T}_n$	The Neumann traction specified on $\partial_2\Omega$.
$\mathbf{B}$	The divergence matrix operator.
h	Specific enthalpy or Convection heat transfer coefficient.
k	Thermal conductivity.
$[k]$	Thermal conductivity tensor represented by a matrix.
p	Energy production per unit volume.
q	Heat flux per unit area.
q_x, q_y, q_z	Components of $\mathbf{q}$.
q_k	The KAF thermal flux per unit area.
q_s	The SAF thermal flux per unit area.
$\mathbf{q}_k$	Vector of KAF thermal flux per unit area.
$\mathbf{q}_s$	Vector of SAF thermal flux per unit area.
T	Temperature.
$\mathbf{T}$	Nodal vector of temperature in an element.

Contents

Acceptance Sheet	ii
Abstract	iii
Dedication	iv
Acknowledgements	v
Nomenclature	vi
Table of Contents	viii
List of Tables	xi
List of Figures	xii
1 Introduction	1
1.1 Background	1
1.1.1 The Residual Type Error Estimates	3
1.1.2 The Recovery Type Error Estimate	5
1.1.3 The Relationship Between Error Estimators and Error Bound	7

1.2	Aims and Organization of the Thesis	8
1.2.1	Aims of the Thesis	8
1.2.2	Organization of the Thesis	8
2	Fundamental Concepts	10
2.1	Kinematically and Statically Admissible Stress	10
2.1.1	Basic Equations	10
2.1.2	Kinematically Admissible Stress (KAS)	12
2.1.3	Statically Admissible Stress (SAS)	13
2.1.4	Error Bound	14
2.1.5	The Relationship Between KAS and SAS	15
2.2	Kinematically and Statically Admissible Thermal Flux	19
2.2.1	Basic Equations	19
2.2.2	Kinematically Admissible Flux (KAF)	22
2.2.3	Statically Admissible Flux (SAF)	24
2.2.4	Error Bound	24
2.2.5	The Relationship Between KAF and SAF	26
3	Discrete SAS Fields	30
3.1	A. Mashaie's Traction Force Algorithm	30
3.2	KAS Nodal Forces	31

3.2.1	KAS Nodal Force on Element	31
3.2.2	KAS Nodal Force on Face	32
3.3	SAS Nodal Forces on Face	34
3.4	SAS Nodal Stress from SAS Nodal Forces	44
3.5	Error Bound on Each Element	50
3.6	SAS with Body Force	51
4	Discrete SAF Fields	57
4.1	Algorithm	57
4.2	KAF Nodal Thermal Loads	58
4.2.1	KAF Nodal Load on Element	58
4.2.2	KAF Nodal Thermal Load on Face	59
4.3	SAF Nodal Thermal Loads on Face	61
4.4	SAF Nodal Thermal Flux	63
4.5	Error Bound on Each Element	68
4.6	SAF with the Transient Thermal Flux	69
4.6.1	Transform $\nabla \cdot \delta \bar{\mathbf{q}}$ into matrix form	70
4.6.2	Transform $\nabla \cdot \bar{\mathbf{q}}$ into matrix form	71
4.6.3	Creating and Solving the System of Equations	73
4.6.4	Different Element Types for KAF and SAF Calculation	74

5	Test Problems for SAS	79
5.1	Divergence Free Test	79
5.2	3DL Block Test	82
5.3	Crack Test	90
5.4	6-node and 15-node Prism Element Text	96
6	Test Problems for SAF	99
6.1	Divergence Free Test	99
6.2	8-node Hexahedral Element Bar Test	101
6.2.1	25 time steps with time step size of 2 seconds	102
6.2.2	30 time steps with time step size of 40 seconds	103
6.3	6-node Prism Element Text	104
7	Conclusions and Contributions	114
7.1	Conclusions	114
7.2	Contributions	115

List of Tables

3.1	The size of the $[\bar{H}]$ matrix for different element types for SAS calculation.	49
4.1	The size of the $[\bar{H}]$ matrix for different element types for SAF calculation.	67
5.1	Number of elements vs. total strain energy for 3DL-Block	84
5.2	Number of elements vs. total relative energy error for 3DL-Block . . .	85
5.3	Maximum and minimum element relative error	87
5.4	3DL-Block calculation results for case two	88
5.5	Maximum and minimum element relative strain energy error	93
5.6	Crack calculation results	93
5.7	Crack calculation results	94
5.8	3DL-Block calculation results for 6-node prism	97
5.9	Maximum and minimum element relative error	98
7.1	The feasibility of SAS and SAF calculation for various element types.	115

List of Figures

2.1	Relationship between KAS and SAS	16
2.2	Stress field and exact solution	18
2.3	Relationship between KAF and SAF	26
2.4	Thermal flux field and exact solution	28
3.1	Geometry of the set of elements and faces containing the node i	37
3.2	3DL-Block	41
5.1	3DL-Block boundary conditions	82
5.2	Number of elements vs. strain energy for case one	86
5.3	Number of elements vs. total relative strain energy error	86
5.4	Number of elements vs. error bound in domain	87
5.5	Number of elements vs. strain energy for case two	88
5.6	3DL-Block strain energy error with 56 elements.	89
5.7	3DL-Block strain energy error with 448 elements.	89

5.8	3DL-Block strain energy error with 3,584 elements.	89
5.9	3DL-Block strain energy error with 28,672 elements.	89
5.10	Boundary conditions for the crack problem	90
5.11	Number of elements vs. strain energy	91
5.12	Number of elements vs. relative strain energy error	92
5.13	Crack strain energy error with 16 elements	95
5.14	Crack strain energy error with 128 elements	95
5.15	Crack strain energy error with 1,024 elements	95
5.16	Crack strain energy error with 8,192 elements	95
5.17	6-node prism element in x, y, z directions	97
6.1	Boundary condition for the 8-node hexahedral element bar test. . . .	102
6.2	Element number vs. thermal flux for the 8-node hexahedral.	106
6.3	Element number vs. increment of thermal flux for the 8-node hexahedral.	107
6.4	Element number vs. thermal flux increment due to bubble function for the 8-node hexahedral.	108
6.5	Element number vs. thermal flux for the 8-node hexahedral.	109
6.6	Number of time steps vs. total energy for the 8-node hexahedral. . .	110
6.7	Number of time steps vs. total error bound for the 8-node hexahedral.	110
6.8	Number of time steps vs. total relative energy error for the 8-node hexahedral.	111

6.9	Boundary condition for the 6-node prism element bar test.	111
6.10	Number of time steps vs. total energy.	112
6.11	Number of time steps vs. total relative energy error.	112
6.12	Number of time steps vs. total error bound.	113

Chapter 1

Introduction

1.1 Background

The finite element method is the most popular method to calculate approximate solutions of many partial differential equations. It is successfully used in many practical applications, e.g. in solid mechanics, thermal analysis, structural analysis, fluid dynamics, and so on.

Naturally, the user will ask the following questions immediately on receiving the discrete solutions: How do we know if the results are accurate enough? If the answer is negative, what should be done next? A posteriori error estimates of finite element solutions will help us answer these questions.

It is now widely accepted that adaptive procedures are advantageous for practical

finite element computations [1]. Error estimation is a key feature of an adaptive procedure. It is used to steer the adaptive process by indicating regions where more or less resolution is needed and to control solution accuracy and stop the adaptive process. An error estimator provides information about the global quality of the solution and the distribution of the error in domain. The global estimate assesses the admissibility of the approximate solution. If it is not admissible, a new mesh must be created following a remeshing criterion based on the error distribution. Elements are added where the approximate solution is less accurate. The goal is a new solution with a uniform error distribution and a specified global error.

There are three main methods of refining an existing finite element solution. The first one is referred to as h-refinement, which involves adding vertices and elements to subdomain or through more global remeshing but has the general aim of increasing the number of vertices and elements in those regions of the domain where some measure of the error is unacceptably high. This refinement is essentially performed by increasing the number of elements while keeping the order of polynomial of interpolation fixed. The opposite of h-refinement is often referred to as p-refinement. This refinement process increases the order of the element polynomial while keeping the elements in the mesh fixed. The third method is referred to as r-refinement, which adapts the mesh in such a way that the topology mesh is not changed, i.e., the number of vertices, elements and their connectivity remains unchanged, but the nodes are moved

and the mesh is continuously deformed, so as to increase the density of vertices in those regions of the domain with the highest errors [8].

A priori error estimates are the main tool for the theoretical study of the finite element method but they cannot provide practical results. In order to obtain numerical approximations of the actual error, it is necessary to use a posteriori error estimators.

This thesis deals with error bounds for finite element solutions based on duality. The real error cannot be greater than an error bound. Because most papers in the finite element method literature are based on error estimates, not error bounds, we begin with a brief review of error estimates for finite element method. Unlike an error bound, an error estimator can be less than the real error.

The error estimates usually used in practical applications are the residual type error estimates and the recovery type error estimates.

1.1.1 The Residual Type Error Estimates

The approximate finite element solution does not satisfy the governing partial differential equation. This lack of fulfillment is called the residual. The residual based error estimators were first introduced by Babuska and Rheinboldt in 1978 [2] and have been since used very effectively and further developed by many others. Here, substantial progress was made as recently as 1993 with the introduction of the residual equilibration by Ainsworth and Oden [3].

The residual type error in the finite element solution is contributed by the residual in the equation of equilibrium, the discontinuity of the stress across the boundaries of elements and the discontinuity of the stress and the traction boundary conditions across the external boundaries.

In solid mechanics, for instance, the kinematically admissible stress (KAS) field σ_k is computed by the displacement finite element method.

The equation

$$r_1 = \int_{\Omega} (\nabla \cdot \sigma_k + f) d\Omega$$

denotes the residual in the equation of equilibrium due to that KAS stress σ_k does not satisfy equilibrium equation pointwise.

$$r_2 = \sum_S \int_S [(n^T \sigma_k)_{element1} - (n^T \sigma_k)_{element2}] dS_{internal}$$

denotes the residual due to the stress jump across the internal boundaries, and

$$r_3 = \sum_S \int_S (n^T \sigma_k - \bar{T}) dS_{external}$$

denotes the residual due to the discontinuity of the stress and the traction boundary conditions across the external boundaries. The total error is the sum of three components of residuals described above:

$$Error = r_1 + r_2 + r_3$$

The residual type estimators have a solid theoretical background but their results have been found to be less robust than recovery type estimates [4]. The application of residual method is costly because the error indicator must be evaluated for all degrees

of freedom corresponding to the next refinement stage [6].

1.1.2 The Recovery Type Error Estimate

The recovery based error estimators (or flux projection estimators) are, on the other hand, more recent having been first introduced by Zernkiewicz and Zhu in 1987 [8]. Again, these were extensively improved by the introduction of new recovery processes. Here, in particular, the so called SPR (or Superconvergent Patch Recovery) method introduced in 1992 by the same authors [9, 10] has significantly improved performance of the recovery based methods.

The main idea of such estimator is to employ a projection technique in order to recover a post-processed quantity (e.g. the stresses) from the finite element solution. The error is then estimated by taking the difference between the recovered solution and the finite element solution.

An example is an elliptic linear elastic problem with the form

$$\nabla \cdot \sigma = \mathbf{B}^T \mathbf{C}^{-1} \mathbf{B} \mathbf{u} = -f \quad \text{in } \Omega$$

together with appropriate boundary conditions on $\partial\Omega$ and definitions of stress and strain: $\sigma = \mathbf{C}^{-1} \varepsilon$, $\varepsilon = \mathbf{B} \mathbf{u}$. Assuming $\mathbf{u}_k$ and σ_k are approximate solutions, $\mathbf{u}$ and σ are exact solutions, the error point wise for displacements is:

$$\mathbf{e} = \mathbf{u} - \mathbf{u}_k$$

and for stresses is:

$$\mathbf{e}_\sigma = \sigma - \sigma_{\mathbf{k}}$$

The error in elements or in whole domain is the integral measure:

$$\begin{aligned} ||\mathbf{e}||_i^2 &= \int_i (\sigma - \sigma_{\mathbf{k}})^T \mathbf{C} (\sigma - \sigma_{\mathbf{k}}) \det \mathbf{J} \\ ||\mathbf{e}||^2 &= \int_\Omega (\sigma - \sigma_{\mathbf{k}})^T \mathbf{C} (\sigma - \sigma_{\mathbf{k}}) d\Omega = \sum_i^n ||\mathbf{e}||_i^2 \end{aligned}$$

where, i represents an element contribution and n the number of total elements.

Because the exact solution, σ , of the problem is usually not available, Zienkiewicz-Zhu [8] use the nodal average stresses $\bar{\sigma}^*$ and the same interpolation function as those used in the displacement, $\mathbf{u}_{\mathbf{k}} = \mathbf{N}\bar{\mathbf{u}}$, and compute a more accurate recovered stress, $\sigma^* = \mathbf{N}\bar{\sigma}^*$. Using this projected stress to replace the exact stress, one can get the practically useful error estimator:

$$||\mathbf{e}||_i^2 = \int_i (\sigma^* - \sigma_{\mathbf{k}})^T \mathbf{C} (\sigma^* - \sigma_{\mathbf{k}}) \det \mathbf{J}$$

Superconvergence is not observed in meshes mixing different element types, therefore flux projection estimates should not be used with such meshes [1]. Moreover, the recovered solution usually shows a poor quality near the boundaries. Nevertheless, flux projection estimators are the more popular and they have been implemented in many commercial finite element codes because of their easy implementation and practical efficiency. Flux projection estimates can be computed using only the existing framework of any classical finite element code.

H. Ohtsubo and M. Kitamura [12] have given some examples of estimators, which are difficult to classify into one of these groups.

1.1.3 The Relationship Between Error Estimators and Error Bound

As described above, the residual error estimator, the recovery error estimator and other type of error estimators can only tell us the finite element solutions in a particular region are more or less accurate than those in the rest of the domain. They can not tell us the error in the finite element approximate solutions with respect to the exact solution, because there is no concept of error bound involved in those error analysis scheme. In particular, the real error can be greater than the error estimate.

The duality method of error analysis, used in this thesis, employs a statically admissible stress field (or thermal flux field) to compute a bound on the error in the finite element solution. This error gives us a bound on the error of the particular problem. Using the error bound, one can more accurately evaluate the finite element approximate solutions and control the processes of mesh refinement. The real error cannot be greater than error bound.

1.2 Aims and Organization of the Thesis

1.2.1 Aims of the Thesis

This thesis is an extension of A. Mashaie's work [13]. A. Mashaie established an algorithm to calculate a statically admissible stress and a bound on the error, particularly with 8-node hexahedral element. He compared the SAS error estimate with the residual error estimate and recovery error estimate.

This thesis will focus on the application of A. Mashaie's algorithm and test an implementation of some algorithms into a current general finite element code, in order to:

1. Calculate a statically admissible stress and error estimators with 6-node prism element, 8-node hexahedral element and 20-node hexahedral element.
2. Calculate statically admissible thermal flux and error estimators with 6-node prism element and 8-node hexahedral element.

1.2.2 Organization of the Thesis

This thesis describes fundamental principles to calculate a statically admissible stress and a statically admissible thermal flux with different element types and solves test problems and presents practical applications

Chapter 1 gives an overview of a variety of posteriori error estimates and presents the organization of the thesis.

Chapter 2 presents the fundamental concept of a statically admissible stress and a kinematically admissible stress and the relationship between them, and a statically admissible thermal flux and a kinematically admissible thermal flux and the relationship between them.

Chapter 3 creates discrete SAS stress fields, develops matrix formulations that are implemented in the code in an existing FEM program and investigates the features of several element types for SAS calculation.

Chapter 4 creates discrete SAF thermal flux fields, develops matrix formulations for implementing code in existing FEM program and investigates the features of several element types for SAF calculation.

Chapter 5 conducts several test problems of SAS to verify the implemented code.

Chapter 6 conducts several test problems of SAF to verify the implemented code.

Chapter 7 summarizes the main results and conclusions.

Chapter 2

Fundamental Concepts

2.1 Kinematically and Statically Admissible Stress

This section, which is based largely on A. Mashaie [13], will define and discuss the kinematically admissible stress and the statically admissible stress. Before doing that, it is necessary to recall some equations which are essential for stress analysis.

2.1.1 Basic Equations

Let u be a displacement field, σ be a stress field, $\partial\Omega$ be boundary of the domain Ω . For generality, we assume mixed boundary conditions prescribed on two complementary sub-boundaries $\partial_1\Omega$ and $\partial_2\Omega$. For example, let the displacement $\bar{u}$ be specified on $\partial_1\Omega$ and the traction $\bar{T}_n$ on $\partial_2\Omega$. We would like to compute the solutions u and σ on

Ω by using the finite element method based on the following equations.

(1) Equation of equilibrium

$$\nabla \cdot \sigma + f = 0 \quad (2.1)$$

Writing in component form

$$\frac{\partial \sigma_{xx}}{\partial x} + \frac{\partial \sigma_{xy}}{\partial y} + \frac{\partial \sigma_{xz}}{\partial z} + f_x = 0$$

$$\frac{\partial \sigma_{xy}}{\partial x} + \frac{\partial \sigma_{yy}}{\partial y} + \frac{\partial \sigma_{yz}}{\partial z} + f_y = 0$$

$$\frac{\partial \sigma_{xz}}{\partial x} + \frac{\partial \sigma_{yz}}{\partial y} + \frac{\partial \sigma_{zz}}{\partial z} + f_z = 0$$

Where, f is the body force with components f_x, f_y, f_z for an x, y, z coordinate system.

(2) Equation of continuity

$$\varepsilon = Bu \quad (2.2)$$

Where, $\varepsilon = \{\varepsilon_x, \varepsilon_y, \varepsilon_z, \gamma_{xy}, \gamma_{yz}, \gamma_{xz}\}^T$ is the strain vector, $u = \{u_x, u_y, u_z\}^T$ is displacement at a point x, y, z in Ω . B is the gradient operator,

$$B = \begin{bmatrix} \frac{\partial}{\partial x} & 0 & 0 \\ 0 & \frac{\partial}{\partial y} & 0 \\ 0 & 0 & \frac{\partial}{\partial z} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} & 0 \\ 0 & \frac{\partial}{\partial z} & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} & 0 & \frac{\partial}{\partial x} \end{bmatrix}$$

(3) Constitutive Equation

$$\sigma = C^{-1}\varepsilon \quad (2.3)$$

where, $\sigma = \{\sigma_{xx}, \sigma_{yy}, \sigma_{zz}, \sigma_{xy}, \sigma_{yz}, \sigma_{xz}\}^T$ is the stress vector, C is the matrix of physical properties, e.g., for a 3D isotropic elastic material,

$$C = \frac{E}{(1+\nu)(1-2\nu)} \begin{bmatrix} 1-\nu & \nu & \nu & 0 & 0 & 0 \\ \nu & 1-\nu & \nu & 0 & 0 & 0 \\ \nu & \nu & 1-\nu & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{(1+\nu)}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{(1+\nu)}{2} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{(1+\nu)}{2} \end{bmatrix}$$

(4) Boundary condition

$$u|_{\partial_1\Omega} = \bar{u} \quad (2.4)$$

$$n^T \sigma|_{\partial_2\Omega} = \bar{T}_n \quad (2.5)$$

2.1.2 Kinematically Admissible Stress (KAS)

A kinematically admissible stress (KAS) field is defined as the stress field, whose displacement field satisfies the equation of continuity, $\varepsilon = Bu$, in the interior and on the boundary $\partial_1\Omega$ of domain Ω , where the displacement boundary condition $u = \bar{u}$

is prescribed. In continuum mechanics, the displacement finite element method is usually used to solve for the displacement field u . Then the KAS stress field σ_k is computed from the displacement field using the following equation:

$$\sigma_k = C^{-1}Bu$$

The computed KAS stress field σ_k need not satisfy the equations of equilibrium at each point in the interior of domain, i.e.,

$$\nabla \cdot \sigma_k + f \neq 0.$$

In addition, the traction vector computed from the KAS field can be a non-equilibrating field across the boundaries of elements, in other words, the elements are not balanced by boundary tractions. In a general case, the traction vector computed from a σ_k stress field has a jump across the boundaries of elements and the associated stress field is statically admissible if and only if the error is zero.

2.1.3 Statically Admissible Stress (SAS)

A statically admissible stress (SAS) field is defined as the stress field that satisfies the static equilibrium conditions in the interior and on the boundaries of domain. In other words, the statically admissible stress field satisfies the equation of equilibrium $\nabla \cdot \sigma + f = 0$ pointwise in the interior of the domain and the stress vector satisfies the traction boundary conditions pointwise, $n^T \sigma = \bar{T}_n$, on the external boundaries $\partial_2 \Omega$ of the domain.

In addition, the traction vector computed from the SAS stress field does not have a jump across the boundaries of elements, in other words, the associated element tractions are a single valued function across the boundaries of interior elements. The elements traction are in equilibrium across the boundaries of elements.

The SAS stress field need not satisfy the continuity equation. In other words, if the displacement field of the SAS stress field exists, then the displacement field might not be continuous across the boundaries of elements and might not satisfy the displacement boundary condition $u = \bar{u}$ on the external boundaries $\partial_1\Omega$ of the domain.

2.1.4 Error Bound

A norm of the stress vector is defined in the strain energy measured by the following inner product:

$$SE = \int_{\Omega} \sigma^T C \sigma d\Omega \quad (2.6)$$

From this equation, KAS strain energy and SAS strain energy in the whole domain Ω are separately expressed as:

$$SE_{KAS} = \int_{\Omega} \sigma_k^T C \sigma_k d\Omega \quad (2.7)$$

$$SE_{SAS} = \int_{\Omega} \sigma_s^T C \sigma_s d\Omega \quad (2.8)$$

where, σ_k and σ_s indicate the KAS stress field and the SAS stress field, respectively.

The element error bound is defined as:

$$\Delta SE_i = \int_{e_i} (\sigma_s - \sigma_k)^T C (\sigma_s - \sigma_k) det J \quad (2.9)$$

The element relative error is defined as:

$$E_i = \frac{\Delta SE_i}{0.5 \times (SE_{KAS} + SE_{SAS})} \quad (2.10)$$

The total error bound in Ω is defined as:

$$\Delta SE = \int_{\Omega} (\sigma_s - \sigma_k)^T C (\sigma_s - \sigma_k) d\Omega \quad (2.11)$$

The total relative error in Ω is defined as:

$$E_{tot} = \frac{\Delta SE}{0.5 \times (SE_{KAS} + SE_{SAS})} \quad (2.12)$$

2.1.5 The Relationship Between KAS and SAS

The first objective is to construct a SAS field in order to compute an error-bound.

Using a displacement finite element method, one can compute a kinematically admissible stress field. To compute a statically admissible stress from kinematically

admissible stress, one has to find some fundamental relationship between KAS and SAS.

Figure (2.1) provides a explicit form of definition of KAS and SAS. A KAS field satisfies the displacement boundary condition and a SAS field satisfies the traction boundary condition.

Figure 2.1: Relationship between KAS and SAS

For easily describing the properties of KAS and SAS, we introduce the concepts of homogeneous and inhomogeneous stress fields.

An inhomogeneous KAS stress field is defined as a KAS stress field with non-zero displacement boundary condition, i.e.

$$\Sigma = \{\sigma | \sigma = C^{-1} B u, u|_{\partial_1\Omega} = \bar{u}\}$$

An homogeneous KAS stress field is defined as a KAS stress field with zero displacement boundary condition, i.e.

$$\Sigma_0 = \{\sigma | \sigma = C^{-1} B u, u|_{\partial_1\Omega} = 0\}$$

An inhomogeneous SAS stress field is defined as a SAS stress field with non-zero traction boundary condition, i.e.

$$\Pi = \{\sigma | \sigma = C^{-1}\varepsilon, \nabla\sigma + f = 0 \text{ in } \Omega, n^T\sigma|_{\partial_2\Omega} = \bar{T}_n\}$$

An homogeneous SAS stress field is defined as a SAS stress field with zero traction boundary condition and zero body force, i.e.

$$\Pi_0 = \{\sigma | \sigma = C^{-1}\varepsilon, \nabla\sigma = 0 \text{ in } \Omega, n^T\sigma|_{\partial_2\Omega} = 0\}$$

From these equations, the relationship between KAS and SAS can be described using two lemmas that has been proved by A. Mashaie [13].

Lemma 2.1:

The homogeneous kinematically and homogeneous statically admissible stress spaces, Σ_0 and Π_0 , are mutually orthogonal with the respect to inner product

$$\int_{\Omega} \sigma_k^T C \sigma_s d\Omega = 0. \quad (2.13)$$

From this lemma, it can be concluded that the intersection of an inhomogeneous KAS field and an inhomogeneous SAS field is the exact solution of problem subject to the five equations (Figure 2.1) at the same time and this solution $(u_{exact}, \sigma_{exact})$ is unique. The geometry of the statically and kinematically admissible stress fields and the exact solution is presented in Figure (2.2).

Figure 2.2: Stress field and exact solution

Lemma 2.2:

The difference between a kinematically admissible stress field and a statically admissible stress field associated with the problem in norm (2.6) is an upper bound for the difference between the kinematically admissible stress field and the exact solution of the problem and for the difference between the statically admissible stress field and the exact solution of the problem in that norm.

In mathematical form:

$$||\sigma_k - \sigma_{exact}||^2 \leq ||\sigma_k - \sigma_s||^2$$

$$||\sigma_s - \sigma_{exact}||^2 \leq ||\sigma_k - \sigma_s||^2$$

where, σ_{exact} is the exact stress solution.

σ_k is a kinematically admissible solution of the problem.

σ_s is a statically admissible solution of the problem.

This relationship can also be expressed as: the hypotenuse of a right-angled triangle is bigger than either of its two legs (Figure 2.2).

2.2 Kinematically and Statically Admissible Thermal Flux

This section will define and discuss the kinematically admissible thermal flux and the statically admissible thermal flux. Before doing that, it's necessary to recall some equations which are essential for thermal analysis.

2.2.1 Basic Equations

Let T be the temperature, $\dot{h}$ be the rate of change of specific enthalpy, p be the rate of heat generation per unit volume, q be the heat flux per unit area, $\partial\Omega$ be the boundary of the domain Ω . For generality, we assume mixed boundary conditions prescribed on two complementary sub-boundaries $\partial_1\Omega$ and $\partial_2\Omega$. For example, let the temperature $\bar{T}$ be specified on $\partial_1\Omega$ and heat flux $\bar{q}$ on $\partial_2\Omega$. We would like to compute the solutions T and q on Ω by using the finite element method based on the following equations [20].

(1) Energy Equation

$$\dot{h} + \nabla \cdot q + p = 0 \quad (2.14)$$

writing in component form:

$$\dot{h} + \frac{\partial q_x}{\partial x} + \frac{\partial q_y}{\partial y} + \frac{\partial q_z}{\partial z} + p = 0 \quad (2.15)$$

where, q_x, q_y and q_z are components of the heat flux rate vector per unit area in Cartesian coordinates (x, y, z) .

(2) Conduction

Conduction is the flow of thermal energy through a solid or fluid due to a temperature gradient. The constitutive equation describing this heat transfer mode is Fourier's law. For an isotropic medium, Fourier's law is:

$$q = -k \frac{\partial T}{\partial n} \quad (2.16)$$

where, k is the thermal conductivity that may be a function of the temperature T and n indicates a normal direction. In Cartesian coordinates, the components of the heat flux q are:

$$q_x = -k \frac{\partial T}{\partial x}, \quad q_y = -k \frac{\partial T}{\partial y}, \quad q_z = -k \frac{\partial T}{\partial z} \quad (2.17)$$

(3) Boundary Conditions

Energy equation (2.14) is solved subject to an initial condition and appropriate boundary conditions. The initial condition consists of specifying the temperature throughout the solid at an initial time. The boundary conditions can take several forms and may be linear or nonlinear and must be specified on all boundaries of the solid. Typical boundary conditions encountered in finite element formulations are the following:

$$T|_{\partial_1\Omega} = \bar{T} \quad (2.18)$$

$$q|_{\partial_2\Omega} = \bar{q} \quad (2.19)$$

where, the surface temperature $\bar{T}$ on boundary $\partial_1\Omega$ is a specified function of boundary position and time. The rate of heat flow $\bar{q}$ across a boundary $\partial_2\Omega$ is specified to be a function of boundary position and time, which can be predicted from convective heat exchange surface or radiation heat exchange surface or a combination of both of them.

(a) Convection BC

The rate of convection heat flux across a boundary is proportional to the difference between the surface temperature and a convective exchange temperature of an adjacent fluid, e.g.,

$$q_v = h\Delta T \quad (2.20)$$

where, h is a convection heat transfer coefficient.

(b) Radiation BC

The rate of radiation heat flow across a boundary is specified in terms of the emitted energy from the surface and the incident radiant thermal energy, emitted and reflected from other solids and/or fluids.

$$q_r = \sigma\epsilon(T_s^4 - T_e^4) \quad (2.21)$$

where, σ is Stefan-Boltzmann constant ($\sigma = 5.67 \times 10^{-8} \text{ w/m}^2 \text{ k}^4$), ϵ is the surface emissivity, T_s is the surface temperature and T_e is ambient temperature.

2.2.2 Kinematically Admissible Flux (KAF)

A kinematically admissible thermal flux (KAF) field is defined as a thermal flux field, whose temperature field satisfies the continuity condition in the interior and on the boundary $\partial_1\Omega$ of domain Ω , where the temperature boundary condition $T = \bar{T}$ is prescribed.

In thermal analysis, the temperature finite element method is usually used to solve for the temperature field T , then the KAF thermal flux q is computed from the temperature field using the following equation:

$$q_k = -[k]BT$$

where, $q_k = \begin{Bmatrix} q_x \\ q_y \\ q_z \end{Bmatrix}$ is thermal flux vector.

$[k] = \begin{bmatrix} k_{xx} & 0 & 0 \\ 0 & k_{yy} & 0 \\ 0 & 0 & k_{zz} \end{bmatrix}$ is the thermal conductivity tensor represented as a matrix.

$B = \begin{bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} \end{bmatrix}$ is the temperature gradient operator matrix.

T is the temperature at point x, y, z in Ω .

The computed KAF thermal flux field q_k need not satisfy the energy equations point-wise in the interior of domain, i.e.,

$$\dot{h} + \nabla \cdot q_k + p \neq 0$$

In addition, the thermal flux vector computed from the KAF field can be a non-equilibrated field across the boundaries of elements. In other words, the KAF thermal flux is not continuous across element boundaries. In the general case, the KAF thermal flux field has a jump across the boundaries of elements and the associated thermal flux field is not statically admissible.

2.2.3 Statically Admissible Flux (SAF)

A statically admissible thermal flux (SAF) field is defined as the thermal flux field that satisfies the energy equation at every point in the interior and on the boundaries of domain. In other words, a statically admissible thermal flux field satisfies the energy equation $\dot{h} + \nabla \cdot q + p = 0$ in the interior of the domain and the flux vector satisfies the flux boundary conditions, $q = \bar{q}$, on the external boundaries $\partial_2\Omega$ of the domain.

In addition, the thermal flux vector in the SAF thermal flux field does not have a jump across the boundaries of elements. In other words, the associated element flux is single valued across the boundaries of interior elements. The element thermal fluxes are in equilibrium across the boundaries of elements.

A SAF thermal flux field need not satisfy temperature continuity condition, in other words, if the temperature in a SAF thermal flux field exists, then the temperature field might not be continuous across the boundaries of elements and might not satisfy the temperature boundary condition $T = \bar{T}$ on the external boundaries $\partial_1\Omega$ of the domain.

2.2.4 Error Bound

The norm of a thermal flux vector is defined in the form of energy measured by the following inner product:

$$SE = \int_{\Omega} q^T k^{-1} q d\Omega \quad (2.22)$$

From this equation, KAF energy and SAF energy in the whole domain Ω are separately expressed as:

$$SE_{KAF} = \int_{\Omega} q_k^T k^{-1} q_k d\Omega \quad (2.23)$$

$$SE_{SAF} = \int_{\Omega} q_s^T k^{-1} q_s d\Omega \quad (2.24)$$

where, q_k and q_s indicate KAF thermal flux and SAF thermal flux separately.

The element error bound is defined as:

$$\Delta SE_i = \int_{e_i} (q_s - q_k)^T k^{-1} (q_s - q_k) det J \quad (2.25)$$

The element relative error is defined as:

$$SE_i = \frac{\Delta SE_i}{0.5 \times (SE_{KAF} + SE_{SAF})} \quad (2.26)$$

The total error bound is defined as:

$$\Delta SE = \int_{\Omega} (q_s - q_k)^T k^{-1} (q_s - q_k) d\Omega \quad (2.27)$$

The total relative error is defined as:

$$SE_{tot} = \frac{\Delta SE}{0.5 \times (SE_{KAF} + SE_{SAF})} \quad (2.28)$$

2.2.5 The Relationship Between KAF and SAF

The reason for constructing a SAF field is to compute the error-bound. Using the temperature finite element method, one can compute a kinematically admissible thermal flux field. To compute a statically admissible thermal flux from a kinematically admissible thermal flux, one has to find some fundamental relationship between KAF and SAF.

Figure (2.3) provides a explicit form of definition of KAF and SAF. A KAF field satisfies the temperature (Dirichlet) boundary condition and a SAF field satisfies the thermal flux (Neumann) boundary condition.

Figure 2.3: Relationship between KAF and SAF

For easily describing the properties of KAF and SAF, we introduce the concepts of homogeneous and inhomogeneous thermal flux fields.

An inhomogeneous KAF thermal flux field is defined as a KAF thermal flux field with a non-zero temperature boundary condition, i.e.

$$\Sigma = \{q|q = -k\frac{\partial T}{\partial n}, T|_{\partial_1\Omega} = \bar{T}\} \quad (2.29)$$

An homogeneous KAF thermal flux field is defined as a KAF thermal flux field with a zero temperature boundary condition, i.e.

$$\Sigma_0 = \{q|q = -k\frac{\partial T}{\partial n}, T|_{\partial_1\Omega} = 0\} \quad (2.30)$$

An inhomogeneous SAF thermal flux field is defined as a SAF thermal flux field with a non-zero thermal flux boundary condition, i.e.,

$$\Pi = \{q|\nabla \cdot q + p + \dot{h} = 0 \text{ in } \Omega, q|_{\partial_2\Omega} = \bar{q}\} \quad (2.31)$$

An homogeneous SAF thermal flux field is defined as a SAF thermal flux field with a zero thermal flux boundary condition, a zero change of specific enthalpy and a zero energy production, i.e.

$$\Pi_0 = \{q|\nabla \cdot q = 0 \text{ in } \Omega, q|_{\partial_2\Omega} = 0\} \quad (2.32)$$

From these equations, the relationship between KAF and SAF can be described using two lemmas which have been proved by A. Mashaie [13].

Lemma 2.3:

The homogeneous kinematically and homogeneous statically admissible thermal flux space, Σ_0 and Π_0 , are mutually orthogonal with the respect to the inner product

$$\int_{\Omega} q_k^T k^{-1} q_s d\Omega = 0. \quad (2.33)$$

From this lemma, it can be concluded that the intersection of KAF field and SAF field is the exact solution of the problem subject to the five constraints in Figure (2.3) and this solution (T_{exact}, q_{exact}) is unique.

Figure 2.4: Thermal flux field and exact solution

Lemma 2.4:

The difference between a kinematically admissible thermal flux field and a statically admissible thermal flux field associated with the problem in the norm (2.6) is an upper bound for the difference between the kinematically admissible thermal flux field and the exact solution of the problem and for the difference between the statically admissible thermal flux field and the exact solution of the problem in that norm.

In mathematical form:

$$\begin{aligned} ||q_k - q_{exact}||^2 &\leq ||q_k - q_s||^2 \\ ||q_s - q_{exact}||^2 &\leq ||q_k - q_s||^2 \end{aligned}$$

where, q_{exact} is an exact thermal flux solution.

q_k is a kinematically admissible solution of the problem.

q_s is a statically admissible solution of the problem.

This relationship can also be expressed as: the hypotenuse of a right-angled triangle is bigger than either of its two legs (Figure 2.4).

Chapter 3

Discrete SAS Fields

The theory in this chapter follows J. Goldak [14][16] and A. Mashaie [13].

3.1 A. Mashaie's Traction Force Algorithm

In order to construct a statically admissible stress field, one must define a stress field that satisfies the conditions described in section (2.1.3). A. Mashaie [13] has developed an algorithm for constructing the SAS field σ_s in a 4-node quad element and an 8-node hexahedral element from a KAS field.

In the following sections, we will derive mathematically the detailed matrix equations being used to develop a computer program. All of these equations can be used for a 4-node tetrahedral element, a 6-node prism element, an 8-node hexahedral element and a 20-node hexahedral element if we set the element type as variable. Unless

explicitly stated, all discussion will refer to an 8-node brick with 4-node quad for each face.

The algorithm consists of the following steps:

1. Solve for the KAS displacement, strain and stress.
2. Solve for the KAS nodal force vector for each element and the nodal force vector for each face. The nodal forces on each face are not in equilibrium.
3. Solve for the SAS nodal forces vector on each face due to tractions.
4. Solve for the SAS stress in each element. Note that this is nodal data in the element. It is not global data. If a node is shared by eight elements, then this node can have a different value of SAS stress for this node for each element.
5. Solve for the error bound in each element and the global error bound.

Our work assumes that the KAS results exist and we will start from step 2 in the following section.

3.2 KAS Nodal Forces

3.2.1 KAS Nodal Force on Element

The vector of KAS nodal force on each element can be evaluated from the equation

$\mathbf{F}_k^e = \int_V \mathbf{B}^T \sigma_k dV$. where, $\sigma_k = \{\sigma_{xx}, \sigma_{yy}, \sigma_{zz}, \sigma_{xy}, \sigma_{yz}, \sigma_{xz}\}^T$ is the stress vector at

the point (ξ, η, ζ) in the reference element. $\mathbf{B}^T$ is the divergence matrix which maps the stress vector at the point (ξ, η, ζ) in the reference element to the vector of nodal loads on that element. For an 8-node hexahedral element, $\mathbf{B}^T$ is $(3 \times 8) \times 6 = 24 \times 6$ matrix.

$$\mathbf{B} = \left[\begin{array}{c} \left[\begin{array}{ccc} \frac{\partial N_i}{\partial x} & 0 & 0 \\ 0 & \frac{\partial N_i}{\partial y} & 0 \\ 0 & 0 & \frac{\partial N_i}{\partial z} \\ \frac{\partial N_i}{\partial y} & \frac{\partial N_i}{\partial x} & 0 \\ 0 & \frac{\partial N_i}{\partial z} & \frac{\partial N_i}{\partial y} \\ \frac{\partial N_i}{\partial z} & 0 & \frac{\partial N_i}{\partial x} \end{array} \right]_{i=1} & \left[\begin{array}{c} \\ \\ \\ \\ \\ \end{array} \right]_{i=2} & \dots & \left[\begin{array}{c} \\ \\ \\ \\ \\ \end{array} \right]_{i=8} \end{array} \right]_{6 \times 24}$$

$\mathbf{F}_k^e$ is $(3 \times 8) \times 1 = 24 \times 1$ vector:

$$\mathbf{F}_k^e = \{P^{e_1}, Q^{e_1}, R^{e_1}, \quad P^{e_2}, Q^{e_2}, R^{e_2}, \quad \dots, \quad P^{e_8}, Q^{e_8}, R^{e_8}\} \quad (3.1)$$

Here, P, Q, R refers to the components of KAS nodal force on the x, y, z axes. The superscript $e_1, e_2, \dots, e_8$ refers to the first, second, ..., eighth node of the element.

3.2.2 KAS Nodal Force on Face

On each face, the vector of nodal loads due to KAS tractions can be evaluated from the equation $\mathbf{F}_k^f = \int_S [\hat{R}] \tau dS$, where, $\tau = \{\tau_x, \tau_y, \tau_z\}^T$ is the KAS traction vector at the point (ξ, η) on the given face which can be calculated using the stress tensor at

that point times the normal vector, $\mathbf{n} = \{n_x, n_y, n_z\}^T$, of the face at the same point.

$$\tau = \begin{Bmatrix} \tau_x \\ \tau_y \\ \tau_z \end{Bmatrix} = \begin{pmatrix} \sigma_{xx} & \sigma_{xy} & \sigma_{xz} \\ \sigma_{xy} & \sigma_{yy} & \sigma_{yz} \\ \sigma_{xz} & \sigma_{yz} & \sigma_{zz} \end{pmatrix} \begin{Bmatrix} n_x \\ n_y \\ n_z \end{Bmatrix} \quad (3.2)$$

and $[\hat{R}]$ is the matrix of face basis function.

$$[\hat{R}] = \begin{bmatrix} R_f^T & 0 & 0 \\ 0 & R_f^T & 0 \\ 0 & 0 & R_f^T \end{bmatrix}_{12 \times 3} \quad (3.3)$$

Here R_f is the vector of face basis functions. For example, $R_f = \{N_1, N_2, N_3, N_4\}$ for a 4-node quad face. Each interior face has contributions from two elements. The average of the vectors of face nodal loads from those two elements is

$$\mathbf{F}_k^f = \frac{1}{2} \left[\int_{S_1} [\hat{R}] \tau dS_1 - \int_{S_2} [\hat{R}] \tau dS_2 \right] \quad (3.4)$$

for a 4-node quad face, $\mathbf{F}_k^f$ is 12×1 vector.

Note that the minus sign reflects the different sign of the outward normals of the two sides of the face.

$$\mathbf{F}_k^f = \{P^{f_1}, P^{f_2}, P^{f_3}, P^{f_4}, \quad Q^{f_1}, Q^{f_2}, Q^{f_3}, Q^{f_4}, \quad R^{f_1}, R^{f_2}, R^{f_3}, R^{f_4}\}^T \quad (3.5)$$

here, P, Q, R refers to the components of KAS nodal force on the x, y, z axes. The

superscript f_1, f_2, f_3, f_4 refers to the first, second, third and fourth node of the face of a 4-node quad.

Each exterior face has contributions from one face. If an exterior face has Neumann BC, use the Neumann tractions, not the KAS traction. If the face has Dirichlet BC, use the KAS traction.

To calculate the average of nodal loads on a face, we have to set the face orientation vector globally so that the first element ID has orientation $+n$ and the second element ID has orientation $-n$. This orientation is global and will not be changed throughout the SAS nodal force calculation.

3.3 SAS Nodal Forces on Face

For a given node, let N be the number of faces shared by this node, M be the number of elements shared by this node,

$\mathbf{X} = \{x^{f_1}, x^{f_2}, \dots, x^{f_N}\}^T$ be the vector of the x-component SAS nodal force on faces $\{f_1, f_2, \dots, f_N\}$

$\mathbf{Y} = \{y^{f_1}, y^{f_2}, \dots, y^{f_N}\}^T$ be the vector of the y-component SAS nodal force on faces $\{f_1, f_2, \dots, f_N\}$

$\mathbf{Z} = \{z^{f_1}, z^{f_2}, \dots, z^{f_N}\}^T$ be the vector of the z-component SAS nodal force on faces $\{f_1, f_2, \dots, f_N\}$

We wish to set up a system of equations in order to solve for the unknown vectors

X, Y, Z.

To do that, form a $M \times N$ matrix $[K_0]$ initialized to all zeros, in which the number of rows M equals to the number of elements shared by this node and the number of columns N equals to the number of faces shared by this node. The entry K_{0ij} of $[K_0]$ associated with row i and column j . Row i refers to element i and column j refers to face j .

For each element shared by a specific face, set the entries in the row for the face in the first element to +1 or -1 depending on whether this element is the first or second element for this face. These face orientations have been defined in section (3.2.2).

Also, form a x-component vector $\mathbf{R}_x^e$ with size $M \times 1$. For each element shared by this node, assign the x-components of KAS element nodal force $\{P^{e_1}, P^{e_2}, \dots, P^{e_M}\}^T$ to $\mathbf{R}_x^e$. $e_1, e_2, \dots, e_M$ refers to the first, second, ..., and Mth element shared by this node.

Use the Singular Value Decomposition (SVD) [18] algorithm to compute the null space of the matrix $[K_0]$. This null space forms a new matrix $[K_n]$. Multiply $[K_n]$ by x-component vector of KAS nodal force on faces sharing this node, $\{P^{f_1}, P^{f_2}, \dots, P^{f_N}\}^T$, we get a new vector $\mathbf{R}_x^f = [K_n]\{P^{f_1}, P^{f_2}, \dots, P^{f_N}\}^T$. Combine matrix $[K_0]$ and $[K_n]$ into $[K] = \begin{bmatrix} K_0 \\ K_n \end{bmatrix}$, and the vector $\mathbf{R}_x^e$ and $\mathbf{R}_x^f$ into $\mathbf{R}_x = \begin{Bmatrix} \mathbf{R}_x^e \\ \mathbf{R}_x^f \end{Bmatrix}$. A. Mashaie [13] has shown that $\mathbf{X}$, the vector of x-component of SAS nodal force on faces shared by this node, can be computed by solving the following system equation:

$$[K]\mathbf{X} = \mathbf{R}_x \quad (3.6)$$

Similarly, we can construct the y-component and the z-component vectors

$$\mathbf{R}_y = \begin{Bmatrix} \mathbf{R}_y^e \\ \mathbf{R}_y^f \end{Bmatrix}$$

and

$$\mathbf{R}_z = \begin{Bmatrix} \mathbf{R}_z^e \\ \mathbf{R}_z^f \end{Bmatrix}$$

to compute vectors of the y-component and the z-component of the SAS nodal force on faces sharing the same node:

$$[K]\mathbf{Y} = \mathbf{R}_y \quad (3.7)$$

$$[K]\mathbf{Z} = \mathbf{R}_z \quad (3.8)$$

where,

$$\mathbf{R}_y^e = \{Q^{e_1}, Q^{e_2}, \dots, Q^{e_M}\}^T$$

$$\mathbf{R}_z^e = \{R^{e_1}, R^{e_2}, \dots, R^{e_M}\}^T$$

$$\mathbf{R}_y^f = [K_n]\{Q^{f_1}, Q^{f_2}, \dots, Q^{f_N}\}^T$$

$$\mathbf{R}_z^f = [K_n]\{R^{f_1}, R^{f_2}, \dots, R^{f_N}\}^T$$

Note that, for the same node, the matrix $[K]$ is the same for the x , y and z systems of equations.

As an example, we will give details of how to create the matrix $[K]$ and the vector $\mathbf{R}_x$ of node i in Figure (3.1).

Figure 3.1: Geometry of the set of elements and faces containing the node i

There are 8 elements $(e_1, e_2, \dots, e_8)$ and 12 faces $(f_1, f_2, \dots, f_{12})$ that share the node i (Figure 3.1), so the size of the matrix $[K_0]$ is 8×12 .

$$[K_0] = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & -1 \end{bmatrix}_{8 \times 12}$$

In this matrix, each column corresponds to a face shared by this node. Each column has two nonzero entries (+1 and -1) that correspond to faces shared by two elements. Each row of this matrix corresponds to the elements shared by this node and in each row the three nonzero entries correspond to the three faces shared by node i located on this element.

Using SVD, compute the null space matrix $[K_n]$ of $[K_0]$:

$$[K_n] = \begin{bmatrix} -0.6455 & 0.3227 & 0.3227 & -0.3227 & -0.3227 & 0.2582 & 0.0645 & -0.0645 & 0.2582 & 0.0645 & -0.0645 & 0.1291 \\ 0.0000 & 0.1376 & -0.1376 & -0.2201 & 0.2201 & -0.3384 & 0.4760 & -0.5585 & -0.0922 & -0.0454 & 0.1279 & 0.4306 \\ 0.0000 & -0.4945 & 0.4945 & -0.0571 & 0.0571 & -0.1772 & -0.3173 & -0.2343 & -0.0286 & 0.5231 & 0.0285 & 0.2058 \\ 0.0000 & -0.2081 & 0.2081 & 0.1080 & -0.1080 & -0.2714 & 0.0633 & -0.1634 & 0.5780 & -0.3699 & 0.4700 & -0.3066 \\ 0.0000 & -0.0761 & 0.0761 & 0.4991 & -0.4991 & -0.3612 & 0.2851 & 0.1380 & 0.0814 & -0.0054 & -0.4177 & 0.2797 \end{bmatrix}$$

The matrix $[K] = \begin{bmatrix} K_0 \\ K_n \end{bmatrix}$ is:

$$[K] = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & -1 \\ -0.6455 & 0.3227 & 0.3227 & -0.3227 & -0.3227 & 0.2582 & 0.0645 & -0.0645 & 0.2582 & 0.0645 & -0.0645 & 0.1291 \\ 0.0000 & 0.1376 & -0.1376 & -0.2201 & 0.2201 & -0.3384 & 0.4760 & -0.5585 & -0.0922 & -0.0454 & 0.1279 & 0.4306 \\ 0.0000 & -0.4945 & 0.4945 & -0.0571 & 0.0571 & -0.1772 & -0.3173 & -0.2343 & -0.0286 & 0.5231 & 0.0285 & 0.2058 \\ 0.0000 & -0.2081 & 0.2081 & 0.1080 & -0.1080 & -0.2714 & 0.0633 & -0.1634 & 0.5780 & -0.3699 & 0.4700 & -0.3066 \\ 0.0000 & -0.0761 & 0.0761 & 0.4991 & -0.4991 & -0.3612 & 0.2851 & 0.1380 & 0.0814 & -0.0054 & -0.4177 & 0.2797 \end{bmatrix}$$

The vector $\mathbf{R}_x^f = [K_n]\{P^{f_1}, P^{f_2}, \dots, P^{f_N}\}^T$ is:

$$\mathbf{R}_x^f = \begin{bmatrix} -0.6455 P^{f_1} + 0.3227 P^{f_2} + 0.3227 P^{f_3} - 0.3227 P^{f_4} - 0.3227 P^{f_5} + 0.2582 P^{f_6} + 0.0645 P^{f_7} - 0.0645 P^{f_8} + 0.2582 P^{f_9} + 0.0645 P^{f_{10}} - 0.0645 P^{f_{11}} + 0.1291 P^{f_{12}} \\ 0.0000 P^{f_1} + 0.1376 P^{f_2} - 0.1376 P^{f_3} - 0.2201 P^{f_4} + 0.2201 P^{f_5} - 0.3384 P^{f_6} + 0.4760 P^{f_7} - 0.5585 P^{f_8} - 0.0922 P^{f_9} - 0.0454 P^{f_{10}} + 0.1279 P^{f_{11}} + 0.4306 P^{f_{12}} \\ 0.0000 P^{f_1} - 0.4945 P^{f_2} + 0.4945 P^{f_3} - 0.0571 P^{f_4} + 0.0571 P^{f_5} - 0.1772 P^{f_6} - 0.3173 P^{f_7} - 0.2343 P^{f_8} - 0.0286 P^{f_9} + 0.5231 P^{f_{10}} + 0.0285 P^{f_{11}} + 0.2058 P^{f_{12}} \\ 0.0000 P^{f_1} - 0.2081 P^{f_2} + 0.2081 P^{f_3} + 0.1080 P^{f_4} - 0.1080 P^{f_5} - 0.2714 P^{f_6} + 0.0633 P^{f_7} - 0.1634 P^{f_8} + 0.5780 P^{f_9} - 0.3699 P^{f_{10}} + 0.4700 P^{f_{11}} - 0.3066 P^{f_{12}} \\ 0.0000 P^{f_1} - 0.0761 P^{f_2} + 0.0761 P^{f_3} + 0.4991 P^{f_4} - 0.4991 P^{f_5} - 0.3612 P^{f_6} + 0.2851 P^{f_7} + 0.1380 P^{f_8} + 0.0814 P^{f_9} - 0.0054 P^{f_{10}} - 0.4177 P^{f_{11}} + 0.2797 P^{f_{12}} \end{bmatrix}$$

The vector $\mathbf{R}_x = \begin{Bmatrix} \mathbf{R}_x^e \\ \mathbf{R}_x^f \end{Bmatrix}$ is:

$$\mathbf{R}_x = \begin{bmatrix} p^{e_1} \\ p^{e_2} \\ p^{e_3} \\ p^{e_4} \\ p^{e_5} \\ p^{e_6} \\ p^{e_7} \\ p^{e_8} \\ -0.6455 p^{f_1} + 0.3227 p^{f_2} + 0.3227 p^{f_3} - 0.3227 p^{f_4} - 0.3227 p^{f_5} + 0.2582 p^{f_6} + 0.0645 p^{f_7} - 0.0645 p^{f_8} + 0.2582 p^{f_9} + 0.0645 p^{f_{10}} - 0.0645 p^{f_{11}} + 0.1291 p^{f_{12}} \\ 0.0000 p^{f_1} + 0.1376 p^{f_2} - 0.1376 p^{f_3} - 0.2201 p^{f_4} + 0.2201 p^{f_5} - 0.3384 p^{f_6} + 0.4760 p^{f_7} - 0.5585 p^{f_8} - 0.0922 p^{f_9} - 0.0454 p^{f_{10}} + 0.1279 p^{f_{11}} + 0.4306 p^{f_{12}} \\ 0.0000 p^{f_1} - 0.4945 p^{f_2} + 0.4945 p^{f_3} - 0.0571 p^{f_4} + 0.0571 p^{f_5} - 0.1772 p^{f_6} - 0.3173 p^{f_7} - 0.2343 p^{f_8} - 0.0286 p^{f_9} + 0.5231 p^{f_{10}} + 0.0285 p^{f_{11}} + 0.2058 p^{f_{12}} \\ 0.0000 p^{f_1} - 0.2081 p^{f_2} + 0.2081 p^{f_3} + 0.1080 p^{f_4} - 0.1080 p^{f_5} - 0.2714 p^{f_6} + 0.0633 p^{f_7} - 0.1634 p^{f_8} + 0.5780 p^{f_9} - 0.3699 p^{f_{10}} + 0.4700 p^{f_{11}} - 0.3066 p^{f_{12}} \\ 0.0000 p^{f_1} - 0.0761 p^{f_2} + 0.0761 p^{f_3} + 0.4991 p^{f_4} - 0.4991 p^{f_5} - 0.3612 p^{f_6} + 0.2851 p^{f_7} + 0.1380 p^{f_8} + 0.0814 p^{f_9} - 0.0054 p^{f_{10}} - 0.4177 p^{f_{11}} + 0.2797 p^{f_{12}} \end{bmatrix}$$

Note that the size of matrix $[K_0]$ is 8×12 , the size of its null space matrix $[K_n]$ is 5×12 . The number of rows of $[K_0]$ plus the number of rows in $[K_n]$ is one bigger than the number of faces shared by this node ($8 + 5 > 12$). The reason for this is that the node i is an interior node (the node is fully surrounded by elements), the rows of matrix $[K_0]$ for such a node are linearly dependent.

For an exterior node (a node on the boundary of a domain is not fully surrounded by elements), the rows of matrix $[K_0]$ are linearly independent. The number of rows of $[K_0]$ plus the number of rows of $[K_n]$ equals the number of faces shared by this node. For instance, node 14 in Figure (3.2) is not fully surrounded by elements, the matrix $[K_0]$ for this node is:

Figure 3.2: 3DL-Block

$$[K_0] = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 1 \end{bmatrix}_{7 \times 12}$$

There are 7 elements and 12 faces sharing this node, so the size of the matrix $[K_0]$ is 7×12 . The null space matrix $[K_n]$ is:

The number of rows of $[K_0]$ plus the number of rows of $[K_n]$ is 12, which equals

$$[K_n] = \begin{bmatrix} -0.6455 & 0.3227 & 0.3227 & -0.3227 & -0.3227 & 0.2582 & 0.0645 & -0.0645 & 0.2582 & 0.0645 & 0.1291 & -0.0645 \\ 0.0000 & -0.3992 & 0.3992 & 0.2407 & -0.2407 & -0.4158 & 0.0166 & -0.1750 & 0.5139 & -0.1147 & -0.0982 & 0.2732 \\ 0.0000 & -0.3220 & 0.3220 & -0.2877 & 0.2877 & -0.0477 & -0.2742 & -0.3355 & -0.2295 & 0.5515 & 0.2772 & 0.0582 \\ 0.0000 & 0.1138 & -0.1138 & 0.1376 & -0.1376 & -0.4155 & 0.5293 & -0.2779 & -0.1402 & 0.0264 & 0.5557 & -0.2778 \\ 0.0000 & 0.1911 & -0.1911 & -0.3909 & 0.3909 & -0.0474 & 0.2385 & -0.4383 & 0.1163 & -0.3074 & -0.0690 & 0.5073 \end{bmatrix}$$

the number of faces shared by this node.

In the matrix $[K_0]$ at node 14, the columns with two nonzero entries (+1 and -1) correspond to the internal faces shared by two elements. The columns with only one nonzero entry (+1) correspond to an exterior faces. At node 14, there are nine interior faces and three exterior faces. For different boundary nodes, there are quite different appearance of $[K_0]$. For comparing them, we list matrices of $[K_0]$ of all nodes in Figure (3.2) at the end of this section.

For node i in (Figure 3.1), the size of the matrix $[K] = \begin{bmatrix} K_0 \\ K_n \end{bmatrix}$ is 13×12 . So we must use SVD to solve each of the systems of equations:

$$[K]\mathbf{X} = \mathbf{R}_x$$

$$[K]\mathbf{Y} = \mathbf{R}_y$$

$$[K]\mathbf{Z} = \mathbf{R}_z$$

By looping over all nodes in the domain, one can compute SAS nodal force on all faces,

$$\mathbf{X} = \{x^{f_1}, x^{f_2}, \dots, x^{f_N}\}^T$$

$$\mathbf{Y} = \{y^{f_1}, y^{f_2}, \dots, y^{f_N}\}^T$$

$$\mathbf{Z} = \{z^{f_1}, z^{f_2}, \dots, z^{f_N}\}^T$$

The SAS nodal forces above are sorted by the faces shared the particular node. To compute an SAS nodal stress, one has to rearrange them by faces of each element. For instance, for a specific face in an 8-node brick element, the nodal force vector of all nodes on this 4-node face could be:

$$\mathbf{F}_s^f = \{x^{n_1}, x^{n_2}, x^{n_3}, x^{n_4}, \quad y^{n_1}, y^{n_2}, y^{n_3}, y^{n_4}, \quad z^{n_1}, z^{n_2}, z^{n_3}, z^{n_4}\}^T \quad (3.9)$$

where, the superscripts n_1, n_2, n_3, n_4 refer to the first, second, third and fourth node of that face.

The matrices of $[K_0]$ of all nodes in Figure (3.2) are:

$$\begin{aligned} & \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \\ & \text{NODE 1, 3, 7, 9, 18, 19, 23, 24, 25, 26} \\ & \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ -1 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & -1 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & -1 & 1 & 1 \end{bmatrix} \\ & \text{NODE 2, 8, 20} \quad \text{NODE 4, 6, 21} \quad \text{NODE 10, 12, 16} \\ & \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} \\ & \text{NODE 15} \quad \text{NODE 17} \quad \text{NODE 22} \end{aligned}$$

$$\begin{array}{c}
\begin{bmatrix}
1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
-1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\
0 & -1 & 0 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & -1 & 0 & -1 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
-1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & -1 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & -1 & -1 & 0 & 1
\end{bmatrix} \\
\text{NODE 5} \qquad \qquad \qquad \text{NODE 11} \\
\\
\begin{bmatrix}
1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & -1 & 0 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & -1 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & -1 & 0 & -1 & 1
\end{bmatrix} \\
\text{NODE 13}
\end{array}$$

3.4 SAS Nodal Stress from SAS Nodal Forces

The SAS stress vector $\sigma_{\mathbf{s}} = \{\hat{\sigma}_{xx}, \hat{\sigma}_{yy}, \hat{\sigma}_{zz}, \hat{\sigma}_{xy}, \hat{\sigma}_{yz}, \hat{\sigma}_{xz}\}^T$ at the point (ξ, η, ζ) in the reference element can be calculated using following equation:

$$\sigma_s = \begin{bmatrix} \hat{\sigma}_{xx} \\ \hat{\sigma}_{yy} \\ \hat{\sigma}_{zz} \\ \hat{\sigma}_{xy} \\ \hat{\sigma}_{yz} \\ \hat{\sigma}_{xz} \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^8 N_i \hat{\sigma}_{xx}^i \\ \sum_{i=1}^8 N_i \hat{\sigma}_{yy}^i \\ \sum_{i=1}^8 N_i \hat{\sigma}_{zz}^i \\ \sum_{i=1}^8 N_i \hat{\sigma}_{xy}^i \\ \sum_{i=1}^8 N_i \hat{\sigma}_{yz}^i \\ \sum_{i=1}^8 N_i \hat{\sigma}_{xz}^i \end{bmatrix} \quad (3.10)$$

defining the vector of SAS nodal stress for an 8-node brick $\hat{\sigma}_s^8$

$$\hat{\sigma}_s^8 = \{\hat{\sigma}_{xx}^1, \hat{\sigma}_{xx}^2, \dots, \hat{\sigma}_{xx}^8, \hat{\sigma}_{yy}^1, \hat{\sigma}_{yy}^2, \dots, \hat{\sigma}_{yy}^8, \hat{\sigma}_{zz}^1, \hat{\sigma}_{zz}^2, \dots, \hat{\sigma}_{zz}^8, \hat{\sigma}_{xy}^1, \hat{\sigma}_{xy}^2, \dots, \hat{\sigma}_{xy}^8, \hat{\sigma}_{yz}^1, \hat{\sigma}_{yz}^2, \dots, \hat{\sigma}_{yz}^8, \hat{\sigma}_{xz}^1, \hat{\sigma}_{xz}^2, \dots, \hat{\sigma}_{xz}^8\}^T \quad (3.11)$$

and the matrix of volume basis function:

$$[\bar{R}] = \begin{bmatrix} R_v & 0 & 0 & 0 & 0 & 0 \\ 0 & R_v & 0 & 0 & 0 & 0 \\ 0 & 0 & R_v & 0 & 0 & 0 \\ 0 & 0 & 0 & R_v & 0 & 0 \\ 0 & 0 & 0 & 0 & R_v & 0 \\ 0 & 0 & 0 & 0 & 0 & R_v \end{bmatrix}_{6 \times 48}. \quad (3.12)$$

Equation 3.10 becomes:

$$\sigma_s = [\bar{R}] \hat{\sigma}_s^8 \quad (3.13)$$

where, $R_v = \{N_1, N_2, \dots, N_8\}$ is the vector of volume basis function.

The SAS traction vector $\tau_{\mathbf{s}} = \{\hat{\tau}_x, \hat{\tau}_y, \hat{\tau}_z\}^T$ on a face can be obtained from the SAS stress tensor by post multiplying this tensor by the normal vector $\mathbf{n} = \{n_x, n_y, n_z\}^T$ of the face:

$$\tau_{\mathbf{s}} = \begin{Bmatrix} \hat{\tau}_x \\ \hat{\tau}_y \\ \hat{\tau}_z \end{Bmatrix} = \begin{pmatrix} \hat{\sigma}_{xx} & \hat{\sigma}_{xy} & \hat{\sigma}_{xz} \\ \hat{\sigma}_{xy} & \hat{\sigma}_{yy} & \hat{\sigma}_{yz} \\ \hat{\sigma}_{xz} & \hat{\sigma}_{yz} & \hat{\sigma}_{zz} \end{pmatrix} \begin{Bmatrix} n_x \\ n_y \\ n_z \end{Bmatrix}. \quad (3.14)$$

Using the stress vector $\sigma_{\mathbf{s}} = \{\hat{\sigma}_{xx}, \hat{\sigma}_{yy}, \hat{\sigma}_{zz}, \hat{\sigma}_{xy}, \hat{\sigma}_{yz}, \hat{\sigma}_{xz}\}^T$, it can be rewritten as:

$$\tau_{\mathbf{s}} = [\overline{N}] \sigma_{\mathbf{s}} \quad (3.15)$$

where,

$$[\overline{N}] = \begin{pmatrix} n_x & 0 & 0 & n_y & 0 & n_z \\ 0 & n_y & 0 & n_x & n_z & 0 \\ 0 & 0 & n_z & 0 & n_y & n_x \end{pmatrix} \quad (3.16)$$

The SAS face nodal load vector

$$\mathbf{F}_{\mathbf{s}}^{f_i} = \{x^{n_1}, x^{n_2}, x^{n_3}, x^{n_4}, y^{n_1}, y^{n_2}, y^{n_3}, y^{n_4}, z^{n_1}, z^{n_2}, z^{n_3}, z^{n_4}\}^T \quad (3.17)$$

can be calculated using following integral:

$$\mathbf{F}_{\mathbf{s}}^{f_i} = \int_{f_i} [\hat{R}] \tau_{\mathbf{s}} ds \quad (3.18)$$

where, $[\hat{R}]$ is the face basis function matrix defined in equation (3.3), i is the face number of that element.

Substitute equation (3.13) and (3.15) into (3.18), we get:

$$\mathbf{F}_s^{f_i} = \int_{f_i} [\hat{R}][\bar{N}][\bar{R}] \hat{\sigma}_s^8 ds = \left(\int_{f_i} [\hat{R}][\bar{N}][\bar{R}] ds \right) \hat{\sigma}_s^8 \quad (3.19)$$

Let

$$H_{f_i} = \int_{f_i} [\hat{R}][\bar{N}][\bar{R}] ds \quad (3.20)$$

It, with the size of 12×48 , maps the vector of the SAS nodal stress to the vector of the SAS nodal force on face f_i .

The integrand can be written:

$$[\hat{R}][\bar{N}][\bar{R}] = \begin{bmatrix} n_x R_f^T R_v & 0 & 0 & n_y R_f^T R_v & 0 & n_z R_f^T R_v \\ 0 & n_y R_f^T R_v & 0 & n_x R_f^T R_v & n_z R_f^T R_v & 0 \\ 0 & 0 & n_z R_f^T R_v & 0 & n_y R_f^T R_v & n_x R_f^T R_v \end{bmatrix} \quad (3.21)$$

Note that the matrices $n_x R_f^T R_v$, $n_y R_f^T R_v$, $n_z R_f^T R_v$ can be integrated separately.

Now create the matrix $[\bar{H}]$ which maps the vector of SAS nodal stresses $\hat{\sigma}_s^8$ to the SAS nodal force on all 6 faces:

$$[\bar{H}] = \left\{ \begin{array}{c} H_{f_1} \\ H_{f_2} \\ H_{f_3} \\ H_{f_4} \\ H_{f_5} \\ H_{f_6} \end{array} \right\}_{72 \times 48}. \quad (3.22)$$

For a given SAS nodal force on the 6 faces,

$$\bar{\mathbf{X}} = \left\{ \begin{array}{c} \mathbf{F}_s^{f_1} \\ \mathbf{F}_s^{f_2} \\ \mathbf{F}_s^{f_3} \\ \mathbf{F}_s^{f_4} \\ \mathbf{F}_s^{f_5} \\ \mathbf{F}_s^{f_6} \end{array} \right\}_{72 \times 1} \quad (3.23)$$

we solve the system of equations

$$[\bar{H}]\hat{\sigma}_s^{\mathbf{8}} = \bar{\mathbf{X}} \quad (3.24)$$

for $\hat{\sigma}_s^{\mathbf{8}}$.

For an 8-node hexahedral element, the size of $[\bar{H}]$ is $72 \times 48 = (3 \times m \times k) \times (n \times 6)$,

m is the number of nodes per face, n is the number of nodes per element and k is the number of face per element. For other types of elements, the size of $[\bar{H}]$ is listed in Table (3.1).

From this table, we notice that, for all types of elements, the system of equations is over determined. Because the number of equations is bigger than the number of unknowns, we must use a SVD solver.

Element Types	Number of Nodes per Element n	Number of Nodes per Face m	Number of Faces per Element k	The Size of $[\bar{H}]$ Matrix for SAS $(3 \cdot m \cdot k) \times (6 \cdot n)$
4-node Tetrahedron	4	3	4	36×24
10-node Tetrahedron	10	6	4	72×60
6-node Prism	6	3.6	5	54×36
15-node Prism	15	7.2	5	108×90
8-node Hexahedron	8	4	6	72×48
20-node Hexahedron	20	8	6	144×120
32-node Hexahedron	32	12	6	216×192

Table 3.1: The size of the $[\bar{H}]$ matrix for different element types for SAS calculation.

3.5 Error Bound on Each Element

For each element, we have computed the SAS nodal stress and the KAS Gauss point stress. In this element, the KAS strain energy, the SAS strain energy and an error bound can be computed using the following element integrals:

$$SE_{KAS_i} = \int_{e_i} \sigma_{\mathbf{k}}^T C \sigma_{\mathbf{k}} \det J \quad (3.25)$$

$$SE_{SAS_i} = \int_{e_i} \sigma_{\mathbf{s}}^T C \sigma_{\mathbf{s}} \det J \quad (3.26)$$

$$\Delta SE_i = \int_{e_i} (\sigma_{\mathbf{s}} - \sigma_{\mathbf{k}})^T C (\sigma_{\mathbf{s}} - \sigma_{\mathbf{k}}) \det J \quad (3.27)$$

The norm of each kind of strain energy in the whole domain is the sum of its components over the total number of elements N .

$$\begin{aligned} \Delta SE_{tot} &= \sum_{i=1}^N \Delta SE_i \\ SE_{totSAS} &= \sum_{i=1}^N SE_{SAS_i} \\ SE_{totKAS} &= \sum_{i=1}^N SE_{KAS_i} \end{aligned}$$

The contribution of element i to the global error in strain energy is:

$$\begin{aligned} E_i &= \frac{\Delta SE_i}{0.5 \times (SE_{totKAS} + SE_{totSAS})} \\ E_{tot} &= \sum_{i=1}^N E_i \end{aligned}$$

We define constants a and b with say $a = 0.1$ and $b = 0.2$, so that,

if $E_i < a \times E_{tot}/N$, then this element could be coarsened.

if $E_i > b \times E_{tot}/N$, then this element could be refined.

If each element makes exactly the same contribution to the total error E_{tot} , then this mesh is optimal. No mesh with this number of elements could provide a more accurate solution.

3.6 SAS with Body Force

In case a body force or inertial force is present, then SAS satisfies the equilibrium equation, $\nabla \cdot \sigma_s + f = 0$, only on the boundaries of each element. To approximate the SAS requirement that σ_s be statically admissible pointwise inside the element, we have to add a bubble function, and then to compute the SAS. It requires an additional step to solve for the coefficients of the bubble function. If an inertial force is present, then it is simply added to the body force.

A node-less shape function $N^B(\xi, \eta, \zeta)$, is called a bubble function if it is zero on the boundary of the element. An example of a bubble function is defined by the expression:

$$N^B(\xi, \eta, \zeta) = (1 - \xi^2)(1 - \eta^2)(1 - \zeta^2) \quad (3.28)$$

The main feature of a bubble function is that it vanishes on the boundaries of the

element. Using a bubble function, the SAS stresses can be computed from equation:

$$\nabla \cdot (\bar{\sigma} + \delta\bar{\sigma}) + f = 0 \quad (3.29)$$

where, $\bar{\sigma}$ denotes the SAS stress inside the element without body force and/or inertial force. $\delta\bar{\sigma} = N^B(\xi, \eta, \zeta)\sigma^B$ denotes the increment in the SAS stress due to a body force and/or an inertial force. σ^B is a vector of unknown stress coefficients. $\sigma_s = \bar{\sigma} + \delta\bar{\sigma}$ is the total SAS stress inside the element.

We noticed that, the bubble function is zero at the nodes on the element boundary. The increment of the SAS stress vector $\delta\bar{\sigma}$ does not affect the SAS stress at nodes, as a result, the nodal force due to tractions on the faces of element is not changed. i.e.

$$\int_V \mathbf{B}^T \sigma^B dV = 0 \quad (3.30)$$

From equation (3.29), we have

$$\nabla \cdot \delta\bar{\sigma} = -\nabla \cdot \bar{\sigma} - f \quad (3.31)$$

To compute the stress coefficients σ^B , we need to transform equation (3.31) into the matrix form.

For any point within the element, we have

$$\nabla \cdot \delta \bar{\sigma} = \begin{pmatrix} \delta \bar{\sigma}_{xx} & \delta \bar{\sigma}_{xy} & \delta \bar{\sigma}_{xz} \\ \delta \bar{\sigma}_{xy} & \delta \bar{\sigma}_{yy} & \delta \bar{\sigma}_{yz} \\ \delta \bar{\sigma}_{xz} & \delta \bar{\sigma}_{yz} & \delta \bar{\sigma}_{zz} \end{pmatrix} \begin{pmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} \end{pmatrix} = N^B \begin{pmatrix} \sigma_{xx}^B & \sigma_{xy}^B & \sigma_{xz}^B \\ \sigma_{xy}^B & \sigma_{yy}^B & \sigma_{yz}^B \\ \sigma_{xz}^B & \sigma_{yz}^B & \sigma_{zz}^B \end{pmatrix} \begin{pmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} \end{pmatrix} \quad (3.32)$$

defining the stress coefficient vector as:

$$\sigma^B = \begin{Bmatrix} \sigma_{xx}^B \\ \sigma_{yy}^B \\ \sigma_{zz}^B \\ \sigma_{xy}^B \\ \sigma_{yz}^B \\ \sigma_{xz}^B \end{Bmatrix} \quad (3.33)$$

and the gradient matrix as:

$$[E^B] = \begin{pmatrix} \frac{\partial N^B}{\partial x} & 0 & 0 & \frac{\partial N^B}{\partial y} & 0 & \frac{\partial N^B}{\partial z} \\ 0 & \frac{\partial N^B}{\partial y} & 0 & \frac{\partial N^B}{\partial x} & \frac{\partial N^B}{\partial z} & 0 \\ 0 & 0 & \frac{\partial N^B}{\partial z} & 0 & \frac{\partial N^B}{\partial y} & \frac{\partial N^B}{\partial x} \end{pmatrix}. \quad (3.34)$$

$\nabla \cdot \delta \bar{\sigma}$ becomes:

$$\nabla \cdot \delta \bar{\sigma} = [E^B] \sigma^B. \quad (3.35)$$

Similarly, we can transform $\nabla \cdot \bar{\sigma}$ to matrix form.

$$\nabla \cdot \bar{\sigma} = \begin{pmatrix} \bar{\sigma}_{xx} & \bar{\sigma}_{xy} & \bar{\sigma}_{xz} \\ \bar{\sigma}_{xy} & \bar{\sigma}_{yy} & \bar{\sigma}_{yz} \\ \bar{\sigma}_{xz} & \bar{\sigma}_{yz} & \bar{\sigma}_{zz} \end{pmatrix} \begin{pmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} \end{pmatrix}. \quad (3.36)$$

There is another form for this equation:

$$\nabla \cdot \bar{\sigma} = \begin{pmatrix} \frac{\partial}{\partial x} & 0 & 0 & \frac{\partial}{\partial y} & 0 & \frac{\partial}{\partial z} \\ 0 & \frac{\partial}{\partial y} & 0 & \frac{\partial}{\partial x} & \frac{\partial}{\partial z} & 0 \\ 0 & 0 & \frac{\partial}{\partial z} & 0 & \frac{\partial}{\partial y} & \frac{\partial}{\partial x} \end{pmatrix} \begin{pmatrix} \bar{\sigma}_{xx} \\ \bar{\sigma}_{yy} \\ \bar{\sigma}_{zz} \\ \bar{\sigma}_{xy} \\ \bar{\sigma}_{yz} \\ \bar{\sigma}_{xz} \end{pmatrix}. \quad (3.37)$$

Recall equation (3.13). Here, it can be re-written as:

$$\bar{\sigma} = [\bar{R}] \hat{\sigma}_s^8. \quad (3.38)$$

Defining the gradient matrix:

$$[E^8] = \begin{pmatrix} \frac{\partial R_v}{\partial x} & 0 & 0 & \frac{\partial R_v}{\partial y} & 0 & \frac{\partial R_v}{\partial z} \\ 0 & \frac{\partial R_v}{\partial y} & 0 & \frac{\partial R_v}{\partial x} & \frac{\partial R_v}{\partial z} & 0 \\ 0 & 0 & \frac{\partial R_v}{\partial z} & 0 & \frac{\partial R_v}{\partial y} & \frac{\partial R_v}{\partial x} \end{pmatrix} \quad (3.39)$$

$\nabla \cdot \bar{\sigma}$ becomes:

$$\nabla \cdot \bar{\sigma} = [E^8] \hat{\sigma}_s^8 \quad (3.40)$$

Equation (3.31) can be re-written as matrix form:

$$[E^B] \sigma^B = -[E^8] \hat{\sigma}_s^8 - f \quad (3.41)$$

where, $f = \{f_x, f_y, f_z\}^T$ is the body force vector.

There are six unknowns in this system of equations. One point in an element can contribute three equations, so we at least need two points to construct and solve this system of equations. We may use the following steps to choose these two particular points for generating the system of equations:

1. If a body force is present for stress analysis, then evaluate the body force

f_x, f_y, f_z at the element centroid.

2. Evaluate the Jacobian matrix $[J] =$ at the element centroid.

3. Evaluate $\begin{Bmatrix} d\xi \\ d\eta \\ d\zeta \end{Bmatrix} = [J]^{-1} \begin{Bmatrix} f_x \\ f_y \\ f_z \end{Bmatrix}$, find $\max(|d\xi|, |d\eta|, |d\zeta|)$ (suppose it is $|d\xi|$),

then scale $(d\xi, d\eta, d\zeta) = (d\xi, d\eta, d\zeta)/|d\xi|$ and

compute the length of $L = \sqrt{(d\xi)^2 + (d\eta)^2 + (d\zeta)^2}$.

4. Compute local coordinates of the 2 points.

$$(\xi_1, \eta_1, \zeta_1) = \frac{1}{\sqrt{3}L}(d\xi, d\eta, d\zeta)$$

$$(\xi_2, \eta_2, \zeta_2) = -(\xi_1, \eta_1, \zeta_1)$$

5. For these two points, compute $f_1 = \{f_x, f_y, f_z\}_1^T$ and $f_2 = \{f_x, f_y, f_z\}_2^T$, E^{81} and E^{82} , E^{B1} and E^{B2} . The superscript $B1$ and $B2$ refers to the bubble functions evaluated at these two points. The superscript 81 and 82 refers to the basis functions of the 8-node brick evaluated at points 1 and 2.

6. Generate the system equation:

$$\begin{bmatrix} E^{B1}(\xi_1, \eta_1, \zeta_1) \\ E^{B1}(\xi_2, \eta_2, \zeta_2) \end{bmatrix} \sigma^B = \begin{bmatrix} E^{81}(\xi_1, \eta_1, \zeta_1) \\ E^{81}(\xi_2, \eta_2, \zeta_2) \end{bmatrix} \hat{\sigma}_s^8 - \begin{bmatrix} f_1(\xi_1, \eta_1, \zeta_1) \\ f_2(\xi_2, \eta_2, \zeta_2) \end{bmatrix} \quad (3.42)$$

Having solved for σ^B , the coefficients of the bubble functions, the SAS stress at the point (ξ, η, ζ) in the element is given by:

$$\sigma_s(\xi, \eta, \zeta) = [\bar{R}] \hat{\sigma}_s^8 + N^B \sigma^B \quad (3.43)$$

where, $[\bar{R}]$ is the matrix of volume basis function defined in equation (3.12) and N^B is bubble function defined in equation (3.28).

The element level error measures can now be evaluated as before using this SAS field.

Chapter 4

Discrete SAF Fields

The theory in this chapter follows J. Goldak [15][16].

4.1 Algorithm

The standard FEM formulation computes a KAF field. Our objective here is to compute a SAF field from the KAF solution by solving local problems. The algorithm consists of the following main steps:

1. Solve for the KAF temperature, the KAF temperature gradient and the KAF thermal flux.
2. Solve for the KAF nodal thermal loads on each element and the nodal thermal loads for each face. The nodal thermal loads on a face are not in equilibrium.

3. Solve for the SAF nodal thermal loads on each face.
4. Solve for the SAF thermal flux in each element. Note that this is nodal data in the element. It is not global data. If a node is shared by eight elements, then this node can have a different value of SAF thermal flux for this node for each element.
5. Solve for the error bound in each element and the global error bound.

It is assumed that the KAF results exist and we will start from step 2.

In the following sections, this method is applied to an 8-node hexahedral element and the corresponding mathematical details of matrix equations used to develop a program are derived. All of these equations are suitable for a 6-node prism element and a 4-node tetrahedral element if the element type is set as variable.

4.2 KAF Nodal Thermal Loads

4.2.1 KAF Nodal Load on Element

The vector of KAF nodal loads on each element can be evaluated with the equation $\mathbf{Q}_k^e = \int_V \mathbf{B}^T \mathbf{q}_k dV$ where, $\mathbf{q}_k = \{q_x, q_y, q_z\}^T$ is the KAF thermal flux vector at the point (ξ, η, ζ) in the reference element. $\mathbf{B}^T$ is the transpose of the gradient matrix which maps the thermal flux vector at the point (ξ, η, ζ) in the reference element to

the vector of nodal loads on that element. For an 8 node hexahedral element, $\mathbf{B}^T$ is the transpose of an 3×8 matrix.

$$\mathbf{B} = \begin{bmatrix} \frac{\partial N_1}{\partial x} & \frac{\partial N_2}{\partial x} & \dots & \frac{\partial N_8}{\partial x} \\ \frac{\partial N_1}{\partial y} & \frac{\partial N_2}{\partial y} & \dots & \frac{\partial N_8}{\partial y} \\ \frac{\partial N_1}{\partial z} & \frac{\partial N_2}{\partial z} & \dots & \frac{\partial N_8}{\partial z} \end{bmatrix}_{3 \times 8}$$

$\mathbf{Q}_k^e$ is 8×1 vector:

$$\mathbf{Q}_k^e = \{P^{e_1}, P^{e_2}, \dots, P^{e_8}\}^T \quad (4.1)$$

here, P refers to the components of KAF nodal load. The superscripts $e_1, e_2, \dots, e_8$ refer to the first, second, ..., eighth node of the element.

4.2.2 KAF Nodal Thermal Load on Face

On each face, the vector of thermal nodal loads due to the KAF scalar thermal flux can be evaluated from the equation $\mathbf{Q}_k^f = \int_S R_f^T \tau dS$. Where, τ is the KAF scalar thermal flux at the point (ξ, η) on the given face, which can be calculated using the thermal flux vector at that point times the normal vector, $\mathbf{n} = \{n_x, n_y, n_z\}^T$, of the face at the same point, $R_f = \{N_1, N_2, N_3, N_4\}$ is a vector of face basis function.

$$\tau = \{q_x, q_y, q_z\} \begin{Bmatrix} n_x \\ n_y \\ n_z \end{Bmatrix} \quad (4.2)$$

Each interior face has contributions from two elements. The average of the two vectors of nodal loads from those two elements is

$$\mathbf{Q}_k^f = \frac{1}{2} [\int_{S_1} R_f^T \tau dS_1 - \int_{S_2} R_f^T \tau dS_2] \quad (4.3)$$

For an 8-node hexahedral element, $\mathbf{Q}_k^f$ is 4×1 vector.

Note that the minus sign reflects the different sign of the outward normals of the two sides of the face.

$$\mathbf{Q}_k^f = \{P^{f_1}, P^{f_2}, P^{f_3}, P^{f_4}\}^T \quad (4.4)$$

Here, P refers to the components of the KAF nodal load. The superscript f_1, f_2, f_3, f_4 refers to the first, second, third and fourth node of the face.

Each exterior face has contributions from one face. If an exterior face has the Neumann BC, use the Neumann scalar thermal flux, not the KAF scalar thermal flux. If the face has the Dirichlet BC, use the KAF scalar thermal flux.

To calculate the average of nodal loads on a face, we have to set a face orientation vector globally so that the first element ID has the orientation $+n$ and the second element ID has the orientation $-n$. This orientation is global and will not be changed through out the SAF nodal load calculation.

4.3 SAF Nodal Thermal Loads on Face

For a given node, let N be the number of faces shared by this node, M be the number of elements shared by this node,

$\mathbf{P} = \{p^{f_1}, p^{f_2}, \dots, p^{f_N}\}^T$ be the vector of SAF nodal load on faces $\{f_1, f_2, \dots, f_N\}$.

We wish to set up a system of equations to solve for the unknown vector $\mathbf{P}$.

To do that, form a $M \times N$ matrix $[K_0]$ initialized to all zeros, in which the number of rows M equals to the number of elements shared by this node and the number of columns N equals to the number of faces shared by this node. The entry K_{0ij} of $[K_0]$ associated with row i and column j . Row i refers to element i and column j refers to face j .

For each element sharing a specific face, set the entries in the row for the face in the first element to +1 or -1 depending on whether this element is the first or the second element for this face. These face orientations have been defined in section (4.2.2).

Also, form a vector $\mathbf{R}_e$ with size $M \times 1$. For each element shared by this node, assign the KAF element nodal load $\{p^{e_1}, p^{e_2}, \dots, p^{e_M}\}^T$ to $\mathbf{R}_e$. $e_1, e_2, \dots, e_M$ refers to the first, second, ..., and Mth element shared by this node.

Use SVD to compute the null space of the matrix $[K_0]$. This null space forms a new matrix $[K_n]$. Multiply $[K_n]$ by the vector of KAF nodal loads on faces sharing this node, $\{p^{f_1}, p^{f_2}, \dots, p^{f_N}\}^T$ to get a new vector $\mathbf{R}_f = [K_n]\{p^{f_1}, p^{f_2}, \dots, p^{f_N}\}^T$. Combine

the matrix $[K_0]$ and $[K_n]$ into $[K] = \begin{bmatrix} K_0 \\ K_n \end{bmatrix}$, and the vector $\mathbf{R}_e$ and $\mathbf{R}_f$ into $\mathbf{R} = \begin{Bmatrix} \mathbf{R}_e \\ \mathbf{R}_f \end{Bmatrix}$. Using the method suggested by A. Mashaie [13] it can be shown that $\mathbf{P}$, the vector of SAF nodal loads on faces shared by this node, can be computed by solving the following systems of equation:

$$[K]\mathbf{P} = \mathbf{R} \quad (4.5)$$

Note that the properties of the matrix $[K]$ here are exactly the as same as those of the matrix $[K]$ in section (3.3), where the SAS nodal forces on faces were computed.

The SAF nodal loads above are sorted by the faces sharing the particular node. To compute the SAF nodal thermal flux, one has to rearrange nodal loads by face. For instance, for a specific face in an 8-node hexahedral element, the nodal load vector of all nodes on this face should be:

$$\mathbf{Q}_s^f = \{p^{n_1}, p^{n_2}, p^{n_3}, p^{n_4}\}^T \quad (4.6)$$

where, the superscript n_1, n_2, n_3, n_4 refers to the first, second, third and fourth node of that face.

4.4 SAF Nodal Thermal Flux

The SAF thermal flux vector $\mathbf{q}_s = \{\hat{q}_x, \hat{q}_y, \hat{q}_z\}^T$ at the point (ξ, η, ζ) in the reference element can be calculated using following equation:

$$\mathbf{q}_s = \begin{bmatrix} \hat{q}_x \\ \hat{q}_y \\ \hat{q}_z \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^8 N_i \hat{q}_x^i \\ \sum_{i=1}^8 N_i \hat{q}_y^i \\ \sum_{i=1}^8 N_i \hat{q}_z^i \end{bmatrix} \quad (4.7)$$

Define the vector of SAF nodal thermal flux for an 8-node brick $\hat{\mathbf{q}}^8$

$$\hat{\mathbf{q}}^8 = \{\hat{q}_x^1, \hat{q}_x^2, \dots, \hat{q}_x^8, \hat{q}_y^1, \hat{q}_y^2, \dots, \hat{q}_y^8, \hat{q}_z^1, \hat{q}_z^2, \dots, \hat{q}_z^8\}^T \quad (4.8)$$

and the matrix of volume basis function:

$$[\overline{R}] = \begin{bmatrix} R_v & 0 & 0 \\ 0 & R_v & 0 \\ 0 & 0 & R_v \end{bmatrix}_{3 \times 24} \quad (4.9)$$

equation (4.7) becomes:

$$\mathbf{q}_s = [\overline{R}] \hat{\mathbf{q}}^8 \quad (4.10)$$

where, $R_v = \{N_1, N_2, \dots, N_8\}$ is the vector of volume basis function.

The SAF scalar thermal flux $\hat{\tau}$ on a face can be obtained using the dot product of the SAF thermal flux vector and the normal vector $\mathbf{n} = \{n_x, n_y, n_z\}^T$ of the face:

$$\hat{\tau} = \{n_x, n_y, n_z\} \cdot \begin{Bmatrix} \hat{q}_x \\ \hat{q}_y \\ \hat{q}_z \end{Bmatrix} \quad (4.11)$$

or

$$\hat{\tau} = \mathbf{n}^T \mathbf{q}_s \quad (4.12)$$

The SAF face nodal load vector

$$\mathbf{Q}_s^{f_i} = \{p^{n_1}, p^{n_2}, p^{n_3}, p^{n_4}\}^T \quad (4.13)$$

can be calculated using the following integral:

$$\mathbf{Q}_s^{f_i} = \int_{f_i} R_f^T \hat{\tau} ds \quad (4.14)$$

where, $R_f = [N_1, N_2, N_3, N_4]$ is the face basis function matrix, i is the face number of that element.

Substituting equation (4.10) and (4.12) into (4.14), we get:

$$\mathbf{Q}_s^{f_i} = \int_{f_i} R_f^T \mathbf{n}^T [\bar{R}] \hat{\mathbf{q}}^8 ds = \left(\int_{f_i} R_f^T \mathbf{n}^T [\bar{R}] ds \right) \hat{\mathbf{q}}^8 \quad (4.15)$$

The matrix

$$H_{f_i} = \int_{f_i} R_f^T \mathbf{n}^T [\bar{R}] ds \quad (4.16)$$

with the size of 4×24 , maps the vector of SAF nodal thermal fluxes to the vector of SAF thermal nodal loads on face f_i .

The integrand can be written:

$$R_f^T \mathbf{n}^T [\bar{R}] = [n_x R_f^T R_v, \quad n_y R_f^T R_v, \quad n_z R_f^T R_v]_{4 \times 24} \quad (4.17)$$

Note that the matrices $n_x R_f^T R_v$, $n_y R_f^T R_v$, $n_z R_f^T R_v$ can be integrated separately.

Now, create the matrix $[\bar{H}]$, that maps the vector of SAF nodal thermal flux $\hat{\mathbf{q}}^8$ to the SAF nodal load on all 6 faces:

$$[\bar{H}] = \left\{ \begin{array}{c} H_{f_1} \\ H_{f_2} \\ H_{f_3} \\ H_{f_4} \\ H_{f_5} \\ H_{f_6} \end{array} \right\}_{24 \times 24} \quad (4.18)$$

For a given SAF nodal load on the 6 faces,

$$\bar{\mathbf{X}} = \begin{Bmatrix} \mathbf{Q}_s^{f_1} \\ \mathbf{Q}_s^{f_2} \\ \mathbf{Q}_s^{f_3} \\ \mathbf{Q}_s^{f_4} \\ \mathbf{Q}_s^{f_5} \\ \mathbf{Q}_s^{f_6} \end{Bmatrix}_{24 \times 1} \quad (4.19)$$

we solve the system of equations

$$[\bar{H}]\hat{\mathbf{q}}^8 = \bar{\mathbf{X}} \quad (4.20)$$

for $\hat{\mathbf{q}}^8$.

For an 8-node hexahedral element, the size of $[\bar{H}]$ is $24 \times 24 = (m \times k) \times (n \times 3)$, m is the number of nodes per face, n is the number of nodes per element and k is the number of faces per element. For other types of elements, the size of $[\bar{H}]$ is listed in Table (4.1).

Note that, from this table, the system for 4-node tetrahedral element has 12 equations with 12 unknowns, the system for 6-node prism element has 18 equations with 18 unknowns, the system for 8-node hexahedral element has 24 equations with 24 unknowns. They can be solved using a regular solver or a SVD solver. But for the 20-node hexahedral element and 32-node hexahedral element, the number of unknowns

Element Types	Number of Nodes per Element n	Number of Nodes per Face m	Number of Faces per Element k	The Size of $[\bar{H}]$ Matrix for SAF $(m \cdot k) \times (3 \cdot n)$
4-node Tetrahedron	4	3	4	12×12
10-node Tetrahedron	10	6	4	72×60
6-node Prism	6	3.6	5	18×18
15-node Prism	15	7.2	5	36×45
8-node Hexahedron	8	4	6	24×24
20-node Hexahedron	20	8	6	48×60
32-node Hexahedron	32	12	6	72×96

Table 4.1: The size of the $[\bar{H}]$ matrix for different element types for SAF calculation.

is greater than the number of equations. Therefore the algorithm as presented cannot work for these two kinds of elements. In this case, the KAF calculation for 20-node hexahedral element is to be combined with the SAF calculation for 8-node hexahedral element as outlined in Section (4.6.4).

4.5 Error Bound on Each Element

For each element, we have computed the SAF nodal thermal flux and the KAF Gauss point thermal flux. In this element, the KAF energy, the SAF energy and an error bound can be computed using the following element integrals:

$$SE_{KAF_i} = \int_{e_i} \mathbf{q}_k^T [k]^{-1} \mathbf{q}_k \det J \quad (4.21)$$

$$SE_{SAF_i} = \int_{e_i} \mathbf{q}_s^T [k]^{-1} \mathbf{q}_s \det J \quad (4.22)$$

$$\Delta SE_i = \int_{e_i} (\mathbf{q}_s - \mathbf{q}_k)^T [k]^{-1} (\mathbf{q}_s - \mathbf{q}_k) \det J \quad (4.23)$$

The norm of each kind of energy in the whole domain is the sum of its components over the total number of elements N .

$$\begin{aligned} \Delta SE_{tot} &= \sum_{i=1}^N \Delta SE_i \\ SE_{totSAF} &= \sum_{i=1}^N SE_{SAF_i} \\ SE_{totKAF} &= \sum_{i=1}^N SE_{KAF_i} \end{aligned}$$

The contribution of element i to the global error in energy is:

$$\begin{aligned} E_i &= \frac{\Delta SE_i}{0.5 \times (SE_{totKAF} + SE_{totSAF})} \\ E_{tot} &= \sum_{i=1}^N E_i \end{aligned}$$

We define constants a and b with say $a = 0.1$ and $b = 0.2$, so that

if $E_i < a \times E_{tot}/N$, then this element could be coarsened.

if $E_i > b \times E_{tot}/N$, then this element could be refined.

If each element makes exactly the same contribution to the total error E_{tot} , then this mesh is optimal and no mesh with this number of elements could provide a more accurate solution.

4.6 SAF with the Transient Thermal Flux

In the case of a heat source or a transient specific enthalpy being present, the SAF satisfies the energy equation, $\dot{h} + \nabla \cdot \mathbf{q} + p = 0$, only on the boundaries of each element. To approximate the SAF requirement that $\mathbf{q}_s$ be statically admissible point-wise inside the element, we have to add a bubble function, and then to compute the SAF. It requires an additional step to solve for the coefficients of the bubble function.

Using the bubble function,

$$N^B(\xi, \eta, \zeta) = (1 - \xi^2)(1 - \eta^2)(1 - \zeta^2)$$

the SAF flux can be computed from the equation:

$$\dot{\mathbf{h}} + \nabla \cdot (\bar{\mathbf{q}} + \delta\bar{\mathbf{q}}) + \mathbf{p} = 0 \quad (4.24)$$

where, $\bar{\mathbf{q}}$ denotes SAF flux inside the element without a heat source or a transient specific enthalpy. $\delta\bar{\mathbf{q}} = N^B(\xi, \eta, \zeta)\mathbf{q}^B$ denotes the increment SAF flux due to a heat

source or a transient specific enthalpy. $\mathbf{q}^B = \begin{Bmatrix} q_x^B \\ q_y^B \\ q_z^B \end{Bmatrix}$ is the vector of unknown flux coefficients. $\mathbf{q}_s = \bar{\mathbf{q}} + \delta\bar{\mathbf{q}}$ is the total SAF flux inside the element.

Because the bubble function at the nodes is equal to zero, the increment in the SAF flux vector $\delta\bar{\mathbf{q}}$ does not affect the SAF flux at nodes on the element boundary. As a result, the nodal load on the faces of a element does not change, i.e.,

$$\int_V \mathbf{B}^T \mathbf{q}^B dV = 0 \quad (4.25)$$

From equation (4.24), we have

$$\nabla \cdot \delta\bar{\mathbf{q}} = -\dot{\mathbf{h}} - \nabla \cdot \bar{\mathbf{q}} - \mathbf{p} \quad (4.26)$$

To compute the flux coefficients $\mathbf{q}^B$, we need to transform equation (4.26) into matrix form and build a system of equations.

4.6.1 Transform $\nabla \cdot \delta\bar{\mathbf{q}}$ into matrix form

Using the relation

$$\delta\bar{\mathbf{q}} = N^B(\xi, \eta, \zeta) \mathbf{q}^B \quad (4.27)$$

we have

$$\nabla \cdot \delta \bar{\mathbf{q}} = \begin{bmatrix} \frac{\partial}{\partial x} & \frac{\partial}{\partial y} & \frac{\partial}{\partial z} \end{bmatrix} \begin{Bmatrix} \delta \bar{q}_x \\ \delta \bar{q}_y \\ \delta \bar{q}_z \end{Bmatrix} = \begin{bmatrix} \frac{\partial}{\partial x} & \frac{\partial}{\partial y} & \frac{\partial}{\partial z} \end{bmatrix} N^B \begin{Bmatrix} q_x^B \\ q_y^B \\ q_z^B \end{Bmatrix} \quad (4.28)$$

let $[E^B] = \begin{bmatrix} \frac{\partial N^B}{\partial x} & \frac{\partial N^B}{\partial y} & \frac{\partial N^B}{\partial z} \end{bmatrix}$, equation (4.28) becomes:

$$\nabla \cdot \delta \bar{\mathbf{q}} = [E^B] \mathbf{q}^B \quad (4.29)$$

where the components of the gradient of the bubble function can be calculated using the relation:

$$\begin{Bmatrix} \frac{\partial N^B}{\partial x} \\ \frac{\partial N^B}{\partial y} \\ \frac{\partial N^B}{\partial z} \end{Bmatrix} = [J]^{-T} \begin{Bmatrix} \frac{\partial N^B}{\partial \xi} \\ \frac{\partial N^B}{\partial \eta} \\ \frac{\partial N^B}{\partial \zeta} \end{Bmatrix} = [J]^{-T} \begin{pmatrix} -2\xi(1-\eta^2)(1-\zeta^2) \\ -2\eta(1-\xi^2)(1-\zeta^2) \\ -2\zeta(1-\xi^2)(1-\eta^2) \end{pmatrix} \quad (4.30)$$

here, $[J]$ is the Jacobian Matrix.

4.6.2 Transform $\nabla \cdot \bar{\mathbf{q}}$ into matrix form

For any point within the element, we have

$$\nabla \cdot \bar{\mathbf{q}} = \begin{bmatrix} \frac{\partial}{\partial x} & \frac{\partial}{\partial y} & \frac{\partial}{\partial z} \end{bmatrix} \begin{Bmatrix} \bar{q}_x \\ \bar{q}_y \\ \bar{q}_z \end{Bmatrix} \quad (4.31)$$

$$\text{let } \bar{\mathbf{q}} = \begin{Bmatrix} \bar{q}_x \\ \bar{q}_y \\ \bar{q}_z \end{Bmatrix} = [\bar{R}] \hat{\mathbf{q}}^8, \text{ where, } [\bar{R}] = \begin{bmatrix} R_v & 0 & 0 \\ 0 & R_v & 0 \\ 0 & 0 & R_v \end{bmatrix}_{3 \times 24} \text{ is the matrix of}$$

volume basis functions and $R_v = [N_1, N_2, \dots, N_8]$ is the vector of volume basis functions,

$\hat{\mathbf{q}}^8 = \{\hat{q}_x^1, \hat{q}_x^2, \dots, \hat{q}_x^8, \hat{q}_y^1, \hat{q}_y^2, \dots, \hat{q}_y^8, \hat{q}_z^1, \hat{q}_z^2, \dots, \hat{q}_z^8\}^T$ is the SAF nodal thermal flux vector,

also let $[E^v] = \begin{bmatrix} \frac{\partial R_v}{\partial x} & \frac{\partial R_v}{\partial y} & \frac{\partial R_v}{\partial z} \end{bmatrix}$ be the vector of the volume physical gradient.

Then substitute these equations into (4.31) and re-arrange to get:

$$\nabla \cdot \bar{\mathbf{q}} = [E^v] \hat{\mathbf{q}}^8 \quad (4.32)$$

where, the components of physical gradient $\frac{\partial R_v}{\partial x}, \frac{\partial R_v}{\partial y}, \frac{\partial R_v}{\partial z}$ can be calculated using the

following relation:

$$\begin{Bmatrix} \frac{\partial R_v}{\partial x} \\ \frac{\partial R_v}{\partial y} \\ \frac{\partial R_v}{\partial z} \end{Bmatrix} = [J]^{-T} \begin{Bmatrix} \frac{\partial R_v}{\partial \xi} \\ \frac{\partial R_v}{\partial \eta} \\ \frac{\partial R_v}{\partial \zeta} \end{Bmatrix} \quad (4.33)$$

Substitute equation (4.29) and (4.32) into (4.26), to get:

$$[E^B]\mathbf{q}^B = -[E^v]\hat{\mathbf{q}}^8 - \dot{\mathbf{h}} - \mathbf{p} \quad (4.34)$$

In this equation, there are three unknowns, q_x^B, q_y^B, q_z^B . We have to find at least three points to build a system of equations to solve for the flux coefficient q_x^B, q_y^B, q_z^B .

4.6.3 Creating and Solving the System of Equations

For 8-node hexahedral element, we choose 8 Gauss points and get the following equation:

$$\begin{Bmatrix} E^{B1} \\ E^{B2} \\ E^{B3} \\ E^{B4} \\ E^{B5} \\ E^{B6} \\ E^{B7} \\ E^{B8} \end{Bmatrix}_{8 \times 3} \mathbf{q}^B = - \begin{Bmatrix} E^{v1} \\ E^{v2} \\ E^{v3} \\ E^{v4} \\ E^{v5} \\ E^{v6} \\ E^{v7} \\ E^{v8} \end{Bmatrix}_{8 \times 24} \hat{\mathbf{q}}^8 - \begin{Bmatrix} \dot{h}_1 \\ \dot{h}_2 \\ \dot{h}_3 \\ \dot{h}_4 \\ \dot{h}_5 \\ \dot{h}_6 \\ \dot{h}_7 \\ \dot{h}_8 \end{Bmatrix}_{8 \times 1} - \begin{Bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \\ p_7 \\ p_8 \end{Bmatrix}_{8 \times 1} \quad (4.35)$$

There are 8 equations and three unknowns in this system of equations. Again, we need to use SVD to solve it.

Finally, the total SAF flux inside the element can be computed by:

$$\mathbf{q}_s = \bar{\mathbf{q}} + \delta \bar{\mathbf{q}}$$

or

$$\mathbf{q}_s = [\bar{R}]\hat{\mathbf{q}}^8 + N^B \mathbf{q}^B.$$

Using matrix form, we have:

$$\begin{Bmatrix} \hat{q}_x \\ \hat{q}_y \\ \hat{q}_z \end{Bmatrix} = \begin{bmatrix} R_v & 0 & 0 \\ 0 & R_v & 0 \\ 0 & 0 & R_v \end{bmatrix} \begin{Bmatrix} \hat{q}_x^1 \\ \hat{q}_x^2 \\ \dots \\ \hat{q}_x^8 \\ \hat{q}_y^1 \\ \hat{q}_y^2 \\ \dots \\ \hat{q}_y^8 \\ \hat{q}_z^1 \\ \hat{q}_z^2 \\ \dots \\ \hat{q}_z^8 \end{Bmatrix} + N^B \begin{Bmatrix} q_x^B \\ q_y^B \\ q_z^B \end{Bmatrix}. \quad (4.36)$$

4.6.4 Different Element Types for KAF and SAF Calculation

As described in section (4.4) and Table (4.1), for the 4-node tetrahedral element, the 6-node prism element and the 8-node hexahedral element, the number of equations in the system of equations (4.20) equals the number of unknowns. These can be

solved using a regular solver or a SVD solver. For a 20-node hexahedral element or for elements with a higher order of basis function, the number of unknowns is bigger than the number of equations and the algorithm cannot work for such elements. Nevertheless, we could use a higher order element to get a more accurate KAF thermal flux field, and then use a lower order element to compute SAF thermal flux field and error. During these calculations, a technique to project data from the higher order element to the lower order element is necessary.

Below, we will give the details of such a technique [17].

The value of a parameter at any point (r, s, t) inside a 3D element can be expressed by:

$$T = a_0 + a_1r + a_2s + a_3t + a_4rs + a_5st + a_6tr + a_7rst$$

Using the matrix form, we have

$$T = [1 \ r \ s \ t \ rs \ st \ tr \ rst] \mathbf{a}$$

where, $\mathbf{a} = \{a_0, a_1, \dots, a_7\}^T$ is a vector of coefficients.

For an 8-node brick, suppose the values at 8 volume Gauss points are known, i.e., $\mathbf{T}_m = \{T_1, T_2, \dots, T_8\}^T$ are known, then the coefficient vector $\mathbf{a}$ can be computed by solving the following equation:

$$\mathbf{T}_m = \begin{bmatrix} 1 & r_1 & s_1 & t_1 & r_1 s_1 & s_1 t_1 & t_1 r_1 & r_1 s_1 t_1 \\ 1 & r_2 & s_2 & t_2 & r_2 s_2 & s_2 t_2 & t_2 r_2 & r_2 s_2 t_2 \\ 1 & r_3 & s_3 & t_3 & r_3 s_3 & s_3 t_3 & t_3 r_3 & r_3 s_3 t_3 \\ 1 & r_4 & s_4 & t_4 & r_4 s_4 & s_4 t_4 & t_4 r_4 & r_4 s_4 t_4 \\ 1 & r_5 & s_5 & t_5 & r_5 s_5 & s_5 t_5 & t_5 r_5 & r_5 s_5 t_5 \\ 1 & r_6 & s_6 & t_6 & r_6 s_6 & s_6 t_6 & t_6 r_6 & r_6 s_6 t_6 \\ 1 & r_7 & s_7 & t_7 & r_7 s_7 & s_7 t_7 & t_7 r_7 & r_7 s_7 t_7 \\ 1 & r_8 & s_8 & t_8 & r_8 s_8 & s_8 t_8 & t_8 r_8 & r_8 s_8 t_8 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \end{bmatrix} \quad (4.37)$$

We write the solution formally as $\mathbf{a} = \left[[B]^T [B] \right]^{-1} [B]^T \mathbf{T}_m$. Where,

$$[B] = \begin{bmatrix} 1 & r_1 & s_1 & t_1 & r_1 s_1 & s_1 t_1 & t_1 r_1 & r_1 s_1 t_1 \\ 1 & r_2 & s_2 & t_2 & r_2 s_2 & s_2 t_2 & t_2 r_2 & r_2 s_2 t_2 \\ 1 & r_3 & s_3 & t_3 & r_3 s_3 & s_3 t_3 & t_3 r_3 & r_3 s_3 t_3 \\ 1 & r_4 & s_4 & t_4 & r_4 s_4 & s_4 t_4 & t_4 r_4 & r_4 s_4 t_4 \\ 1 & r_5 & s_5 & t_5 & r_5 s_5 & s_5 t_5 & t_5 r_5 & r_5 s_5 t_5 \\ 1 & r_6 & s_6 & t_6 & r_6 s_6 & s_6 t_6 & t_6 r_6 & r_6 s_6 t_6 \\ 1 & r_7 & s_7 & t_7 & r_7 s_7 & s_7 t_7 & t_7 r_7 & r_7 s_7 t_7 \\ 1 & r_8 & s_8 & t_8 & r_8 s_8 & s_8 t_8 & t_8 r_8 & r_8 s_8 t_8 \end{bmatrix} \quad (4.38)$$

Each row of $[B]$ is associated with one Gauss point. If we have an n-node brick with more than 8 Gauss points, just add one row to $[B]$ for each additional Gauss

point. Thus $[B]$ is an $m \times 8$ matrix where $m \geq 8$ is the number of Gauss points.

Similarly, we have

$$[A] = \begin{bmatrix} 1 & r_1 & s_1 & t_1 & r_1 s_1 & s_1 t_1 & t_1 r_1 & r_1 s_1 t_1 \\ 1 & r_2 & s_2 & t_2 & r_2 s_2 & s_2 t_2 & t_2 r_2 & r_2 s_2 t_2 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 1 & r_n & s_n & t_n & r_n s_n & s_n t_n & t_n r_n & r_n s_n t_n \end{bmatrix} \quad (4.39)$$

Each row of $[A]$ is associated with one node. Thus $[A]$ is an $n \times 8$ matrix where $n \geq 8$ is the number of nodes. The nodal values $\mathbf{T}_n = [A] \mathbf{a} = [A] \left[[B]^T [B] \right]^{-1} [B]^T \mathbf{T}_m \equiv [P] \mathbf{T}_m$. Thus the matrix

$$[P] \equiv [A] \left[[B]^T [B] \right]^{-1} [B]^T \quad (4.40)$$

projects the Gauss point data to the points defined by $[A]$ that could be nodes or even a different set of Gauss points.

For instance, the following matrix will project the data of 27 Gauss points in 20-node hexahedral element to the 8 Gauss points of 8-node hexahedral element.

$$\begin{bmatrix} a & b & c & b & d & e & c & e & f & b & d & e & d & g & h & e & h & j & c & e & f & e & h & j & f & j & i \\ c & b & a & e & d & b & f & e & c & e & d & b & h & g & d & j & h & e & f & e & c & j & h & e & i & j & f \\ c & e & f & b & d & e & a & b & c & e & h & j & d & g & h & b & d & e & f & j & i & e & h & j & c & e & f \\ f & e & c & e & d & b & c & b & a & j & h & e & h & g & d & e & d & b & i & j & f & j & h & e & f & e & c \\ c & e & f & e & h & j & f & j & i & b & d & e & d & g & h & e & h & j & a & b & c & b & d & e & c & e & f \\ f & e & c & j & h & e & i & j & f & e & d & b & h & g & d & j & h & e & c & b & a & e & d & b & f & e & c \\ f & j & i & e & h & j & c & e & f & e & h & j & d & g & h & b & d & e & c & e & f & b & d & e & a & b & c \\ i & j & f & j & h & e & f & e & c & j & h & e & h & g & d & e & d & b & f & e & c & e & d & b & c & b & a \end{bmatrix}$$

where,

$$a = 0.7105688459041913 \quad b = 0.2654297024536922$$

$$c = -0.1797094409968069 \quad d = 0.0991500364120312$$

$$e = -0.0671296296296298 \quad f = 0.0454501817375474$$

$$g = 0.0370370370370370 \quad h = -0.0250759623379571$$

$$i = -0.0114947718301164 \quad j = 0.0169777049537155$$

Using this method, the only information required is the (r, s, t) coordinates of each Gauss point and of each node and the scalar data for each Gauss point $\mathbf{T}_m$. One need not know the basis functions of element.

Furthermore, this projection matrix must project a constant function and a linear function in the space of $(1, r, s, t, rs, st, tr, rst)$ exactly.

Chapter 5

Test Problems for SAS

5.1 Divergence Free Test

As described in section (2.1.2), the KAS stress need not satisfy the equation of equilibrium. If the KAS stress, in a particular case, does satisfy this equation, the KAS stress will exactly equal the SAS stress at every point in the interior domain. Moreover, both KAS and SAS equal the exact solution and the error is identically zero.

Assuming the body force $f = 0$, the equilibrium equation becomes:

$$\nabla \cdot \sigma = 0$$

or writing in component form:

$$\begin{bmatrix} \sigma_{xx} & \sigma_{xy} & \sigma_{xz} \\ \sigma_{xy} & \sigma_{yy} & \sigma_{yz} \\ \sigma_{xz} & \sigma_{yz} & \sigma_{zz} \end{bmatrix} \begin{pmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} \end{pmatrix} = \begin{bmatrix} \frac{\partial \sigma_{xx}}{\partial x} + \frac{\partial \sigma_{xy}}{\partial y} + \frac{\partial \sigma_{xz}}{\partial z} \\ \frac{\partial \sigma_{xy}}{\partial x} + \frac{\partial \sigma_{yy}}{\partial y} + \frac{\partial \sigma_{yz}}{\partial z} \\ \frac{\partial \sigma_{xz}}{\partial x} + \frac{\partial \sigma_{yz}}{\partial y} + \frac{\partial \sigma_{zz}}{\partial z} \end{bmatrix} = 0. \quad (5.1)$$

At Gauss points we set the value of the KAS stress as $(\sigma_{zz}, \sigma_{yz}, \sigma_{xz})$ equal to zero or constants (b_0, b_1, b_2) and $(\sigma_{xx}, \sigma_{yy}, \sigma_{xy})$ to the following functions of (x, y) . The coefficients $(a_0, \dots, a_6)$ can be chosen arbitrarily.

$$\begin{pmatrix} \sigma_{zz} \\ \sigma_{yz} \\ \sigma_{xz} \end{pmatrix} = \begin{pmatrix} b_0 \\ b_1 \\ b_2 \end{pmatrix} \text{ and } \begin{pmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{xy} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & y & 0 & x & 0 \\ 0 & 1 & 0 & 0 & x & 0 & y \\ 0 & 0 & 1 & 0 & 0 & -y & -x \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \end{pmatrix}$$

This stress is exactly divergence free, i.e., $\nabla \cdot \sigma = 0$. Therefore the SAS stress should exactly equal the KAS stress at any point (x, y, z) . Also the projected KAS stress at face Gauss points should equal the stress evaluated at the face Gauss point (x, y, z) .

The above equation sets the z components of the stress tensor to be constant and x and y component to be a first order polynomial of (x, y) . Symmetrically, to run this test, we also can use equations which set the x or y components of the stress tensor

to be constant and y and z or x and z components, respectively, to be a first order polynomial of (y, z) or (x, z) ; i.e.,

$$\begin{pmatrix} \sigma_{yy} \\ \sigma_{xy} \\ \sigma_{yz} \end{pmatrix} = \begin{pmatrix} b_0 \\ b_1 \\ b_2 \end{pmatrix} \text{ and } \begin{pmatrix} \sigma_{xx} \\ \sigma_{zz} \\ \sigma_{zx} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & z & 0 & x & 0 \\ 0 & 1 & 0 & 0 & x & 0 & z \\ 0 & 0 & 1 & 0 & 0 & -z & -x \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \end{pmatrix}$$

or

$$\begin{pmatrix} \sigma_{xx} \\ \sigma_{xy} \\ \sigma_{zx} \end{pmatrix} = \begin{pmatrix} b_0 \\ b_1 \\ b_2 \end{pmatrix} \text{ and } \begin{pmatrix} \sigma_{yy} \\ \sigma_{zz} \\ \sigma_{yz} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & z & 0 & y & 0 \\ 0 & 1 & 0 & 0 & y & 0 & z \\ 0 & 0 & 1 & 0 & 0 & -z & -y \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \end{pmatrix}$$

This test allows all data to be checked at each stage of the computation. It requires the code to set the values of KAS stress at volume Gauss point. This should work for any mesh.

5.2 3DL Block Test

Figure 5.1: 3DL-Block boundary conditions

The purpose of this test is to compare the error estimate associated with a coarse and a fine mesh.

A. Mashaie has done a test with a 2D L-shaped domain. Here, we will use a more general case, a 3D L-shaped domain (3DL-Block), and apply two different BCs, displacement BCs and traction BCs. The geometry and the physical boundary conditions associated with this problem are illustrated in figure (5.1).

The material properties of this test are: $E = 69 \text{ GPa}$, $\nu = 0.33$.

Case one: Displacement BC $u = 0,000018m$

There are no traction boundary conditions in this case. The displacement on the left side is $u = 0$ and on the right side is $u = 0.000018\text{ m}$. Set $v = w = 0$ at point A and $v = 0$ at point B to constrain the rigid body motion.

Initially, the 3DL-block is discretized into 7 elements with uniform 8-node hexahedral elements. For each mesh refinement, the number of elements is doubled in all x, y, z directions. Tables (5.1) and (5.2) present the calculation results with different numbers of elements.

In this table (5.2), one can see that, when the number of elements is increased, both the total KAS strain energy and the total SAS strain energy is decreased. That means that both the KAS and the SAS strain energy are over valued with a coarse mesh. When the number of elements tends to infinity, theoretically, the total KAS strain energy should equal to the total SAS strain energy. This statement agrees with our calculation results (Figure 5.2)

In addition, the finer the mesh, the smaller the total relative strain energy error and the error bound (Figure 5.3 and 5.4). Table (5.3) lists the maximum and minimum of element relative error with different numbers of elements.

When the number of elements is increased, the minimum of element relative error decreases much steeper than the maximum of element relative error. This is because the maximum value is located near the singularity. By means of mesh refinement,

Number of	Total KAS	Total SAS	Total Error Bound (J)
Elements	Strain Energy (J)	Strain Energy (J)	
N	$\sum_{i=1}^N \int_{e_i} \sigma_k^T C \sigma_k \det J$	$\sum_{i=1}^N \int_{e_i} \sigma_s^T C \sigma_s \det J$	$\sum_{i=1}^N \int_{e_i} (\sigma_s - \sigma_k)^T C (\sigma_s - \sigma_k) \det J$
7	4.5750E-02	4.5155E-02	7.2395E-04
56	4.5090E-02	4.4811E-02	3.8078E-04
448	4.4771E-02	4.4686E-02	1.6136E-04
3584	4.4638E-02	4.4613E-02	6.6777E-05
28672	4.4584E-02	4.4577E-02	2.7272E-05

Table 5.1: Number of elements vs. total strain energy for 3DL-Block

the element relative error decreases faster in elements far from singularity than in elements near the singularity.

Figure (5.6) to (5.9) shows how the singularity areas change with the changes in the number of elements.

Case two: right face with load $751000n/m^2$

The displacement on the left side face is the same as those in case one. On the right side face, instead of displacement, a traction boundary condition, $751000N/m^2$ in x direction, is applied. Table (5.4) shows the calculation results with different number of elements. Contrary to case one, both the total KAS strain energy and the total

Number of Elements	Total Relative Energy Error
N	$\frac{\sum_{i=1}^N \int_{e_i} (\sigma_s - \sigma_k)^T C (\sigma_s - \sigma_k) \det J}{0.5 \times (\sum_{i=1}^N \int_{e_i} \sigma_k^T C \sigma_k \det J + \sum_{i=1}^N \int_{e_i} \sigma_s^T C \sigma_s \det J)}$
7	1.5928E-02
56	8.4711E-03
448	3.6075E-03
3584	1.4964E-03
28672	6.1174E-04

Table 5.2: Number of elements vs. total relative energy error for 3DL-Block

SAS strain energy is increased with each increment in the number of elements. This means both of KAS and SAS have lower values in the coarse mesh than those in the fine mesh with traction BCs. When the number of elements tends to infinity, the total KAS strain energy tends to the same value as the total SAS strain energy (Figure 5.5).

Figure 5.2: Number of elements vs. strain energy for case one

Figure 5.3: Number of elements vs. total relative strain energy error

Figure 5.4: Number of elements vs. error bound in domain

Number of Element N	Maximum of Element Relative Error $\frac{\int_{e_i} (\sigma_s - \sigma_k)^T C (\sigma_s - \sigma_k) \det J}{\frac{1}{2} \left(\sum_{i=1}^N \int_{e_i} \sigma_k^T C \sigma_k \det J + \sum_{i=1}^N \int_{e_i} \sigma_s^T C \sigma_s \det J \right)}$	Minimum of Element Relative Error $\frac{\int_{e_i} (\sigma_s - \sigma_k)^T C (\sigma_s - \sigma_k) \det J}{\frac{1}{2} \left(\sum_{i=1}^N \int_{e_i} \sigma_k^T C \sigma_k \det J + \sum_{i=1}^N \int_{e_i} \sigma_s^T C \sigma_s \det J \right)}$
7	4.6953E-03	6.1624E-04
56	6.2990E-04	9.9716E-07
448	1.0550E-04	1.0509E-08
3584	1.9753E-05	1.8089E-10
28672	3.8283E-06	3.4997E-12

Table 5.3: Maximum and minimum element relative error

Number of Elements	Total KAS Strain Energy (J)	Total SAS Strain Energy (J)	Total Error Bound (J)	Total Relative Strain Energy Error
7	4.6102E-02	4.5373E-02	7.4986E-04	1.6395E-02
56	4.6747E-02	4.6432E-02	3.5181E-04	7.5512E-03
448	4.7060E-02	4.6971E-02	1.6055E-04	3.4149E-03
3584	4.7197E-02	4.7172E-02	7.2319E-05	1.5327E-03
28672	4.7255E-02	4.7249E-02	3.2262E-05	6.8276E-04

Table 5.4: 3DL-Block calculation results for case two

Figure 5.5: Number of elements vs. strain energy for case two

Figure 1: 3DL-Block strain energy error with 56 elements.

Figure 3: 3DL-Block strain energy error with 3,584 elements.

Figure 2: 3DL-Block strain energy error with 448 elements.

Figure 4: 3DL-Block strain energy error with 28,672 elements.

5.3 Crack Test

Figure 5.10: Boundary conditions for the crack problem

The purpose of this test is to test the statically admissible stress fields and the effect of the singularity.

The material properties of this test are: $E = 69 \text{ GPa}$, $\nu = 0.33$

The geometry and the physical boundary conditions associated with this problem are illustrated in Figure (5.10). We use one fourth of the whole domain to discretize and compute. The displacement on the face xoy is $w = 0$, on the face $yo z$ is $u = 0$ and half of the face xoz is $v = 0$ and another half of it is free face where the crack locates. The traction on top face in y direction is 69000 N/m^2 .

Figure 5.11: Number of elements vs. strain energy

Initially, the one fourth of the domain is discretized into 16 elements with uniform 8-node hexahedral element. In each mesh refinement, the number of element is doubled in all x, y, z directions.

When the number of elements is increased, the total KAS and SAS strain energy increase and the total relative strain energy error goes down. According to the trend of lines in (Fig. 5.11 and 5.12), when the number of element tends to infinity, the total KAS and SAS strain energy tend to a constant value and the difference between them tends to zero. The total relative strain energy error decreases to a very small number.

Figure 5.12: Number of elements vs. relative strain energy error

Table (5.5) illustrates the maximum and minimum of the element relative error. The maximum value is adjacent to the crack tip, and the minimum value is far from crack tip. To refine the mesh from 16 to 65536 elements, the maximum of relative error decreases from 2.7×10^{-2} to 1.1×10^{-4} , the ratio between them is 245. At the same time, the minimum of the relative error decreases from 4.2×10^{-3} to 4.1×10^{-11} , this ratio is 10^8 . This result means the uniform mesh refinement does not efficiently reduce the relative error in the region of the singularity .

Number of Element N	Maximum of Element Relative Error $\frac{\int_{e_i} (\sigma_s - \sigma_k)^T C (\sigma_s - \sigma_k) detJ}{\frac{1}{2} (\sum_{i=1}^N \int_{e_i} \sigma_k^T C \sigma_k detJ + \sum_{i=1}^N \int_{e_i} \sigma_s^T C \sigma_s detJ)}$	Minimum of Element Relative Error $\frac{\int_{e_i} (\sigma_s - \sigma_k)^T C (\sigma_s - \sigma_k) detJ}{\frac{1}{2} (\sum_{i=1}^N \int_{e_i} \sigma_k^T C \sigma_k detJ + \sum_{i=1}^N \int_{e_i} \sigma_s^T C \sigma_s detJ)}$
16	2.6986E-02	4.2237E-03
128	6.8844E-03	3.2419E-05
1024	1.7673E-03	1.0490E-07
8192	4.4169E-04	2.6038E-09
65536	1.1021E-04	4.1252E-11

Table 5.5: Maximum and minimum element relative strain energy error

Number of Elements N	Total KAS Strain Energy (J) $\sum_{i=1}^N \int_{e_i} \sigma_k^T C \sigma_k detJ$	Total SAS Strain Energy (J) $\sum_{i=1}^N \int_{e_i} \sigma_s^T C \sigma_s detJ$	Total Error Bound (J) $\sum_{i=1}^N \int_{e_i} (\sigma_s - \sigma_k)^T C (\sigma_s - \sigma_k) detJ$
16	3.2130E-03	3.2730E-03	5.8225E-04
128	3.6588E-03	3.6989E-03	3.4561E-04
1024	3.9259E-03	3.9611E-03	1.8364E-04
8192	4.0681E-03	4.0902E-03	9.3388E-05
65536	4.1408E-03	4.1530E-03	4.6886E-05

Table 5.6: Crack calculation results

Number of Elements	Total Relative Energy Error
N	$\frac{\sum_{i=1}^N \int_{e_i} (\sigma_s - \sigma_k)^T C (\sigma_s - \sigma_k) \det J}{0.5 \times (\sum_{i=1}^N \int_{e_i} \sigma_k^T C \sigma_k \det J + \sum_{i=1}^N \int_{e_i} \sigma_s^T C \sigma_s \det J)}$
16	1.7954E-01
128	9.3946E-02
1024	4.6569E-02
8192	2.2894E-02
65536	1.1306E-02

Table 5.7: Crack calculation results

Figure 1: Crack strain energy error with
16 elements

Figure 3: Crack strain energy error with
1,024 elements

Figure 2: Crack strain energy error with
128 elements

Figure 4: Crack strain energy error with
8,192 elements

5.4 6-node and 15-node Prism Element Text

The purpose of this test is to compare the results of 6-node and 15-node prism element calculation with the results of 8-node hexahedral element.

The material properties of this test are: $E = 69 \text{ } GP_a$, $\nu = 0.33$.

An 8-node hexahedral element can be divided into two triangle prisms. There are three ways to do that, i.e., cutting the 8-node hexahedral element in x, y, or z direction separately.

In this calculation, we use a 3DL-block mesh with 448 8-node hexahedral elements, or 896 6-node (or 15-node) prism elements. The boundary conditions are exactly the same as those in section (5.2), case one. The calculation results are listed in Table (5.8) and (5.9).

From the result, one can see that the total KAS strain energy, the total SAS strain energy and the total error bound of strain energy for the 6-node prism element are almost 100 times larger than the 8-node hexahedral element. In other words, the 8-node hexahedral element is much accurate than the 6-node prism element, when the singularity exists.

Moreover, the 6-node prism element in x direction is a little bit more accurate than those in y and z direction, because the non-zero displacement boundary condition is in x direction. The 15-node prism element is less accurate than the 6-node prism element.

Figure 5.17: 6-node prism element in x, y, z directions

Element Types	Total KAS Strain Energy (J)	Total SAS Strain Energy (J)	Total Error Bound (J)	Total Relative Energy Error
8-node Hexahedron	4.4771E-02	4.4686E-02	1.6136E-04	3.6075E-03
6-node Prism in x	4.8122	4.8013	2.2340E-02	4.6477E-03
6-node Prism in y	4.8322	4.8371	3.4683E-02	7.1738E-03
6-node Prism in z	4.8336	4.8330	3.6564E-02	7.5651E-03
15-node Prism in x	4.7895	5.1358	2.7446E-01	5.5306E-02

Table 5.8: 3DL-Block calculation results for 6-node prism

Element Types N	Maximum of Element Relative Error $\frac{\int_{e_i} (\sigma_s - \sigma_k)^T C (\sigma_s - \sigma_k) \det J}{\frac{1}{2} \left(\sum_{i=1}^N \int_{e_i} \sigma_k^T C \sigma_k \det J + \sum_{i=1}^N \int_{e_i} \sigma_s^T C \sigma_s \det J \right)}$	Minimum of Element Relative Error $\frac{\int_{e_i} (\sigma_s - \sigma_k)^T C (\sigma_s - \sigma_k) \det J}{\frac{1}{2} \left(\sum_{i=1}^N \int_{e_i} \sigma_k^T C \sigma_k \det J + \sum_{i=1}^N \int_{e_i} \sigma_s^T C \sigma_s \det J \right)}$
8-node Hexahedron	1.0550E-04	1.0509E-08
6-node Prism in x	7.8403E-05	2.8073E-08
6-node Prism in y	1.7135E-04	5.0570E-09
6-node Prism in z	1.6394E-04	4.5171E-09
15-node Prism in x	1.7135E-04	5.0570E-09

Table 5.9: Maximum and minimum element relative error

Chapter 6

Test Problems for SAF

6.1 Divergence Free Test

As described in section (2.2.2), the KAF thermal flux need not satisfy the energy equation. If, in a particular case, the KAF thermal flux does satisfy energy equation, the KAF thermal flux will exactly equal the SAF thermal flux at every point in the interior domain. Moreover, both KAF and SAF equal the exact solution and the error is identically zero.

Assuming the rate of specific enthalpy $\dot{h} = 0$ and the energy production rate $p = 0$, the energy equation becomes Laplace's equation:

$$\nabla \cdot q = 0$$

or writing in component form:

$$\frac{\partial q_x}{\partial x} + \frac{\partial q_y}{\partial y} + \frac{\partial q_z}{\partial z} = 0 \quad (6.1)$$

We set the value of the KAF thermal flux at Gauss points to q_x equal to zero or constants a_0 and (q_y, q_z) to the following function of (y, z) . The coefficients (a_0, a_1, a_2, a_3) can be chosen arbitrarily.

$$\begin{pmatrix} q_x \\ q_y \\ q_z \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & -z \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} \quad (6.2)$$

This thermal flux is exactly divergence free, i.e., $\nabla \cdot q = 0$. Therefore the SAF thermal flux should exactly equal the KAF thermal flux at any point (x, y, z) . Also the projected KAF thermal flux at face Gauss points should equal the thermal flux evaluated at the face Gauss point (x, y, z) .

The equation (6.2) sets the x components of thermal flux to be constant and in y and z components to be first order polynomial of (y, z) . Symmetrically, we also can use equations which set the y or z components of thermal to be constant and x and z

or x and y components to be first order polynomial of (x, z) or (x, y) to run this test:

$$\begin{pmatrix} q_x \\ q_y \\ q_z \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -z \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} \quad (6.3)$$

or

$$\begin{pmatrix} q_x \\ q_y \\ q_z \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & -y \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} \quad (6.4)$$

This test allows all data to be checked at each stage of the computation. It requires the code to set the values of the KAF thermal flux at volume Gauss point. This should work for any mesh and any type of element.

6.2 8-node Hexahedral Element Bar Test

The purpose of this test is to test the SAF thermal flux and to investigate how the bubble function works with a transient thermal flux.

The thermal conductivity of the material is: 109 W/mk .

Figure (6.1) illustrates the mesh of ten 8-node hexahedral elements bar and bound-

Figure 6.1: Boundary condition for the 8-node hexahedral element bar test.

ary condition of the test. The size of domain is $0.1 \times 0.01 \times 0.01 mm$. The temperature on the left side is $300K$ and the thermal flux on the right side is $-1000W/m^2$. The top, bottom, front and back faces are thermal insulated. The initial temperature of the whole domain is $300K$.

We conduct two kinds of calculations in this test. One is to compute with 25 time steps with a time step size of 2 seconds, so as to show microscopically the behaviors of the transient thermal flux and how the bubble function works. The second kind of calculation uses 30 time steps and a time step size of 40 seconds to reveal macroscopically the relationship between the KAF and the SAF thermal flux and error.

6.2.1 25 time steps with time step size of 2 seconds

During the first time step, the thermal flux acts as a thermal impact to the right side of the bar. The SAF thermal flux in this side reach $-350W/m^2$ (Fig. 6.2). In the left

side of the bar, due to not enough time for the heat to transfer to this side, the thermal flux still remains at zero. In other words, the left side of the bar is still in steady state. The increment of SAF thermal flux due to the bubble function in the left side is zero and in the right side $125W/m^2$ (Fig. 6.3). The effect of the bubble function on the side at steady state is zero. As time passes, the peak value of the increment of SAF thermal flux due to bubble function goes down and, at the same time, moves from the right side toward the left side. The SAF thermal flux in both sides gradually increases, but still remains smaller than the BC value $-1000W/m^2$. Finally, with enough time steps, the whole bar will reach the steady state and coefficients of the bubble function goes to zero.

6.2.2 30 time steps with time step size of 40 seconds

In this calculation, the time step size is 40 seconds. In the first time step, the SAF thermal flux in the right side is $-832W/m^2$ and the left side is $-160W/m^2$. The difference in the SAF thermal flux with bubble functions and without bubble functions is about $189W/m^2$ in the right side and $34W/m^2$ in the left side (Figure 6.4). In the second time step, the SAF thermal flux difference drops down very steeply to $12W/m^2$ in the right side and increases a small amount in the left side. For the last time step, it reaches steady state. The SAF thermal flux in the whole domain reaches the BC value $-1000W/m^2$ and the effect of the bubble function is zero.

The total SAF energy without bubble functions is almost the same as the total KAF energy in each time step (Fig. 6.6). Those values, in the first 10 time steps, are slightly more than the total SAF energy with bubble function.

The total error bounds and total relative error, during the steady state (about after the 9th time step), are exactly same with bubble function as without it. But in the transient state, the total error bounds and total relative error without bubble functions have lower values than with bubble functions (Fig. 6.7 and Fig. 6.8).

6.3 6-node Prism Element Text

The purpose of this test is to compare the results of a 6-node prism element calculation with the results of 8-node hexahedral element.

To facilitate the comparison with the results of calculation of the 8-node hexahedral element, we use the same domain, the same boundary condition, the same material and the same number of time steps and time step size as those used in section (6.2) to conduct this test. Figure (6.9) illustrates the 6-node prism element mesh.

The total KAF energy for both 8-node hexahedral element mesh and 6-node prism element mesh is almost the same (Fig. 6.10) in each time step. The SAF energy for the 6-node prism element mesh is a little bit smaller than those for the 8-node hexahedral element mesh in the beginning of the test. When the time step is more than 9, the

total KAF and SAF energy for two kinds of element tend to be equal.

The total relative energy error and total error bound for the 6-node prism element mesh are greater than those for the 8-node hexahedral element mesh during the first 9 time steps. From the 10th time step, it is very close to the steady state. The error for these two kinds of mesh is the same and approaches zero.

From this test, we can conclude that, for the case of transient thermal flux, the 8-node hexahedral element mesh is more accurate than 6-node prism element mesh.

Figure 6.2: Element number vs. thermal flux for the 8-node hexahedral.

Figure 6.3: Element number vs. increment of thermal flux for the 8-node hexahedral.

Figure 6.4: Element number vs. thermal flux increment due to bubble function for the 8-node hexahedral.

Figure 6.5: Element number vs. thermal flux for the 8-node hexahedral.

Figure 6.6: Number of time steps vs. total energy for the 8-node hexahedral.

Figure 6.7: Number of time steps vs. total error bound for the 8-node hexahedral.

Figure 6.8: Number of time steps vs. total relative energy error for the 8-node hexahedral.

Figure 6.9: Boundary condition for the 6-node prism element bar test.

Figure 6.10: Number of time steps vs. total energy.

Figure 6.11: Number of time steps vs. total relative energy error.

Figure 6.12: Number of time steps vs. total error bound.

Chapter 7

Conclusions and Contributions

7.1 Conclusions

1. The duality method in error analysis employing a statically admissible stress field or a statically admissible thermal flux field can compute a bound on the error.
2. The SAS calculation is suitable for most types of elements. The SAF calculation is only feasible for some of them. Table (7.1) lists the feasibility of SAS and SAF calculation for several element Types.
3. Uniform mesh refinement does not efficiently reduce the relative error in the singular region. This clearly shows the value of adaptive mesh refinement

Element Types	SAS Calculation		SAF Calculation	
	The Size of $[\bar{H}]$	works?	The Size of $[\bar{H}]$	works?
4-node Tetrahedron	36×24	✓	12×12	✓
10-node Tetrahedron	72×60	✓	72×60	✓
6-node Prism	54×36	✓	18×18	✓
15-node Prism	108×90	✓	36×45	✓
8-node Hexahedron	72×48	✓	24×24	✓
20-node Hexahedron	144×120	✓	48×60	×
32-node Hexahedron	216×192	✓	72×96	×

Table 7.1: The feasibility of SAS and SAF calculation for various element types.

4. Using the bubble function, it is possible to compute an error estimate when a body force or inertial force is present in stress analysis, or a heat source or a transient specific enthalpy is present in the thermal analysis.

7.2 Contributions

My main contribution was developing the test problems, helping to debug the C++ code and interpreting the results of test problems.

The theory presented in this thesis was taken from the Ph.D. thesis of Akbar Mashaie and from internal reports written by Professor Goldak. The exception is

page 40, where to discuss the size of null space of the matrix $[K_0]$, and page 116, where to discuss the feasibility of SAS and SAF calculation for various element types.

The C++ code to compute the statically admissible stress and thermal flux was written by Shaodong Wang. I wrote C code to compute the SAS Nodal Forces on Faces for one node. The C code evaluated the following steps.

1. For each node, form a matrix K of size $N \times N$ initialized to all zeroes and a vector P of size $N \times 1$. N is the number of faces shared by this node.
2. For each element $E(i)$ shared by this face, set the entries in the row (i) for the faces in the first element to $+1$ or -1 depending on whether this element is the first or second element for this face ID.
3. For each element $E(i)$, assign the nodal load component $P(i) = Q(h)$ from the vector. Q were evaluated by a preprocessing loop over elements and these values cached. It has size equal to the number of elements times number of nodes per element times 3.
4. Repeat steps 2 and 3 for each element that is shared by this node.
5. For each face, fetch the nodal force on the face at this node due to the average traction, i.e., the jump in the traction across the face. This is a vector of size M for this node. The nodal values on each face were evaluated by a preprocessing loop over faces and these values cached. It has size equal to the number of faces

times number of nodes per face times 3. The order should be entity, space, time and component. Entity is type face. Space refers to (x,y,z) . Time is zero or 1.

6. Use SVD to compute the null space of the matrix K . The null space appears to result from neglecting the conservation of angular momentum associated with the jumps in tractions on element faces.

Bibliography

- [1] Pedro Diez, Juan Jose Egozcue, Antonio Huerta, *A posteriori error estimation for standard finite element analysis*, Comput. Methods Appl. Mech. Engrg. 163 (1998) 141-157.
- [2] I. Babuska and W.C. Rheinboldt, *A posteriori error estimates for the finite element method*, Int. J.Numer. Methods Engrg. 12 (1978) 1597-1615.
- [3] M. Ainsworth and J.T. Oden, *A unified approach to a-posteriori error estimation using element residual methods*, Numer. Math. 65 (1993)23-50.
- [4] I. Babuska, T. Strouboulis and C.S. Upadhyay, *A model study of a posteriori error estimators for linear problems. Error estimation in the interior of patch wise uniform grids of triangles*, Comput. Methods Appl. Mech. Engrg. 114 (1994) 307-378.
- [5] I. Babuska, T. Strouboulis, C.S. Upadhyay, S.K. Gangaraj and K. Copps, *Validation of a posteriori error estimators by numerical approach*, Int. J. Numer.

Methods Engrg. 37 (1994) 357-374.

- [6] O. C. Zienkiewicz and Alan Craig, *Adaptive Refinement, Error Estimates, Multi-grid Solution, and Hierarchic Finite Element Method Concepts, Accuracy Estimates and adaptive Refinements in Finite Element Computations*, Edited by; I. Babuska, O. C. Zienkiewicz, J. Gago, E.R. de A. Oliverra, 1986, John Wiley & Sons Ltd, pp 25-59.
- [7] R.E. Bank and A. Weiser, *Some a posteriori error estimators for elliptic partial differential equations*, Math. Comput. 44 (1985) 283-301.
- [8] O.C. Zienkiewicz and J.Z. Zhu, *A simple error estimator and adaptive procedure for practical engineering analysis*, Int. J. Numer. Methods Engrg. 24 (1987) 337-357.
- [9] O.C. Zienkiewicz and J.Z. Zhu, *The superconvergent patch recovery and a posteriori error estimates*, Part 1: The recovery technique, Int. J.Numer. Methods Engrg. 33 (1992)1331-1364.
- [10] O.C. Zienkiewicz and J.Z. Zhu, *The superconvergent patch recovery and a posteriori error estimates*, Part 2: Error estimates and adaptivity, Int. J. Numer. Methods Engrg. 33 (1992)1365-1382.

- [11] P. Ladeveze, J.-P. Pelle and Ph. Rougeot, *Error estimation and mesh optimization for classical finite elements*, Engrg. Comput. 8 (1991) 69-80.
- [12] H. Ohtsubo and M. Kitamura, *Numerical investigation of element-wise a posteriori error estimation in two and three-dimensional elastic problems*, Int. J. Numer. Methods Engrg. 34 (1992) 969-977.
- [13] Akbar Mashaie, *Error Estimates for Finite Element Solutions of Elliptic Boundary Value Problems*, PH.D Thesis, Carleton University, 1991.
- [14] J. Goldak, *The Notes of Discrete SAS Field*, 2003.
- [15] J. Goldak, *The Notes of Discrete SAF Field*, 2003.
- [16] J. Goldak, *Personal Communication*, 2003.
- [17] J. Goldak, *The Notes of Project Data Between Two Kinds of Elements*, 2003.
- [18] William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling, *Numerical Recipes in C: the art of scientific computing*, Cambridge University Press, 1992.
- [19] J.P. Holman, *Heat Transfer*, 7th ed., McGraw-Hill, 1990.
- [20] Kenneth H. Huebner, Earl A. Thornton, Ted G. Byrom, *The Finite Element Method for Engineers*, 3rd ed., Wiley, 1995.

- [21] O.C. Zienkiewicz, *The Finite Element Method*, London, McGraw-Hill, 1977.
- [22] O.C. Zienkiewicz and R.L. Taylor, *The Finite Element Method*, Vol.2: Solid Mechanics, 5th ed., Springer-Verlag, 1998.