IMPLEMENTATION OF B-SPLINES IN A CONVENTIONAL FINITE ELEMENT FRAMEWORK

A Thesis

by

BRIAN CHRISTOPHER OWENS

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

May 2009

Major Subject: Aerospace Engineering

IMPLEMENTATION OF B-SPLINES IN A CONVENTIONAL FINITE ELEMENT FRAMEWORK

A Thesis

by

BRIAN CHRISTOPHER OWENS

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

May 2009

Major Subject: Aerospace Engineering

ABSTRACT

Implementation of B-splines in a
Conventional Finite Element Framework. (May 2009)

Brian Christopher Owens, B.S., Texas A&M University
Chair of Advisory Committee: Dr. John D. Whitcomb

The use of B-spline interpolation functions in the finite element method (FEM) is not a new subject. B-splines have been utilized in finite elements for many reasons. One reason is the higher continuity of derivatives and smoothness of B-splines. Another reason is the possibility of reducing the required number of degrees of freedom compared to a conventional finite element analysis. Furthermore, if B-splines are utilized to represent the geometry of a finite element model, interfacing a finite element analysis program with existing computer aided design programs (which make extensive use of B-splines) is possible.

While B-splines have been used in finite element analysis due to the aforementioned goals, it is difficult to find resources that describe the process of implementing B-splines into an existing finite element framework. Therefore, it is necessary to document this methodology. This implementation should conform to the structure of conventional finite elements and only require exceptions in methodology where absolutely necessary. One goal is to implement B-spline interpolation functions in a finite element framework such that it appears very similar to conventional finite elements and is easily understandable by those with a finite element background.

The use of B-spline functions in finite element analysis has been studied for advantages and disadvantages. Two-dimensional B-spline and standard FEM have been compared. This comparison has addressed the accuracy as well as the computational efficiency of B-spline FEM. Results show that for a given number of degrees of freedom, B-spline FEM can produce solutions with lower error than standard FEM. Furthermore, for a given solution time and total analysis time B-spline FEM will typically produce solutions with lower error than standard FEM. However, due to a more coupled system of equations and larger elemental stiffness matrix, B-spline FEM will take longer per degree of freedom for solution and assembly times than standard FEM. Three-dimensional B-spline FEM has also been validated by the comparison of a three-dimensional model with plane-strain boundary conditions to an equivalent two-dimensional model using plane strain conditions.

To my parents, for their love and support

ACKNOWLEDGMENTS

I wish to express my gratitude to my advisor Dr. John Whitcomb for supporting me financially and academically. I'd like to thank him for the opportunity to work under him as an undergraduate. It truly had a great impact on my choice to pursue graduate studies. Without his patience and constant guidance, this work would have never been complete. His emphasis on quality has encouraged me to work diligently and thoroughly in my research.

I would also like to thank Dr. Walter Haisler and Dr. Xin-Lin Gao for taking the time to serve on my thesis committee. Furthermore, I would like to thank the Aerospace Department staff, especially Ms. Karen Knabe for their every day assistance during my graduate studies.

I would like to acknowledge Bhavya Aggarwal for her research in applying B-splines to plane elasticity problems. Without her groundwork in this area, this thesis may not have been possible. She was of great assistance as I struggled to understand the concepts of B-splines and their application in finite elements. I would like to thank Julian Varghese for his constant assistance as I adapted to research and graduate school. I cannot thank him enough for his patience and helpfulness. I would also like to thank Deepak Goyal for sharing his experience with me. Furthermore, I'd like to thank my teammates Kevin Maxwell and Ross McLendon for their support.

I am thankful to all my friends for their support outside of my research and academics, especially Dasia, Nick, David, Forrest, Shalom, Luis, and Jehon. I would also like to thank my parents, grandparents, brothers, and sisters for their lifelong support and encouragement.

This work was supported by Clarkson Aerospace. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of Clarkson Aerospace.

This work was supported by the US Air Force Office of Scientific Research, funded by Contract Ref No. FZ9550-07-1-0207. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the US Air Force Office of Scientific Research.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

FIGURE Page

CHAPTER I

INTRODUCTION

A. Overview

Throughout history engineers and mathematicians have discretized complex systems into smaller components to gain a better understanding of the system. Engineers typically decompose a larger physical region into smaller regions that have a simple behavior that is easier to understand. After gaining insight on the behavior of these smaller regions, they may be reassembled to understand the behavior of the overall system [1]. The finite element method(FEM) has roots in this methodology. The concept of FEM began in the early 1940's and finally reached the familiar form used today in the late 1950's [2].

The finite element method approximates the solution to partial differential equations (PDE) over a domain. This domain is discretized into sub-domains known as "elements". The boundary of an element is represented by distinct points known as "nodes". These nodes are points at which the approximate solution of the PDE is obtained [2]. Since the creation of the finite element method, it has been applied to a wide range of problems including structural analysis, fluid flow, and electromagnetism.

Finite element analysis typically employs Lagrangian or Hermite interpolation functions. These functions are easy to implement and can provide sufficient accuracy. However, they are relatively inefficient and can be expensive computationally. Furthermore, if a smooth solution with continuity of higher-order derivatives is desired these interpolation functions become even less efficient. For models with many

This thesis follows the style and format of Journal of Applied Mechanics.

degrees of freedom inefficiencies involved with interpolation functions can drastically increase the time it takes to solve the system of equations in a finite element analysis.

B-splines are polynomial curves that are defined piecewise over a domain [3]. These curves have the ability to maintain smoothness and continuity of higher-order derivatives [4] while maintaining efficiency. There are various reasons to motivate the use of B-spline functions in place of Lagrangian or Hermite interpolation functions. First, B-splines are commonly used in computer aided design (CAD) to accurately model complex geometry. Therefore, if B-splines are implemented in the finite element method interfacing CAD geometry with a finite element analysis is more convenient. Also, the use of a single B-spline function instead of multiple piecewise Lagrangian or Hermite functions reduces the overall degrees of freedom in an analysis. This reduction in degrees of freedom has the potential to reduce the time for a finite element analysis significantly.

B-spline functions have already been implemented in the finite element method. Those who undergo the task of implementing B-splines in finite elements typically note the increased smoothness, accuracy, and computational savings [5, 6, 7] compared to conventional finite elements. B-spline FEM has many differences with conventional finite elements, such as the need for special pre-processing and post-processing routines. However, there are also many similarities between the two methods, such as the basic formulation of the two methods. If one is not careful, it is easy to make B-spline finite elements appear confusing and much more different than conventional finite elements.

This work will focus on implementing B-spline functions within a conventional finite element framework using a clear and concise methodology. All attempts will be made to use the existing framework where possible. Therefore, exceptions required due to the use of B-spline functions will only be created where absolutely necessary.

The overall goal is to create a method that appears very similar to conventional finite elements and can be easily understood by those with a finite element background. Pre-processing tools will also be created to ease the difficulties of mesh generation associated with B-spline finite elements. Additional post-processing routines to calculate displacement information for B-spline FEM will be presented. Finally, a comparison between B-spline and conventional finite element analysis will be completed with regards to accuracy and computational efficiency.

B. Literature Review

B-Spline curves were created as an improvement over Bézier curves in the 1970's. This effort to produce splines that contained local support was led by Riesenfeld [8]. Since B-Splines have local support the shape of a particular segment of the curve can be altered without affecting the overall curve. This gives one more control over the shapes produced by B-Spline curves [4]. Cox and de Boor [9] discovered a recursive formula for the definition of B-Spline basis functions of any order. B-Splines were extended to non uniform rational B-splines(NURBS) initially by Versprille [10], and later by Piegel and Tiller [11, 12] in the 1980's. NURBS associates a weight with each control point or basis function that allows for greater shape control. Currently, NURBS curve/surface representation is the standard in computer aided graphics and design.

B-splines may have been created for the purpose of curve and surface representation but much work has been done to apply the advantages of B-splines in numerical methods such as finite elements. Some work has focused on interfacing CAD systems with finite element analysis by utilizing B-splines within a finite element analysis [13]. The B-spline utilized in the finite element analysis are the same functions used for geometry representation in CAD. In this work Kagan et al. have focused on extending geometric design directly to mechanical analysis. Kagan et al. presents convergence data for various degrees of B-spline functions.

Non-uniform rational B-splines have also been used to enhance the finite element method by Huerta et al. [6]. A CAD description of the boundary of a model is used in the finite element analysis. Conventional interpolation functions are utilized on the interior and special piecewise polynomial functions are implemented at the NURBS boundary. The computational efficiency of utilizing NURBS in FEM is noted as well as the ability to obtain an accurate solution using a coarser mesh and high-order interpolation.

Inoue et al. have also implemented NURBS within the finite element method to aid in product shape design [14]. The goal of this work is the integration of finite element analysis with computer aided design software. This study involves a shell finite element that has exactly the same geometry as the NURBS surface for the manufactured product. The NURBS finite element method is applied to bending analyses of plates and shells. NURBS FEM approximated solutions to buckling analysis with lower error than ordinary FEM. The authors suggest that this may be due to the precise geometry modeled in NURBS FEM.

Hughes et al. have performed significant work in the implementation of NURBS in finite elements. Studies of this implementation have focused on refinement and continuity [15]. In this implementation the NURBS parameters from CAD are used in the finite element analysis. However, refinement through knot insertion and/or degree elevation is used to obtain a better finite element solution while still maintaining the precise geometry created through CAD. It appears for most cases that an increase in smoothness provided by NURBS increases the accuracy of the solution.

Others offer further improvements on the B-spline finite element method. Kagan et al. have developed methods for adaptive refinement of B-spline finite elements [16]. This work presents various refinement methods and also addresses continuity of the solution approximation. Li and Wang also presented a new 8-node quadrilateral spline finite element [17]. Simplifications of derivatives, integrals and products of shape functions are also presented. Results suggest better performance than a standard 8-node isoparametric element.

Splines have also been applied to numerical analysis methods other than finite elements. Kumar applied splines to a finite difference boundary value problem [18]. Second order convergence of the spline finite difference method is noted as well as the efficiency of splines in solving the boundary value problem. Kumar also applied fourth-order splines to a finite difference boundary value problem and noted fourth-order convergence [19].

Mohammadi et al. also used cubic-splines to solve a system of boundary value problems [20]. This method is applied to multiple problems and convergence behavior is studied. The error in the solutions of various fourth-order boundary value problems analyzed with the cubic-spline method was shown to be lower than other methods employed for solving the system of boundary value problems.

Kadalbajoo and Aggarwal have used a fitted mesh B-spline method for solving singularly perturbed boundary value problems [21]. A uniform mesh is created and in conjunction with the use of their B-spline method a tridiagonal linear system is achieved. Results show uniform convergence with better results than existing methods.

Brown et al. focused on the accuracy of B-spline finite element approximation to a PDE surface [22]. Data transfer between CAD systems is addressed. Both refinement and degree elevation were studied to achieve a better approximation. For their particular application of surface approximation there was no significant difference using periodic or non-periodic B-spline functions.

Caglar et al. compared B-spline approximation with finite element, finite difference, and finite volume methods for a two-point boundary value problem [23]. Results show lower maximum error norms for B-spline approximation than all other methods. The conclusion is made that B-spline functions are better suited to approximate a smooth solution than the other methods.

There have been various applications of numerical methods using splines. Pullman and Schaff performed analysis of a cross-ply laminate with a circular hole using a 3D spline variational method [24]. Results show good agreement between the spline variational method and ordinary finite element method. It was determined that the spline variational method can reduce the number of degrees of freedom by a factor of 3-5 while maintaining interlaminar stress distributions comparable to ordinary finite elements.

Mizusawa has applied a spline element method to analyze the bending of skew plates [25]. Accuracy and convergence were compared to other numerical methods. The spline element method showed good agreement with other numerical methods. It was also observed that use of high-ordered splines and a mesh grading technique were effective in improvement of accuracy.

Leung and Au applied spline finite elements to beams and plates. The advantages of splines including computational efficiency, flexibility in modeling different boundary conditions, and the variation diminishing property splines possess are noted[26]. Good agreement is observed between reference methods including finite strip solutions. When examining the lower number of degrees of freedom the efficiency of splines is further emphasized.

Kong and Cheung have applied spline finite strip analysis to shear-deformable plates [27]. This type of analysis was proposed to study thick laminated composite plates. The use of cubic B-splines allowed for the flexibility to meet various continuity requirements for classical plate analysis. There is very good agreement between analytical and existing numerical solutions.

PengCheng et al. utilized a multivariable spline element method to perform vibration analysis of plates [28]. It was determined that the multivariable spline element method was very effective in achieving a solution with high accuracy. Good convergence characteristics are also noted.

Gupta et al. also used cubic B-splines finite elements for shell analysis [29]. The second-derivative continuity of B-splines is well suited for shell analysis where cubic Hermite functions are not sufficient. The use of cubic B-splines eliminates stress discontinuities in results. The overall reduction in degrees of freedom provided by the use of B-splines is also noted.

Kim et al. used a B-spline finite element method to shape and analyze a torque converter clutch disk [30]. An algorithm is utilized to reposition ”master nodes” that define the B-spline curve geometry to fall directly on the surface of the model. A mesh smoothing technique is also applied to achieve a better finite element solution. This method was utilized to design the disk with optimum shape, providing a significant weight reduction.

Zhong and Yuqiu have used spline elements for the analysis of tall buildings [31]. Tall buildings of arbitrary shape, such as structures with irregular openings and boundaries, can be analyzed using this method. The order of B-spline functions is carefully chosen in different dimensions to increase computational efficiency. Results show good approximations with only a few elements for both static and dynamic analysis.

Kutluay and Esen used a B-spline finite element method to analyze a thermistor problem involving electrical conductivity [32]. Cubic B-splines were used in conjunction with a Galerkin method. Data is presented for results at a variety of mesh refinements. The approximated solution of the analysis converges to the known exact solution.

Aksan has also used quadratic B-splines to approximate the solution of a 1D non-linear Burgers' equation [33]. The Burgers' equation was converted to a set of non-linear ordinary differential equations. Each equation was then solved by means of a quadratic B-spline finite element method. The high accuracy of the B-spline FEM solution approximation was noted. Gardner et al. also approximated the solution of Burgers' equation using a cubic B-spline finite element method [34]. Gardner et al. determined that cubic B-spline FEM produced more accurate results than other methods.

Patlashenko and Weller employed cubic B-spline collocation methods for analysis of panels under thermal and mechanical loading [35]. This method was applied to one and two-dimensional problems. The analysis also studied the non-linear induced response of panels. Again, the efficiency and possible acceleration of convergence through the use of B-splines is noted.

C. Scope of Research

This work continues the research begun by Bhavya Aggarwal at Texas A&M University [36]. Aggarwal laid the ground work for implementing B-spline functions in the in-house Alpha finite element software. While this work was very important, there were limitations of its implementation. B-splines were implemented for one and two-dimensional analysis. However, only uniform B-spline functions were implemented. This restricted finite element meshes to be uniform and control of mesh refinement was limited. Also, there were no pre-processing tools to set up a B-spline finite element analysis and models were quite time-consuming to make. Finally, the implementation did not integrate well with the existing Alpha software. In many ways the implementation was a kind of stand alone application that existed within the Alpha project, but did not make use of existing routines in Alpha.

The current research seeks to improve upon and refine Aggarwal's work. Chapter II of this thesis presents the formulation of a conventional elastic finite element method. Chapter III of this thesis will present the theory associated with Lagrangian and Hermite interpolation and introduce B-spline approximation. The theory and application of B-splines will be discussed. Furthermore, the formulations required for implementing B-spline functions in the finite element method for two and three-dimensional problems will be presented in chapter IV.

The new implementation of B-spline finite elements will reside in the Beta finite element software (an improved version of Alpha). Through the use of object oriented programming (OOP) the implementation will be simplified. OOP allows existing functionality to be inherited for a new implementation. Therefore, the new implementation of B-splines will be much leaner. New functionality will only be created where absolutely necessary. The new implementation of B-spline finite elements in Beta will be also expanded to include non-uniform B-spline functions, allowing the use of graded meshes in analyses. Furthermore, the method will be extended to the capability of three-dimensional analysis. Pre-processing utilities will be developed to expedite the creation of models for B-spline finite element analysis. Additional post-processing capabilities are also required to produce meaningful data from B-spline finite element analysis. These additional pre/post-processing routines will be explained in detail.

After a detailed explanation concerning the implementation of B-splines in the finite element method, two and three dimensional problems will be analyzed in chapter V. The results of the two-dimensional B-spline finite element analysis will be compared to conventional two-dimensional finite element analysis. The two methods will be compared with respect to accuracy and computational efficiency. The convergence characteristics of the two methods will also be studied. The capability of the three-dimensional B-Spline finite element method will also be presented as well as a validation of three-dimensional results. Finally, chapter VI presents conclusions and recommendations for future work.

CHAPTER II

FORMULATION OF THE ELASTIC FINITE ELEMENT METHOD

The main processes of a finite element analysis are: pre-processing, analysis, and post-processing. Pre-processing involves information about the mesh. This includes the overall geometry of the model, mesh refinement, and assignment of material to elements. Material data is also defined in this step. Finally, boundary conditions, including loads, constraints, and multi-point constraints are specified.

Analysis involves the calculation of elemental stiffness matrices and load vectors. These are obtained by expressing the governing differential equation in the weak form. The individual element matrices and load vectors are then assembled to create a global stiffness matrix and load vector for the model. Constraints specified as part of the boundary conditions may be applied by modification of the stiffness matrix and load vector. Finally, the systems of equations is solved and nodal displacements are obtained.

The final step of a finite element analysis involves post-processing. The displacements from the previous step may be used to calculate strains and stresses. These values are calculated at the quadrature points of an element because these points have the lowest error from approximations involving the interpolation functions. Quadrature point strains and stresses may be extrapolated to the nodes of an element so that they may be visualized on the mesh.

The following sections present the development of a three-dimensional elasticity formulation and a reduction to a two-dimensional elasticity formulation. The main steps of pre-processing, analysis, and post-processing are also explained.

A. Three-dimensional Elasticity Formulation

The formulation of an elastic finite element method begins with the equilibrium equations expressed using index notation

$$\sigma_{ij,j} + f_i = 0 \tag{2.1}$$

The use of index notation simplifies the formulation. Here σ_{ij} are stresses and f_i are body force components. The term $\sigma_{ij,j}$ represents the partial derivative of σ_{ij} with respect to x_j $(\frac{\partial \sigma_{ij}}{\partial x_j})$. These equilibrium equations will be expressed in weak form by first multiplying by test functions δu_i and integrating over the domain (Ω). In three dimensions the domain is a volume, in two dimensions the domain is an area.

$$\int_\Omega \delta u_i \left(\sigma_{ij,j} + f_i\right) d\Omega = 0 \tag{2.2}$$

Integration by parts yields

$$\int_\Gamma \left(\delta u_i \sigma_{ij} n_j\right) d\Gamma + \int_\Omega \left(f_i \delta u_i - \delta u_{i,j} \sigma_{ij}\right) d\Omega = 0 \tag{2.3}$$

For the three-dimensional case Γ represents the domain of a surface integral. For the two-dimensional case Γ represents the domain of a line integral. Use of Cauchy's stress formula (Eq. (2.4)) results in Eq. (2.5).

$$T_i = \sigma_{ij} n_j \tag{2.4}$$

$$\int_\Omega \left(f_i \delta u_i\right) d\Omega + \int_\Gamma \left(T_i \delta u_i\right) d\Gamma - \int_\Omega \left(\delta u_{i,j} \sigma_{ij}\right) d\Omega = 0 \tag{2.5}$$

Introduction of kinematic relations (Eq. (2.6)) allows for the weak form to be expressed as Eq. (2.7).

$$\epsilon_{ij} = \begin{cases} u_{i,j} & i = j \\ u_{i,j} + u_{j,i} & i \neq j \end{cases} \tag{2.6}$$

$$\int_{\Omega} (f_i \delta u_i)\, d\Omega + \int_{\Gamma} (T_i \delta u_i)\, d\Gamma - \int_{\Omega} (\delta \epsilon_{ij} \sigma_{ij})\, d\Omega = 0 \tag{2.7}$$

Since the terms δu_i are independent the weak form equations may be combined

$$\sum_i \left[\int_{\Omega} (f_i \delta u_i)\, d\Omega + \int_{\Gamma} (T_i \delta u_i)\, d\Gamma - \int_{\Omega} (\delta u_{i,j} \sigma_{ij})\, d\Omega \right] = 0 \tag{2.8}$$

Finite element analysis will employ a Galerkin approximation such that

$$\delta u_i = \sum_{m=1}^{n} \delta u_i^m N_m \tag{2.9}$$

Here the superscript m denotes the index of nodes for an element, m will range from one to the number of nodes of an element. Nodal displacements (u_i^m) will be collected in a degree of freedom (DOF) vector q_α such that

$$q_\alpha = [u_1^1, u_2^1, u_3^1, ..., u_1^n, u_2^n, u_3^n] \tag{2.10}$$

The functions δu_i and $\delta \epsilon_{ij}$ may be expressed in terms of the DOF vector (q_α).

$$\delta u_i = \frac{\partial u_i}{\partial q_\alpha} \delta q_\alpha \tag{2.11}$$

$$\delta \epsilon_{ij} = \frac{\partial \epsilon_{ij}}{\partial q_\alpha} \delta q_\alpha \tag{2.12}$$

Substitution of Eqs. (2.11) and (2.12) into Eq. (2.7) and canceling the δq_α term yields

$$\sum_\alpha \left[\sum_i \left[\int_{\Omega} \left(f_i \frac{\partial u_i}{\partial q_\alpha} \right) d\Omega + \int_{\Gamma} \left(T_i \frac{\partial u_i}{\partial q_\alpha} \right) d\Gamma - \int_{\Omega} \left(\sigma_{ij} \frac{\partial \epsilon_{ij}}{\partial q_\alpha} \right) d\Omega \right] \right] = 0 \tag{2.13}$$

Note that the last integral of Eq. (2.13) may be expressed using Voigt notation as

$$\int_{\Omega} \left(\sigma_k \frac{\partial \epsilon_k}{\partial q_\alpha} \right) d\Omega \tag{2.14}$$

A strain-displacement matrix ($B_{k\alpha}$) may be introduced that relates displacements and strain via the kinematic relations.

$$B_{k\alpha} = \frac{\partial \epsilon_k}{\partial q_\alpha} = \left[\begin{array}{ccc|ccc|c|ccc} \frac{\partial N_1}{\partial x_1} & 0 & 0 & \frac{\partial N_2}{\partial x_1} & 0 & 0 & & \frac{\partial N_n}{\partial x_1} & 0 & 0 \\ 0 & \frac{\partial N_1}{\partial x_2} & 0 & 0 & \frac{\partial N_2}{\partial x_2} & 0 & & 0 & \frac{\partial N_n}{\partial x_2} & 0 \\ 0 & 0 & \frac{\partial N_1}{\partial x_3} & 0 & 0 & \frac{\partial N_2}{\partial x_3} & \cdots & 0 & 0 & \frac{\partial N_n}{\partial x_3} \\ \frac{\partial N_1}{\partial x_2} & \frac{\partial N_1}{\partial x_1} & 0 & \frac{\partial N_2}{\partial x_2} & \frac{\partial N_2}{\partial x_1} & 0 & \cdots & \frac{\partial N_n}{\partial x_2} & \frac{\partial N_n}{\partial x_1} & 0 \\ 0 & \frac{\partial N_1}{\partial x_3} & \frac{\partial N_1}{\partial x_2} & 0 & \frac{\partial N_2}{\partial x_3} & \frac{\partial N_2}{\partial x_2} & & 0 & \frac{\partial N_n}{\partial x_3} & \frac{\partial N_n}{\partial x_2} \\ \frac{\partial N_1}{\partial x_3} & 0 & \frac{\partial N_1}{\partial x_1} & \frac{\partial N_2}{\partial x_3} & 0 & \frac{\partial N_2}{\partial x_1} & & \frac{N_n}{\partial x_3} & 0 & \frac{\partial N_n}{\partial x_1} \end{array}\right] \tag{2.15}$$

$$\epsilon_k = B_{k\alpha} q_\alpha \tag{2.16}$$

Therefore the integral of Eq. (2.14) may be expressed as

$$\int_\Omega \left(B_{k\alpha}\sigma_k\right) d\Omega = \int_\Omega \left(B^T \sigma\right) d\Omega \tag{2.17}$$

The stress tensor (σ_k) may be expressed in terms of displacements (q_α) by utilizing kinematic and constitutive relations. The constitutive matrix (C_{kl}) relates stress and strain. Note that thermal strains are assumed to be zero for this formulation.

$$\sigma_k = C_{kl}\epsilon_l = C_{kl} B_{l\alpha} q_\alpha \tag{2.18}$$

The constitutive matrix is calculated via material properties. For an orthotropic material, the constitutive matrix may be calculated as follows, all other elements of the constitutive matrix are zero. If necessary this matrix can be transformed to reflect a rotation of the material coordinate system relative to the global coordinate system.

$$\begin{aligned}
c_{11} &= \frac{1-\nu_{23}\nu_{32}}{E_{22}E_{33}\Delta} \\
c_{22} &= \frac{1-\nu_{31}\nu_{13}}{E_{33}E_{11}\Delta} \\
c_{33} &= \frac{1-\nu_{12}\nu_{21}}{E_{11}E_{22}\Delta} \\
c_{12} = c_{21} &= \frac{\nu_{21}+\nu_{31}\nu_{23}}{E_{22}E_{33}\Delta} \\
c_{13} = c_{31} &= \frac{\nu_{31}+\nu_{21}\nu_{32}}{E_{22}E_{33}\Delta} \\
c_{23} = c_{32} &= \frac{\nu_{32}+\nu_{31}\nu_{12}}{E_{33}E_{11}\Delta} \\
c_{44} &= G_{23} \\
c_{55} &= G_{31} \\
c_{66} &= G_{12} \\
\Delta &= \frac{1-\nu_{12}\nu_{21}-\nu_{23}\nu_{32}-\nu_{31}\nu_{13}-2\nu_{12}\nu_{23}\nu_{31}}{E_{11}E_{22}E_{33}}
\end{aligned} \tag{2.19}$$

Finally the integral of Eq. (2.14) may be expressed in terms of the strain-displacement matrix, constitutive matrix, and nodal displacements. The integral $\int_\Omega B^T CBd\Omega$ is termed the stiffness matrix (K).

$$\int_\Omega \left(B^T CBq\right) d\Omega = Kq \tag{2.20}$$

The remaining integrals of Eq. (2.13) contain the term $(\frac{\partial u_i}{\partial q_\alpha})$. This term may also be expressed as

$$\frac{\partial u_i}{\partial q_\alpha} = \left[\begin{array}{ccc|ccc|ccc|c|ccc}
N_1 & 0 & 0 & N_2 & 0 & 0 & N_3 & 0 & 0 & & N_n & 0 & 0 \\
0 & N_1 & 0 & 0 & N_2 & 0 & 0 & N_3 & 0 & \ldots & 0 & N_n & 0 \\
0 & 0 & N_1 & 0 & 0 & N_2 & 0 & 0 & N_3 & & 0 & 0 & N_n
\end{array}\right] \tag{2.21}$$

Therefore the load vector (F_α) may be expressed as

$$F_\alpha = \int_\Omega f_i \frac{\partial u_i}{\partial q_\alpha} d\Omega + \int_\Gamma T_i \frac{\partial u_i}{\partial q_\alpha} d\Gamma \tag{2.22}$$

In a finite element formulation the domain of interest is the volume or area of an element (Ω_e), and the load vector may be expressed in terms of interpolation functions by use of Eq. (2.21).

$$F_\alpha = \begin{bmatrix} \int_{\Omega_e} f_1 N_1 d\Omega_e + \int_{\Gamma_e} T_1 N_1 d\Gamma_e \\ \int_{\Omega_e} f_2 N_1 d\Omega_e + \int_{\Gamma_e} T_2 N_1 d\Gamma_e \\ \int_{\Omega_e} f_3 N_1 d\Omega_e + \int_{\Gamma_e} T_3 N_1 d\Gamma_e \\ \int_{\Omega_e} f_1 N_2 d\Omega_e + \int_{\Gamma_e} T_1 N_2 d\Gamma_e \\ \int_{\Omega_e} f_2 N_2 d\Omega_e + \int_{\Gamma_e} T_2 N_2 d\Gamma_e \\ \int_{\Omega_e} f_3 N_2 d\Omega_e + \int_{\Gamma_e} T_3 N_2 d\Gamma_e \\ \vdots \\ \int_{\Omega_e} f_1 N_n d\Omega_e + \int_{\Gamma_e} T_1 N_n d\Gamma_e \\ \int_{\Omega_e} f_2 N_n d\Omega_e + \int_{\Gamma_e} T_2 N_n d\Gamma_e \\ \int_{\Omega_e} f_3 N_n d\Omega_e + \int_{\Gamma_e} T_3 N_n d\Gamma_e \end{bmatrix} \tag{2.23}$$

This results in a system of equations in which nodal displacements are the unknowns

$$Kq = F \tag{2.24}$$

Elements in a finite element analysis can take on complex geometries. Therefore, elements are typically transformed to a normalized coordinate system (ξ_i). The domain of the normalized element is $\bar{\Omega}_e$. This normalized domain ranges from (-1,-1,-1) to (1,1,1). With this transformation the integrals of Eqs. (2.20) and (2.22) will take the form

$$K^{(e)} = \int_{-1}^{1} \int_{-1}^{1} \int_{-1}^{1} B^T C B |J| d\bar{\Omega}_e \tag{2.25}$$

$$F^{(e)} = \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} f_i \frac{\partial u_i}{\partial q_\alpha}|J|d\bar{\Gamma}_e + \int_{-1}^{1}\int_{-1}^{1} T_i \frac{\partial u_i}{\partial q_\alpha}|J|d\bar{\Gamma}_e \tag{2.26}$$

The term $|J|$ is the determinant of the Jacobian. The Jacobian may be expressed as

$$J_{ij} = \frac{\partial x_j}{\partial \xi_i} = x_j^k \frac{\partial N_k}{\partial \xi_i} \tag{2.27}$$

Integrals are calculated numerically using a Gaussian quadrature rule. A master element (see Fig. 1) may be defined in the normalized coordinate system, and interpolation function derivatives may be easily calculated at integration points with respect to the normalized coordinate system ($\frac{\partial N_m}{\partial \xi_i}$). Since the strain-displacement matrix is a function of interpolation function derivatives with respect to a global coordinate system ($\frac{\partial N_m}{\partial x_i}$), these normalized derivatives must be transformed by using the Jacobian.

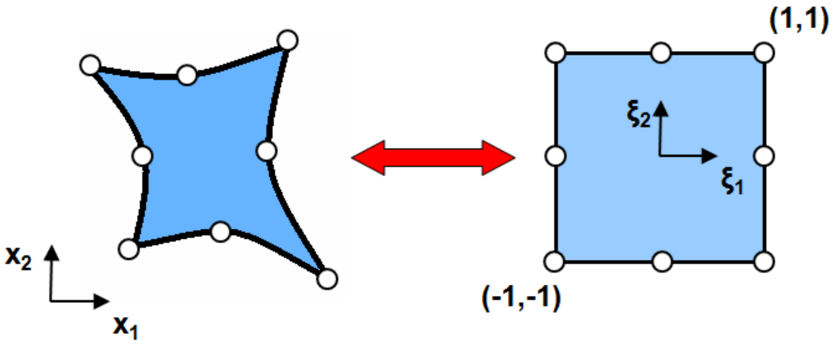


Fig. 1 Two-dimensional master element in normalized coordinate system

$$\frac{\partial N_m}{\partial x_i} = \frac{\partial \xi_j}{\partial x_i}\frac{\partial N_m}{\partial \xi_j} = J_{ij}^{-1}\frac{\partial N_m}{\partial \xi_j} \tag{2.28}$$

Finally, boundary conditions are a critical part of the elasticity formulation. Essential boundary conditions specify a displacement (u_i) and natural boundary conditions specify a traction (T_i).

B. Two-dimensional Elasticity Formulation

For a plane elasticity formulation many of the equations presented in the three-dimensional elasticity formulation hold true. The strain-displacement matrix may be reduced to

$$B_{k\alpha} = \frac{\partial \epsilon_k}{\partial q_\alpha} = \left[\begin{array}{cc|cc|c|cc} \frac{\partial N_1}{\partial x_1} & 0 & \frac{\partial N_2}{\partial x_1} & 0 & & \frac{\partial N_n}{\partial x_1} & 0 \\ 0 & \frac{\partial N_1}{\partial x_2} & 0 & \frac{\partial N_2}{\partial x_2} & \cdots & 0 & \frac{\partial N_n}{\partial x_2} \\ \frac{\partial N_1}{\partial x_2} & \frac{\partial N_1}{\partial x_1} & \frac{\partial N_2}{\partial x_2} & \frac{\partial N_2}{\partial x_1} & & \frac{\partial N_n}{\partial x_2} & \frac{\partial N_n}{\partial x_1} \end{array}\right] \tag{2.29}$$

The constitutive relations for a plane elasticity must assume plane stress (Eq. (2.30)) or plane strain ((Eq. 2.31)) conditions. Again, the constitutive matrix may be calculated via material properties. For plane stress, elements of the constitutive matrix are shown in Eq. (2.33). For plane strain, elements of the constitutive matrix are shown in Eq. (2.34).

$$\sigma_{33} = 0, \sigma_{13} = 0, \sigma_{23} = 0 \tag{2.30}$$

$$\epsilon_{33} = 0, \epsilon_{13} = 0, \epsilon_{23} = 0 \tag{2.31}$$

$$\begin{aligned} c_{11} &= \frac{E_{11}}{1 - \nu_{12}\nu_{21}} \\ c_{22} &= \frac{E_{22}}{1 - \nu_{12}\nu_{21}} \\ c_{12} = c_{21} &= \nu_{12}c_{22} \\ c_{66} &= G_{12} \end{aligned} \tag{2.32}$$

$$\begin{aligned}
c_{11} &= \frac{1-\nu_{23}\nu_{32}}{E_{22}E_{33}\Delta} \\
c_{22} &= \frac{1-\nu_{31}\nu_{13}}{E_{33}E_{11}\Delta} \\
c_{12}=c_{21} &= \frac{\nu_{21}+\nu_{31}\nu_{23}}{E_{22}E_{33}\Delta} \\
c_{66} &= G_{12} \\
\Delta &= \frac{1-\nu_{12}\nu_{21}-\nu_{23}\nu_{32}-\nu_{31}\nu_{13}-2\nu_{12}\nu_{23}\nu_{31}}{E_{11}E_{22}E_{33}}
\end{aligned} \tag{2.33}$$

The resulting load vector for a two-dimensional formulation is also reduced

$$F_\alpha = \begin{bmatrix}
\int_{\Omega_e} f_1 N_1 d\Omega_e + \int_{\Gamma_e} T_1 N_1 d\Gamma_e \\
\int_{\Omega_e} f_2 N_1 d\Omega_e + \int_{\Gamma_e} T_2 N_1 d\Gamma_e \\
\int_{\Omega_e} f_1 N_2 d\Omega_e + \int_{\Gamma_e} T_1 N_2 d\Gamma_e \\
\int_{\Omega_e} f_2 N_2 d\Omega_e + \int_{\Gamma_e} T_2 N_2 d\Gamma_e \\
\vdots \\
\int_{\Omega_e} f_1 N_n d\Omega_e + \int_{\Gamma_e} T_1 N_n d\Gamma_e \\
\int_{\Omega_e} f_2 N_n d\Omega_e + \int_{\Gamma_e} T_2 N_n d\Gamma_e
\end{bmatrix} \tag{2.34}$$

C. Pre-processing

Pre-processing in a conventional finite element analysis is relatively straight forward. Mesh generation involves the definition of model geometry using large macro-elements. The number of elements for each dimension of a macro-element may be specified and the nodes and elements can be constructed to "mesh" that particular macro-element. This is done for each marco-element used in the model geometry definition. Afterward, duplicate nodes on macro-element interfaces are removed so the mesh is completely "tied" together. Note that the mesh refinement need not be uniform, details of this implementation in a mesh generator are dependent on the developer.

A mesh consists of a listing of nodes and a listing of elements. Each node has a node number and coordinates associated with it. Each element has an element number and a connectivity list which contains the nodes which represent an element.

As stated in the formulation, boundary conditions must be applied to a model. In the case of essential boundary conditions displacements are specified at nodes. In the case of natural boundary conditions, a traction or distributed load is specified on the surface (or edge) of an element. This distributed load may then be resolved into equivalent nodal loads. Equivalent nodal loads for a two-dimensional element are computed via the second integral of Eq. (2.22). Through this methodology, equivalent nodal loads may be calculated for a given traction on an the surface of an element. If a traction is applied to multiple adjacent elements, the nodal forces are summed for coincident nodes as part of the assembly process (see Fig. 2). A body force (f_i) may be applied to an element by calculating an equivalent nodal load using the first integral of Eq. (2.22). Again, if elements share nodes the nodal forces are summed as part of the assembly process. Finally, if a concentrated load is applied to a node this load is added directly to the corresponding DOF of the load vector.

During pre-processing materials are defined. A constitutive matrix can be calculated from material properties using the relations in the formulation sections. Materials are also assigned to elements during this step.

D. Analysis

Pre-processing in a finite element defines the geometry of elements in a mesh, and equivalent nodal loads may be calculated from specified tractions and body forces using the relationships developed in the elasticity formulation.

The elemental stiffness matrix for each element may be calculated via Eq. (2.25).

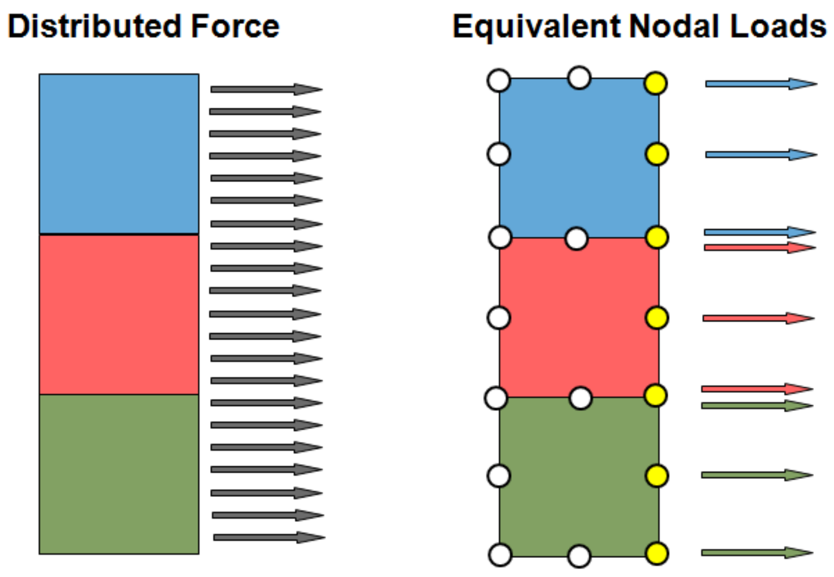


Fig. 2 Distributed force represented by equivalent nodal loads.

The assembly process then combines all elemental stiffness matrices and load vectors into a global stiffness matrix and load vector. The assembly process sums the components of the elemental stiffness matrices that share degrees of freedom. A typical implementation of this process is illustrated in Fig. 3.

The implementation begins by calculating a degree of freedom list for an element. This will be useful for assembly purposes. Integration point data is also determined at this time. Next, the interpolation functions and interpolation function derivatives are calculated at the given integration point. The Jacobian may be calculated from the interpolation function derivatives and nodal coordinates of the element. These interpolation function derivatives may be transformed to be with respect to the global coordinate system by using the inverse of the Jacobian. Next, the contribution of a particular integration point to the integration factor or element volume is calculated. The contribution to the element volume is calculated by multiplying the contribution to the determinant of the Jacobian by the weight of the current integration point. The contribution to the strain-displacement ($B_{k\alpha}$) matrix may be calculated using the global interpolation function derivatives. Finally, the contribution of an integration point to the elemental stiffness matrix and load vector may be calculated. Once an element stiffness matrix and load vector are calculated, they are assembled in the global stiffness matrix and load vector by means of the degree of freedom list.

Upon assembly of the global stiffness matrix and load vector, constraints may be specified by altering the resulting system of equations. The analysis proceeds by solving the system of equations for the unknown displacements. Next, these displacements may be post-processed to obtain stresses or strains.

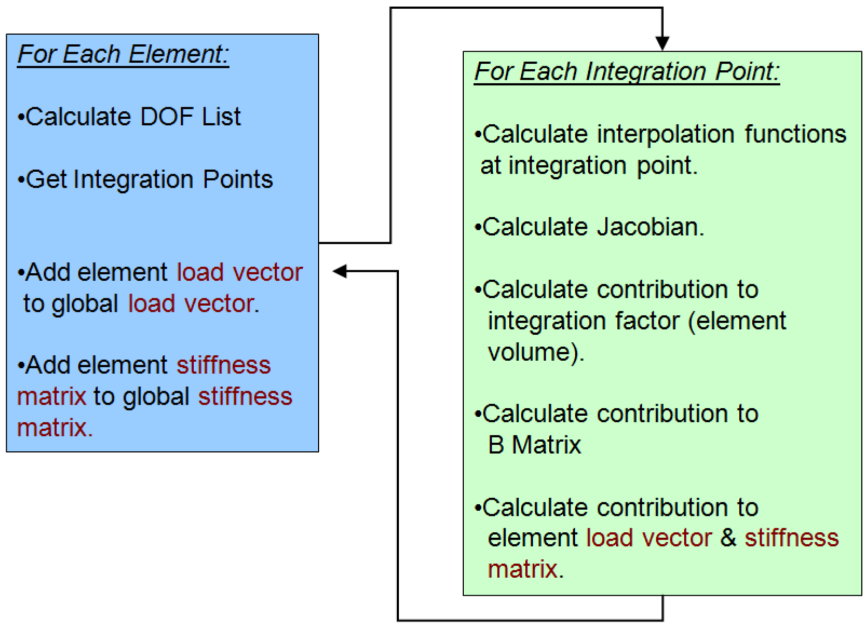

For Each Element:
•Calculate DOF List
•Get Integration Points
•Add element load vector to global load vector.
•Add element stiffness matrix to global stiffness matrix.
For Each Integration Point:
•Calculate interpolation functions at integration point.
•Calculate Jacobian.
•Calculate contribution to integration factor (element volume).
•Calculate contribution to B Matrix
•Calculate contribution to element load vector & stiffness matrix.

Fig. 3 Implementation of elemental calculations and assembly

E. Post-processing

A finite element analysis produces displacement values at the nodes of a mesh. While displacement values are useful for visualizing deformed shapes of a model, many times they are not sufficient. Strains and stresses are desirable to give insight on how a model behaves under certain loading scenarios. Stress and strain values may be compared to allowable values for the material system, thus giving insight into degradation and failure. Stress and strain values are calculated after the analysis through post-processing.

The first step in post-processing for strain and stress is to calculate displacement derivatives. These derivatives will be calculated at the quadrature points of an element since these points have the lowest error due to the approximation introduced by interpolation functions. Displacement derivatives for a particular quadrature point ($\hat{\xi_k}$) are calculated via Eq. (2.35).

$$u_{i,j}(\hat{\xi_k}) = u_i^m N_{m,j}(\hat{\xi_k}) \tag{2.35}$$

After calculating the displacement derivatives at quadrature points, the strains at quadrature points may be easily calculated using kinematic relations. Similarly, the stresses at quadrature points may be calculated using constitutive relations.

The usefulness of quadrature point strains and stresses is limited since these values cannot be visualized directly on the mesh. For this reason, quadrature point values must be extrapolated to the nodes of an element. First consider an interpolating polynomial

$$\phi_i = a + b\xi_i + c\eta_i + d\xi_i\eta_i ... = H_{i\alpha}\Delta_\alpha \tag{2.36}$$

$$G(\xi_i, \eta_i) = \begin{bmatrix} 1 & \xi_i & \eta_i & \xi_i\eta_i & ... \end{bmatrix} \tag{2.37}$$

$$H_i = G(\xi_i, \eta_i) \tag{2.38}$$

$$\Delta = \begin{bmatrix} a & b & c & d & ... \end{bmatrix} \tag{2.39}$$

Let ϕ^N represent a nodal value and ϕ_i^Q represent the quadrature point values. The matrix $H_{i\alpha}$ in Eq. (2.40) is calculated by evaluating G at the local coordinates of the quadrature points. Eq. (2.40) shows the interpolation of quadrature point values. The local coordinates of a node are represented by ξ^N and η^N. Eq. (2.41) shows the interpolation of a nodal value. By relating Eqs. (2.40) and (2.41), a relation between quadrature point values and a nodal value is obtained (Eq. (2.42)). This allows for the extrapolation from quadrature point values to a nodal value. The matrix GH^{-1} is an extrapolation matrix.

$$\phi_i^Q = H_{i\alpha}\Delta_\alpha \tag{2.40}$$

$$\phi^N = G_\alpha(\xi^N, \eta^N)\Delta \tag{2.41}$$

$$\phi^N = G(\xi^N, \eta^N)H^{-1}\phi_i^Q \tag{2.42}$$

CHAPTER III

INTERPOLATION

This chapter will review various methods of interpolation. Specifically, the methods of Lagrangian and Hermite interpolation and B-spline approximation will be discussed. The concept and use of each method will be presented. Also, the advantages and limitations of each method will be discussed.

A. Lagrangian Interpolation

Lagrangian interpolation [37] provides a relatively easy way to interpolate a set of values with any order function. This set of values will be termed "nodal values". The interpolated curve fit will pass exactly through the nodal values. Lagrangian interpolation can be done using a single curve fit of higher order, or it may be done using multiple piecewise curve-fits of lower order. The method of using a single curve fit of higher order will be discussed first.

Lagrangian interpolation works by creating a set of functions used for interpolation. There is a function for every nodal value to be interpolated within a particular domain. The interpolating polynomial is constructed from a set of nodal values and basis functions ($\Psi_j(x)$) by means of Eq. (3.1). Similarly, if the interpolated derivative is desired, Eq. (3.2) uses basis function derivatives $\Psi_j'(x)$. The formula for calculating basis functions is presented in Eq. (3.3). The interpolating polynomial passes through nodal values because the associated basis function will have a value of one and all other basis functions will have a value of zero at nodal coordinate ξ_j (Eq. (3.4)). The degree of the basis functions is defined as d. If a single curve is to interpolate a set of n nodal values then basis functions of order $d = n - 1$ must be used. Degree is related to the order k such that $k = d + 1$.

$$u(x) = \sum u_j \Psi_j(x) \tag{3.1}$$

$$u'(x) = \sum u_j \Psi'_j(x) \tag{3.2}$$

$$\Psi_i(x) = \frac{(x - x_1)(x - x_2)...(x - x_{k+1})}{(x_i - x_1))(x_i - x_2)...(x_i - x_{k+1})} \tag{3.3}$$

$$\Psi_i(\xi_j) = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases} \tag{3.4}$$

See Fig. 4 for an example of Lagrangian interpolation. Since seven nodal values are being interpolated, sextic basis functions ($d = 6$) must be used to interpolate all nodal values with one curve. However, use of higher-order functions can create oscillation in the curve fit. Therefore interpolation between two points may not represent the behavior of the interpolated value in the domain accurately.

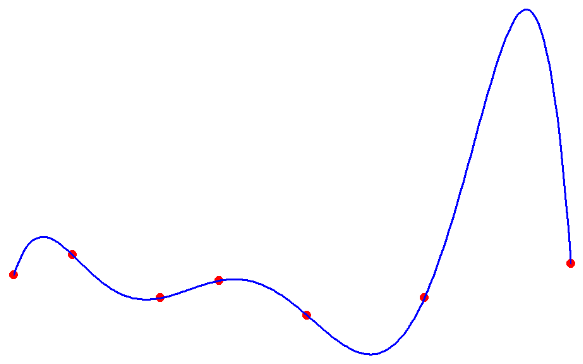


Fig. 4 Lagrangian sextic interpolation of seven nodal values

As stated before, the interpolation may also be done by using lower order basis functions in a piecewise manner. If a piecewise interpolation method is used the domain must be split into segments or elements. To generalize the formulas a local coordinate system will be imposed on the element that ranges from -1 to 1. The physical coordinate system may be mapped to a local coordinate system by using Eq. (3.5). The basis function formulas may be redefined in the local element coordinate system as presented in Eq. (3.6). Fig. 5 illustrates the normalized coordinate system of an arbitrary one-dimensional quadratic element (spanning from x_i to x_{i+2}). Figs. 6 and 7 display the basis functions for a linear and quadratic element respectively.

$$\xi = \frac{2x - (x_i + x_{k+i})}{x_{k+i} - x_i} \tag{3.5}$$

$$\Psi_i(\xi) = \frac{(\xi - \xi_1)(\xi - \xi_2)...(\xi - \xi_{k+1})}{(\xi_i - \xi_1))(\xi_i - \xi_2)...(\xi_i - \xi_{k+1})} \tag{3.6}$$

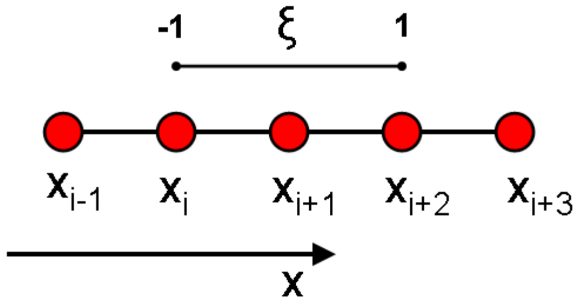


Fig. 5 Nodes and normalized coordinate system of quadratic element (d=2)

An example of piecewise interpolation is presented in Fig. 8. A set of seven nodal values will be interpolated using piecewise quadratic interpolation. Since there are three nodes per quadratic element and elements will share nodes at the boundaries, there will be three elements in this piecewise interpolation. Interpolation between

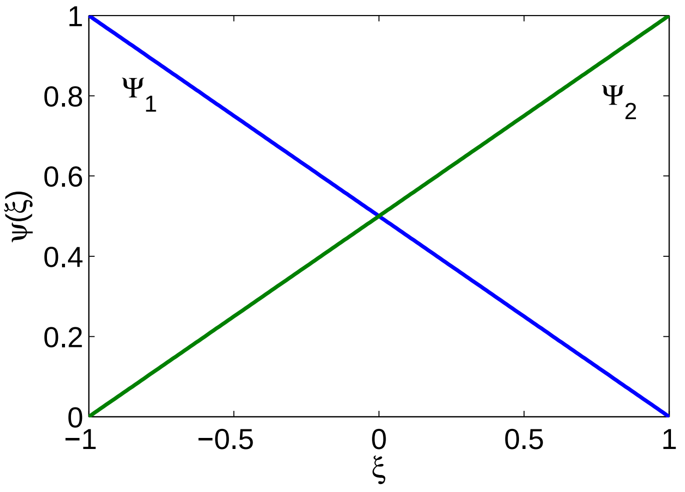


Fig. 6 Basis functions of a linear element

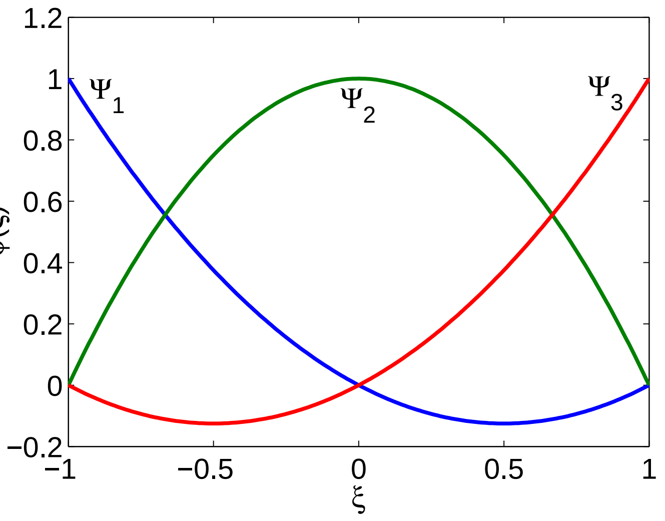


Fig. 7 Basis functions of a quadratic element

nodal values does not oscillate as seen in Fig. 4. However, the curve fit between adjacent elements is not as smooth, and the derivatives will be discontinuous at the nodes of adjacent elements.

The use of a single higher-order function to curve fit a set of nodal values has merit in the fact that the interpolation is smooth and continuity of derivatives at the nodes is maintained. However, limitations arise in the fact that the curve fit may not be accurate due to the oscillation caused by the higher-order function (see Fig. 4). While piecewise interpolation using lower-ordered functions eliminates the oscillation, smoothness and continuity of derivatives is lost. That is, for a node at the interface of quadratic elements there is C^0 continuity(continuity of the value) but not C^1 continuity(continuity of the first derivative).

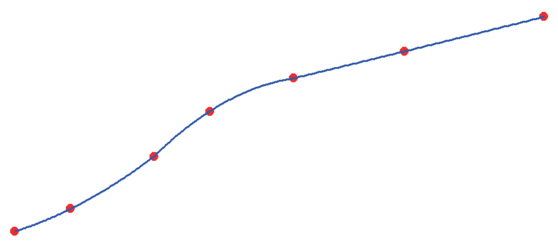


Fig. 8 Lagrangian piecewise quadratic interpolation of seven nodal values

B. Hermite Interpolation

Hermite interpolation [37] offers an improvement over Lagrangian interpolation. C^1 continuity can be maintained at the nodes on the interface of elements. However to construct an interpolating polynomial, nodal derivative values must be known in addition to nodal values. Eq. (3.7) presents the formula for constructing a polynomial using Hermite interpolation. Here u_i and u_i' are the nodal values and nodal derivative values respectively. The term $\Psi_i^0(\xi)$ is the ith basis function corresponding to the ith nodal value and the term $\Psi_i^1(\xi)$ is the ith basis function corresponding to the ith nodal derivative value. The term h_e is the length of a one-dimensional Hermite element. The basis functions used for a cubic Hermite interpolation are displayed in Eq. (3.8) Since four parameters are required (nodal value and nodal derivative value at two nodes) cubic interpolation is applied to a two node element.

$$u(\xi) = \sum u_i \Psi_i^0(\xi) + \sum \frac{h_e}{2} u_i' \Psi_i^1(\xi) \tag{3.7}$$

$$\begin{aligned}
\Psi_1^0(\xi) &= \tfrac{1}{4}(2 - 3\xi + \xi^3) \\
\Psi_2^0(\xi) &= \tfrac{1}{4}(2 + 3\xi - \xi^3) \\
\Psi_1^1(\xi) &= \tfrac{1}{4}(1 - \xi - \xi^2 + \xi^3) \\
\Psi_2^1(\xi) &= \tfrac{1}{4}(-1 - \xi + \xi^2 + \xi^3)
\end{aligned} \tag{3.8}$$

The basis functions for a cubic Hermite element are displayed in Fig. 9. An example of interpolation using cubic Hermite elements is presented in Fig. 10. Note that arbitrary nodal derivative values were used for this particular example. To use Hermite interpolation for curve fitting a data set, the derivatives of nodal values must be known. If Hermite functions are to be used in finite elements the derivatives of nodal values must also be determined as part of the finite element solution.

The capability of Hermite interpolation functions to maintain C^1 continuity comes at a computational cost. A finite element analysis employing Lagrangian functions will only solve for the values (i.e. displacement) at a node. This value is a degree of freedom (DOF). Cubic Hermite interpolation requires values and derivatives at each node and therefore requires more DOFs. Even though a cubic Hermite element has fewer nodes than a quadratic Lagrangian element the increased DOF per node of a Hermite element typically increases the overall DOFs of a finite element analysis. Also, for certain applications the excess continuity at nodes is not desirable. For example , in FEA if two adjacent elements have different material assignments in general stress is not continuous across the element boundary. Therefore, cubic Hermite elements are not well suited for this particular situation.

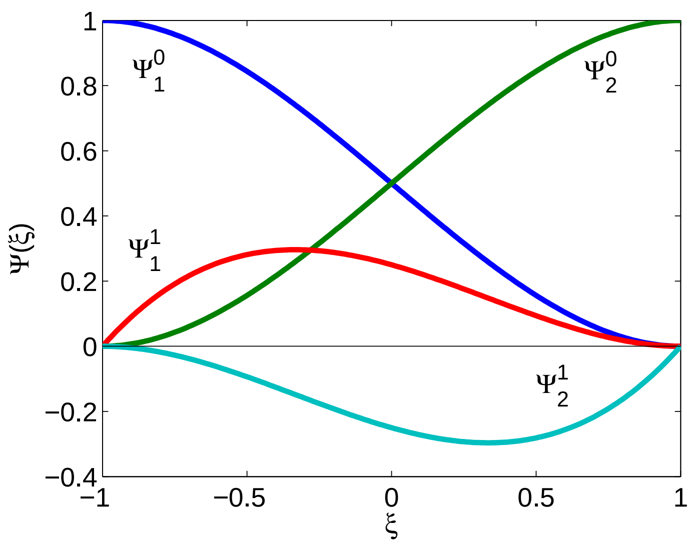


Fig. 9 Basis functions of a cubic Hermite element

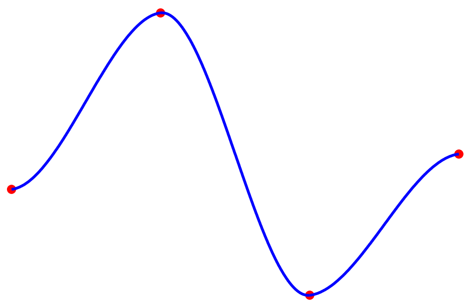


Fig. 10 Cubic Hermite interpolation of four nodal values

C. B-spline Approximation

1. Overview

B-splines [4] are piecewise polynomial curves that typically provide a better curve fit than other interpolation methods. B-splines possess a variation-diminishing property which means that as the order of a B-spline function is increased it does not create oscillation in the entire curve. B-splines also have local support, which means that a portion of the B-spline curve may be modified without affecting the shape of the whole curve. Also, B-splines maintain smoothness and continuity of higher-order derivatives. Again this can be problematic in the case of FEA where continuity across elements with different material groups is undesirable.

It is important to note that a B-spline curve fit typically does not pass through the values being interpolated. Therefore, B-spline functions are not interpolatory, but instead give an approximation of a curve fit. A B-spline curve may be forced to pass through a certain point by changing some parameters but this forcing action will reduce the continuity at the point to C^0 continuity.

Instead of interpolating at nodal values as with Lagrangian interpolation, B-splines approximate the value of a function using "control point" values. The control point values may not have a precise location within the domain (except at the boundaries of the domain). Therefore, they are not nodal variables but rather coefficients that influence the shape of the curve. Eq. (3.9) displays the equation used for constructing a B-spline curve. The polynomial $u(t)$ is constructed by means of control point values (a_i) and basis functions $(N_{i,k}(t))$ of order k. The basis functions are evaluated at value (t) located in a knot coordinate system.

$$u(t) = \sum a_i N_{i,k}(t) \tag{3.9}$$

The basis functions ($N_{i,k}(t)$) for a B-spline curve are defined by a recursive formula. For step basis functions ($k = 1$) Eq. (3.10) is used. For any other order basis function Eq. (3.11) is used. Fig. 11 illustrates the recursive dependencies of the B-spline basis functions. A single cubic B-spline basis function will require the calculation of nine other lower order B-splines. Note that the basis functions are defined in a different coordinate system than the typical normalized coordinate system (-1 to 1) used for Lagrangian and Hermite basis functions. Basis functions are defined in a knot coordinate system (t). Basis functions will be discussed further in the basis function section.

$$N_{i,1}(t) = \begin{cases} 1 & t_i \le t < t_{i+1} \\ 0 & \text{otherwise} \end{cases} \tag{3.10}$$

$$N_{i,k}(t) = \frac{(t - t_i)N_{i,k-1}(t)}{t_{i+k-1} - t_i} + \frac{(t_{i+k} - t)N_{i+1,k-1}(t)}{t_{i+k} - t_{i+1}} \tag{3.11}$$

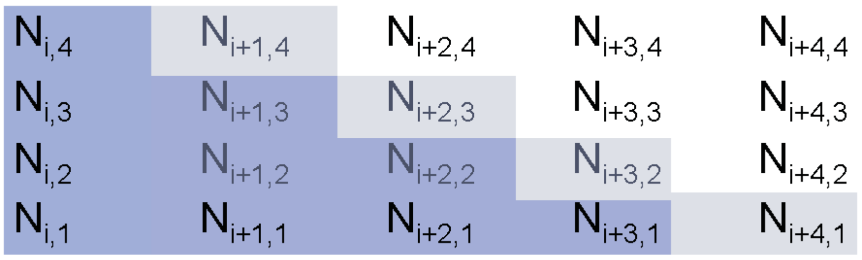

$N_{i,4}$ $N_{i+1,4}$ $N_{i+2,4}$ $N_{i+3,4}$ $N_{i+4,4}$
$N_{i,3}$ $N_{i+1,3}$ $N_{i+2,3}$ $N_{i+3,3}$ $N_{i+4,3}$
$N_{i,2}$ $N_{i+1,2}$ $N_{i+2,2}$ $N_{i+3,2}$ $N_{i+4,2}$
$N_{i,1}$ $N_{i+1,1}$ $N_{i+2,1}$ $N_{i+3,1}$ $N_{i+4,1}$

Fig. 11 Recursive dependency of B-spline basis functions

A knot vector ($\vec{t}$) is a vector of increasing real numbers (knot values) that exist in the knot coordinate system. The knot vector has a very strong influence on the shape of B-spline curves. Knot vectors can be constructed with various spacing of knot values and knot values can be repeated to alter the shape of the curve. More on knot vectors will be explained in the knot vector section.

For B-spline functions the order (k) is defined to be one more than the degree of the basis functions. For a linear function $k = 2$, for a quadratic function $k = 3$, and so on. B-spline curves have the property of maintaining C^{k-2} continuity in general.

Lagrangian interpolation requires that a set of n values be interpolated by a function of degree $n - 1$. B-spline approximation does not have such a strict dependency between number of values/control points and degree. The extra parameter of the knot vector provides greater flexibility. The relation of knot vector length (l), number of control points (n), and order (k) is displayed in Eq. (3.12). This relation subdivides the domain into $l - 1$ regions or knot intervals.

$$l = n - k + 2 \tag{3.12}$$

2. Knot Vector

B-spline curves are defined by a knot vector. This knot vector is a vector of knot values (real numbers in increasing order). The knot vector subdivides a domain into sub-regions or knot intervals similar to the division of a domain into elements for piecewise interpolation. Different types of knot vectors include uniform knot vectors, non-uniform knot vectors, and open knot vectors. Uniform knot vectors have even spacing of knot values whereas non-uniform knot vectors do not. Open knot vectors have $k - 1$ multiplicity of the first and last knot values.

A periodic knot vector is essentially a uniform, non-open knot vector. An example of a periodic knot vector is presented in Eq. (3.13). An example of a non-uniform knot vector is shown in Eq. (3.14) and an example of a open uniform knot vector for a cubic B-spline ($k = 4$) is shown in Eq. (3.15). Open knot vectors may also be non-uniform.

$$\vec{t}_{periodic} = [0, 1, 2, 3, 4, 5, 6] \tag{3.13}$$

$$\vec{t}_{non-uniform} = [0, 0.5, 1.2, 3, 3.7, 4.2, 5.5] \tag{3.14}$$

$$\vec{t}_{open} = [0, 0, 0, 0, 1, 2, 3, 4, 5, 6, 6, 6, 6] \tag{3.15}$$

Repetition of a knot value has the effect of drawing the curve closer to a specific control point. If a knot value is repeated $k-1$ times the B-spline curve will pass through the associated control point. Therefore in the case of an open B-spline (constructed from an open knot vector) the first and last control points will be interpolated. However multiplicity of knot values also decreases the continuity at the associated location of the B-spline curve. If knot values are repeated the function will have continuity of C^s at the associated knot value, where Eq. (3.16) presents the relationship between continuity (C^s), order (k) and multiplicity (m) of knot values. Therefore open B-splines have C^0 continuity at the first and last knot values in the knot vector.

$$s = k - m - 1 \tag{3.16}$$

3. Basis Functions

Different types of B-spline basis functions are dictated by the different types of knot vectors. Fig. 12 displays a single periodic basis function for a cubic B-spline. Note that the basis function spans over k intervals. This span is the local support region for that particular basis function. Modification of a control point will not modify the curve outside of the associated basis function's local support region.

Fig. 13 displays multiple basis functions for a periodic B-spline. Fig. 14 displays the piecewise segments of k basis functions present within a single knot interval. For periodic basis functions all functions are the same, but they are offset in the knot coordinate direction. Eq. (3.17) explains this characteristic.

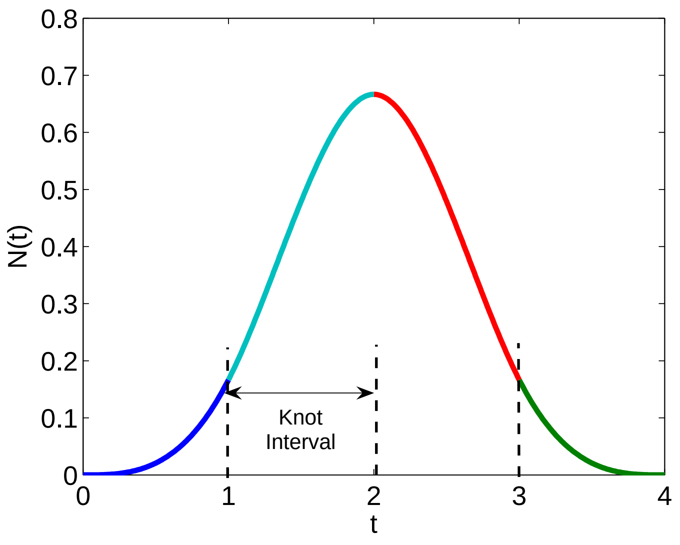


Fig. 12 A single cubic B-spline periodic basis function

$$N_{i,k}(t) = N_{i-1,k}(t-1) = N_{i+1,k}(t+1) \tag{3.17}$$

Fig. 15 displays the basis functions for an open B-spline with a uniform knot vector $\vec{t} = [0,0,0,0,1,2,3,4,4,4,4]$.. The first and last basis functions have a value of one at the first and last knot values respectively. This ensures that the B-spline curve will interpolate the first and last control points. Fig. 16 displays the basis functions for an open B-spline with the knot vector $\vec{t} = [0,0,0,0,1,2.4,3,4,4,4,4]$.

As shown in Fig. 14 there are k basis functions associated with a single knot interval. Therefore, there will also be k control points associated with a single interval. While one-dimensional piecewise Lagrange interpolation will only share a single node

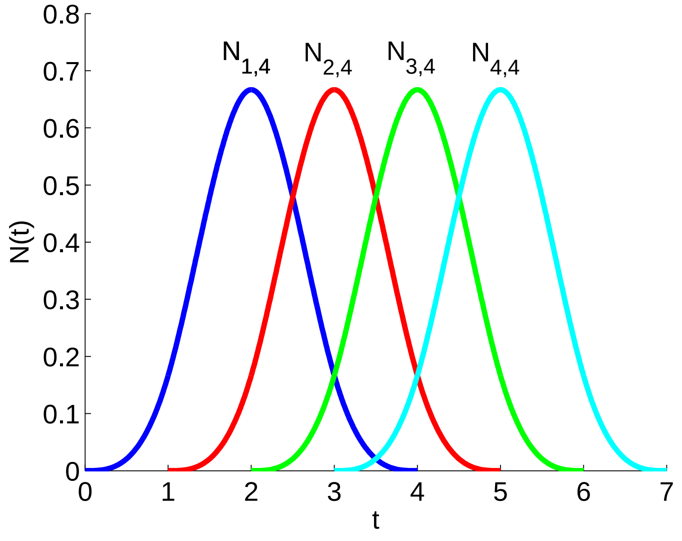


Fig. 13 Cubic B-spline uniform periodic basis functions

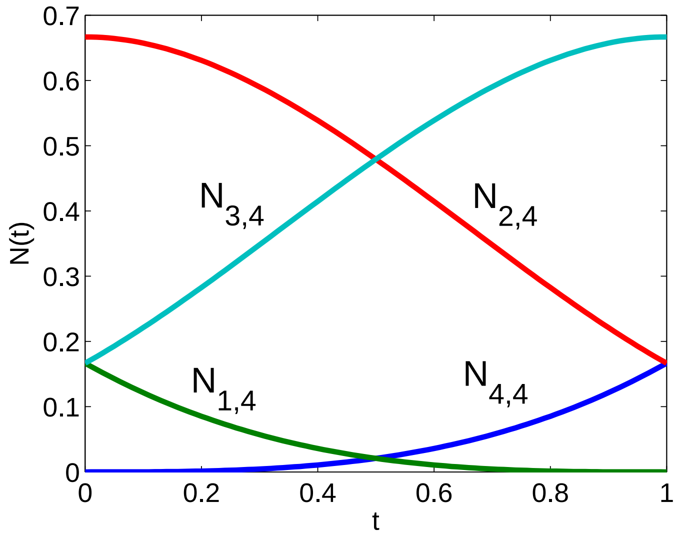


Fig. 14 Cubic B-spline basis function pieces in a single knot interval

between adjacent elements, there is much more overlap in shared control points among knot intervals. This overlap is illustrated in Fig. 17 for a cubic open B-spline with four knot intervals. Using the relations of Eq. (3.12), it can be determined that

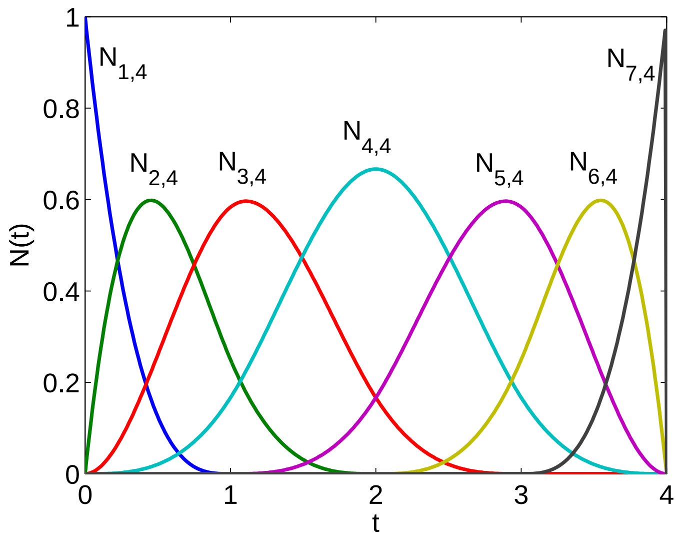


Fig. 15 Cubic B-spline open uniform basis functions

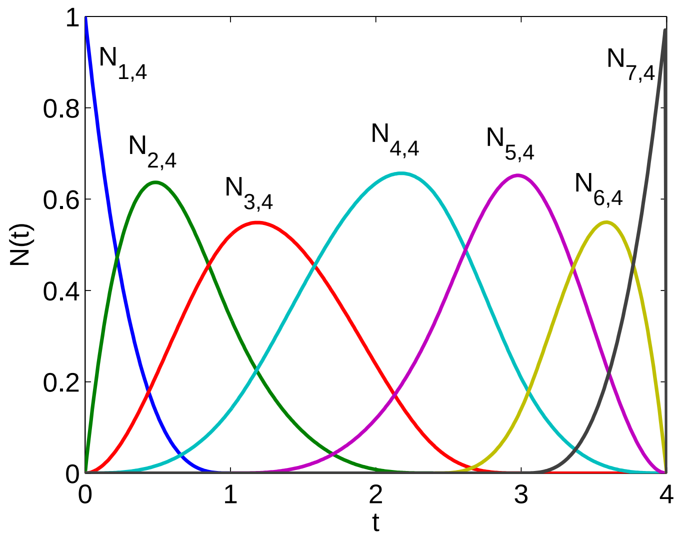


Fig. 16 Cubic B-spline open non-uniform basis functions

there our eight control points associated with this B-spline. The first knot interval $[t_0, t_1]$ is associated with control points a_0, a_1, a_2, a_3, the second knot interval $[t_1, t_2]$ is associated with control point a_1, a_2, a_3, a_4, and so on for other knot intervals.

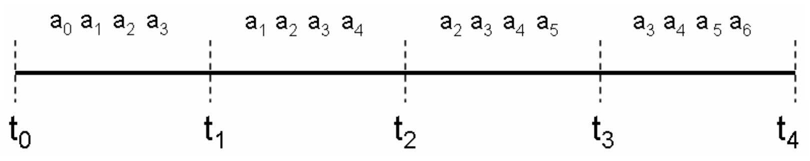


Fig. 17 Association of control points with knot intervals for 1D B-spline

4. Non-recursive Calculation of B-spline Basis Functions

B-spline basis functions may be calculated using a recursive formula. However, if B-spline functions are to be evaluated many times during an analysis, a recursive formula is undesirable. Therefore, non-recursive equations were developed for basis functions of a particular knot interval of an open cubic B-spline with an arbitrary knot vector (see Eqs. (3.18) to (3.21)). These equations require the knot vector, the knot interval in which the basis functions are being calculated (the knot interval "i" corresponds to the interval $[t_i, t_{i+1}]$), and the knot value t at which the basis functions are being evaluated.

$$N_{1,4}(t) = \frac{(t_{i+1} - t)^3}{(t_{i+1} - t_{i-2})(t_{i+1} - t_{i-1})(t_{i+1} - t_i)} \tag{3.18}$$

$$\begin{aligned} N_{2,4}(t) &= \frac{(t-t_{i-2})(t_{i+1}-t)^2}{(t_{i+1}-t_{i-2})(t_{i+1}-t_{i-1})(t_{i+1}-t_i)} \\ &+ \frac{(t_{i+2}-t)(t-t_{i-1})(t_{i+1}-t)}{(t_{i+2}-t_{i-1})(t_{i+1}-t_{i-1})(t_{i+1}-t_i)} \\ &+ \frac{(t_{i+2}-t)^2(t-t_i)}{(t_{i+2}-t_i)(t_{i+1}-t_i)(t_{i+2}-t_{i-1})} \end{aligned} \tag{3.19}$$

$$\begin{aligned} N_{3,4}(t) &= \frac{(t_{i+1}-t)(t-t_{i-1})^2}{(t_{i+2}-t_{i-1})(t_{i+1}-t_{i-1})(t_{i+1}-t_i)} \\ &+ \frac{(t_{i+2}-t)(t-t_{i-1})(t-t_i)}{(t_{i+2}-t_{i-1})(t_{i+2}-t_i)(t_{i+1}-t_i)} \\ &+ \frac{(t-t_i)^2(t_{i+3}-t)}{(t_{i+3}-t_i)(t_{i+2}-t_i)(t_{i+2}-t_i)} \end{aligned} \tag{3.20}$$

$$N_{4,4}(t) = \frac{(t-t_i)^3}{(t_{i+2}-t_i)(t_{i+1}-t_i)(t_{i+3}-t_i)} \tag{3.21}$$

5. Constructing Curves with B-splines

As stated before, B-splines are largely used to construct curves and surfaces for computer aided design (CAD). When B-splines are used for this purpose control points serve only to modify the shape of the curve. There is no interest in whether the control points fall on the curve or the proximity of control points to the curve. Open B-splines may still be utilized to force C^0 continuity between different curves. Fig. 18 displays a B-spline curve created using a collection of control points.

B-splines may also be used to provide a curve fit of data as with Lagrangian or Hermite interpolation. For this application the control points are very similar to nodal values in interpolation. However, values are not interpolated but approximated. Interpolation will occur at knot values that have $k-1$ multiplicity. A curve fit using B-spline approximation is shown in Fig. 19.

Fig. 20 shows two B-spline curves. The curves are approximating two almost

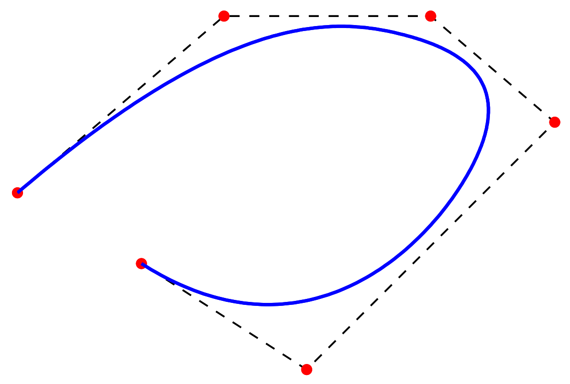


Fig. 18 Cubic B-spline used to create an arbitrary curve

identical sets of control points. However, one set of contains point A whereas the other contains point A'. Notice that the B-spline curves are identical in portions outside of the local support regions. The two curves only have differences in a number of k knot intervals. A change in one point of the set of values would change the shape of the entire curve for non-piecewise Lagrangian or Hermite interpolation.

Previous examples of constructing curves with B-splines have used a given set of control points to produce a B-spline curve. However, sometimes a specific curve geometry is desired and the control points and basis functions must be determined to produce a B-spline curve that represents the desired geometry. Rogers [4] has developed an extensive resource for B-splines that addresses this issue.

Consider a set of points (p_j) at knot values (t_j). These will be the points that the B-spline curve should be forced to pass through. The B-spline curve will be produced by control points values (a_i) and basis functions $(N_{i,k}(t))$ of order k.

$$p_j = a_i N_{i,k}(t_j) \tag{3.22}$$

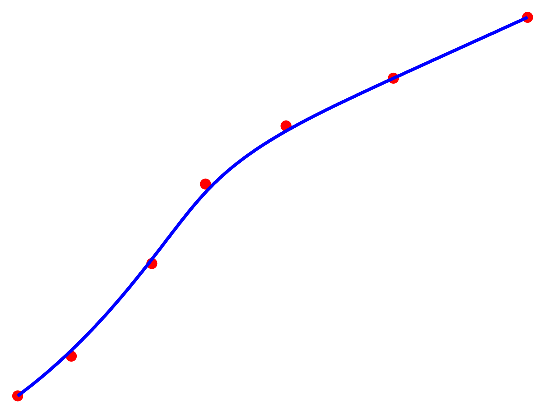


Fig. 19   Cubic B-spline used to curve fit seven values

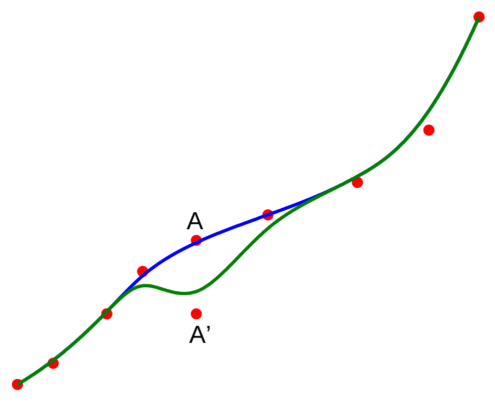


Fig. 20   Local support of a cubic B-spline curve

Eqs. (3.22) to (3.24) represent a system of equations which may be solved to determine the appropriate control point values (a_i) for the desired geometry (p_j). In general, the number of control points ($n+1$) must satisfy the following criteria: $k \leq n+1 \leq j$ where j is the number of points provided for the desired geometry. Using an arbitrary knot vector and order for the B-spline curve, the basis functions may be evaluated at the knot values t_j. Upon calculation of the basis functions, and given the desired geometry (p_j), the system of equations may be solved (Eq. (3.25)) for the necessary control point values (a_i).

$$[P] = [N][A] \tag{3.23}$$

$$\begin{aligned} P^T &= [p_1, p_2, p_3, ...p_{n+1}] \\ A^T &= [a_1, a_2, a_3, ...a_{n+1}] \end{aligned} \tag{3.24}$$

$$N = \begin{bmatrix} N_{1,k}(t_1) & ... & ... & N_{n+1,k}(t_1) \\ \vdots & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ N_{1,k}(t_{n+1}) & ... & ... & N_{n+1,k}(t_{n+1}) \end{bmatrix}$$

$$[A] = [N]^{-1}[P] \tag{3.25}$$

Other possibilities for constructing a specific geometry include using a least-squares approach which seeks to minimize the error between the desired geometry and the geometry produced by B-spline curves. If one is seeking to model an analytical function with a B-spline curve, then error may be calculated as the difference between the B-spline and analytical functions. This error would then be minimized by changing control point values or basis functions until a specified tolerance is reached.

D. Interpolation in Higher Dimensions

The concepts of interpolation developed in the previous sections are for one-dimension. However, generally the ability to interpolate in higher dimensions is required. There are different ways to interpolate in higher dimensions. Two of these ways are by using tensor product of one-dimensional interpolation functions and "serendipity" elements.

For simplicity two-dimensional interpolation will be discussed but the concepts may be extended to three-dimensional space. Only quadrilateral elements will be discussed. An important property of multi-dimensional interpolation is that the basis function associated with a node must have a value of one when evaluated at the coordinates of the node and all other basis functions must have a value of zero when evaluated at the coordinates of the node.

First, two-dimensional interpolation by using a tensor product of one-dimensional basis functions will be discussed. Let $n_i(\xi)$ represent a one dimensional basis function. If the normalized coordinate system has coordinates of (ξ, η) then the two-dimensional tensor product is expressed in Eq. (3.26). If basis function derivatives are necessary they may also be calculated via a tensor product (Eqs. (3.27) and (3.28)). Fig. 21 shows an element using a tensor product of quadratic basis functions. The principal is the same for using a tensor-product of one-dimensional basis functions in three dimensions.

$$N_{ij}(\xi, \eta) = n_i(\xi) n_j(\eta) \tag{3.26}$$

$$\left(\frac{\partial N_{ij}}{\partial \xi} \right) = \frac{dn_i}{d\xi} n_j(\eta) \tag{3.27}$$

$$\left(\frac{\partial N_{ij}}{\partial \eta} \right) = n_i(\xi) \frac{dn_j}{d\eta} \tag{3.28}$$

The next two-dimensional interpolation method uses "serendipity elements".

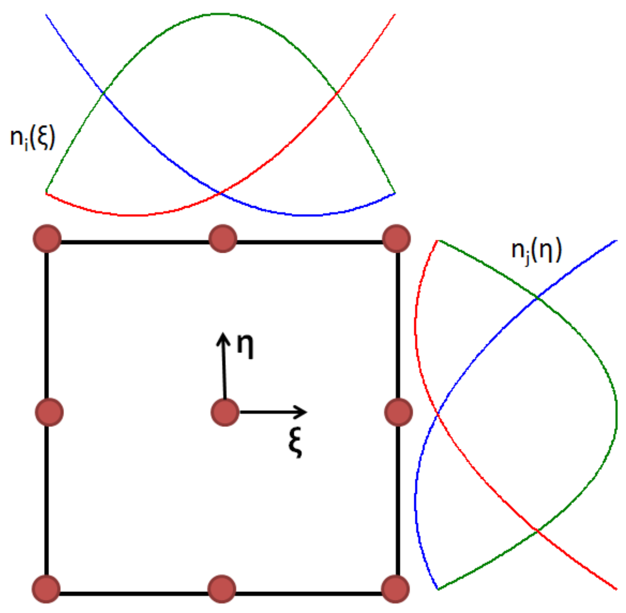


Fig. 21 Two-dimensional tensor product of quadratic basis function

These elements contain no interior nodes. For example, the element illustrated in Fig. 21 would be a serendipity element if the interior node was removed. In fact, very little accuracy is gained by placing this node at the interior, and it is usually removed in favor of reducing the number of nodes per element [37]. This reduction in nodes per element could reduce the overall number of nodes in a finite element analysis considerably. Construction of the basis functions for serendipity elements is more involved than that of the tensor product method. However, Becker et al. have documented the process for a two-dimensional quadrilateral element [37]. Three-dimensional serendipity elements without interior nodes also exist and are commonly used due to their reduction in number of nodes per element.

E. B-spline Approximation in Higher Dimensions

There are two types of elements or domains in B-spline approximation. The first is a “cluster”. The one-dimensional basis functions for a knot vector will span the edges of a cluster. A cluster may be thought of as having two coordinate systems: a global coordinate system (x_i) and a knot coordinate system (t_i). The choice of knot vectors will sub-divide the cluster into smaller domains. These sub-domains are termed “patches”. See Fig. 22 for a cluster that has been subdivided into patches. Also note the one-dimensional basis functions spanning the entire edge of the cluster.

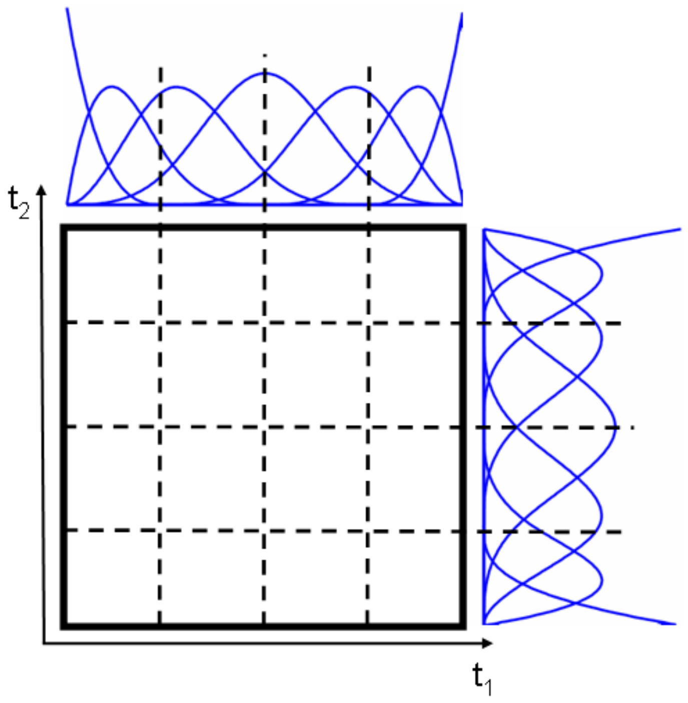


Fig. 22 B-spline basis functions defined over edge of a cluster with knot intervals sub-dividing cluster into patches (knot coordinate system)

Although multiple basis functions span the edge of a cluster, a patch will only be associated with a certain number of basis functions. In one-dimension a single knot interval is associated with k basis functions (4 functions for the case of cubic B-splines). Once the associated one-dimensional basis functions for each dimension a particular patch are selected a higher dimension B-spline approximation for that patch may be obtained by means of a tensor product. This tensor product will produce two-dimensional B-spline basis functions used for approximation in two-dimensions. Therefore, in two-dimensions a patch will be associated with 16 two-dimensional basis functions and 16 control points.

It is difficult to give a control point a physical location since it may be shared among multiple patches. One cannot simply give a control point a coordinate within the domain of an associated patch since multiple patches could share the same control point. Therefore, control point coordinates are arbitrary except on a cluster boundary. For clarity, control points should be given a coordinate within or on the edge of the cluster they are local to. Fig. 23 shows a field of control points resulting from the tensor product of basis functions spanning the edges of a cluster. Note that the number of control points along the edge of the cluster are related to the order of the basis functions and the number of patches or knot intervals on the edge by Eq. (3.12).

F. Summary

B-spline approximation is indeed a satisfactory alternative to the use of Lagrangian or Hermite interpolation functions. Many of the problems of Lagrangian interpolation, including the oscillation of a non-piecewise curve fit and lack of local support for non-piecewise polynomials can be resolved by using B-splines. Hermite interpolation provides an improvement over Lagrangian interpolation with respect to smoothness and continuity of derivatives. However, use of Hermite interpolation in

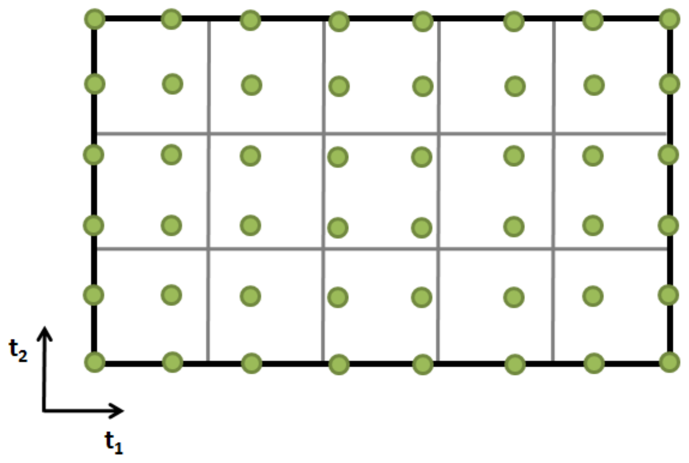


Fig. 23   Field of control points in a two-dimensional cluster

FEM requires solving for nodal values as well as nodal derivatives. Furthermore, Hermite interpolation may increase continuity unacceptably such as enforcing continuity of stress across elements with different materials.

Some of these interpolation methods may be better suited for certain applications than others. The order of the partial differential equation used in FEA will have certain continuity requirements and limitations. In finite element analysis quadratic Lagrangian interpolation is a convenient method that yields sufficient accuracy. However, discontinuities do exist in the derivatives between elements. While this is beneficial in the case of adjacent element with different materials, smoothness could be desired among adjacent elements of the same material. Use of Hermite interpolation functions in finite element analysis would offer and improvement in the solution and solution derivatives but at a certain computational expense. Furthermore, cubic Hermite elements may provide undesirable excessive continuity in certain situations.

B-splines may offer the best performance, a good approximation for the computational cost. The use of open B-splines in clusters will allow for smoothness and continuity of derivatives on the interior of a cluster while maintaining C^0 continuity at the cluster boundaries. This way clusters may be associated with different material groups while maintaining discontinuity of stress across two different materials. The ability to easily adjust the continuity of B-splines through knot repetition is strong motivation for the use of B-splines in FEA. With this in mind, B-spline approximation might offer a better solution due to its flexibility and smoothness.

The use of B-spline approximation in finite element analysis has the possibility to significantly reduce the number of degrees of freedom in an analysis. Piecewise Lagrangian interpolation creates a need for a large number of nodes and elements. However, the use of B-splines allows a domain that was once modeled as many elements to be modeled as a “super element” or “cluster”. This cluster will contain many smaller regions (defined by knot intervals) known as “patches” that are analogous to a conventional element. However, patches share many more degrees of freedom with other patches in the same cluster than conventional elements. This sharing of many degrees of freedom among patches can significantly reduce the number of degrees of freedom in an analysis. The concept of implementing B-splines in a finite element analysis will be discussed in much more detail in further chapters.

CHAPTER IV

IMPLEMENTATION OF THE B-SPLINE FINITE ELEMENT METHOD

This chapter will present the implementation of the B-spline finite element method into an existing finite element framework. Chapter II defined the basic formulation for an elastic finite element method. Chapter III explained the use of B-spline approximation in higher dimensions. Special considerations required by the implementation will be explained. Explanation of the implementation will be segmented into pre-processing, analysis, and post-processing.

A. Pre-processing in the B-spline Finite Element Method

Mesh generation for B-spline finite elements involves the same type of "super-element" mesh as in conventional finite elements. This mesh is termed the "cluster mesh" (see Fig. 24). Knot vector data is also specified for each edge of a cluster. Two additional "meshes" are created based on the cluster mesh and knot vector data. These meshes are the "control point mesh" and the "patch mesh". The patch mesh (see Fig. 25) is analogous to a conventional finite element mesh, however the typical elements are known as patches. The control point mesh (see Fig. 26) has little in common with a typical finite element mesh. The control point mesh is basically a listing of control points associated with a cluster. These control points are the points at which the solution is obtained for the resulting system of equations in a finite element analysis. This solution must be extrapolated to the nodes of the patch mesh during post-processing to obtain displacements.

A knot vector is specified along each edge of a cluster. This knot vector is the same vector that is used for construction of the B-spline function along a cluster edge. Knot vectors subdivide the cluster into patches (see Fig. 27). Use of non-uniform knot

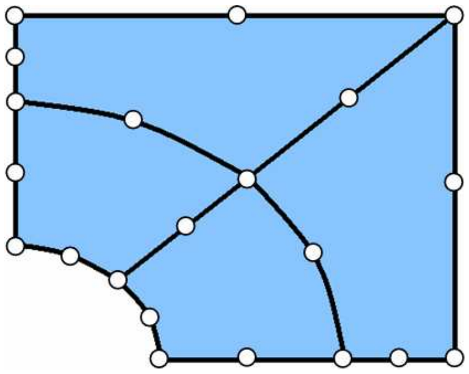


Fig. 24 Example of a two-dimensional cluster mesh

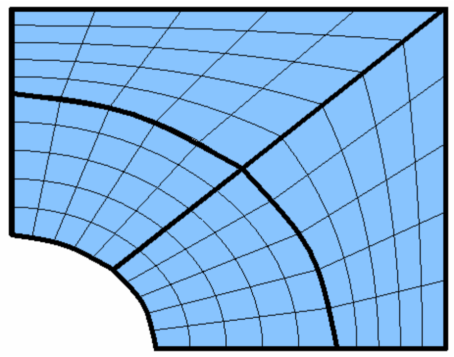


Fig. 25 Example of a two-dimensional patch mesh

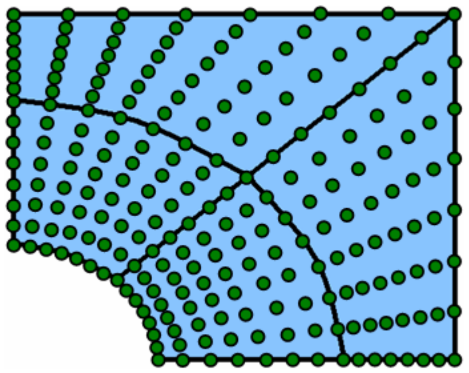


Fig. 26 Example of a two-dimensional control point mesh

vectors will yield non-uniform mesh refinement. A patch mesh is created by meshing the patches of each cluster. After the meshing of each cluster, duplicate nodes along cluster boundaries are removed. The patch mesh is very useful in visualization of data and obtaining information about the a patch geometry during elemental calculations.

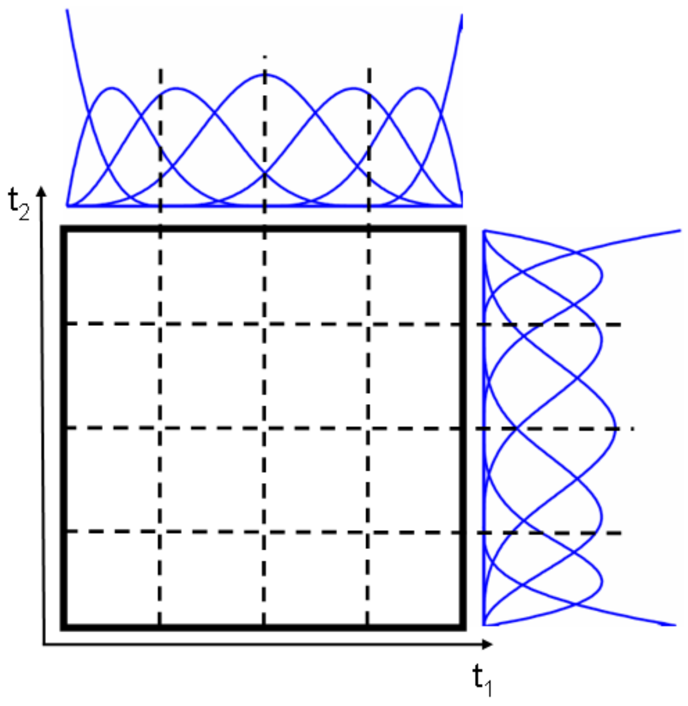


Fig. 27 B-spline basis functions defined over two-dimensional cluster and patch domains.

The control point mesh is harder to visualize. Note that control points are not necessarily associated with a specific physical location even if the control point falls on the boundary of a cluster. For example in a two-dimensional cluster, if the control point is on the edge of the cluster one coordinate is fixed and the other coordinate is arbitrary. Furthermore, if a control point is on the corner of a cluster, both coordinates are fixed. While it is true that control points do not always have a definite physical location, it is convenient to give each control point a unique location for mesh generation purposes. This way mesh generation utilities may be utilized to remove duplicate nodes along cluster boundaries so clusters may share control points on a boundary and the control point mesh can be “tied” together. Therefore, during generation of the control point mesh, all control points are given a physical location. A scheme was developed that would ensure the correct control points were located on cluster boundaries and no control points on the interior of the cluster would have the same coordinate. A grid of control points is generated for each cluster and the cluster’s connectivity is a simple listing of control points that are local to that cluster. Therefore, the control point mesh has the same number of elements as the cluster mesh. After creating a grid of control points for each cluster and specifying the connectivity listing, duplicate control points along shared cluster boundaries are eliminated.

Since open cubic B-splines functions are being used, the functions have C^0 continuity at cluster boundaries. Using relations defined in the chapter III (Eq. (3.12)), one may determine the number of control points along a dimension of a cluster given the number of knot intervals or patches along that dimension. For an open cubic B-spline, the number of control points is 3 greater than the number of knot intervals. For example, the patch mesh in Fig. 25 has 5 patches along each edge of a cluster. Therefore, the control point mesh in Fig. 26 has 8 control points along each edge of a cluster.

In conventional finite elements, material assignments may be made to every element in a mesh. In B-spline finite elements, material assignments are only made to clusters. Patches local to a cluster automatically assume the material assignment made to the parent cluster.

In conventional finite element methods loads, constraints, and multi-point constraints (MPCs) may be applied to a model by specifying the load or constraint value for a particular node number, coordinate, or plane. The same is true for B-spline finite elements. However, constraints and loads are applied to the control points. It is most logical to apply loads or constraints to the boundary of a cluster since control points have a definite physical location on those boundaries. Therefore, it is important that control points on the boundary of a cluster be located on the physical boundary of a cluster.

Distributed loads are slightly more involved for a B-spline finite element analysis. Overall the methodology is the same as that for conventional finite elements. However, the interpolation functions being utilized are B-spline basis functions and there is more overlap in degrees of freedom among patches (see Fig. 28). The equivalent control point loads are calculated using the same methodology as conventional finite elements (Eq. (2.23)).

A pre-processing utility has been developed for the B-spline finite element method. The utility requires input of a cluster mesh, the knot vectors of clusters, and basic information for distributed forces. The utility outputs the control point mesh, patch mesh, and equivalent control point loads with the associated control point numbers.

In conventional finite elements the integration regions or elements are related to nodes by a mesh. However, in B-spline finite elements the patches serve as the integration regions, and control points must be related to a patch. This poses a dilemma since patch information is stored in the patch mesh and control point information is stored in the control point mesh. Therefore, information between two meshes must be shared.

Furthermore, to evaluate the B-spline basis functions the correct basis functions

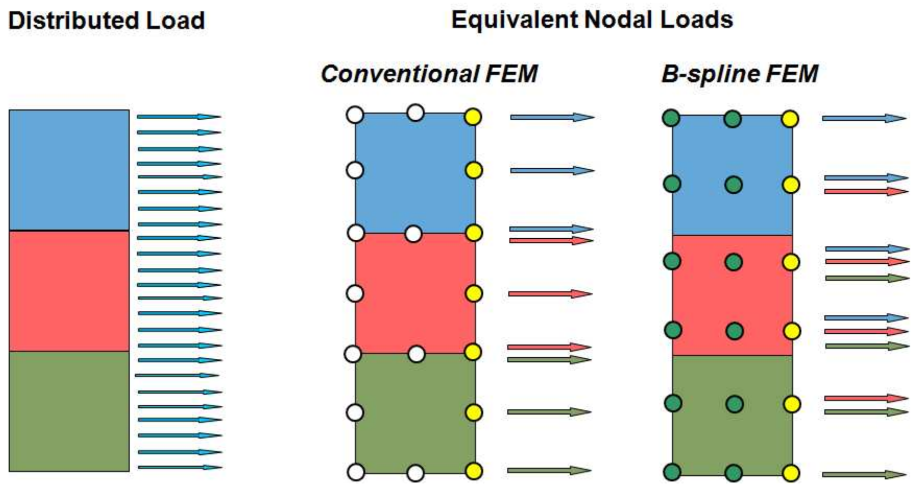


Fig. 28 Equivalent nodal loads of distributed force applied to conventional finite elements and B-spline FEM patches

must be selected for a particular patch. Once the correct piecewise functions are chosen for each dimension the tensor product will produce B-spline basis functions for higher dimensions. Therefore, there must be an association between the patch mesh and knot intervals.

The control point mesh is created by the pre-processing utility. Therefore, a local control point numbering with a specific pattern may be specified within the connectivity of control points in a cluster. This pattern allows for particular information to be extracted. A local numbering is also assumed for patches within a cluster. This local patch numbering is generated from the number of knot intervals on each edge of a cluster. Figs. 29 and 30 show the pattern of local numbering for a two-dimensional patch and control point mesh respectively. Numbering for three-dimensional meshes is handled in a similar manner.

The particular knot intervals associated with a patch may be extracted by using knowledge of a local patch number, and the number of knot intervals or patches along

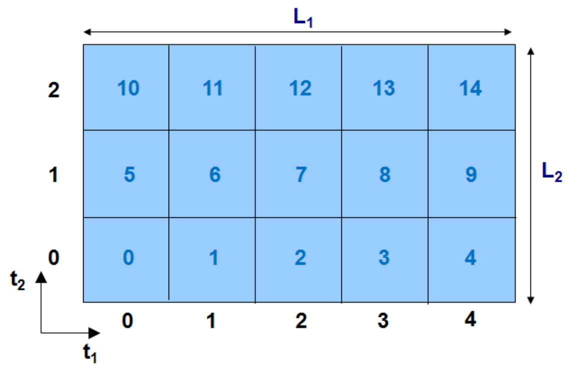


Fig. 29 Local numbering of patches within a cluster

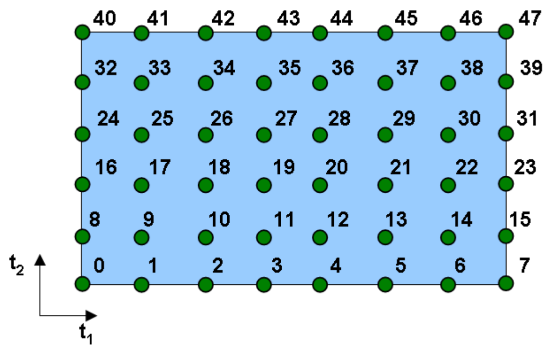


Fig. 30 Local numbering of control points within a cluster

each edge of a cluster (L_1 and L_2). This also requires knowledge of which cluster a patch belongs to. However, association of clusters and patches is easily handled due to a consistent pattern of patch numbering in the pre-processing utility. Determination of knot intervals associated with a patch is done via Eqs. (4.1) and (4.2).

$$t_2\,interval = floor\Big(\frac{local\ patch\ \#}{L_2}\Big) \tag{4.1}$$

$$t_1\,interval = floor\Big(local\ patch\ \# - L_1(t_2\,interval)\Big) \tag{4.2}$$

If a local control point numbering within a patch is utilized (see Fig. 30), association of control points with a patch is fairly straightforward. Eq. (4.3) determines the local control point numbers associated with a patch. Here $nCP1$ is the number of control points in the t_1 direction, $I1$ and $I2$ are the interval numbers of a patch for the t_1 and t_2 directions respectively, and the indices i and j range from 0 to 3 as in the coded implementation. The number of control points per interval ($numCPPerInterval$) must also be known, which is 4 for a cubic B-spline. These local control point numbers may then be referenced to the connectivity of a cluster in the control point mesh to obtain global control point numbers associated with a patch.

$$\begin{aligned} localCPList_{\alpha} &= I1 + j + nCP1(i + I2) \\ \alpha &= i + (numCPPerInterval)j \\ i, j &= 0, 1, 2, 3 \end{aligned} \tag{4.3}$$

In a similar manner the knot intervals associated with a three-dimensional patch may be calculated by Eqs. 4.4 through 4.6. Furthermore, the local numbering of control points associated with a three-dimensional patch may be calculated using Eq. (4.7).

$$interval\, t_3 = floor\Big(\frac{local\, patch\, \#}{L_1 L_2}\Big) \tag{4.4}$$

$$interval\, t_2 = floor\Big(\frac{local\, patch\, \# - (interval\, t_3)L_1 L_2}{L_1}\Big) \tag{4.5}$$

$$interval\, t_1 = floor(local\, patch\, \# - (interval\, t_3)L_1 L_2 - (interval\, t_2)L_2) \tag{4.6}$$

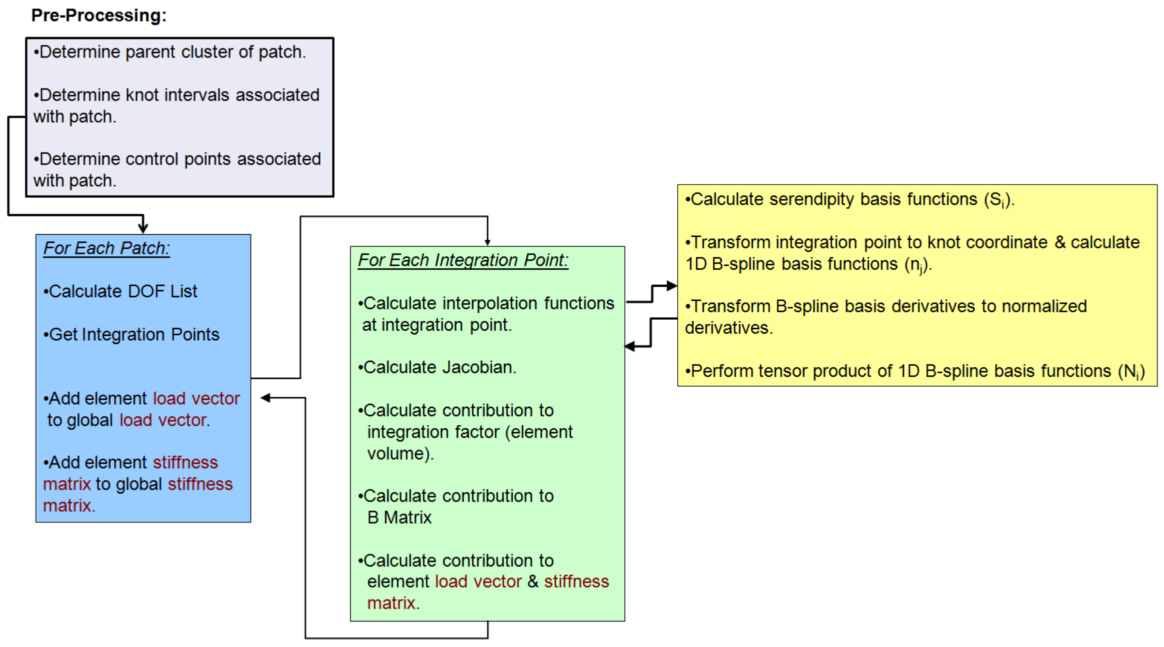
$$\begin{aligned} localCPList_{\alpha} &= I1 + k + nCP1(j + I2) + (i + I3)(nCP1)(nCP2) \\ \alpha &= i + (numCPPerInterval)j + (numCPPerInterval)^2 k \\ i, j, k &= 0, 1, 2, 3 \end{aligned} \tag{4.7}$$

B. Analysis in a B-spline Finite Element Method

Many aspects of analysis in the B-spline finite element method are very similar if not identical to a conventional finite element method. However, there are some differences which must be implemented. Through the use of object oriented design, existing functionality of a finite element framework may be inherited and implementations necessary for B-splines may be added. This results in an implementation in the existing framework which is leaner and more manageable than creating a separate framework for B-spline FEM. The basic structure of the B-spline FEM analysis implementation is presented in Fig. 31. Comparisons of Figs. 3 and 31 shows there is much in common between the B-spline implementation and the conventional finite element framework. Overall, the conventional framework remained in tact with very small or no modifications at all. Special pre-processing steps were implemented for B-spline FEM as well as some very localized functions that dealt with the calculations of the B-spline basis functions and the use of a knot coordinate system.

A typical finite element analysis employs the use of two coordinate systems. That is a global coordinate system and a normalized coordinate system for a particular element. The B-spline finite element method utilizes both of these coordinate systems and an additional coordinate system (the knot coordinate system). Fig. 32 displays the three coordinate systems involved in a B-spline finite element analysis (for a one-dimensional case). Note that the transformation between normalized and knot coordinate systems is a simple linear transformation.

A critical difference between B-spline FEM and conventional FEM is the use of a a sub-parametric formulation. In conventional finite elements, the shape functions



Pre-Processing:
•Determine parent cluster of patch.
•Determine knot intervals associated with patch.
•Determine control points associated with patch.
For Each Patch:
•Calculate DOF List
•Get Integration Points
•Add element load vector to global load vector.
•Add element stiffness matrix to global stiffness matrix.
For Each Integration Point:
•Calculate interpolation functions at integration point.
•Calculate Jacobian.
•Calculate contribution to integration factor (element volume).
•Calculate contribution to B Matrix
•Calculate contribution to element load vector & stiffness matrix.
•Calculate serendipity basis functions (S_i).
•Transform integration point to knot coordinate & calculate 1D B-spline basis functions (n_j).
•Transform B-spline basis derivatives to normalized derivatives.
•Perform tensor product of 1D B-spline basis functions (N_i)

Fig. 31 Structure of analysis for B-spline FEM Implementation

used to describe element geometry are often the same functions use to approximate the solution or displacements. This is known as an iso-parametric formulation. However, in the current implementation B-spline functions are not utilized to describe element geometry. Quadratic serendipity element shape functions (S_i) are used to describe geometry. These serendipity elements are of lower order than the cubic B-splines basis functions (N_i) used to approximate the solution.

Through pre-processing the patches have been associated with control points. Therefore, degree of freedom lists may be generated for a particular patch. Such a list is crucial for assembly. Pre-processing in B-spline finite elements also associates knot intervals of a cluster with a patch. The knot intervals are required to use Eqs. (3.18) through (3.21).

In conventional finite elements basis functions are evaluated at integration points

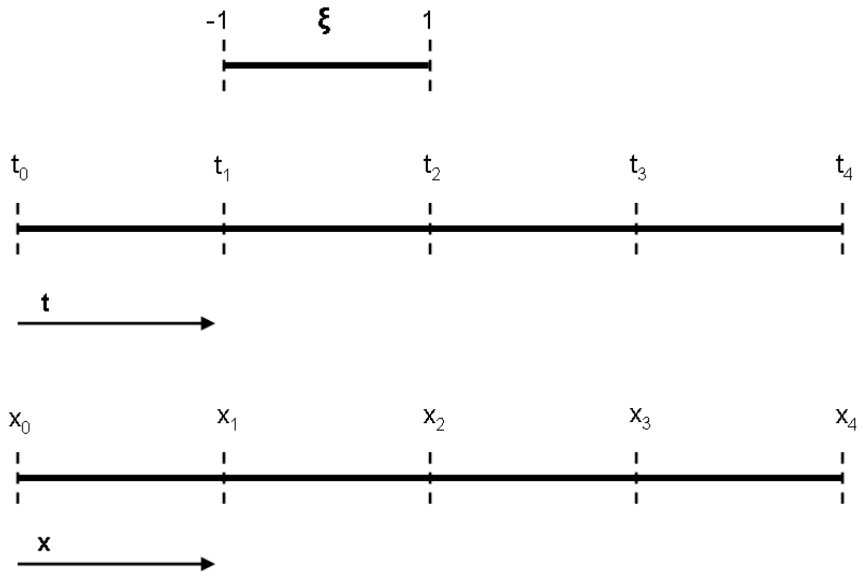


Fig. 32 Three coordinate systems of a B-spline finite element analysis (from top to bottom: normalized, knot, and physical/global coordinate system)

in the normalized coordinate system. However, B-spline basis functions are evaluated at a knot value. Therefore, the integration point in the normalized coordinate system must be transformed to the knot coordinate system. This coordinate transformation from a normalized coordinate (ξ) to a knot coordinate (t) is illustrated in Fig. 33 and implemented using Eq. (4.8). This transformation also requires the knot interval number (i) and knot vector values (t_i), which are available after pre-processing.

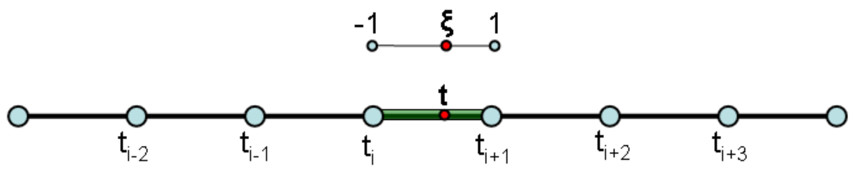


Fig. 33 Definition of knot intervals for generic 1D B-spline basis function equations

$$t = \frac{t_i + t_{i+1} + \xi(t_{i+1} - t_i)}{2} \tag{4.8}$$

After determining the knot intervals associated with a patch and mapping the integration point to the knot coordinate system, one-dimensional B-spline basis functions can be evaluated for each edge of a patch. A tensor product forms the basis function and basis function derivatives (as shown in Eq. (3.22) through (3.24)) used in the finite element analysis. It is convenient to think of the tensor product basis functions as a two-dimensional array (N_{ij}). In a coded implementation it is more efficient to store as a one-dimensional array (N_m). Therefore, the tensor product will be stored in a one-dimensional array with the order of function values corresponding to the order of control points in the list associated with a patch. The same type of storage occurs for a three-dimensional tensor product.

Since a conventional finite element framework assumes basis functions derivatives with respect to a normalized coordinate system it is advantageous to transform the B-spline basis function derivatives with respect to the normalized coordinate system. This step restricts the use of a knot coordinate system to a very localized part of the implementation. Therefore, it allows for more of the existing finite element framework to be utilized as well as bridge the gap between the two methods. The transformation of basis function derivatives is done via Eq. (4.9). This transformation is applied to the one-dimensional B-spline basis functions before the tensor product.

$$\frac{dn_j}{d\xi} = \frac{dn_j}{dt}\frac{dt}{d\xi} = \frac{dn_j}{dt}\left(\frac{t_{i+1} - t_i}{2}\right) \tag{4.9}$$

Calculation of the Jacobian uses the serendipity element shape functions (S_i) that define patch geometry and patch node coordinates from the patch mesh. Therefore, the process is similar to Eq. (2.27) but the term (N_i) is replaced with (S_i). The B-spline basis function derivatives must be transformed to be with respect to the global coordinate system using the inverse Jacobian. This is done in an identical manner as Eq. (2.28).

Upon calculation of global B-spline basis function derivatives, the strain-displacement matrix may be calculated. Since the constitutive matrix is unaffected by the B-spline implementation the stiffness matrix of a patch may also be calculated. The assembly process for B-spline FEM is the same as in conventional FEM. However, there is much more overlap between degrees of freedom among patches. See Fig. 34 for an example of the associated of control points among two-dimensional patches.

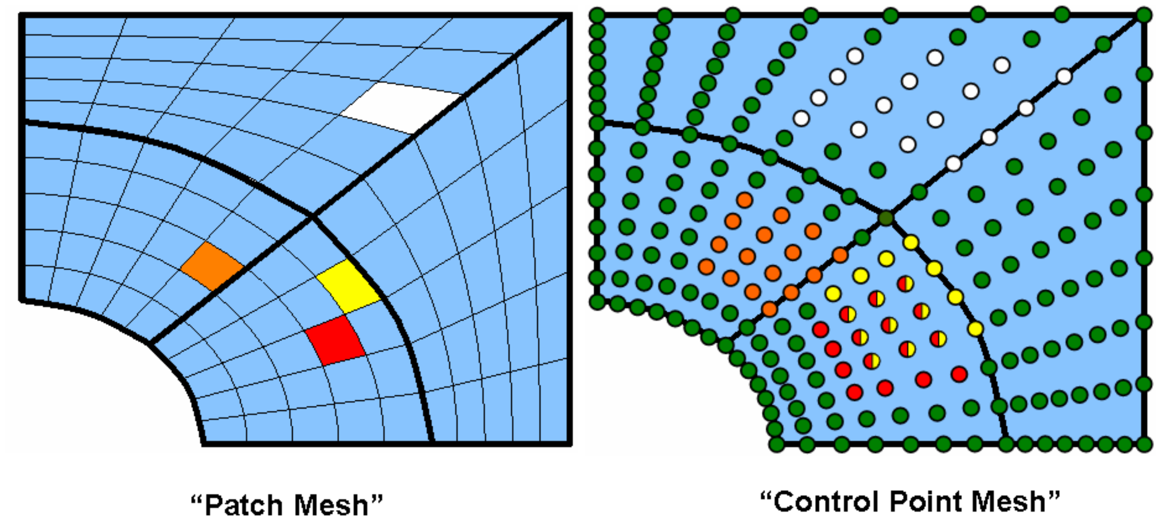


Fig. 34 Association of patches and control points for a 2D model

C. Post-processing in a B-spline Finite Element Method

The B-spline finite element method requires a unique post-processing step. B-spline FEM produces values at control points. These values are not displacements since control points do not have a precise physical location. Instead control point values and B-spline basis functions approximate the displacement field within a cluster. Therefore, the values of the control points associated with a particular patch must be interpolated at that patch's nodes to obtain nodal displacements that can be visualized on the patch mesh. This interpolation is done via Eq. (4.10). Let $\bar{u}$ be the displacement at a patch node, u be the solution at a control point, m is the index of a patch node, and j is the index of the control point and associated B-spline basis function.

$$(\bar{u}_\alpha)_m = (u_\alpha)_j N_j(\xi_m, \eta_m, \zeta_m) \tag{4.10}$$

Post-processing of strains and stresses is almost identical to the conventional finite element method. However, the global derivatives of B-spline basis functions and solution values at control points are utilized to calculate displacement derivatives at quadrature points. Quadrature point strains and stresses are calculated in the same manner as conventional finite elements. Extrapolation from quadrature points to nodes is also handled in the same manner described in chapter II.

CHAPTER V

RESULTS

This chapter will present a comparison of results for a two-dimensional plane strain elasticity analysis using both B-spline FEM and conventional FEM. A demonstration of three-dimensional analysis capability using B-spline FEM will also be presented.

A. Two-dimensional Analysis of Square Array of Fibers Configuration

This section will discuss the results of a two-dimensional plane strain analysis using B-spline FEM. Furthermore, results will also be compared to standard FEM. The particular configuration is a unit cell of a square array of circular fibers. The configuration of the model will be described including geometry, boundary conditions, and material properties. An error analysis tool developed to compare two meshes of a configuration will also be explained. Finally, a convergence study will be conducted for standard and B-spline FEM.

1. Configuration

The configuration is a unit cell of a square array of circular fibers (see Fig. 35). The configuration has a fiber volume fraction of 0.636. By exploiting symmetry, this model may be reduced to a 1/4 unit cell (see Fig. 36). The boundary conditions are also depicted in Fig. 36. Plane strain conditions are utilized. The specified displacement $\bar{u_1}$ is 0.01μm. The materials utilized in this configuration are an AS-4 carbon fiber and resin. See Table 1 for material properties (here the x_3 direction is aligned with the length of the fiber).

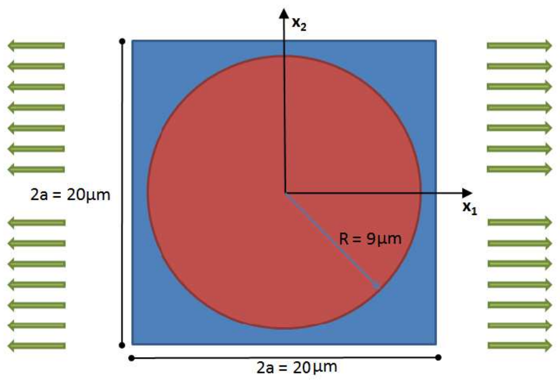


Fig. 35 Two-dimensional unit cell of a square array of circular fibers

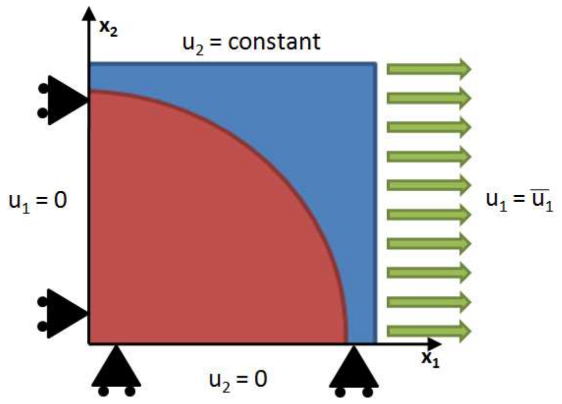


Fig. 36 Quarter unit cell of a square array of circular fibers

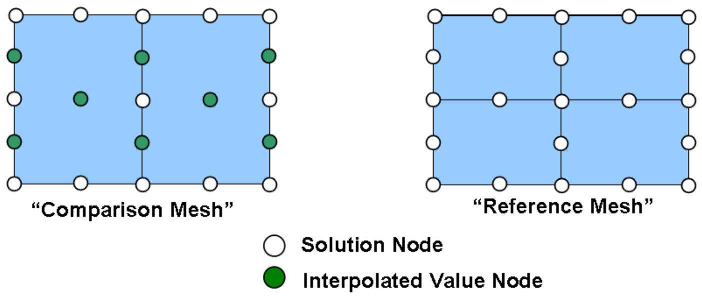


Fig. 37 Mapping of reference mesh to comparison mesh

Table 1 Material system properties

	AS-4 Fiber	Resin
E_{11}	16.55 GPa	2.96 GPa
E_{22}	16.55 GPa	2.96 GPa
E_{33}	227.53 GPa	2.96 GPa
ν_{12}	0.25	0.38
ν_{23}	0.0145	0.38
ν_{13}	0.0145	0.38
G_{12}	6.89 GPa	1.07 GPa
G_{23}	24.82 GPa	1.07 GPa
G_{13}	24.82 GPa	1.07 GPa

2. Error Analysis Utility

An error analysis utility was developed to compare stress distributions between two different meshes of a configuration. The utility accepts a mesh and stress distribution for both a reference solution and a solution that is being evaluated against the reference solution as inputs. For comparisons of B-spline FEM and standard FEM, the most refined standard FEM solution will serve as the reference solution.

Every node of the reference mesh is mapped to the comparison mesh (see Fig. 37). Every node in the reference mesh will either be on the boundary or interior of an element in the comparison mesh. The physical coordinates of the nodes of the comparison mesh element and reference mesh node are known and the normalized coordinate of the reference mesh node may be calculated. After calculation of the normalized coordinate of the reference mesh node, a stress value may be interpolated for this reference node.

After mapping all the nodes of the reference mesh to the comparison mesh and interpolating stress values, a point-wise stress comparison may be made for every node in the reference mesh. These point stress comparisons are used to calculate maximum and average error along with other statistical data. An error contour is also created to assist in visualizing which regions of a model contain the most error.

3. Convergence Study

A variety of meshes were created for the configuration described above. Each B-spline model was composed of 5 clusters (3 for the fiber and 2 for the matrix). Mesh refinement ranged from very coarse to extremely refined. The coarsest B-spline model had 194 DOF, the most refined B-spline model had 377914 DOF. The coarsest standard model had 154 DOF, the most refined standard model had 1108994 DOF. Here standard model refers to the use of quadratic serendipity elements. The most refined standard model served as the reference solution. Fig. 38 shows an example of a mesh used for this model. A B-spline finite element analysis was completed for each mesh refinement and the corresponding patch mesh was used for a standard finite element analysis. In this way, the convergence behavior of B-spline FEM could be compared to standard FEM. Figs. 38 through 40 show example contours for σ_{11}, σ_{22}, and σ_{12} for moderate refinement. These contours show nodal stresses that have been calculated by extrapolation of stresses at integration points.

The error analysis utility was used to compare all mesh refinements for both B-spline and standard finite elements to the solution of the most refined standard analysis. An average error for each stress component was calculated by averaging the point-wise error for all nodes. The error is simply the absolute value of differences in point-wise stresses. Fig. 41 presents a log-log plot of average error vs. DOF.

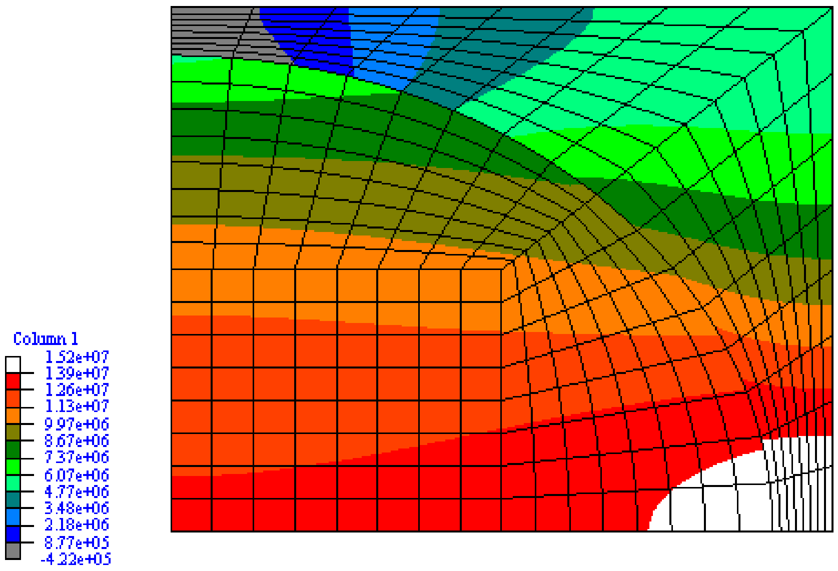


Fig. 38 σ_{11} contours for two-dimensional B-spline analysis

For the same number of degrees of freedom a lower error is obtained for all stress components of the B-spline model compared to the standard model. The σ_{22} and σ_{12} stress components have a much lower error for the B-spline solution compared to the standard solution, with the improvement for σ_{11} being not as drastic. Also, for higher DOF there is some oscillation in the average error of σ_{11} for both methods. This might be explained by round-off error in highly refined solutions. By examining the slopes of the lines, it is clear that B-spline FEM has a higher convergence rate than the standard FEM. For lower degrees of freedom the standard models have a convergence rate (slopes in Fig. 41) of approximately -1.005. The B-spline models exhibit a convergence rate of approximately -1.551. Therefore, the B-spline models converge at a rate approximately 54% faster than the standard models. For higher degrees of freedom there appears to be some leveling off to a much lower convergence rate.

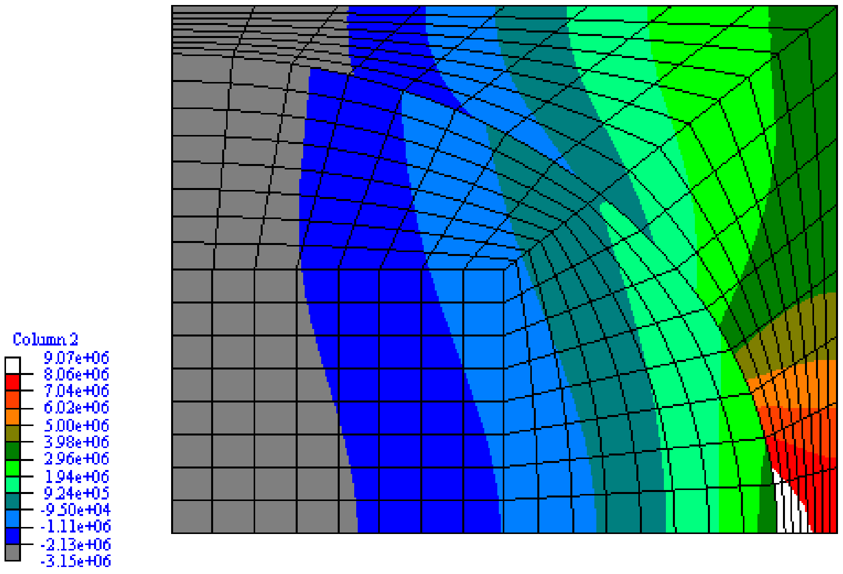


Fig. 39 σ_{22} contours for two-dimensional B-spline analysis

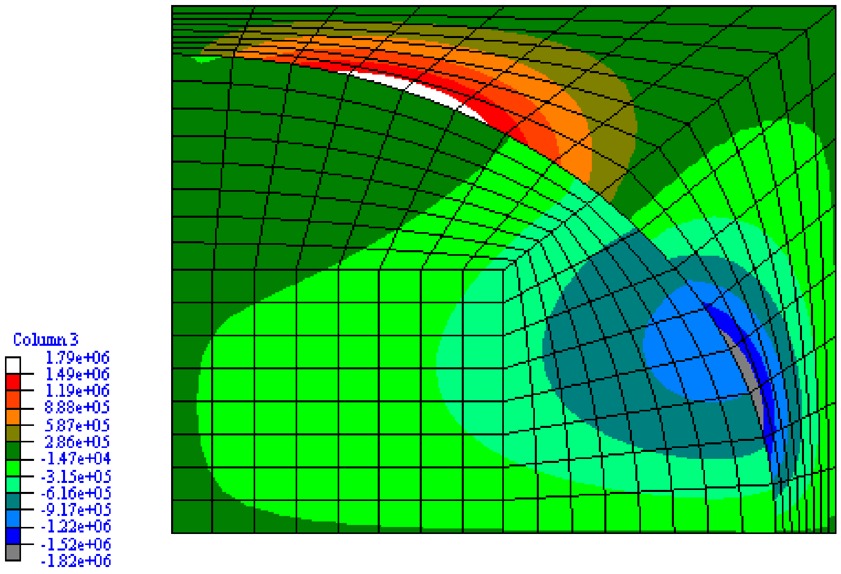


Fig. 40 σ_{12} contours for two-dimensional B-spline analysis

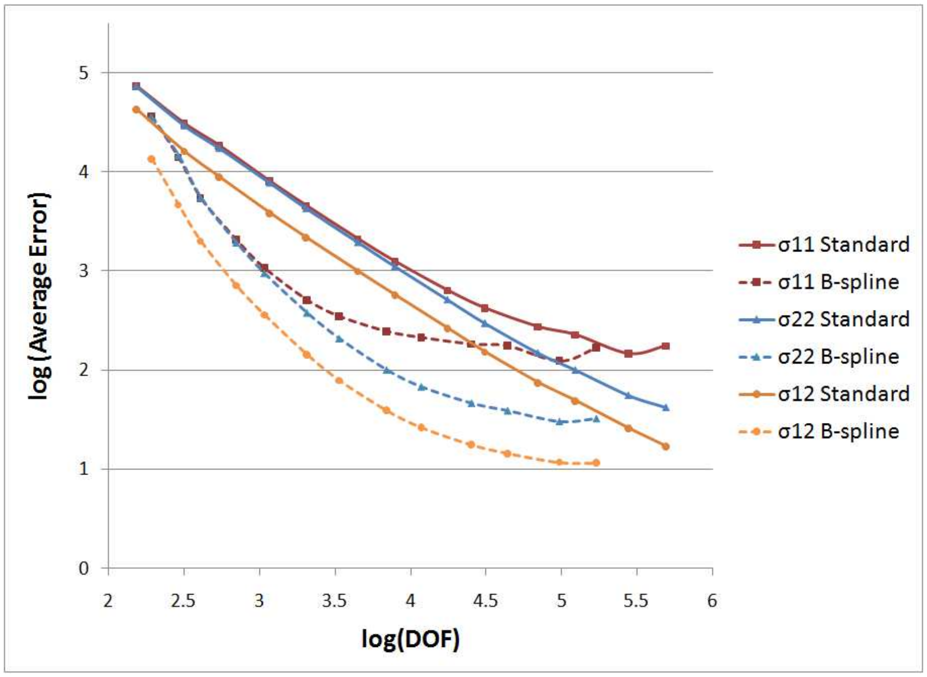


Fig. 41 Log-log plot of average error vs. DOF for standard FEM and B-spline FEM

Figs. 42 and 43 present semi-log plots of average error vs. solution time and total analysis time respectively. The solutions were obtained using the MKL Pardiso direct solver [38]. Results indicate that for a given solution time the B-spline FEM will produce a lower average error than the standard FEM. Again, there appears to be some oscillation of average error at high DOF, perhaps due to round-off error. For a given total analysis time, B-spline FEM will produce lower average error than the standard FEM for σ_{22} and σ_{12}. There is a region for σ_{11} where standard FEM produces a lower error for a given total analysis time. However, this is due to the oscillations in average error for very refined solutions. Considering all stress components, the overall trend is that B-spline FEM produces a lower average error for a given total analysis time than standard FEM.

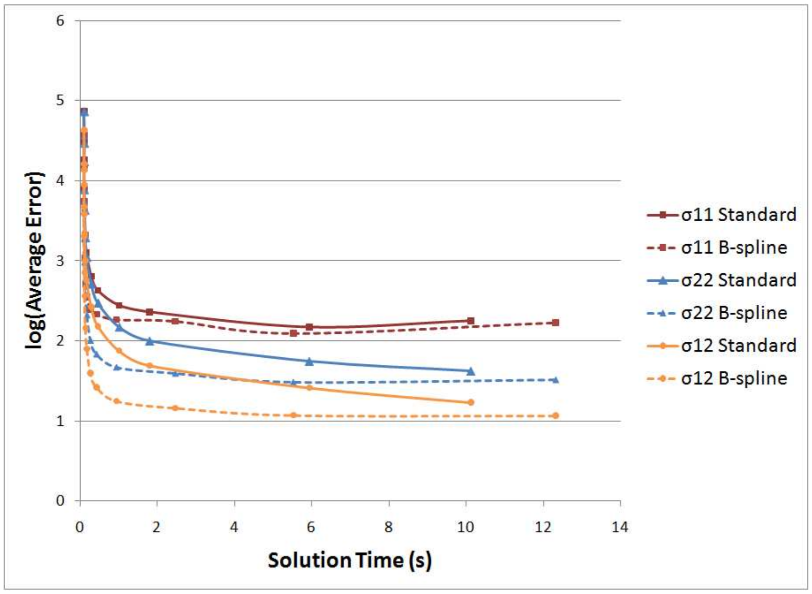


Fig. 42 Semi-log plot of average error vs. solution time for standard and B-spline analysis

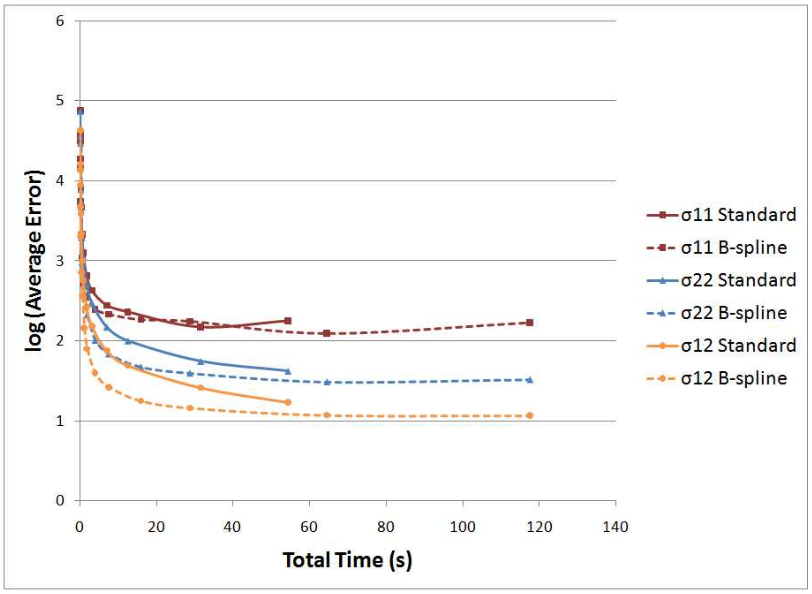


Fig. 43 Semi-log plot of average error vs. total time for standard and B-spline analysis

A B-spline analysis will typically involve less degrees of freedom than a standard analysis with comparable mesh refinement. However, this reduction in degrees of freedom does not necessarily mean a lower solution time. Since many degrees of freedom are shared between patches, the resulting system of equations is more coupled than a standard analysis and solution time will be longer (see Fig. 44). Furthermore, an element stiffness is much larger for a B-spline patch. For example, a standard two-dimensional analysis using quadratic serendipity elements will have a 16 x 16 element stiffness matrix, but the B-spline analysis using cubic B-splines will have a 32 x 32 stiffness matrix. Therefore, calculation of the element stiffness matrix and assembly takes longer (see Fig. 45). The trends of assembly time vs. DOF shown in Fig. 45 are approximately linear and show that for a given number of DOF the B-spline FEM will take over fourteen times longer to assemble than standard FEM. Also, if displacements are required a post-processing step must calculate displacement at patch mesh nodes from control point values. This additional post-processing step will increase the overall analysis time.

Figs. 46 and 47 show a breakdown of analysis time for B-spline FEM and standard FEM respectively. Allocation is a minimal part of run-time for each method. Post-processing includes calculation of stress for both methods and calculation of displacements for B-spline FEM. For standard FEM, post-processing of stress is a relatively minimal part of analysis, and writing stress values to output files is a very significant part of the run-time. In B-spline FEM post-processing of stress is still relatively minimal and post-processing of displacements takes a quarter of the run-time. If one is only interested in stress values, the displacement post-processing step may be omitted to reduce run-time. Output of post-processing data is still significant for B-spline FEM. For standard FEM assembly of the load vector and stiffness matrix is minimal compared to the time spent solving the system of equations. However,

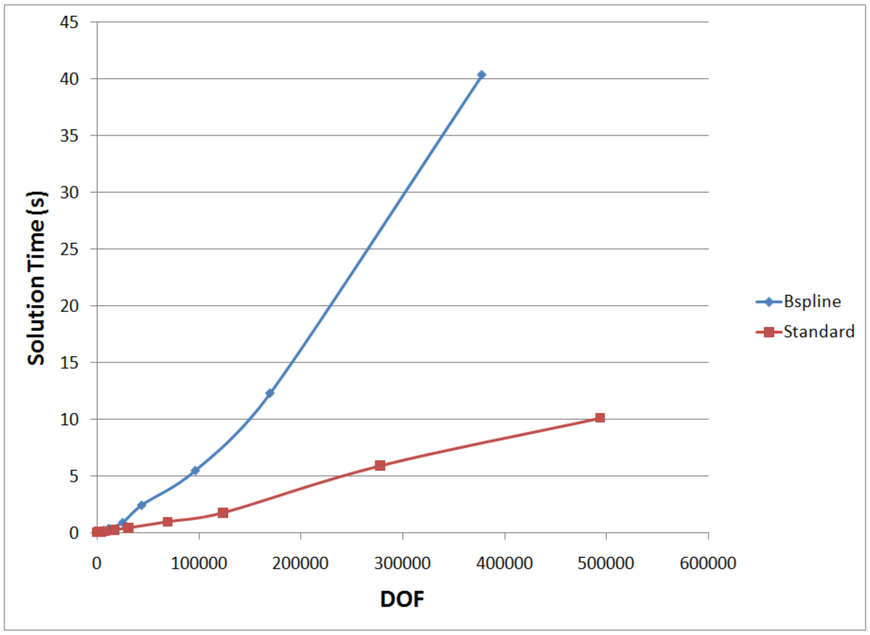


Fig. 44 Plot of solution time vs. DOF

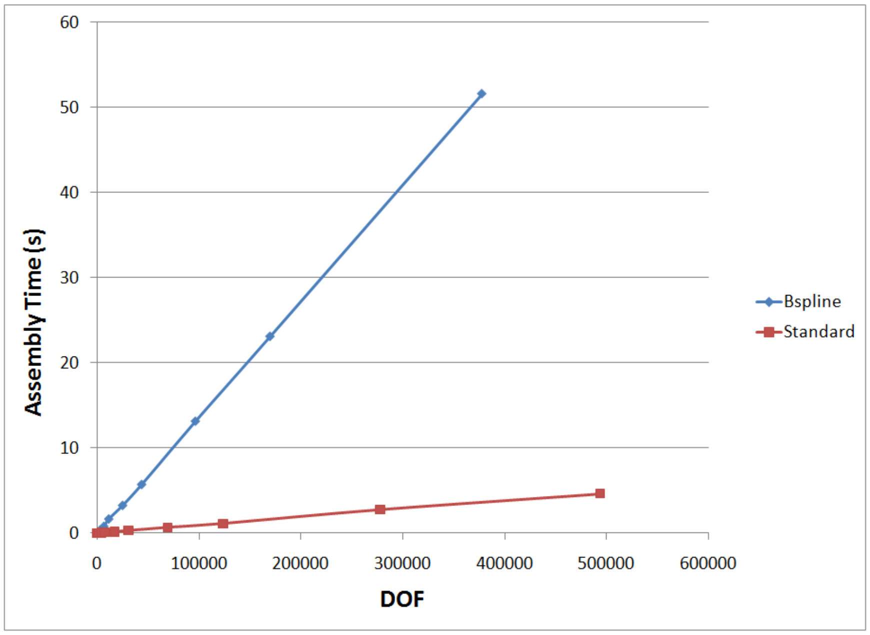


Fig. 45 Plot of stiffness matrix assembly time vs. DOF

for B-spline FEM assembly requires a greater portion of run-time than the solver. Attempts to optimize the calculation of the element stiffness matrix and assembly might assist in reducing analysis time significantly.

Another advantage of using B-splines in FEM is the increased continuity of a solution. Cubic B-splines will maintain continuity of strains and stresses between patches within a cluster. For a standard FEM analysis using quadratic serendipity elements, nodal stresses and strains are not continuous between elements of the same material. Therefore, stresses must be averaged at nodes to smooth the solution for visualization. Figs. 48 and 49 show un-smoothed stress contours for B-spline and standard quadratic analysis respectively. Notice the increased smoothness of the B-spline contour. Furthermore, each cluster is associated with a single material, and clusters have C^0 continuity at boundaries. Therefore, there is no risk of improperly maintaining continuity of stress across the interface of two materials.

B. Two-dimensional Analysis of an L-shaped Cross-section

A second configuration was studied to observe the performance of B-spline FEM and standard FEM in the presence of a singularity in the stress distribution of a model. This particular configuration was an L-shaped cross-section. This configuration will be described in detail, and results of a convergence study will be presented.

1. Configuration

The configuration for the L-shaped cross-section is shown in Fig. 50. The top of the beam has been constrained in both the x_1 and x_2 direction. A downward displacement load in the x_2 direction has been specified. Plane strain conditions were also imposed. Material properties of AISI 1005 steel have been specified for

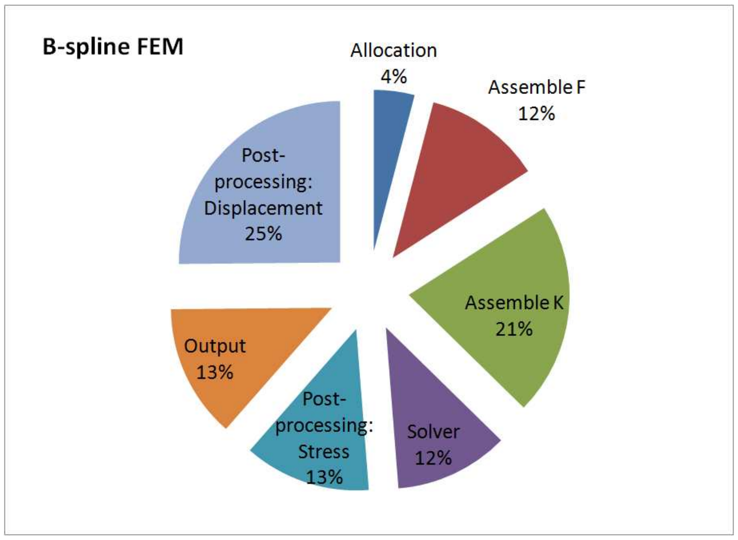


Fig. 46 Subdivision of analysis time for B-spline FEM(81920 patches, 170042 DOF)

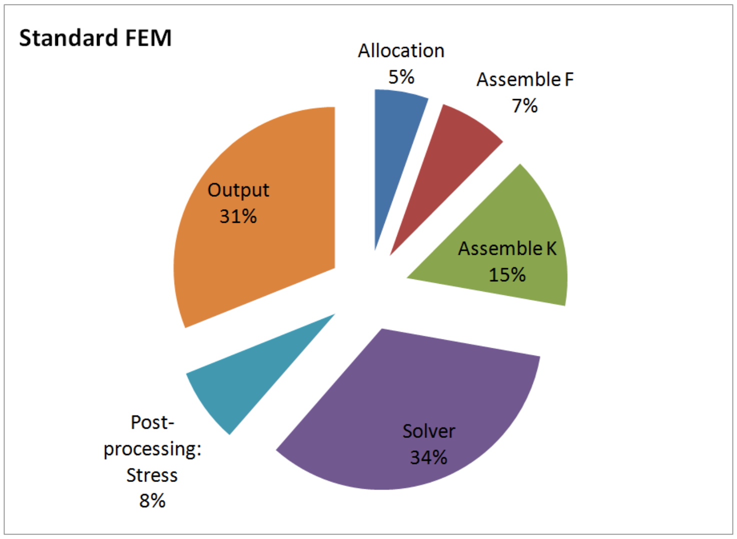


Fig. 47 Subdivision of analysis time for standard FEM(81920 elements, 493507 DOF)

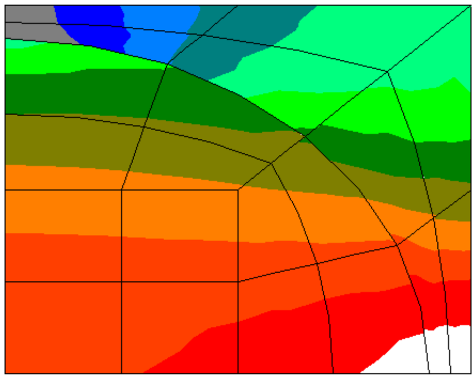


Fig. 48 B-spline FEM un-smoothed σ_{11} contour

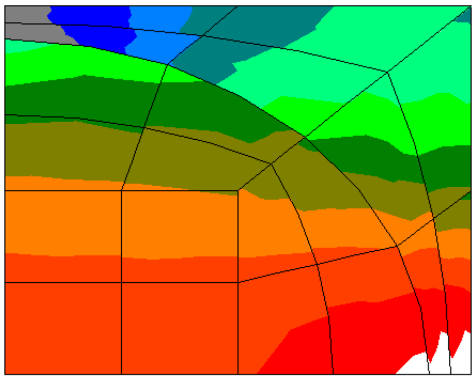


Fig. 49 Standard FEM un-smoothed σ_{11} contour

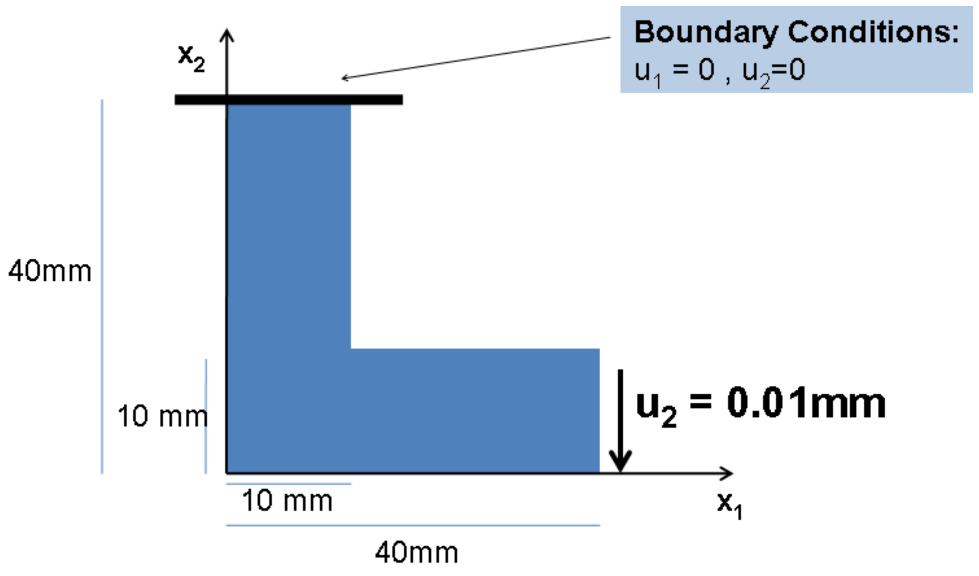


Fig. 50 L-Shaped cross-section

the configuration. This isotropic material has a Young's Modulus (E) of 200 GPa, a Poisson's ratio (ν) of 0.29, and a shear modulus (G) of 80 GPa.

2. Convergence Study

A variety of mesh refinements were constructed for the convergence study. For the B-spline models mesh refinement ranged from 990 DOF to 203154 DOF. For the standard models mesh refinement ranged from 2010 DOF to 661962 DOF. Again, the standard model uses quadratic serendipity elements. The most refined standard model serves as a reference solution to which other refinements are compared. Fig. 51 shows a typical mesh refinement for this configuration. The mesh is graded around the area at which a singularity in the stress distribution is expected. Figs. 52 through 54 show stress contours for the reference solution.

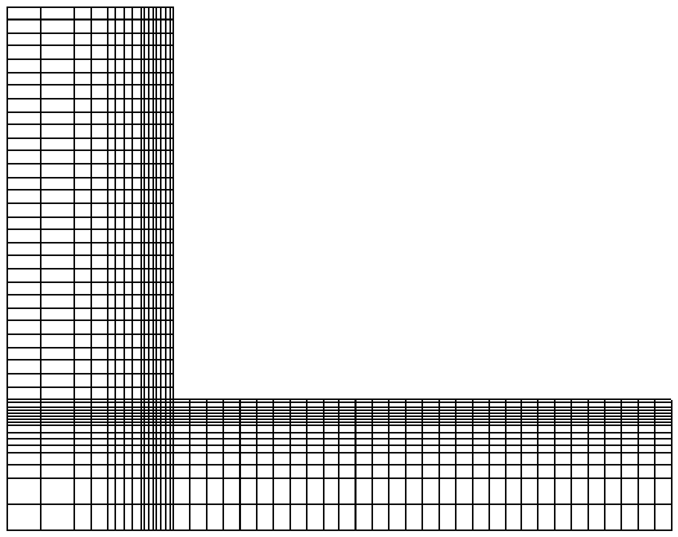


Fig. 51 Typical mesh refinement for L-Shaped cross-section

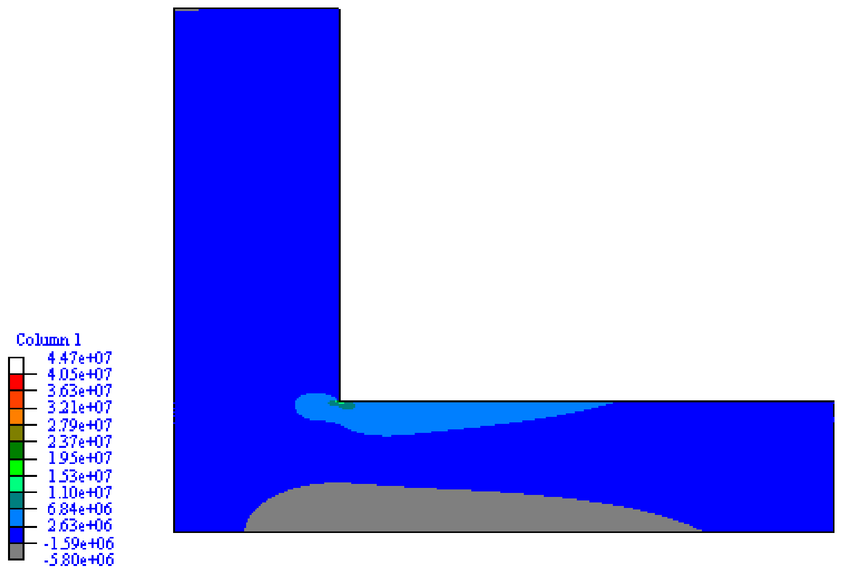


Fig. 52 σ_{11} contours for L-shaped cross-section

The stress at a particular point was extracted for the various mesh refinements of the standard and B-spline models. The existing error analysis tool was modified to extract stresses at coordinates. In the event that multiple elements shared the same coordinate, the stress values were averaged since the model was composed of one material. The singularity was located at coordinate (10.0,10.0). Two points were sampled for stress, coordinate (9.9,9.9) and coordinate (9.0,9.0). An error was calculated by taking the absolute value of the difference between a stress value of a particular mesh and the reference mesh.

Fig. 55 shows the convergence of σ_{11} for coordinate (9.9,9.9). This coordinate is very close to the singularity. The convergence for both methods is not very smooth at low and highly refined meshes. There is however a region at moderate refinements where the convergence behavior is smoother for both B-spline and standard models. In this region the B-spline model has a slightly higher convergence rate. Also, at

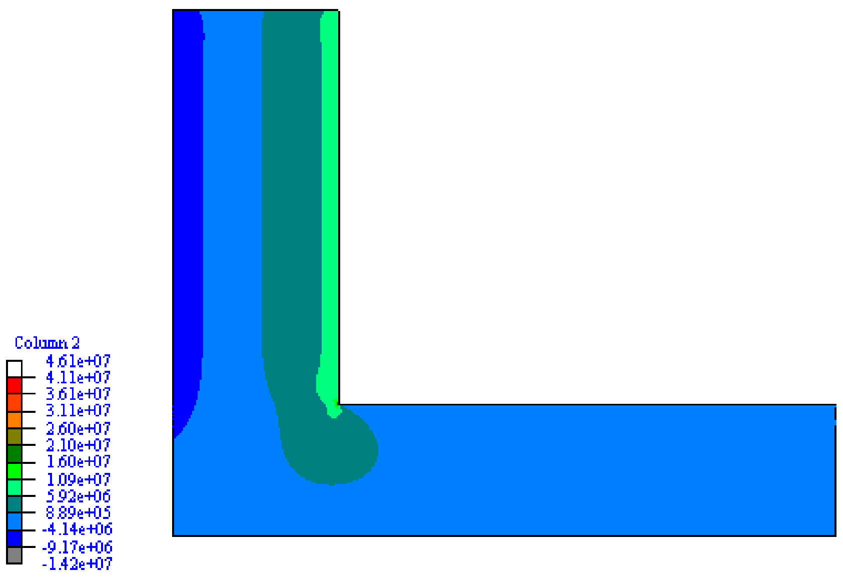


Fig. 53 σ_{22} contours for L-shaped cross-section

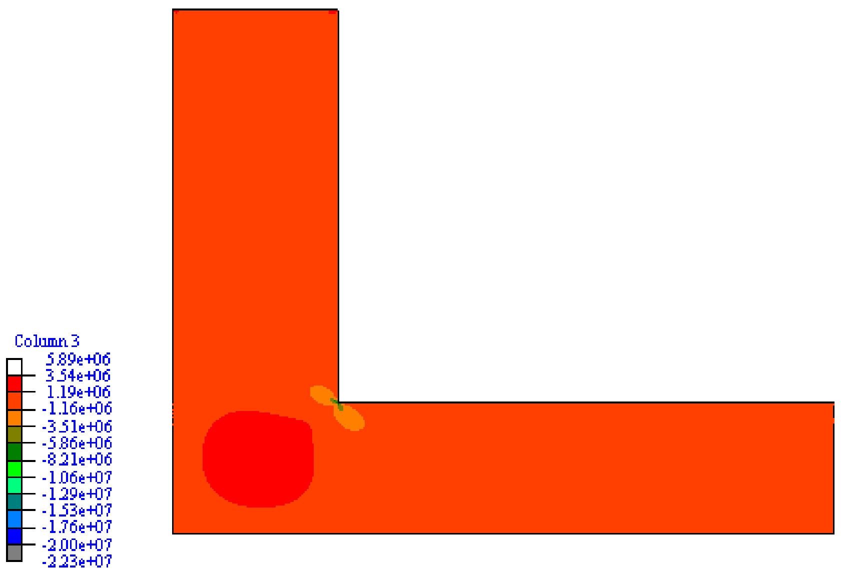


Fig. 54 σ_{12} contours for L-shaped cross-section higher mesh refinements the error of the B-spline model increases rapidly. This may be due to the B-spline model actually predicting a more accurate solution than the reference solution from which error is calculated. Convergence behavior of σ_{22} is very similar to σ_{11} and is not being shown. Convergence behavior of σ_{12} is not very smooth at all, and it is difficult to draw conclusions on convergence rates for either method. However, despite the oscillation of convergence behavior the B-spline model appears to consistently have a lower error than the standard model for a given number of degrees of freedom.

In an attempt to observe smoother convergence behavior a coordinate further from the singularity was examined. The coordinate of (9.0,9.0) should still be close enough to capture stress behavior around the singularity. Fig. 56 shows convergence of σ_{11} and σ_{12} for B-spline and standard models. The convergence of σ_{22} is very similar to σ_{11} and is not being shown. With regards to σ_{11} the B-spline model has a higher convergence rate than the standard model. Also, the B-spline model achieves lower errors at much lower degrees of freedom than the standard model. With regards to σ_{12} the B-spline also has a higher convergence rate than the standard model. Interestingly, the standard model predicts lower values of error initially. However at higher degrees of freedom the B-spline model predicts much lower error. It should be noted that the error for both stress components of the B-spline model shown in Fig. 56 reach a minimum and sharply increase. This is most likely due to the B-spline actually giving a better prediction of stress than the reference solution from which error is calculated.

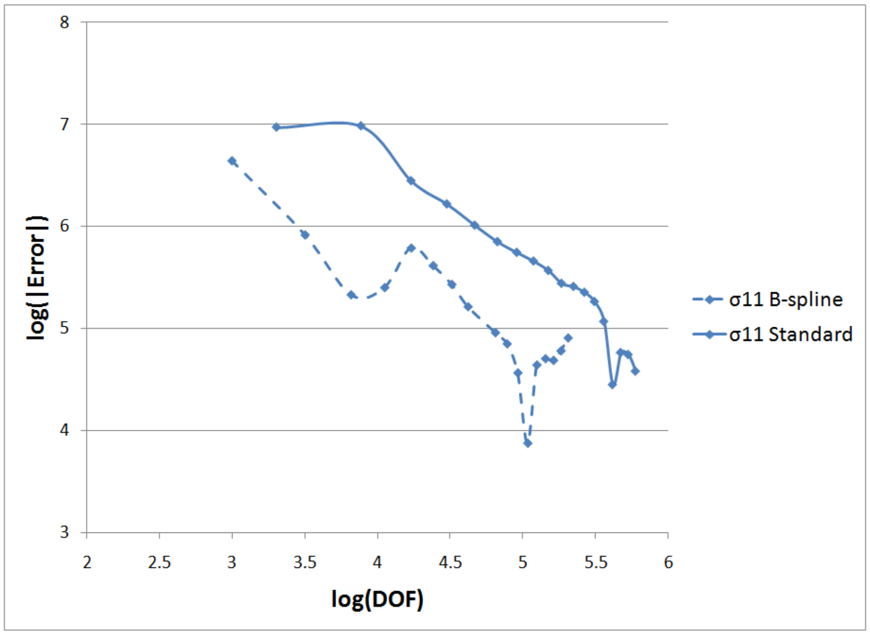


Fig. 55 Convergence of σ_{11} for coordinate (9.9,9.9)

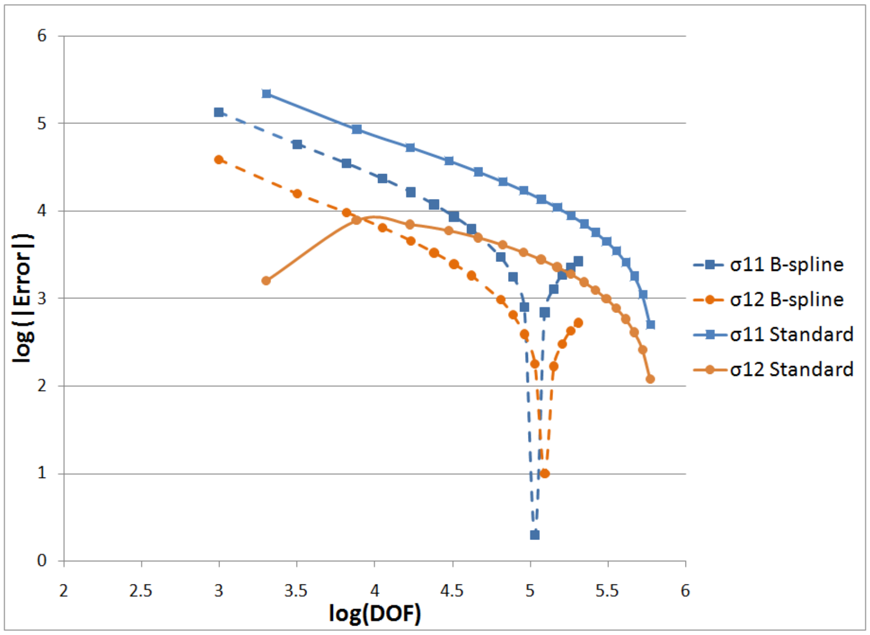


Fig. 56 Convergence of σ_{11} and σ_{12} for coordinate (9.0,9.0)

C. Demonstration of Three-dimensional B-spline Finite Element Method Capability

A three-dimensional model was created to demonstrate the three-dimensional capability of the B-spline finite element method. The overall configuration is the same as the two-dimensional square array of circular fibers unit cell, but with an extrusion in the x_3 direction. In addition to the boundary conditions of the two-dimensional model, plane strain conditions have been imposed so there is no variation of stress with respect to thickness (x_3). Non-uniform meshing was utilized with greater refinement at the fiber-matrix interface. Note that only one element was used through the thickness since results were not expected to vary through the thickness.

Since there is no variation through the thickness of the configuration, these results may be compared to the two-dimensional results. The in-plane σ_{11} distribution of Fig. 57 may be compared to the stress contours of Fig. 38. Similarly, the in-plane σ_{22} and σ_{12} stress distributions of Figs. 58 and 59 may be compared to the two-dimensional contours in Figs. 39 and 40 respectively. Fig. 60 shows the σ_{33} contour for the three-dimensional model.

Comparison of the two-dimensional and three-dimensional results serves to validate the three-dimensional results. Since a non-uniform mesh was utilized, there is also a validation of the use of non-uniform mesh refinement in three-dimensions.

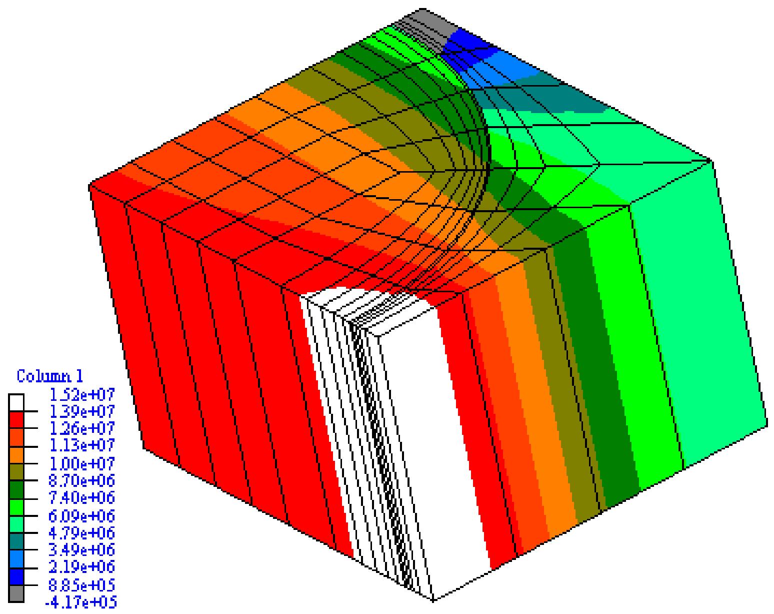


Fig. 57 σ_{11} contours from three-dimensional B-spline analysis

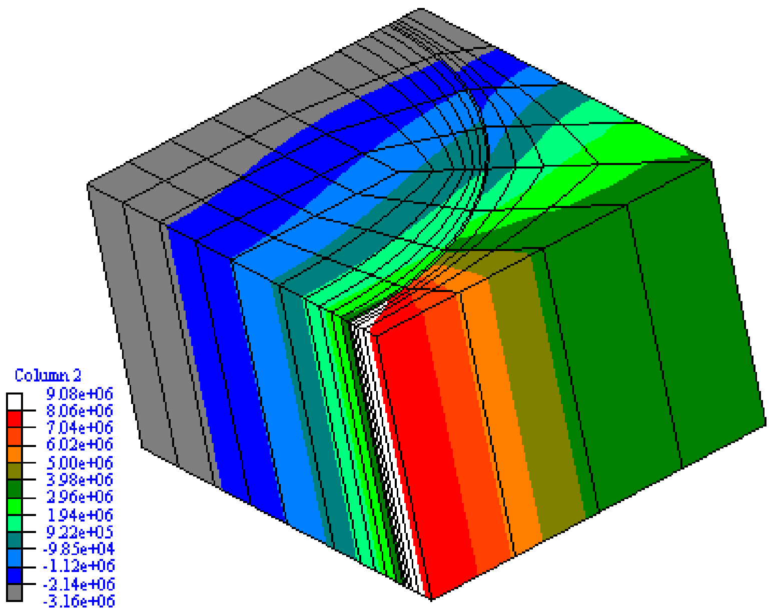


Fig. 58 σ_{22} contours from three-dimensional B-spline analysis

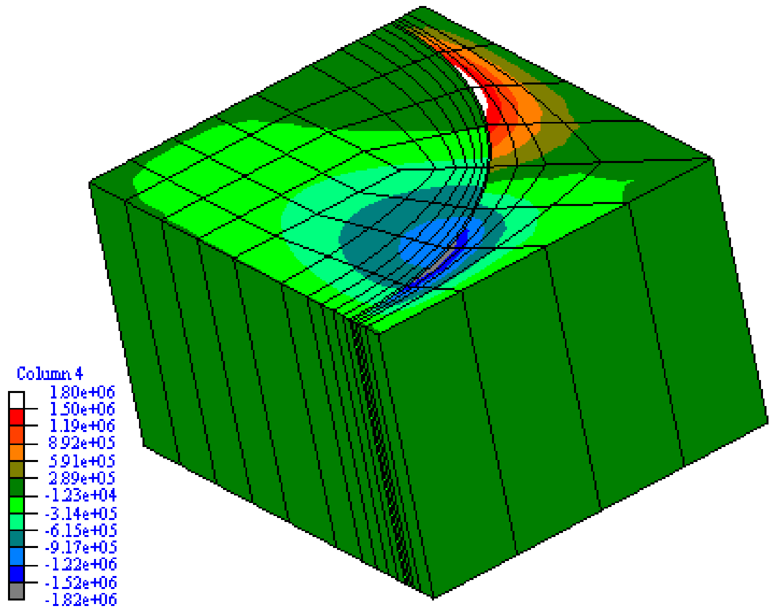


Fig. 59 σ_{12} contours from three-dimensional B-spline analysis

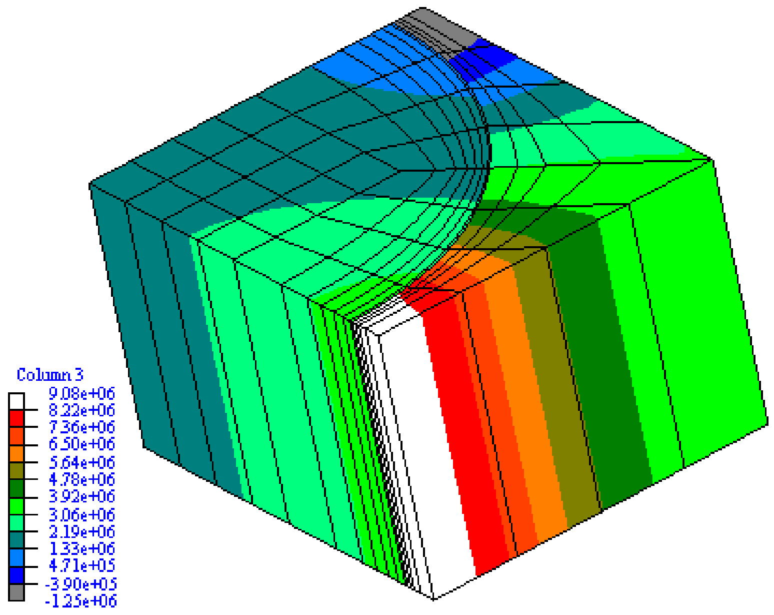


Fig. 60 σ_{33} contours from three-dimensional B-spline analysis

CHAPTER VI

CONCLUSIONS AND FUTURE WORK

B-spline functions were successfully implemented in a conventional finite element framework. This implementation made use of object oriented programming (OOP). The use of OOP allowed for a lean implementation that used existing parts of the framework that did not vary because of the use of B-splines. This new implementation is easily managed and can take advantage of future upgrades and optimizations in the core framework.

Furthermore, the pre-processor significantly expedites the development of models using B-spline FEM. Use of the pre-processor and analysis input file requires no specific knowledge of B-spline theory. Therefore, even those without any kind of background in B-splines may utilize B-spline FEM.

Results showed that B-spline FEM has the ability to achieve much lower average error with respect to a reference solution at much lower DOF than a standard FEM analysis employing quadratic serendipity elements. Also, results suggest that it is possible to achieve lower average error at lower solution and total analysis times than standard FEM. Therefore, B-splines can provide an increased efficiency with respect to error.

B-splines do provide a reduction in the number of DOF for finite element analysis. However, for a given number of DOF B-spline FEM will have longer solution times and total analysis times than for standard FEM. Results show that solution times for B-spline FEM may take longer than a standard analysis with more DOF due to the increased coupling among equations of a B-spline analysis. Furthermore, for a two-dimensional analysis the size of the elemental stiffness matrix for B-spline FEM is four times larger than that of standard FEM using quadratic serendipity elements.

Therefore, assembly for B-spline FEM will take longer than standard FEM (over fourteen times longer for a given number of DOF). Preliminary studies of the three-dimensional B-spline FEM show an even greater assembly time compared to two-dimensional B-spline FEM.

Future work can involve resolving some of the inefficiencies of B-spline FEM. This may include optimization of assembly since this is a major limitation with respect to analysis time for B-spline FEM. Furthermore, B-splines may be well suited to allow for parallelization of assembly. Since the mesh is divided into clusters, each cluster may be assembled by a separate processor. Then each cluster stiffness matrix and load vector may be assembled into the global stiffness matrix and load vector. Such a method would avoid the possibility of memory access problems during assembly.

Other future work could include describing the cluster and patch geometries using B-splines. This kind of implementation would be iso-parametric. If B-splines were utilized for describing geometry, this could allow an interface between CAD models used for design and FEM models used for analysis. Precise geometry could be analyzed instead of a less accurate approximation of geometry.

Finally, the coded implementation has been written in a way that it is general with respect to the order of the B-spline functions utilized. Therefore, quadratic B-splines or even higher ordered B-splines may be implemented within the same framework. However, implementation of other order B-spline functions would also require modification of mesh generators and pre-processing utilities.

REFERENCES

[1] Zienkiewicz, O., Taylor, R., and Zhu, J., 2005, *The Finite Element Method: Its Basis and Fundamentals*, 6th ed., Elsevier Butterworth-Heinemann, Burlington, MA.

[2] Reddy, J., 1993, *An Introduction to the Finite Element Method*, 2nd ed., McGraw Hill, Inc., New York.

[3] Farin, G., 2002, *Curves and Surfaces for CAGD: A Practical Guide*, 5th ed., Morgan Kaufmann Publishers, San Francisco, CA.

[4] Rogers, D. F., 2001, *An Introduction to NURBS: With Historical Perspective*, Morgan Kaufman Publishers, New York.

[5] Pöschl, W., 1997, "B-spline Finite Elements and Their Efficiency in Solving Relativistic Mean Field Equations," Computer Physics Communications, **112**, November, pp. 42–66.

[6] Sevilla, R., Fernández-Méndez, S., and Huerta, A., 2008, "NURBS-enhanced Finite Element Method(NEFEM)," International Journal for Numerical Methods in Engineering, **76**(1), October, pp. 56–83.

[7] Wendel, S., 1993, "Two-dimensional B-spline Finite Elements and Their Application to the Calculation of Solitons," Archiv fur Elektrotechnik, **76**, pp. 427–435.

[8] Riesenfeld, R., 1972, "Application of B-spline Approximation to Geometric Problems of Computer Aided Design," Ph.D. thesis, Syracuse University, Syracuse, NY.

[9] de Boor, C., 1972, "On the Calculation with B-splines," Journal of Approximation Theory, **6**, pp. 50–62.

[10] Verspille, K., 1975, "Computer-aided Design Applications of the Rational B-spline Approximation Form," Ph.D. thesis, Syracuse University, Syracuse, NY.

[11] Piegel, L., and Tiller, W., 1987, "Curve and Surface Constructions Using Rational B-splines," CAD, **19**, pp. 485–498.

[12] Tiller, W., 1983, "Rational B-splines for Curve and Surface Representation," IEEE Computer Graphics and Applications, **3**(6), September, pp. 61–69.

[13] Kagan, P., Fischer, A., and Bar-Yoseph, P., 1998, "New B-spline Finite Element Approach for Geometrical Design and Mechanical Analysis," International Journal for Numerical Methods in Engineering, **41**, pp. 435–458.

[14] Inoue, K., Kikuchi, Y., and Masuyama, T., 2005, "A NURBS Finite Element Method for Product Shape Design," Journal of Engineering Design, **16**(2), April, pp. 157–174.

[15] Cottrell, J., Hughes, T., and Reali, A., 2007, "Studies of Refinement and Continuity in Isogeometric Structural Analysis," Computer Methods in Applied Engineering and Sciences, **196**, pp. 4160–4183.

[16] Kagan, P., Fischer, A., and Bar-Yoseph, P., 2003, "Mechanically Based Models: Adaptive Refinement for B-spline Finite Element," International Journal for Numerical Methods in Engineering, **57**, pp. 1145–1175.

[17] Li, C.-J., and Wang, R.-H., 2006, "A New 8-node Quadrilateral Spline Finite Element," Journal of Computational and Applied Mathematics, **195**, pp. 54–65.

[18] Kumar, M., 2003, "A Second Order Spline Finite Difference Method for Singular Two-point Boundary Value Problems," Applied Mathematics and Computation, **142**, pp. 283–290.

[19] Kumar, M., 2003, "A Fourth-order Spline Finite Difference Method for Singular Two-point Boundary Value Problems," International Journal of Computer Mathematics, **80**(12), pp. 1499–1504.

[20] Rashidinia, J., Mohammadi, R., Jalilian, R., and Ghasemi, M., 2007, "Convergence of Cubic-spline Approach to the Solution of a System of Boundary-value Problems," Applied Mathematics and Computation, **192**, pp. 319–331.

[21] Kadalbajoo, M., and Aggarwal, V., 2005, "Fitted Mesh B-spline Method for Solving a Class of Singular Singularly Perturbed Boundary Value Problems," International Journal of Computer Mathematics, **82**(1), January, pp. 67–76.

[22] Brown, J., Bloor, M., Bloor, M. S., and Wilson, M., 1998, "The Accuracy of B-spline Finite Element Approximations to PDE Surfaces," Computer Methods in Applied Mechanics and Engineering, **158**, pp. 221–234.

[23] Caglar, H., Caglar, N., and Elfaituri, K., 2006, "B-spline Interpolation Compared with Finite Difference, Finite Element and Finite Volume Methods which Applied to Two-point Boundary Value Problems," Applied Mathematics and Computation, **175**, pp. 72–79.

[24] Pullman, D., and Schaff, J., 1998, "A Comparison of 3-D spline Variational and Finite-element Solutions for a Cross-ply Laminate with a Circular Hole," Mechanics of Composite Materials and Structures, **5**, pp. 309–325.

[25] Mizusawa, T., 1994, "Application of the Spline Element Method to Analyze the Bending of Skew Plates," Computers and Structures, **53**(2), pp. 439–448.

[26] Leung, A., and Au, F., 1990, "Spline Finite Elements for Beam and Plate," Computers and Structures, **37**(5), pp. 717–729.

[27] Kong, J., and Cheung, Y., 1993, "Application of the Spline Finite Strip to the Analysis of Shear-deformable Plates," Computers and Structures, **46**(6), pp. 985–988.

[28] PengCheng, S., PeiXing, H., and Yongxia, L., 1992, "Vibration Analysis of Plates Using the Multivariable Spline Element Method," International Journal of Solids and Structures, **29**(24), pp. 3289–3295.

[29] Gupta, A., Kiusalaas, J., and Saraph, M., 1991, "Cubic B-spline for Finite Element Analysis of Axisymmetric Shells," Computers and Structures, **38**(4), pp. 621–632.

[30] Hyun, S., Kim, C., Son, J., Shin, S., and Kim, Y., 2004, "An Efficient Shape Optimization Method Based on FEM and B-spline Curves and Shaping a Torque Converter Clutch Disk," Finite Elements in Analysis and Design, **40**, pp. 1803–1815.

[31] Zhong, F., and Yuqiu, L., 1991, "Linear Analysis of Tall Buildings Using Spline Elements," Engineering Structures, **13**, January, pp. 27–33.

[32] Kutluay, S., and Esen, A., 2004, "A B-spline Finite Element Method for the Thermistor Problem with the Modified Electrical Conductivity," Applied Mathematics and Computation, **156**, pp. 621–632.

[33] Aksan, E., 2006, "Quadratic B-spline Finite Element Method for Numerical Solution of the Burgers' Equation," Applied Mathematics and Computation, **174**, pp. 884–896.

[34] Ali, A., Gardner, G., and Gardner, L., 1992, "A Collocation Solution for Burgers' Equation Using Cubic B-spline Finite Elements," Computer Methods in Applied Mechanics and Engineering, **100**, pp. 325–337.

[35] Patlashenko, I., and Weller, T., 1993, "Cubic B-spline Collocation Method for Nonlinear Static Analysis of Panels Under Mechanical and Thermal Loadings," Computers and Structures, **49**, pp. 89–96.

[36] Aggarwal, B., 2006, "B-spline Finite Elements for Plane Elasticity Problems," MS thesis, Texas A&M University, College Station, TX.

[37] Becker, E., Carey, G., and Oden, J., 1981, *Finite Elements: An Introduction*, Vol. 1, Prentice-Hall, Englewood Cliffs, NJ.

[38] Intel Corporation, 2008. *Intel Math Kernel Library Reference Manual*, August.

VITA

Brian Christopher Owens was born in Dallas, Texas. He obtained his Bachelor of Science degree in Aerospace Engineering from the Dwight Look College of Engineering, Texas A&M University, College Station, Texas. in May 2007. He began graduate studies in Aerospace Engineering at Texas A&M University in August 2007. He plans to pursue a PhD degree at Texas A&M University, researching in the area of oxidation of composite materials.

Departmental Address:
Texas A&M University
Department of Aerospace Engineering
H.R. Bright Building, Rm. 701, Ross Street - TAMU 3141
College Station, TX 77843-3141

Phone: (972) 845-1594
E-mail: brian_owens@tamu.edu bowens1@gmail.com