5-2018

Numerical Evaluation of Energy Release Rate at Material Interfaces for Fatigue Life Predictions

Robert L. Hendrickson
Utah State University

NUMERICAL EVALUATION OF ENERGY RELEASE RATE AT MATERIAL INTERFACES FOR FATIGUE LIFE PREDICTIONS

by

Robert L. Hendrickson

A thesis submitted in partial fulfillment
of the requirements for the degree

of

MASTER OF SCIENCE

in

Mechanical Engineering

Approved:

Ling Liu, Ph.D. Major Professor	Thomas Fronk, Ph.D. Committee Member
Ryan Berke, Ph.D. Committee Member	Mark R. McLellan, Ph.D. Vice President for Research and Dean of the School of Graduate Studies

UTAH STATE UNIVERSITY
Logan, Utah

2018

ABSTRACT

Numerical Evaluation of Energy Release Rate at Material Interfaces for Fatigue Life Predictions

by

Robert L. Hendrickson, Master of Science

Utah State University, 2018

Major Professor: Ling Liu, Ph.D.
Department: Mechanical and Aerospace Engineering

Delamination is an important failure mechanism for composite laminates. Failure by delamination can be predicted for fast fracture or fatigue delamination. A prediction of fatigue life, in composite laminates, improves if an accurate energy release rate (G) can be calculated. One approach of G calculation is the Virtual Crack-Closure Technique (VCCT), which assumes that the work required to propagate a crack is the same as the energy to close the crack. VCCT calculates G from the nodes on the delamination front and corresponding nodes behind the delamination front. If a delamination front does not exist exactly on the nodes of the finite element (FE) mesh, the model needs to be re-meshed to conform the mesh to the crack geometry at every increment of crack growth. Currently, this re-meshing method is arguably the most accurate, but it suffers from the low computational efficiency. The purpose of this thesis is to develop and verify a new approach that does not require re-meshing yet has acceptable accuracy. Springs are added adaptively, as part of an interface element, based on the real delamination front. This allows G to be calculated at the real front. Displacements are obtained from shape functions rather than using the displacement at the nodes in the finite element mesh. The use of shape functions also distributes stiffness to nodes and does not increase the degrees of freedom (DOF) of the interface elements.

A new spring model uses an 8-node interface element that was verified with two classical delamination examples. The approach, after future refinement, may provide a simple yet effective module for high-fidelity fatigue analysis considering delaminations.

(102 pages)

PUBLIC ABSTRACT

Numerical Evaluation of Energy Release Rate at Material Interfaces for Fatigue Life Predictions

Robert L. Hendrickson

Composite materials are becoming popular in almost all industries. Carbon-fiber and glass-fiber composites are used in aircraft, sports equipment, boats, prosthetics, and wind turbine blades. In all these applications, the composites are subjected to different loads. Loads can take the form of impact or cyclic/fatigue loading, both of which decrease the strength of composites as micro-cracks grow through the composite. Composite laminates are made up of fiber plies (thin layers of fiber) and the fibers are surrounded by a resin like epoxy. It is common for laminates to fail because of delamination growth (plies peeling apart). Small delaminations do not fail a composite, but as delaminations grow, the composite weakens and eventually fails. Composites behave differently than metals do, and failure analysis is more complicated because of the various directions of fibers. Numerical methods (specifically Finite Element Analysis) exist for predicting when failure will occur, but improvements are needed to make these numerical methods more accurate and efficient. The method created, for this thesis, is computationally efficient because it doesn't require the analyst or computer to adjust the simulation based on where the delamination is (or what kind of shape it is). Energy values are extracted directly from the delamination front and not averaged from nearby locations.

Dedicated to Heidi Hendrickson

ACKNOWLEDGMENTS

I am thankful for the many hours of help that I received from my advisor Dr. Ling Liu. He has been enthusiastic and helpful whenever I needed guidance. I would like to thank Dr. Thomas Fronk and Dr. Ryan Berke for being part of my committee and supporting me on this project as well.

I would like to thank Global Engineering and Materials, Inc. (GEM) for allowing me to work with them on this project. Xiaodong Cui and Jim Lua answered many questions and directed me to where I might find solutions. I would like to acknowledge that this work was supported by GEM.

I am thankful to my classmates, friends, and family for encouraging me. I especially want to thank and acknowledge my wife, Heidi, who has supported me all the way.

Robert Hendrickson

CONTENTS

LIST OF FIGURES

NOTATION

$AMATRX$	Stiffness Matrix in Abaqus UEL
Δ	Change (difference between maximum and minimum)
G	Strain Energy Release Rate (G_I, G_{II}, G_{III})
G_C	Critical Strain Energy Release Rate (G_{IC}, G_{IIC}, G_{IIIC})
G_0	Analytical Solution of Strain Energy Release Rate (G_{I0}, G_{II0}, G_{III0})
I	One
II	Two
III	Three
R	Stress Ratio
RHS	Right-Hand-Side Vector (residual force vector in Abaqus UEL)
T_G	Glass Transition Temperature
Θ	Angle in Degrees
U	Displacement Vector in Abaqus UEL
UEL	User Element (defined for use with Abaqus)
$\sim$	About

ACRONYMS

AFEM	Augmented Finite Element Method
ARALL	Aramid Reinforced Aluminum Laminates
DCB	Double Cantilever Beam (specimen)
DIC	Digital Image Correlation
DOF	Degrees of Freedom
ENF	End-Notched Flexure (specimen)
FE	Finite Element
FEA	Finite Element Analysis
FNM	Floating Node Method
FRP	Fiber-Reinforced Polymer
GEM	Global Engineering and Materials Inc.
GLARE	Glass Reinforced
LEFM	Linear Elastic Fracture Mechanics
MMB	Mixed Mode Bending (specimen)
SFRC	Short-Fiber Reinforced Composites
TCB	Triple Cantilever Beam
UTS	Ultimate Tensile Strength
VCCT	Virtual Crack-Closure Technique
XFEM	Extended Finite Element Method

CHAPTER 1

INTRODUCTION

1.1 Significance of Composites

Choice of material is an important step in mechanical design. Some considerations are strength (for structural components), weight (for fuel efficiency), electrical conductivity, heat transfer, impact (for shielding/outer material), fatigue life (for cyclic loading), and cost which is usually the most important factor. Composites are materials that are made to meet requirements that aren't available in most traditional materials.

1.1.1 What Are Composites?

Composite structures consist of more than one material. Some examples of composites are reinforced concrete, plywood, and layered materials in the sandwich construction. Other examples that can be found, and exist naturally, are wood and bones [2].

Fiber-reinforced composites generally have a strong fiber (carbon/graphite, steel, glass, etc.) surrounded by a matrix material (polymer, ceramic, metal, etc.). Composites may be hybrid (more than one kind of fiber) or made from natural fibers (from plants). The matrix binds fibers together and keeps them aligned. The matrix helps prevent buckling of fibers, but transfers loads to them. Strength of the fiber composite depends on the quality of the fiber-matrix bond and the volume fraction of the fibers and matrix [3].

Fiber-reinforced polymers (FRP) are made with either a thermo-set or thermo-plastic resin as the matrix. A thermo-plastic matrix and certain fibers (glass) have a glass transition temperature (T_G). These materials are weak and amorphous at temperatures high above the T_G but are brittle below this temperature. Other materials that can be used as a matrix are ceramics and metals. An epoxy matrix is generally very brittle and modifying the matrix with rubber or silica particles has proven to increase the toughness and fatigue

life of FRP. Some combinations of particles have a greater toughening effect in a matrix than either of them has individually [4].

Carbon, glass, and aramid fibers have gained significant popularity since they began being used in the 1900s. Carbon fiber has the highest tensile strength out of glass, carbon, and aramid fibers. Carbon fiber is popular for many applications because the tensile strength is not affected by moisture or high temperatures. Glass fibers are used in about 90% of FRP according to [4]. Aramid fibers like Kevlar resist damage and fatigue more than other fibers. These fibers are good for bullet proof vests, and other impact resistant applications. All three of these fibers have great strength properties. Fiber composites can be made by doing a wet-layup, pre-preg layup, filament winding, resin infusion, and more. Some examples of commercial composite products made for aircraft are ARALL (Aramid Reinforced Aluminum Laminates) and GLARE (Glass Reinforced aluminum). Reference [5] discusses these and other hybrid composites ideal for aircraft and spacecraft.

Fibers are only useful in composites if the load is transferred to them effectively. The fibers must be long enough to transfer the load, and the critical length is a function of the fiber ultimate tensile strength (UTS), fiber diameter, and fiber-matrix bond strength. Fiber composites can be made of short fibers, long uni-directional fibers, or laminates (stacked plies of fiber). Each of these three types of fiber composites have a different degree to which they can be customized to meet requirements.

Short-fiber reinforced composites (SFRC) are much cheaper than the other two fiber composites mentioned (laminates and uni-directional). The critical fiber length for common fibers like carbon and glass is usually on the order of one millimeter. Short fibers are easy to use in plastic injection molding to make a structural material stronger and stiffer, but the orientations are random or aligned with the flow of the plastic. Stiffness and strength (of SFRC) become less predictable with random fiber orientation, and thus require advanced modeling techniques [6].

Fibers in uni-directional fiber composites create high bending and tensile strength in one direction (fiber direction), but the directions perpendicular to the strong direction are

generally weak [3] (see Fig. 1.1). Stiffness and strength of these composites are significantly more predictable than short fiber composites.

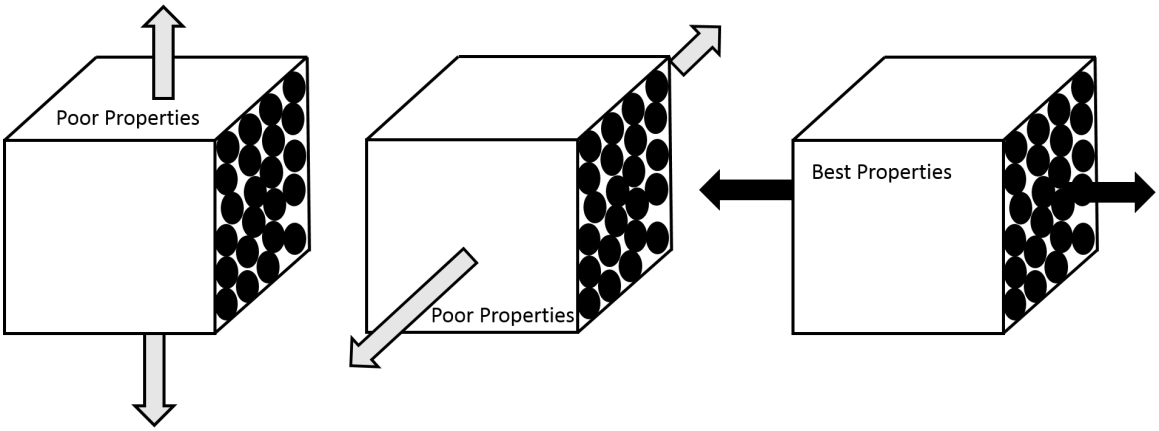

Fig. 1.1: Properties of fiber composites are dependent on fiber direction. Shear properties are poor as well (not shown).

Laminates are made of plies or laminae of uni-directional or woven fiber (Fig. 1.2). A ply is one layer of fiber in a composite laminate. Plies are laid up at angles that provide stiffness and strength in the appropriate directions. Laminates provide strength in directions that uni-directional fiber composites cannot. The stiffness and strength of laminates can be predicted for each individual ply or as a smeared property (average across the whole laminate). The interface between plies is important in this thesis as it is where delamination is likely to occur [7].

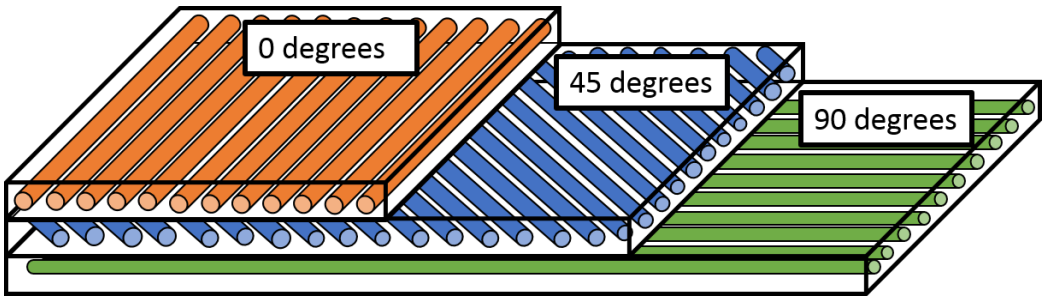

Fig. 1.2: Example of a laminate with uni-directional fiber plies.

1.1.2 Advantages of Composites

High bending stiffness makes fiber-reinforced polymers (FRP) ideal for many applications. Laying up uni-directional plies of fiber at different angles creates strength in appropriate directions. When one fiber is weak compared to surrounding fibers, stress is distributed through the stronger fibers that surround it.

Composites are lightweight and have higher specific strength (strength divided by density) than other materials. In many structural materials, strength increases with weight (e.g. steel is strong but heavy). Molding a composite can reduce the number of parts/fasteners that are otherwise used in a metal part/assembly. Composites absorb impact very well, but each damage decreases the life cycle more than normal use does. Fiber composites generally have a residual strength after impact damage because only some of the fibers/layers failed.

Poisson's ratio has a big influence on fiber composites because of differences in matrix and fiber properties. When a composite is loaded in tension, the fiber-direction strain is much less than the negative strain in the transverse directions [3]. This is illustrated in Fig. 1.3 with a transverse load, but this is a characteristic of loading in the fiber direction also. Fiber composites can be designed to have zero thermal expansion in certain directions. Many natural composites like trees, bones, and muscles have self-healing properties. Research is being done to improve healing in polymers and fibers of composites. Methods for recycling fiber composites are also improving [8, 9].

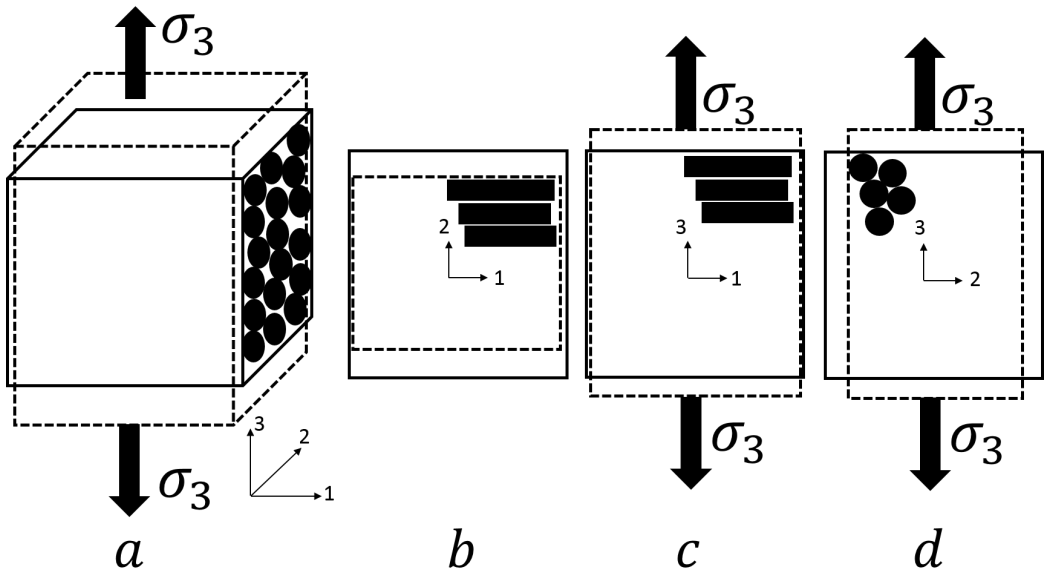

Fig. 1.3: Strain is larger in the transverse direction compared to that along the fiber direction. (a) overall view, (b) view in 1-2 plane, (c) view in 1-3 plane, (d) view in 2-3 plane.

1.1.3 Applications of Composites

Race-cars, airplanes, and rockets all benefit from better fuel efficiency if lighter materials are used. According to [10], Boeing 787 Dreamliner and Airbus A350 are made of composite parts that make up more than 50% of the overall weight. Other applications where light-weight composites have shown great potential include wind turbine blades, fuel tanks, sports equipment, and prosthetic limbs.

1.2 Composite Failure

1.2.1 Why is Failure Important to Study?

Study of failure is important for safety and cost analysis. The safety of people is a main concern in engineering. Damage may generate costs for inspections, repairs, and materials. In composites, many damages are almost visibly undetectable. It is important to know when

and how failure will occur. As illustrated in Fig. 1.4, smaller damages can be detected in a lab than can be detected in service (during regular operation). The zones, in Fig. 1.4, are not fixed. Their sizes can be determined by physical and computational testing, and a safety factor can be chosen based on the damage propagation zone [11, 12].

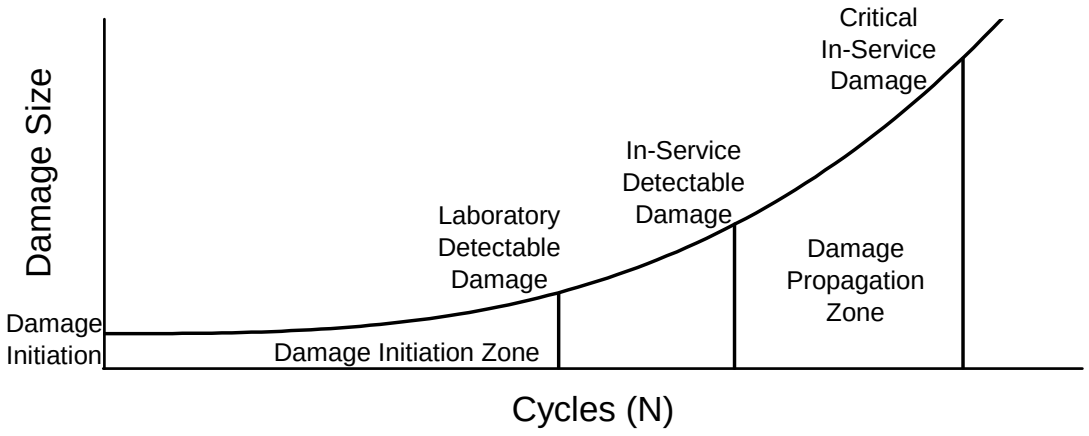

Fig. 1.4: Damage analysis and detection is used to predict fatigue failure.

Damage can initiate and grow from fatigue, manufacturing-induced initial flaws, or accidental damages. In aircraft (and other structures), design tolerance must consider damage during flight. Damage in flight is generally from impact with birds and hail [11]. Some well-known fracture failures involve rivets in ships (Titanic) or rivets too close to a square-shaped window (Comet disaster), welds in ships, fuselage failures of airplanes, and a molasses tank that flooded Boston [13].

1.2.2 Major Failure Mechanisms

There are many failure criteria that exist to determine failure in fiber composites. One example is the Tsai-Wu criterion which determines if any lamina in a composite laminate will fail under certain boundary and loading conditions (due to fiber failure) [14]. This is like the Von Mises criterion used for studying metal yielding. This doesn't work for everything because it assumes perfect bonding between fibers and the matrix. It assumes that there are

no initial delaminations, voids, or impurities in the composite. Composite plates/laminates generally have an assumed plane stress condition because the two orthogonal directions (in the laminate plane) have much greater stresses than the third orthogonal direction. That is not always accurate when the stress becomes 3-dimensional near the edge. Fracture mechanics and damage mechanics are important to study residual strength and stiffness of structures. Failure can occur at microscopic and macroscopic levels, both of which have been studied with fracture mechanics extensively [3].

Fracture in composites can initiate from micro-cracking during the curing process. It is also common for composites to have gas or impurities in a layup that remain there when cured. Fig. 1.5 shows a few defects that can occur from manufacturing. Resin-rich areas and wrinkles in fibers also commonly occur in manufacturing [15].

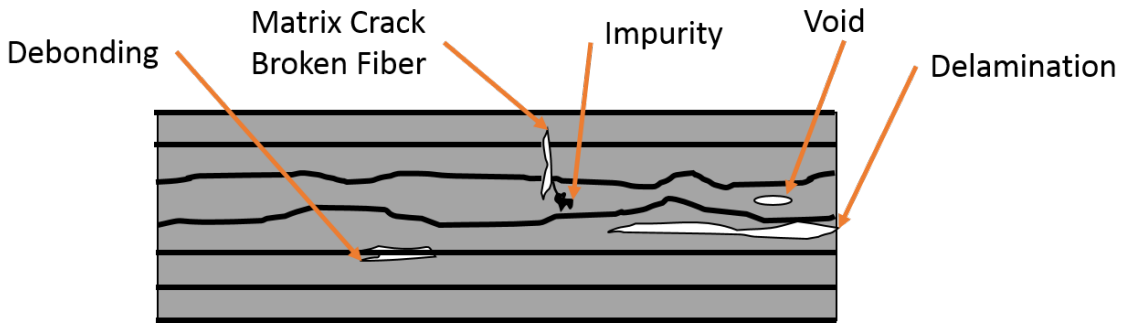

Fig. 1.5: Illustration of fiber composite defects.

When a fiber breaks, increased shear stress may cause debonding along one or more fibers. This can cause the bridging effect that slows or stops the crack growth. Fig. 1.6 illustrates bridging leading to fiber breakage and fiber pull-out [16]. Reference [17] explains how stitching can improve impact resistance by bridging delaminations.

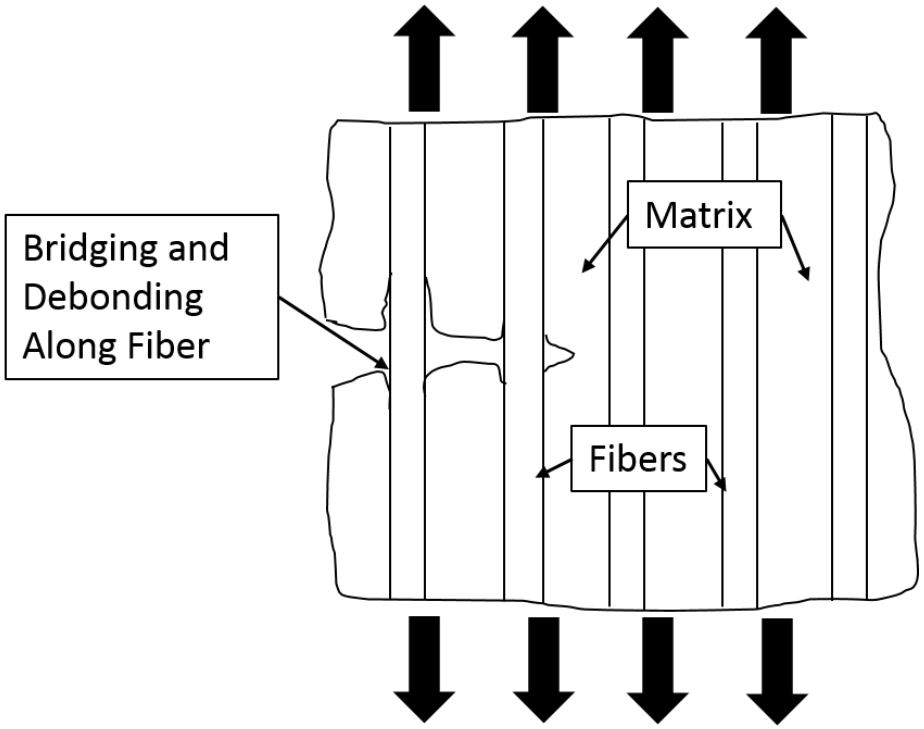

Fig. 1.6: Illustration of fiber bridging.

Fracture in laminated composites may be interlaminar or translaminar. Interlaminar failure occurs when fiber plies separate from each other (delamination) and is mostly based on matrix properties. Delamination is when plies of fiber separate from each other (Fig. 1.7). Translaminar failure occurs across the plies and is related to fiber properties. It occurs when the fibers are in tension. More specific forms of fracture are fiber/matrix debonding, delamination, and fiber pull-out. Fiber bridging, crack deflection, friction, and crack tip void formation are some mechanisms that can help suspend crack growth and improve the fracture toughness of fiber laminates [13].

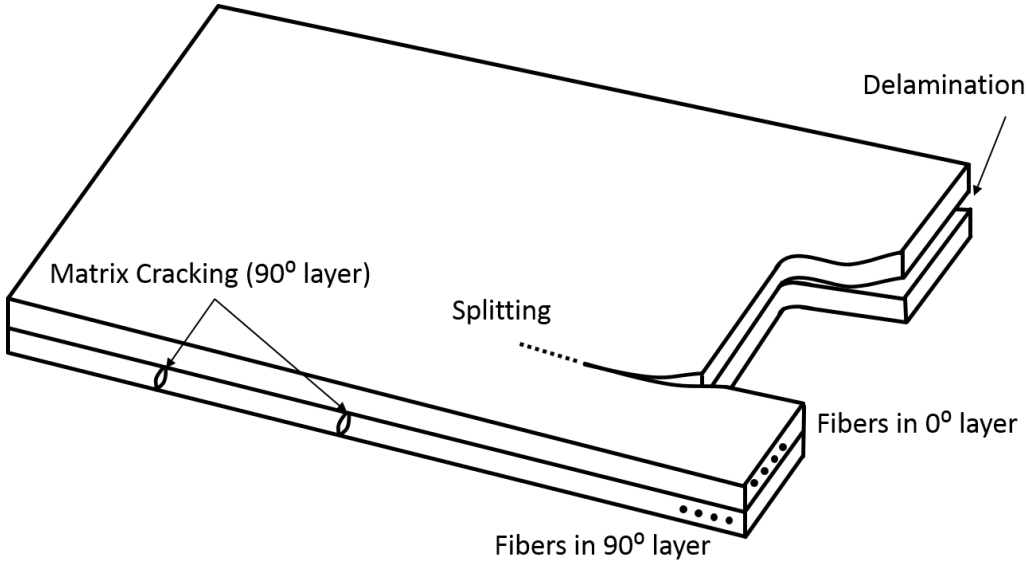

Fig. 1.7: Delamination occurs when fiber plies separate from each other.

Fracture can happen as fast fracture or fatigue fracture. The latter is the most common failure mechanism in composites and can exist in the form of thermal, corrosion, and/or stress-induced fatigue. Impacts are another common factor that accelerate fracture in fatigue environments. Other damage contributions come from chemicals, water absorption, and ultraviolet radiation [16–22].

1.2.3 Delamination and its Significance

This thesis is on delamination. Delamination is a form of fracture. Energy release rate (G) is commonly used to predict fracture, it is the energy dissipated per unit surface area created. There are three modes of fracture, and the critical energy release rate for each mode is described as G_{IC}, G_{IIC}, and G_{IIIC}. The three fracture modes can be described as opening (Mode I), in-plane shear or sliding (Mode II), and out-of-plane shear or tearing (Mode III). The critical energy release rate means that fast fracture occurs when the energy release rate equals this value.

Delamination between fibers plies and interfaces is common because fiber bridging hardly occurs there. Less energy is required for fracture to occur at an interface. In aircraft

parts, aluminum might be bonded to carbon or aramid fiber plies (like ARALL mentioned before) and that is where debonding can take place. Certain features of a composite are likely to accelerate delamination/crack growth, such as corners or bolt holes. These stress concentrations help direct delamination paths as well.

Fatigue is important in composites because voids, micro-cracks, and other defects often exist from manufacturing (sometimes due to curing and cooling rates). Voids and initial delaminations propagate when undergoing a cyclic loading. Composites have high residual strength which causes delaminations to grow slowly at first; composites can be used for many cycles with small delaminations. The delamination growth rate begins to increase as delaminations get bigger. The life (number of cycles to failure) can be predicted through experimental studies or computer simulation.

1.3 Study of Delaminations

Delamination simulation in composites is important because it is a common mode of failure. Linear elastic fracture mechanics (LEFM) is assumed in delamination of composites if the matrix is brittle. Fibers are generally brittle and do not have a yield point. An elastoplastic fracture model may be more appropriate when analyzing a tough matrix [23].

1.3.1 Experimental Approaches

There are many approaches for inspecting composites that could also be used to experimentally study them. Non-destructive test methods are important for inspections. These may include visual inspection, ultrasonic measurements, thermal imaging (thermography), tomography methods, and acoustic methods [15, 24–27].

Experimental study of fracture and delaminations include inspection methods that measure the size of damage. An approach for studying fracture is Digital Image Correlation (DIC). DIC is becoming more popular and efficient in engineering as camera and computer technologies improve. Image correlation is done by comparing several subsequent images to determine displacement and strain fields. DIC is a surface method, and some delaminations might not be detectable from the surface. Figure 12 in [28] (Nikishkov, et al.) shows how

DIC can be used to experimentally study fracture by comparing a DIC test specimen with a finite element model [28–30].

Sandwich specimens are popular in measuring fracture of adhesion. These specimens are the Double Cantilever Beam (DCB), End-Notched Flexure (ENF), Mixed Mode Bending (MMB), and the modified MMB shown in Fig. 1.8 [13]. These have analytical solutions for the strain energy release rate. These solutions assume that the adhesive does not deform much, but the equations could be adjusted for large deformation.

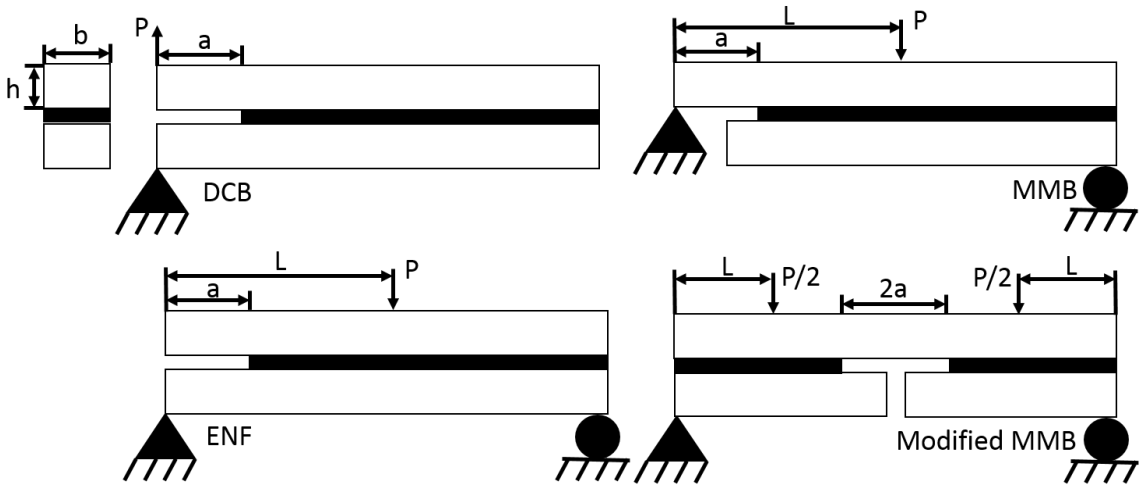

Fig. 1.8: Different fracture modes are simulated with laminated-beam specimens.

G_{I0} in Eq. 1.1, is the analytical solution of G for the DCB where P is the load at failure, a is the crack length, b and h are the width and height, and E_s is the modulus of the substrate/adherend [13]. The DCB has been used in experimental studies of delamination in composites, mostly with uni-directional fibers, but [31] studies multi-direction fiber composites with a DCB. ENF and MMB are also studied experimentally in [32–34].

$$G_{I0} = \frac{12P^2a^2}{b^2h^3E_s} \tag{1.1}$$

1.3.2 Computational Approaches

Cohesive models, nodal release methods, and the eXtended Finite Element Method (XFEM) are computational approaches to simulate fracture and delamination. All these computational approaches use finite element analysis (FEA). In FEA, the degrees of freedom (DOF) of the system are the nodes of the finite element mesh. It is problematic (causes errors) when the nodes do not conform to delamination fronts. Re-meshing of models is expensive in fatigue analysis. Adjusting the following approaches, to make analyses independent of mesh, is done with the goal of making fatigue analysis more efficient.

Cohesive zone models use an elastoplastic fracture approach. Cohesive elements must be placed in between all the elements where fracture/damage might take place. A double cantilever beam (DCB) only needs one layer of cohesive elements, but other models might need cohesive elements to be placed between all the existing structural elements. This approach is useful in fatigue when a material is not brittle and does not have a sharp crack tip. This method doesn't require an initial crack like LEFM does. The element fails, and new crack surfaces are formed when the area under the traction-displacement curve is equal to the critical energy release rate. Fig. 1.9 shows a bilinear representation of the traction-displacement curve that is often used. This assumes linear softening and a linear modulus of elasticity [35–37].

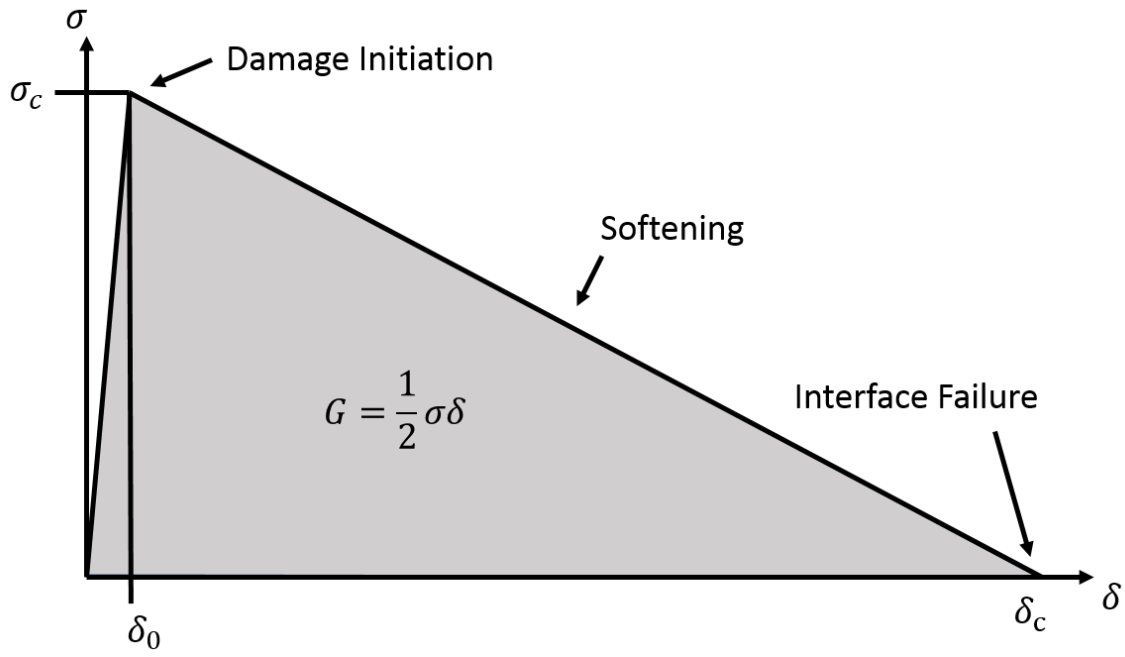

Fig. 1.9: Illustration of cohesive zone model damage (stress vs. displacement).

Nodal release methods calculate the strain energy release rate (G) by calculating a force, area of surfaces created, and a displacement. Nodes at interfaces are released when the G is high enough to propagate the crack. Virtual Crack-Closure Technique (VCCT) is popular and this nodal release method is discussed in Sec. 1.4 as part of the method of spring models that is validated in this thesis [38–40].

XFEM allows cracks/delaminations to grow through an element (it was developed more recently than other approaches). It is not restricted to nodes, geometry conforming mesh, or interface elements. XFEM has many formulations, but all of them involve creating enriched nodes around the crack front. Displacements are interpolated with enrichment functions in addition to regular shape functions to account for the discontinuities (displacement and strain) in the cut element (extra degrees of freedom (DOF) are used). The displacements are used in the calculation of G. XFEM is not generally used for delamination. However, some researchers have created interface elements based on the enrichment functions of XFEM to model delamination [41].

1.3.3 New Methods and Improvements (Literature Review)

Many researchers have worked on creating analysis methods for delaminations where the calculation does not depend on re-meshing. Many of these methods require extra degrees of freedom (DOF) that are activated around a crack front. Integration is also required to calculate stiffness in interface elements that are not simple spring elements. This thesis validates models that do not increase the DOF.

Chen et al. discussed modeling delamination growth in [42]. Their approach uses a Floating Node Method (FNM) in which an enriched ply element was created. In FNM, cracks are simply modeled because the DOF can be assigned to edges or surfaces. The extra DOF exist in the element but are not tied to the nodes. The crack is modeled in an element when floating DOF are allocated to the crack front on the element edges (2-dimensional crack example). The element is divided into a cohesive crack subdomain and two bulk subdomains whose coordinates and displacements are defined by the floating DOF. Elements were created with the FNM to use with various forms of tensile failure in fiber composites [42].

Static condensation is used to decrease computational expense by condensing internal DOF of higher-order elements [43]. Augmented Finite Element Method (AFEM) similarly condenses internal DOF. It creates two sub-domains to model a crack with the internal DOF. The equilibrium equations are used to solve displacements at internal and external nodes (using the principle of virtual work). In the final equilibrium equation, the internal displacements do not exist [44, 45].

Latifi proposed using the level set method to track delamination front growth, through elements, independently of mesh [46]. A level set field is updated with the velocity component of the Paris law (fatigue law that is discussed in Sec. 1.4 as part of the methods for spring models studied in this thesis). The model, in this level set method, has two sub-models that are solved. The cracked laminate sub-model has a kinematic formulation that solves the displacement field, and then the other sub-model uses that displacement to compute G. This method is not computationally expensive, even though a second system

of equations must be solved, because the second system is linear and is only in proximity of the delamination front [46].

De Carvalho et al. presented research that included a progressive nodal release method. As a delamination front moves through elements in this spring model, a kinematic constraint is applied to the top and bottom nodes directly behind the crack front. The force at this node pair decreases linearly to zero as the delamination reaches the next node pair; at the same time, the displacement jump is linearly increased. An 8-node interface element was used in the formulation [47].

In [47], Gaussian averaging was used to correct for crack distortions that can grow in fatigue analysis. The fatigue method is based on a growth increment, but it is mentioned that the same method could be adjusted for a cycle-based fatigue analysis. The progressive nodal release method that was used prevents crack growth and crack healing when loading conditions vary [47].

1.4 GEM Framework of Delamination Fatigue Analysis

The GEM framework is used to model fatigue independently of mesh. It was created by GEM (Global Engineering and Materials Inc.) and is made up of three methods: Virtual Crack-Closure Technique (VCCT), Paris law, and the description of the delamination front. In the GEM approach, VCCT and Paris law are the foundation, but the description of the delamination front is the key to extracting forces, displacements, and surface areas to be used with VCCT. At specified points along the delamination front, VCCT calculates energy release rate (G) from inputs of force, displacement, and surface area associated with its point. A crack growth increment (a) and G are used as inputs for the Paris law, which in turn outputs the number of cycles for the delamination to grow. The GEM method describes the delamination front with nodes close to the real delamination front (virtual front) and these nodes make up the FE front. An FE front is used because nodes are the degrees of freedom (DOF) where forces and displacements are calculated.

1.4.1 Virtual Crack-Closure Technique (VCCT)

The Virtual Crack-Closure Technique (VCCT), which outputs G to determine if a delamination will grow, assumes that the work required to close a crack is equal to the energy required to open the crack. The real VCCT approach requires that the mesh conform to the crack front, but the VCCT approach in spring models does not require a conforming mesh. Similar approaches are the crack extension methods and other crack closure methods that physically extend or close the crack. 2D models are ideal for simulating crack growth because they use much less computation. Fig. 1.10 shows the modeled crack in the deformed view where forces (Z_i and X_i), displacements (w_l, w_{l^*}, u_l, and u_{l^*}), and area ($\Delta a \times$ unit width) can be used to calculate the G (Eq. 1.2). G is equal to the energy ($Force \times Displacement$) over the crack surfaces created (two surfaces are created) [38–40]. Krueger in [38] gives an extensive overview of VCCT and how it can be used for delamination. This work includes computational and experimental studies.

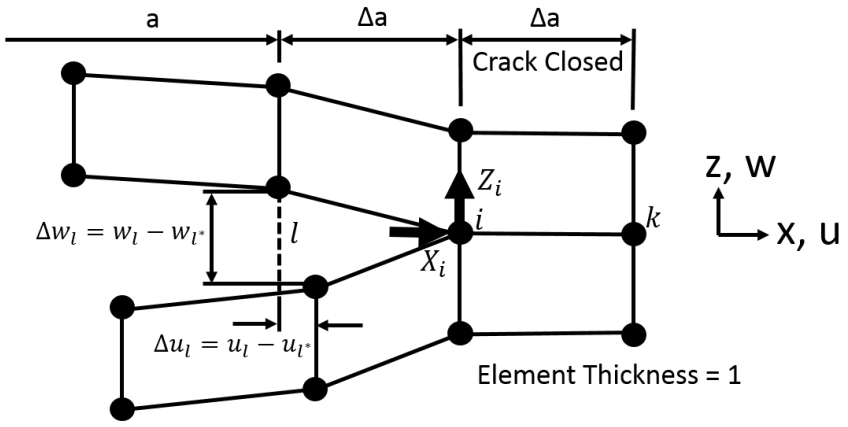

Fig. 1.10: Illustration of 2D VCCT.

$$G_I = \frac{-Z_i(w_l - w_{l^*})}{2\Delta a}$$
$$G_{II} = \frac{-X_i(u_l - u_{l^*})}{2\Delta a} \tag{1.2}$$

If a propagation analysis is to be done, two nodes are set to the same coordinates with multi-point constraints where the crack can grow (interface). Otherwise, only one node is necessary at that interface. See G_I and G_{II} in Eq. 1.2 for 4-node 2D elements as shown in Fig. 1.10 (note that G_{III} is assumed to be zero because of the plane stress/plane strain assumption).

As seen in the Fig. 1.10, it is assumed that Δa in front and behind the crack tip are equal. There are corrections to the calculation method if the lengths of the elements are not the same (see [38]). 2D analysis assumes a unit width of the elements.

The 3D model example is shown in Fig. 1.11, and it is assumed that the element width b is equal for all elements on the crack front. Again, [38] is referenced for corrections if the widths are different. The equations for 3D 8-node elements remain the same for 4 node plate/shell type elements (see Fig. 1.11 and Eq. 1.3). Forces are Z_{Li}, X_{Li}, and Y_{Li}, and the displacements are w_{Ll}, w_{Ll^*}, u_{Ll}, u_{Ll^*}, v_{Ll}, and v_{Ll^*}. Forces, displacements, and surface area ($b \times \Delta a$) determine the value of G. If the delamination is an arbitrary shape, and not a straight line, then a local/natural coordinate system at the crack tip should be defined. See equations in [38] for elements with mid-side nodes and singularity elements.

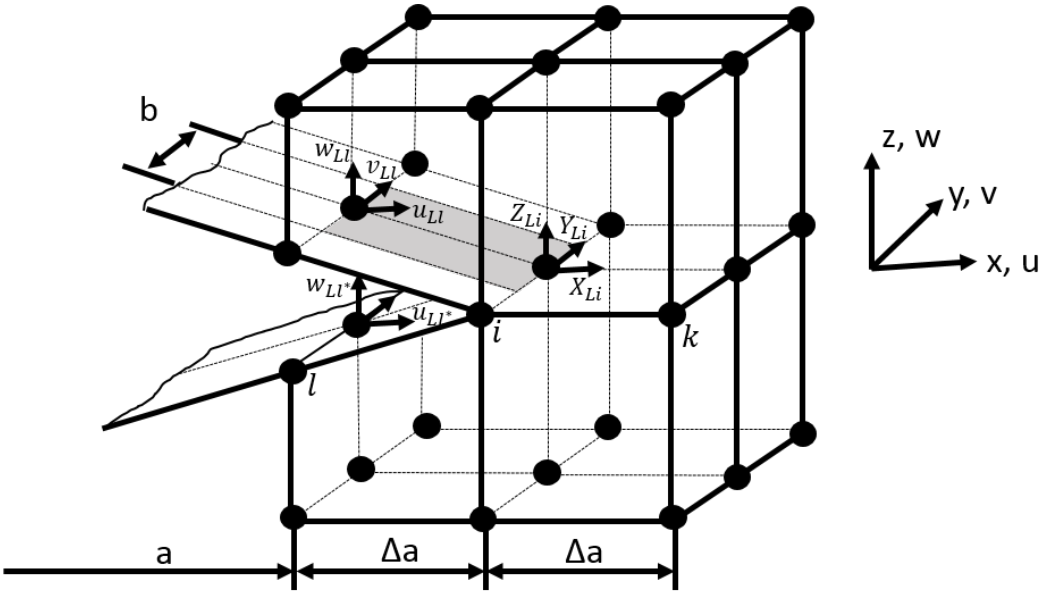

Fig. 1.11: Illustration of 3D VCCT.

$$G_I = \frac{-Z_{Li}(w_{Ll} - w_{Ll^*})}{2b\Delta a}$$

$$G_{II} = \frac{-X_{Li}(u_{Ll} - u_{Ll^*})}{2b\Delta a} \tag{1.3}$$

$$G_{III} = \frac{-Y_{Li}(v_{Ll} - v_{Ll^*})}{2b\Delta a}$$

Sharp-corner delaminations do not work well with this method, but a mesh can be refined to have a small round-corner delamination. For bi-material interfaces (like composites that have directional properties), accurate results will only be achieved if appropriate element sizes are used. The length Δa should be determined based on the initial crack length a and the sameness of material. Element height should be determined by comparing ply thickness to element length (Δa). Shell elements decrease computation time, but they are not always appropriate to use near delaminations. For example, composites that have

delaminations on various layers or near edges are better modeled with 3D elements [38].

VCCT in a Spring Model

A spring model is used by creating interfaces where it is known that the delamination could occur. This interface is made by creating stiff spring elements between the two sides of the interface. An initial delamination can be specified where the springs are removed or reduced to zero stiffness. The GEM spring model (GEM framework) can also initiate a delamination in an area of high stress (still at the interface but where there is no existing delamination), but that is not the focus of this project.

The springs have stiffness in three directions and displacement is based on Hooke's law $F = kx$ (constitutive equation). As described in the VCCT approach, a force, displacement, and area are used to calculate energy release rate G (Eq. 1.3). The force is extracted from the spring, the displacement is taken from a specified distance behind and normal to the delamination front (usually based on element length), and the area is based on the size of the elements as described in [47]. The spring model is based on linear elastic fracture mechanics (LEFM). High forces at the crack front are used to determine G which determines if a crack will grow (forces are concentrated at crack fronts) [48].

1.4.2 Paris Law

The Paris law, which predicts fatigue life, has several forms of expression. Eq. 1.4 is the form that is used for this thesis. C and m are material properties that must be found experimentally. G_C is also a material constant, namely the critical energy release rate ($G_C = G_{IC} + G_{IIC} + G_{IIIC}$). Energy release rate G is to be calculated, and its value determines the crack speed da/dN ($G = G_I + G_{II} + G_{III}$) [34, 35].

$$\frac{da}{dN} = C\left(\frac{\Delta G}{G_C}\right)^m \tag{1.4}$$

The stress intensity factor (for fracture in metals) is related to G. In analysis of composites, it is more useful to use G. Unlike brittle/fast fracture, a crack in fatigue loading

propagates at much lower G values than G_C. It is most important to consider ΔG when designing because the varying load is what causes the failure. Composites have fatigue thresholds like some metals, but they can vary with temperature or loading configuration [11, 13, 35].

An analysis can be done by incrementing either the number of cycles N or the crack growth a. Integration can solve the analytical solution. The integral in Eq. 1.5 is used for incrementing crack growth a (in an analysis, a is not infinitesimal and the solution is not integrated). The energy release rate is generally a function of the crack length a, but not always.

$$N = C^{-1} \left(\frac{1}{G_C} \right)^{-m} \int_{a_0}^{a_1} \Delta G^{-m} da \tag{1.5}$$

Eq. 1.5 can also take the form of Eq. 1.6 where $\Delta G = G_{max}(1 - R^2)$. R is the stress ratio $\sigma_{min}/\sigma_{max}$ which equals $\sqrt{G_{min}/G_{max}}$. The integrand has been simplified in this form of the equation.

$$N = C^{-1} \left(\frac{1 - R^2}{G_C} \right)^{-m} \int_{a_0}^{a_1} (G_{max})^{-m} da \tag{1.6}$$

1.4.3 Description of the Delamination Front in a Spring Model

The purpose of fatigue simulation with the GEM spring model is to make an accurate prediction of fatigue life that does not depend on the shape of the finite element mesh. The true front is represented by a virtual front (it does not coincide with the DOF of the model). The FE front is the node approximation of the virtual front (shown in green in Fig. 1.12). The GEM method is described below:

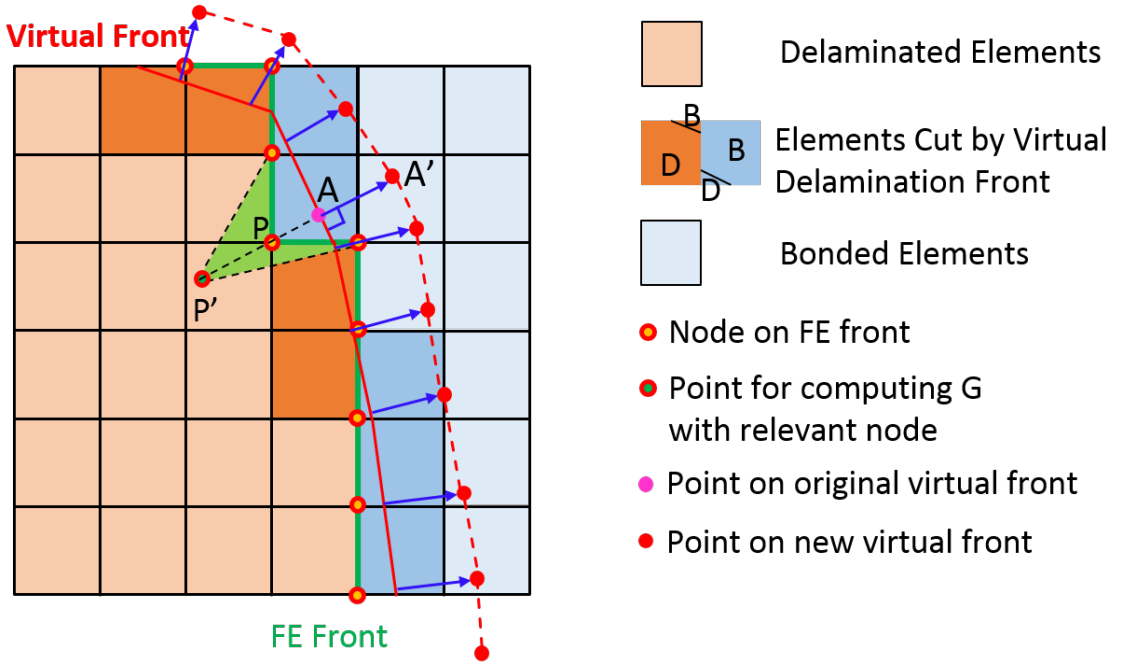

Fig. 1.12: Illustration of the virtual delamination front used in the GEM spring model (top view of interface).

1. The red line is the virtual front. Elements are determined to be debonded if they are behind the front and bonded if they are ahead of the front (elements described are those below the interface).

   (a) Elements cut by the front are determined to be bonded based on the element area traversed by the front. For example, if over half of the element is traversed by the front then the element is considered debonded (shown as dark orange) otherwise it is bonded (dark blue).

   (b) Bonded elements have springs placed at all their nodes.

   (c) Nodes that are shared by bonded and debonded elements become the FE front (green line).

2. The FE front is used to calculate G at the node DOF using the VCCT approach

   (a) A force is extracted from each point on the FE front (for example point P).

(b) The displacement comes from the point P'. Point P and point P' lie on a line normal to the virtual front which originates at point A. Point P' is usually an element length behind point P but can be changed in the code. The displacement is interpolated from the four surrounding nodes.

(c) The surface area is determined by 3 points on the FE front along with point P'.

3. Once G is determined, it is projected to the virtual front (point A) where it determines the crack speed from the Paris law (crack speed is represented by the blue vectors)

4. Point A' is determined from point A based on the value of the crack increment (normally an element length) scaled by the crack speed. Point A' lies on the line normal to the front at point A.

5. All the points to describe the new virtual front are determined (red), and the next increment of crack growth uses the new virtual front.

1.5 Importance of Accurate G Calculations in Fatigue Failure Analysis

It is important that the calculation of G is accurate. The Paris law is discussed in Sec. 1.4 (Sec. 1.4.2) as part of the methods for spring models that are studied in this thesis. It is used for fatigue analysis and life prediction. Error in G is compounded in the fatigue analysis after many increments of crack growth. An exponent exists in the Paris law that also compounds the error of G.

For each increment of crack growth in a finite element analysis, G determines the crack speed at every point on the delamination front. The point with the highest G is where the delamination is propagated the furthest (specified increment of crack growth). The number of cycles is determined from the crack growth and the crack speed. A new G is calculated at every increment of crack growth. If G is not accurate, then it affects the calculated number of cycles increasingly during each increment. In this thesis, the accuracy of G is determined with analytical solutions of specific types of cracks.

A critical issue with the GEM spring model is that a FE front (true front approximated by nodes) is very different from the true front. Another issue is that G is approximated at the true front by a close node on the finite element mesh (FE front). These issues cause errors in the G calculation.

1.6 Summary

Fiber composites make up a significant number of products in industry. They are advantageous because of their ability to have directional strength which allows them to be light-weight. They are used for sports, medical, energy, military, and aerospace applications. Composites can resist impact, but their overall strength is lessened. Delaminations that begin to grow at the interfaces of fiber plies are a major failure mechanism. Delamination can be analyzed with energy release rate (G), but the analysis methods available are not always accurate, efficient, or easy.

Using a finite element approach, many methods require that the finite element mesh be geometry conforming to the delamination front. In many cases, this is difficult because of the complicated geometry. In propagation, it is computationally expensive to re-mesh, and errors in G are compounded. A solution for this problem is a new spring model that places stiff springs at the delamination front (and behind) and calculates G based on the VCCT approach (Sec. 1.4.1). A new interface element is required that uses shape functions to move the springs to the delamination front location to output accurate forces and displacements. An accurate G calculation along the delamination front is critical to having an accurate propagation analysis.

CHAPTER 2

RESEARCH OBJECTIVES

- Develop a user-element subroutine (UEL), as part of a new framework, that simplifies the G calculation by eliminating the FE front (Chap. 3)
- Verify the accuracy of energy release rate (G) in the GEM spring model (described in Sec. 1.4) with analytical solutions, VCCT, and/or J-integral (Sec. 4.1)
- Verify that the new framework has close accuracy of G (Sec. 4.2)
- Compare new framework with GEM framework (Sec. 4.3)
- Study the effect of a mesh-independent energy release rate on the predicted fatigue life (Sec. 4.4)

CHAPTER 3

NEW FRAMEWORK FOR EVALUATION OF G WITHOUT FE FRONT

A new model was created, to increase the accuracy of G. This model is like the GEM model in that forces are extracted from spring elements to solve for G using the VCCT approach. The new model requires an 8-node interface element, rather than a 2-node element, to approximate displacements between nodes using shape functions (Fig. 3.2). These shape functions simulate a spring in the middle of the element by distributing the stiffness to the nodes (there are no added DOF in the model). A stiffness matrix is required in the finite element analysis, and it is created from the distributed stiffness and the original nodal stiffness. The force is then extracted from the simulated spring location, with the approximated stiffness and displacement, to calculate G. The new framework does not currently include fatigue.

Fig. 3.1 shows a comparison of the VCCT calculation in the GEM spring model and the new spring model along with a normal VCCT approach. The thick black line is the delamination front and the region to the left of the front is the delaminated region. The area for the G-calculation is green (see Sec. 1.4.1 and 1.4.3)

	Abaqus VCCT	GEM Spring Model	New Spring Model
Force	From nodes on front	From nodes on FE front	At real front (not on nodes)
Displacement	At node	Determined by FE front	Determined by real front
Surface Area	Determined by element size at front	Determined from FE front	Determined by element edge spacing

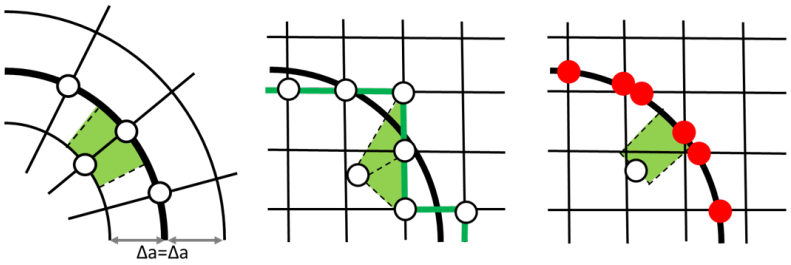

Fig. 3.1: A VCCT approach is used in both the GEM spring model and the new spring model. Forces, displacements, and areas are determined differently in each framework (see Sec. 1.4.1, 1.4.3, and 3.1).

3.1 An 8-Node Interface Element (Abaqus UEL)

The purpose of this 8-node interface element is to distribute a spring stiffness to surrounding nodes and get correct displacements anywhere in the element (see Sec. A.1 for the UEL subroutine outline). Spring forces are calculated using Hooke's law ($F = kx$ where k is the spring stiffness and x is the displacement). The spring at point P, in Fig. 3.2, could be placed on the true front to extract a force at the front. The spring P is only placed in the center of the element for demonstration; the new method adds springs to the element edges to cut the element (see Fig. 3.3). The P_{top} displacement is calculated using shape functions of the top 4-node element, and the P_{bot} displacement is calculated with the shape functions from the bottom 4-node element. The change in spring length is calculated by using the displacements of P_{top} and P_{bot}. The forces on point P are distributed to N1, N2, N3, and N4 by use of the shape functions.

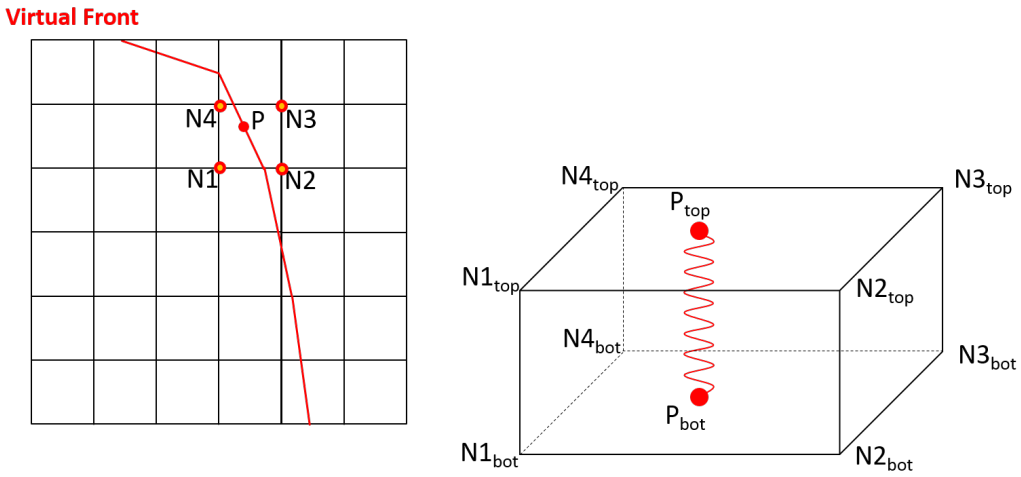

Fig. 3.2: Virtual springs are added to elements and are simulated by distributing the stiffness to the nodes of the top and bottom 4-node elements. It was decided to only place springs on the element edges as shown in Fig. 3.3.

A 2-node UEL (Abaqus user-defined element) cannot be used in the new model because the displacements from N1 to N4 are needed for interpolating the displacement of spring P (Fig. 3.2). Forces are needed at the crack front. Fig. 3.3 shows where the springs are inside of the new 8-node interface UEL.

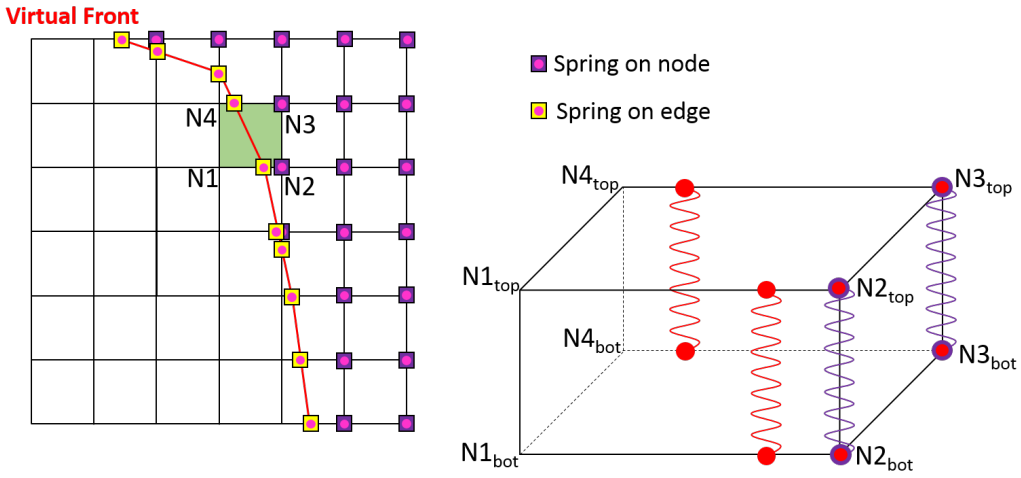

Fig. 3.3: Top and isometric views of 8-node interface element (Abaqus UEL) where springs are added to the element edges and along the delamination front.

3.2 Stiffness Matrix from Assigned Spring Stiffness

Abaqus UEL requires that a stiffness matrix ($AMATRX$) and a residual force vector (RHS) be defined for a static analysis (Eq. 3.1). The U displacement vector is solved in the analysis. Natural coordinates are required to distribute the spring stiffness to nodes and create the stiffness matrix. Node springs and springs on the front are assigned stiffness that are consistent with the overall stiffness of the interface.

$$RHS = -AMATRX \times U \tag{3.1}$$

A stiffness matrix K_{front} contains stiffnesses that have been distributed to nodes from the front springs (on the delamination front). It is added to the nodal stiffness matrix K_{nodal} to create the element stiffness matrix (Eq. 3.2). K_{nodal} is a 24x24 matrix because each node has three degrees of freedom (Eq. 3.3).

$$AMATRX = K_{nodal} + K_{front} \tag{3.2}$$

$$K_{nodal} = \begin{bmatrix}
k1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -k1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & k1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -k1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & k1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -k1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & k2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -k2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & k2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -k2 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & k2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -k2 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & k3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -k3 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & k3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -k3 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & k3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -k3 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & k4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -k4 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & k4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -k4 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & k4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-k1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & k1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -k1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & k1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -k1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & k1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -k2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & k2 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -k2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & k2 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -k2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & k2 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -k3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & k3 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -k3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & k3 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -k3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & k3 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -k4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & k4 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -k4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -k4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix} \tag{3.3}$$

K_{nodal} is based on Hooke's law for a simple spring ($F = kx$). Fig. 3.4 shows the displacements that correspond to each spring. To find the force on the $k2$ spring in the z-direction, the stiffness $k2$ is multiplied by the difference in displacement of the node on top and bottom of the spring; the force is $k2 \times (U(18) - U(6))$ on the top node, and $k2 \times (U(6) - U(18))$ on the bottom node (note that the residual force is the negative value of this force (Eq. 3.1)).

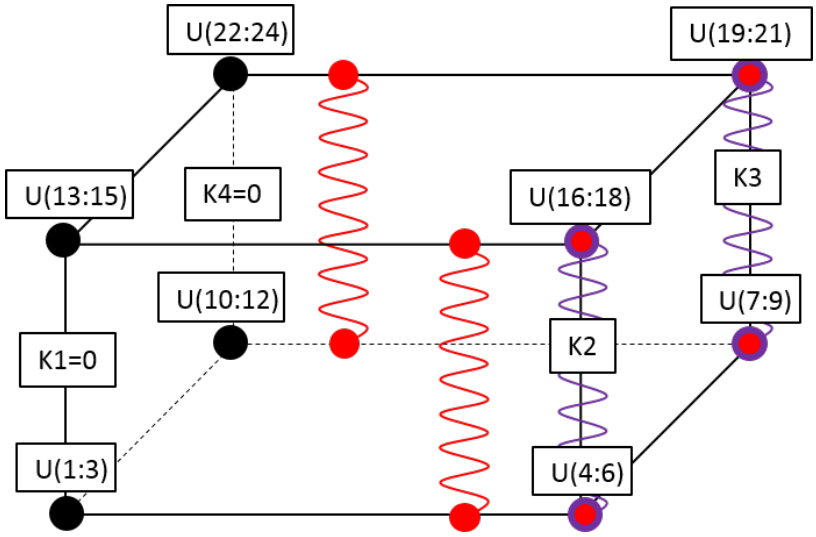

Fig. 3.4: Each node in the interface element has 3 DOF and the stiffness and displacements shown make up the U vector and the K_{nodal} matrix.

K_{front} was derived in the same way that consistent nodal loads are solved (found in finite element texts such as [43]). A nodal reaction force is equal to its corresponding shape function multiplied by the force applied to the element; the shape function is evaluated at the coordinates of the applied force. The reaction force vector is equal to the transpose of its shape function vector multiplied by the applied force.

In the 8-node element formulation, a force is applied to the element by means of a spring. Each spring applies a force to the top and bottom of the element. Multiple forces applied internally require a shape function matrix N instead of a vector. F' is the force vector simulated at the front springs and is given by Eq. 3.4 (prime (') means associated with a front spring). The displacement at a front spring U' is in terms of the nodal displacements U because it is interpolated. K' is a stiffness matrix formed in the same manner as K_{nodal} but with only two springs which are labeled $k1'$ and $k2'$ (see Eq. 3.5).

$$\begin{aligned} F' &= K'U' \\ F' &= K'NU \end{aligned} \tag{3.4}$$

$$K' = \begin{bmatrix} k1' & 0 & 0 & 0 & 0 & 0 & -k1' & 0 & 0 & 0 & 0 & 0 \\ 0 & k1' & 0 & 0 & 0 & 0 & 0 & -k1' & 0 & 0 & 0 & 0 \\ 0 & 0 & k1' & 0 & 0 & 0 & 0 & 0 & -k1' & 0 & 0 & 0 \\ 0 & 0 & 0 & k2' & 0 & 0 & 0 & 0 & 0 & -k2' & 0 & 0 \\ 0 & 0 & 0 & 0 & k2' & 0 & 0 & 0 & 0 & 0 & -k2' & 0 \\ 0 & 0 & 0 & 0 & 0 & k2' & 0 & 0 & 0 & 0 & 0 & -k2' \\ -k1' & 0 & 0 & 0 & 0 & 0 & k1' & 0 & 0 & 0 & 0 & 0 \\ 0 & -k1' & 0 & 0 & 0 & 0 & 0 & k1' & 0 & 0 & 0 & 0 \\ 0 & 0 & -k1' & 0 & 0 & 0 & 0 & 0 & k1' & 0 & 0 & 0 \\ 0 & 0 & 0 & -k2' & 0 & 0 & 0 & 0 & 0 & k2' & 0 & 0 \\ 0 & 0 & 0 & 0 & -k2' & 0 & 0 & 0 & 0 & 0 & k2' & 0 \\ 0 & 0 & 0 & 0 & 0 & -k2' & 0 & 0 & 0 & 0 & 0 & k2' \end{bmatrix} \tag{3.5}$$

The vector containing front spring displacements U' is pre-multiplied by the same shape function matrix N used in consistent nodal loads (Eq. 3.6). This is how the displacements are interpolated. The displacements in the vector U' can be visualized in Fig. 3.5.

$$U' = NU \tag{3.6}$$

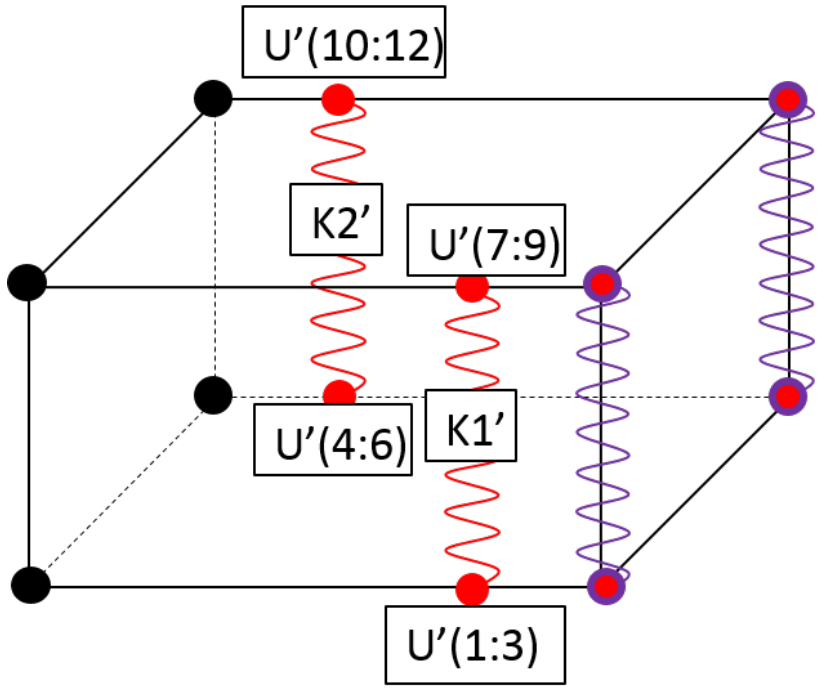

Fig. 3.5: The displacements on top and bottom of the front springs make up the U' vector as shown ($U' = NU$). The K' Matrix is made of the spring stiffnesses $k1'$ and $k2'$ (Eq. 3.5).

The springs, inside the element, distribute forces to the nodes as shown in Eq. 3.7 (RHS is the residual force vector). Again, using Hooke's law (constitutive equation), force equals a stiffness matrix multiplied by a displacement vector. The matrix K_{front} has been created.

$$\begin{aligned} RHS &= -K_{nodal}U - N^T K' NU \\ RHS &= -K_{nodal}U - K_{front}U \end{aligned} \tag{3.7}$$

The stiffness matrix from the front springs K' is pre- and post-multiplied by a shape function matrix N to create K_{front}. This multiplication allows addition of the two stiffness matrices K_{nodal} and K_{front}. Unlike K_{nodal}, the K_{front} matrix is nearly full because internal displacements are in terms of the nodal displacements. However, K_{front} is not as full when

springs are placed on element edges because their displacements are only interpolated from two nodes.

$$K_{front} = N^T K' N \tag{3.8}$$

The shape functions distribute the front spring stiffness to the nodes (see Eqs. 3.9 and 3.10 for the shape function matrix N and its transpose N^T). In the N matrix, $N_1(2')$ is the first shape function (N_1) evaluated at the natural coordinates of the second front spring ($2'$). Other values in the shape function are similarly calculated. Natural coordinates must be converted from global coordinates before the $AMATRX$ can be calculated.

$$N = \begin{bmatrix}
N_1(1') & 0 & 0 & N_2(1') & 0 & 0 & N_3(1') & 0 & 0 & N_4(1') & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & N_1(1') & 0 & 0 & N_2(1') & 0 & 0 & N_3(1') & 0 & 0 & N_4(1') & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & N_1(1') & 0 & 0 & N_2(1') & 0 & 0 & N_3(1') & 0 & 0 & N_4(1') & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
N_1(2') & 0 & 0 & N_2(2') & 0 & 0 & N_3(2') & 0 & 0 & N_4(2') & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & N_1(2') & 0 & 0 & N_2(2') & 0 & 0 & N_3(2') & 0 & 0 & N_4(2') & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & N_1(2') & 0 & 0 & N_2(2') & 0 & 0 & N_3(2') & 0 & 0 & N_4(2') & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & N_1(1') & 0 & 0 & N_2(1') & 0 & 0 & N_3(1') & 0 & 0 & N_4(1') & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & N_1(1') & 0 & 0 & N_2(1') & 0 & 0 & N_3(1') & 0 & 0 & N_4(1') & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & N_1(1') & 0 & 0 & N_2(1') & 0 & 0 & N_3(1') & 0 & 0 & N_4(1') \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & N_1(2') & 0 & 0 & N_2(2') & 0 & 0 & N_3(2') & 0 & 0 & N_4(2') & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & N_1(2') & 0 & 0 & N_2(2') & 0 & 0 & N_3(2') & 0 & 0 & N_4(2') & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & N_1(2') & 0 & 0 & N_2(2') & 0 & 0 & N_3(2') & 0 & 0 & N_4(2')
\end{bmatrix} \tag{3.9}$$

$$N^T = \begin{bmatrix}
N_1(1') & 0 & 0 & N_1(2') & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & N_1(1') & 0 & 0 & N_1(2') & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & N_1(1') & 0 & 0 & N_1(2') & 0 & 0 & 0 & 0 & 0 & 0 \\
N_2(1') & 0 & 0 & N_2(2') & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & N_2(1') & 0 & 0 & N_2(2') & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & N_2(1') & 0 & 0 & N_2(2') & 0 & 0 & 0 & 0 & 0 & 0 \\
N_3(1') & 0 & 0 & N_3(2') & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & N_3(1') & 0 & 0 & N_3(2') & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & N_3(1') & 0 & 0 & N_3(2') & 0 & 0 & 0 & 0 & 0 & 0 \\
N_4(1') & 0 & 0 & N_4(2') & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & N_4(1') & 0 & 0 & N_4(2') & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & N_4(1') & 0 & 0 & N_4(2') & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & N_1(1') & 0 & 0 & N_1(2') & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & N_1(1') & 0 & 0 & N_1(2') & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & N_1(1') & 0 & 0 & N_1(2') \\
0 & 0 & 0 & 0 & 0 & 0 & N_2(1') & 0 & 0 & N_2(2') & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & N_2(1') & 0 & 0 & N_2(2') & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & N_2(1') & 0 & 0 & N_2(2') \\
0 & 0 & 0 & 0 & 0 & 0 & N_3(1') & 0 & 0 & N_3(2') & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & N_3(1') & 0 & 0 & N_3(2') & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & N_3(1') & 0 & 0 & N_3(2') \\
0 & 0 & 0 & 0 & 0 & 0 & N_4(1') & 0 & 0 & N_4(2') & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & N_4(1') & 0 & 0 & N_4(2') & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & N_4(1') & 0 & 0 & N_4(2')
\end{bmatrix} \tag{3.10}$$

3.2.1 Transformation from Global Coordinates to Natural Coordinates

For any point on the front, global coordinates are known. There are eight values in Eq. 3.9 that must be solved; these are shape functions for a Q4 element. This matrix is used in the calculation of Kfront (Eq. 3.8) which represents spring stiffnesses distributed to the element nodes. Each element can be represented by natural coordinates so that the interpolating shape functions are the same for every element. An approach is needed to convert the known global coordinates to natural coordinates to solve the value of each shape function.

The natural coordinates must be known to create an accurate shape function matrix. A Q4 element has bilinear shape functions. A computational method was developed to convert global coordinates into natural coordinates. This method does not require an optimization and/or an iteration method (line search and steepest descent methods have been used in other analysis software).

Shape functions are in terms of natural coordinates. Shape functions in Eq. 3.11 range between 0 and 1 and natural coordinates range between -1 and 1. When $\xi = 1$ and $\eta = 1$,

$N_3 = 1$ and $N_1 = N_2 = N_4 = 0$. The Q4 shape functions can be visualized in Fig. 3.6.

$$
\begin{aligned}
N_1 &= \frac{(1-\xi)(1-\eta)}{4} \\
N_2 &= \frac{(1+\xi)(1-\eta)}{4} \\
N_3 &= \frac{(1+\xi)(1+\eta)}{4} \\
N_4 &= \frac{(1-\xi)(1+\eta)}{4}
\end{aligned}
\tag{3.11}
$$

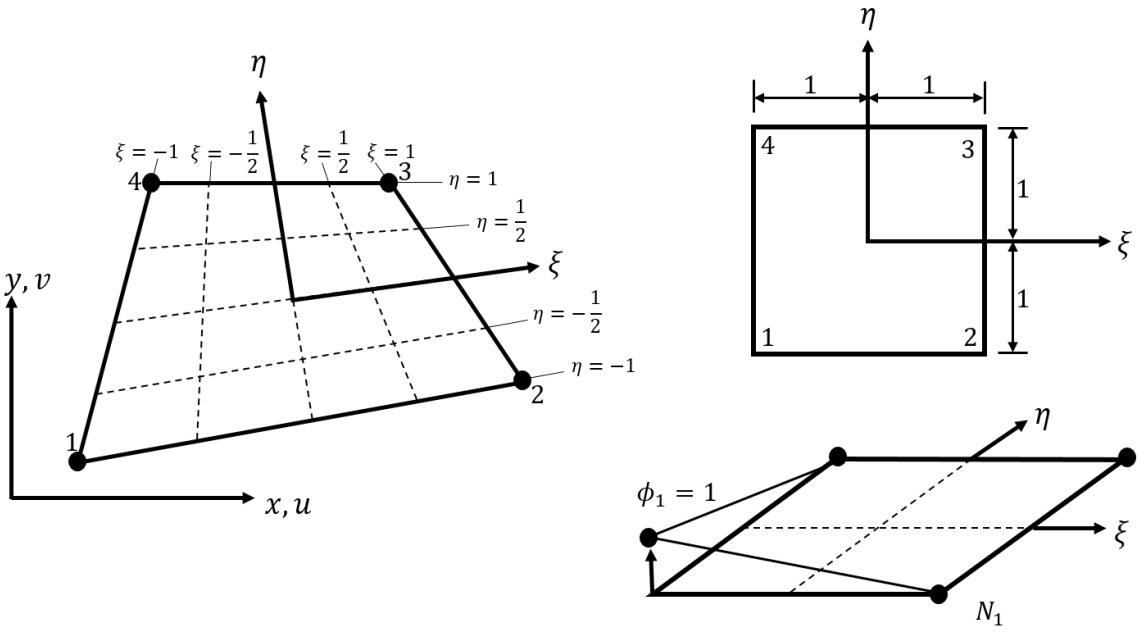

Fig. 3.6: Q4 Shape functions are bilinear (x and y are global coordinates and ξ and η are natural coordinates).

The method for changing global coordinates to natural coordinates is setting slopes equal (Fig. 3.7). The bilinear shape functions make it possible to set one of the natural coordinates to be constant to create a straight line. Three points are used to find the line slopes. ξ is calculated in one of three ways depending on the orientation of the element in the global coordinate system.

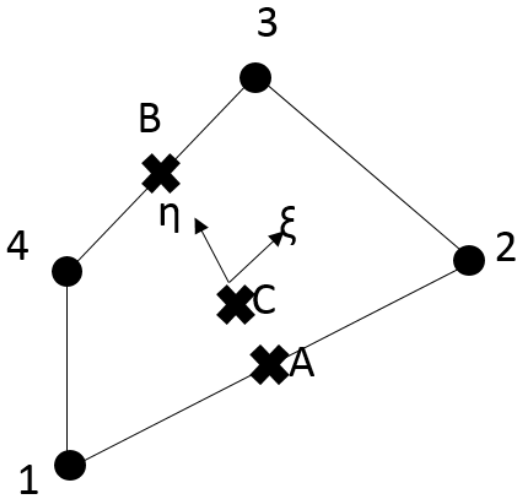

Fig. 3.7: The slopes of AC and BC are equal for a bilinear problem.

The method for vertical slopes uses the slopes from nodes 2 to 3 and nodes 1 to 4 as shown in Eq. 3.12 (Fig. 3.7). The global coordinates for point 1 are x_1 and y_1. The coordinates for other points are labeled likewise.

$$\begin{aligned} m_{23} &= \frac{y_3 - y_2}{x_3 - x_2} \\ m_{14} &= \frac{y_4 - y_1}{x_4 - x_1} \end{aligned} \tag{3.12}$$

If slopes in Eq. 3.12 are vertical, then $x_3 - x_2 = x_4 - x_1 = 0$ and ξ is calculated from Eq. 3.13. This equation uses the shape functions from Eq. 3.11 and $x = \sum_{i=1}^{4} N_i x_i$ (Eq. 3.14) where x is the x-value (global coordinate) of point A and point C in Fig. 3.7. Note that two of the shape functions equal zero because point A is on an element edge.

$$\xi = \frac{2(x - x_1)}{x_2 - x_1} - 1 \tag{3.13}$$

$$x = \sum_{i=1}^{4} N_i x_i$$
$$y = \sum_{i=1}^{4} N_i y_i \tag{3.14}$$

ξ is calculated from Eqs. 3.15 and a quadratic equation if the slopes m_{23} and m_{14} are not vertical. Again, the Eq. 3.14 is used to find the x and y global coordinates of points A and B in Fig. 3.7 (A_x, A_y, B_x, and B_y in Eq. 3.15). The equation has been simplified to isolate ξ.

$$\begin{aligned} A_x &= \frac{(x_2 + x_1)}{2} + \frac{(x_2 - x_1)}{2}\xi = A_{1x} + A_{2x}\xi \\ B_x &= \frac{(x_3 + x_4)}{2} + \frac{(x_3 - x_4)}{2}\xi = B_{1x} + B_{2x}\xi \\ A_y &= \frac{(y_2 + y_1)}{2} + \frac{(y_2 - y_1)}{2}\xi = A_{1y} + A_{2y}\xi \\ B_y &= \frac{(y_3 + y_4)}{2} + \frac{(y_3 - y_4)}{2}\xi = B_{1y} + B_{2y}\xi \end{aligned} \tag{3.15}$$

Using Eq. 3.15 and setting the slopes equal (Eq. 3.16), Eq. 3.17 is derived. The coordinates x and y, in Eqs. 3.17 and 3.16, are the global coordinates of point C. The variables A, B, and C in Eq. 3.17 are new constants from Eq. 3.16 in the form $A\xi^2 + B\xi + C\xi = 0$.

$$\begin{aligned} m_{AC} &= \frac{y - A_y}{x - A_x} \\ m_{CB} &= \frac{B_y - y}{B_x - x} \end{aligned} \tag{3.16}$$

$$\begin{aligned} C &= (B_{1x} - A_{1x})y + (A_{1y} - B_{1y})x + B_{1y}A_{1x} - B_{1x}A_{1y} \\ B &= (B_{2x} - A_{2x})y + (A_{2y} - B_{2y})x + B_{1y}A_{2x} + B_{2y}A_{1x} - A_{2y}B_{1x} - A_{1y}B_{2x} \\ A &= A_{2x}B_{2y} - A_{2y}B_{2x} \end{aligned} \tag{3.17}$$

If A from Eq. 3.17 is equal to zero, then ξ is calculated with Eq. 3.18. Otherwise, ξ is calculated from Eq. 3.19. The root between -1 and 1 is used.

$$\xi = \frac{-C}{B} \tag{3.18}$$

$$\xi = \frac{-B \pm \sqrt{B^2 - 4AC}}{2A} \tag{3.19}$$

After ξ is found, η is found using Eq. 3.22 (where the constants are solved in Eqs. 3.20 and 3.21). The constants in Eqs. 3.20 and 3.21 are derived from Eq. 3.14. The equation is long when all the shape functions are multiplied through; it is broken into 5 equations. Eq. 3.22 contains two equations, and if the denominator of the first one is equal to zero, then the second equation is used.

$$\begin{aligned}
c_1 &= c_2\eta + c_3\xi + c_4\xi\eta \\
c_1 &= 4x - \sum x_i \\
c_2 &= -x_1 - x_2 + x_3 + x_4 \\
c_3 &= -x_1 + x_2 + x_3 - x_4 \\
c_4 &= x_1 - x_2 + x_3 - x_4
\end{aligned} \tag{3.20}$$

$$\begin{aligned}
k_1 &= k_2\eta + k_3\xi + k_4\xi\eta \\
k_1 &= 4y - \sum y_i \\
k_2 &= -y_1 - y_2 + y_3 + y_4 \\
k_3 &= -y_1 + y_2 + y_3 - y_4 \\
k_4 &= y_1 - y_2 + y_3 - y_4
\end{aligned} \tag{3.21}$$

$$\eta = \frac{c_1 - c_3\xi}{c_2 + c_4\xi} \quad \text{or} \quad \eta = \frac{k_1 - k_3\xi}{k_2 + k_4\xi} \tag{3.22}$$

3.2.2 Stiffness Assigned to Front Springs and Node Springs

Two approaches were created for assigning stiffness to springs (necessary for stiffness matrix). Both approaches have advantages and disadvantages, but future work could improve stiffness assignment. Spring stiffness is assigned at the delamination front and at bonded nodes to keep the overall stiffness of the model accurate. Two springs are added to the edges of an element where that element is intersected by a delamination front. For solving intersection points, all element sides are parallel or perpendicular to the global axes and element sides are of equal length.

The algorithm for solving intersection points was created for a circular delamination; future improvement must be made to this algorithm for fatigue analysis. The circle equation is given in Eq. 3.23; x and y are global coordinates for a spring on an element edge and r is the radius of the circular delamination. The terms x_0 and y_0 are the coordinates at the center of the circle.

$$(x - x_0)^2 + (y - y_0)^2 = r^2 \tag{3.23}$$

Edges are made orthogonal to the global axes so that one of the coordinates remains constant. When an element is cut, the program loops through the edges to determine which edge is cut (one node debonded and the other bonded). The edge has one constant coordinate. If y is constant, x is evaluated from Eq. 3.24. If x is constant, y is solved using Eq. 3.25. These two equations are formed from the circle equation (Eq. 3.23). The correct solution to either of these equations is the one that provides coordinates between the node coordinates (the nodes that define the element edge).

$$x = x_0 \pm \sqrt{r^2 - (y - y_0)^2} \tag{3.24}$$

$$y = y_0 \pm \sqrt{r^2 - (x - x_0)^2} \tag{3.25}$$

There are three cases of cut elements as seen in Fig. 3.8. Areas are distributed to the nodal and front springs in a different way for each case. The input stiffness for each spring in the model has units of $Force/Length^3$. Every spring has an area associated with it to calculate a spring stiffness with units of $Force/Length$.

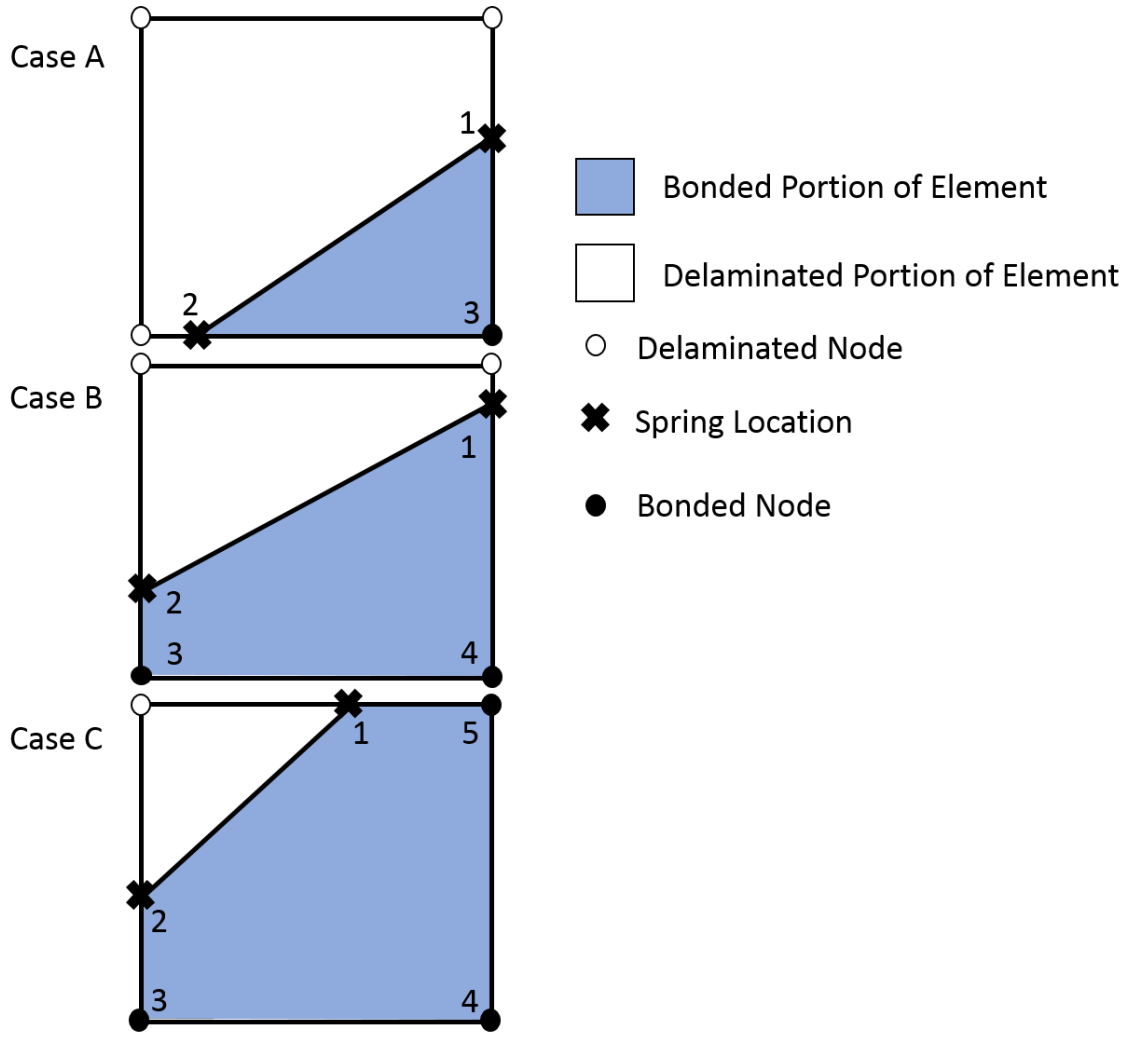

Fig. 3.8: Elements that are intersected by a delamination are cut. There are three cases for cut elements (3-point, 4-point, 5-point).

An initial approach to assigning stiffness has a discrepancy but shows approximately how area is distributed (Fig. 3.9). Area and stiffness were initially distributed by finding the centroid of the shape created by the internal springs and bonded nodes (midpoints between springs were also found). Two triangle areas were calculated and distributed to each spring and node in the bonded part of the element. It was found that there was a discrepancy in the stiffness distribution when different types of cut elements (3-point, 4-point, and 5-point cut elements) approached the same delamination shape.

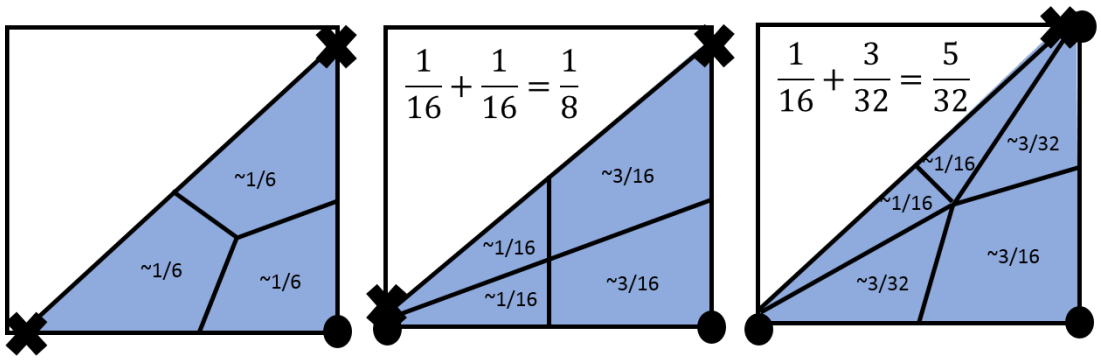

Fig. 3.9: A discrepancy is shown for the area distribution if the centroid is used (bonded portion of the element is blue).

Another area assignment approach was created that fixed the discrepancy. A visual representation cannot be easily made because it is completely based on equations. The stiffness/area distribution shown in Fig. 3.10 assumes that each corner of the element has a square area associated with it. It was used to formulate equations that are continuous as a 3-point or 5-point cut element approaches and becomes a 4-point cut element. Stiffness is assigned to the front springs and the bonded nodes, while delaminated nodes are assigned zero stiffness.

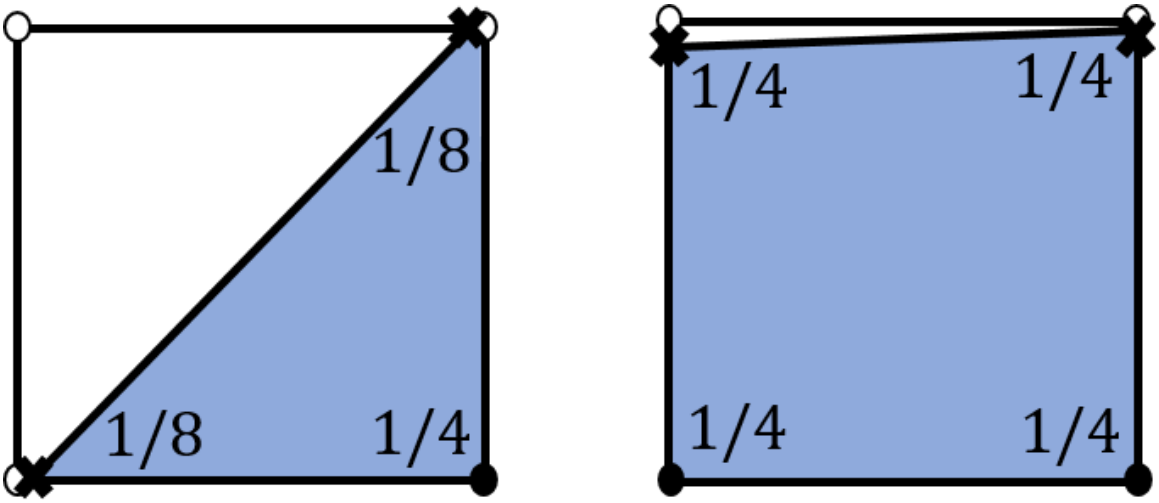

Fig. 3.10: As the bonded area (blue) approaches half of the element area (entire square), the stiffness adds up to half of the fully bonded stiffness. As the element approaches a fully bonded area, each corner is assigned one-fourth of the total stiffness.

For a 3-point cut element (Fig. 3.11), internal springs approach one-eighth stiffness as the bonded length (a or b) approaches the element length l (Eq. 3.26 and Fig. 3.10). $A_{spring,1}$ is the fraction of the element area assigned to the spring at point 1 in Fig. 3.11, and likewise $A_{spring,2}$ is the fraction assigned to point 2. $A_{node,3}$ is the area fraction distributed to the node at point 3. The total bonded area fraction is $x_1 x_2/2$. The stiffness of the springs at each of these points is equal to their area fraction multiplied by the total element stiffness of a fully bonded element. Stiffness, at each point, goes to zero as the element becomes fully debonded.

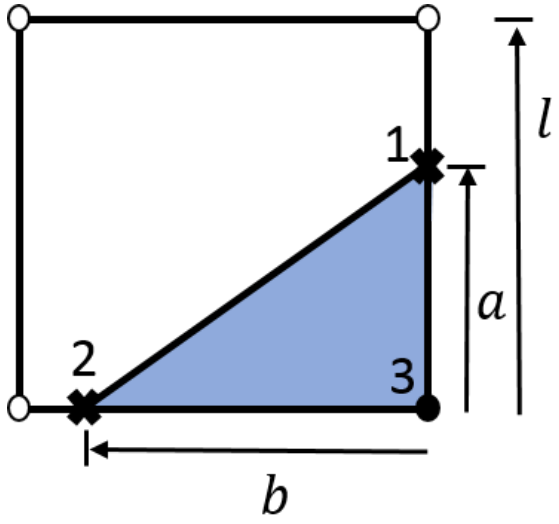

Fig. 3.11: Areas for a 3-point cut element are distributed with Eq. 3.26 (bonded portion of element is blue).

$$
\begin{aligned}
x_1 &= a/l \\
x_2 &= b/l \\
A_{spring,1} &= x_1^2 x_2/8 \\
A_{spring,2} &= x_2^2 x_1/8 \\
A_{node,3} &= (4x_1x_2 - x_1^2x_2 - x_2^2x_1)/8
\end{aligned}
\tag{3.26}
$$

The 4-point cut element is shown in Fig. 3.12 and its equations are shown in Eq. 3.27. The values of a and b are the bonded length and l is the element length. $A_{spring,1}$ and $A_{spring,2}$ are equal to the area fraction assigned to points 1 and 2 in Fig. 3.12. $A_{node,3}$ and $A_{node,4}$ are the area fractions distributed to the nodes at points 3 and 4, respectively. The stiffness of the springs at these points is determined from the fully bonded element total stiffness and the area fraction as described for the 3-point cut element. The total bonded area fraction is $(x_1 + x_2)/2$. A 4-point cut element can simulate an element that is near fully debonded or fully bonded as well as a triangle shape cut element (see Fig. 3.10).

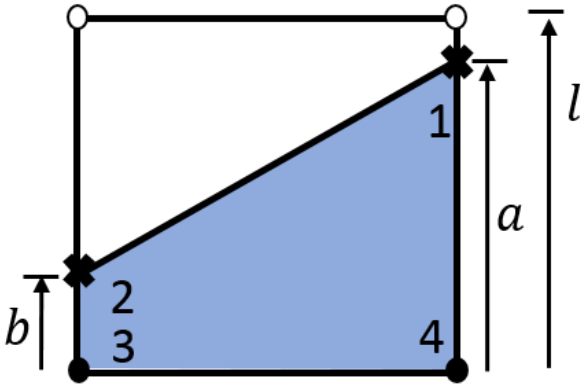

Fig. 3.12: Areas for a 4-point cut element are distributed with Eq. 3.27 (bonded portion of the element is blue).

$$
\begin{aligned}
x_1 &= a/l \\
x_2 &= b/l \\
A_{spring,1} &= (x_1 + x_2)/8 \\
A_{spring,2} &= (x_1 + x_2)/8 \\
A_{node,3} &= x_2/4 \\
A_{node,4} &= x_1/4
\end{aligned}
\tag{3.27}
$$

The 5-point cut element is shown in Fig. 3.13 and equations are shown in Eq. 3.28. The values of a and b are the bonded length and l is the element length. $A_{spring,1}$ and $A_{spring,2}$ are equal to the area fraction assigned to points 1 and 2 in Fig. 3.13. $A_{node,3}$, $A_{node,4}$, and $A_{node,5}$ are the area fractions distributed to the nodes at points 3, 4, and 5, respectively. The stiffness of the springs at these points is determined as described for the 3-point cut element. The total bonded area fraction is $\left(1 - \frac{(1-x_1)(1-x_2)}{2}\right)$. A 5-point cut element can simulate a fully bonded element or a triangle shape cut element (see Fig. 3.10).

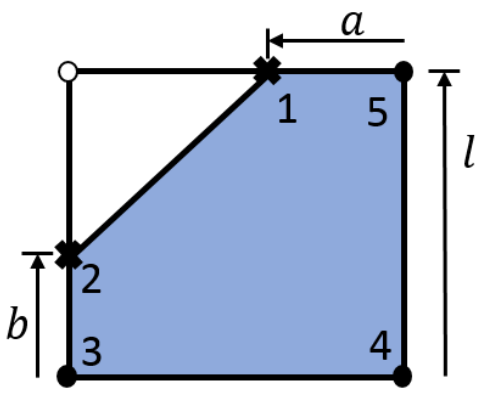

Fig. 3.13: Areas for a 5-point cut element are distributed with Eq. 3.28 (bonded portion of the element is blue).

$$
\begin{aligned}
x_1 &= a/l \\
x_2 &= b/l \\
A_{spring,1} &= \left(\frac{1}{2} - \frac{(1-x_1)(1-x_2)}{2}\right)\frac{x_1}{4} \\
A_{spring,2} &= \left(\frac{1}{2} - \frac{(1-x_1)(1-x_2)}{2}\right)\frac{x_2}{4} \\
A_{node,3} &= \left(\frac{1}{2} - \frac{(1-x_1)(1-x_2)}{2}\right)\frac{2-x_2}{4} + \frac{1}{8} \\
A_{node,4} &= \frac{1}{4} \\
A_{node,5} &= \left(\frac{1}{2} - \frac{(1-x_1)(1-x_2)}{2}\right)\frac{2-x_1}{4} + \frac{1}{8}
\end{aligned}
\tag{3.28}
$$

3.3 VCCT in New Spring Model

The VCCT calculation uses a force, displacement, and area to calculate energy release rate (Sec. 1.4.1 and Fig. 3.14). Fig. 3.14 illustrates a normal VCCT method (that has a geometry conforming mesh) to be compared with a similar VCCT method used in the new mesh-independent spring model.

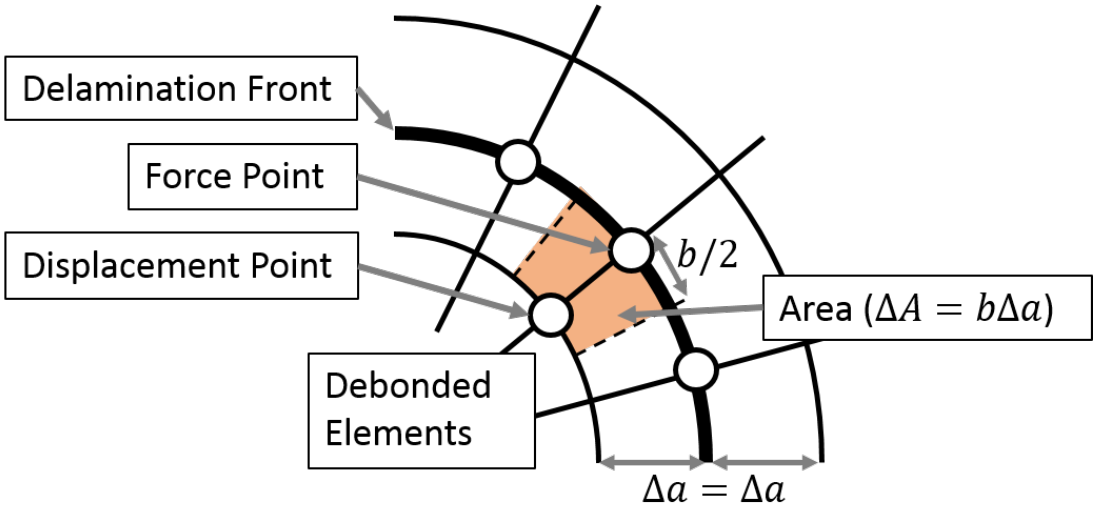

Fig. 3.14: VCCT calculation that could be used for a geometry conforming mesh.

Fig. 3.15 illustrates how the new UEL calculates G using a VCCT method. The spring location, where the force is extracted, is not located on a node. The natural coordinates, of the force point, are needed for nodal stiffness distribution and to determine the displacement at that point (force is this displacement multiplied by spring stiffness). Natural coordinates of the displacement point are needed for the same reason. The global to natural coordinate subroutine described in Sec. 3.2.1 must be used twice for each G calculation point.

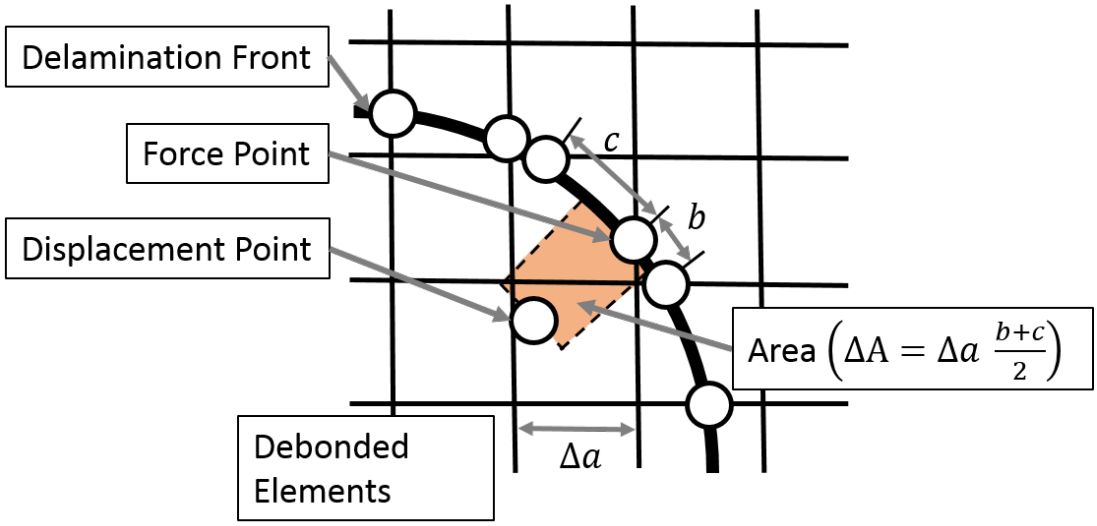

Fig. 3.15: VCCT calculation used in the new interface element (Abaqus UEL). Compare with Fig. 3.14.

CHAPTER 4
RESULTS AND DISCUSSION

4.1 Evaluation of GEM Spring Model

The GEM spring model is efficient in calculating energy release rate (G). The accuracy of the G calculation was determined by comparing it with analytical models and other finite element models. The purpose of the GEM spring model is to calculate G independently of the finite element mesh (for fatigue life predictions). The new framework does not currently include fatigue and that is a plan for future work. For verification, isotropic materials were used (with elastic properties like those of steel). The modulus of elasticity was set to $161,000\ N/mm^2$ and Poisson's ratio was set to 0.3 for all models.

4.1.1 Circular Crack in Infinite Solid (Mode I Crack)

The circular crack was used to solve mode I energy release rate in an infinite solid. Two simplifications were made for computational efficiency: a large cylinder was used instead of an infinite solid and symmetry was used to model only a quarter of the crack. The model shown in Fig. 4.1 is compared with the analytical solution of G for an infinite cylinder which is given in Eq. 4.1 [49]. The load applied to this model was a tensile stress $\sigma = 10\ N/mm^2$. The crack radius was $a = 8\ mm$, Poisson's ratio was $\nu = 0.3$, and the modulus of elasticity was $E = 161,000\ N/mm^2$. The cylinder had a radius of 50 mm and a height of 100 mm. The analytical value for the problem is $G_{I0} = 0.005757\ N/mm$ ($G_{I0} = 5.757\ J/m^2 = 5.757 \times 10^{-6}\ J/mm^2$).

$$G_{I0} = \frac{4\sigma^2 a}{\pi E}(1-\nu^2) \tag{4.1}$$

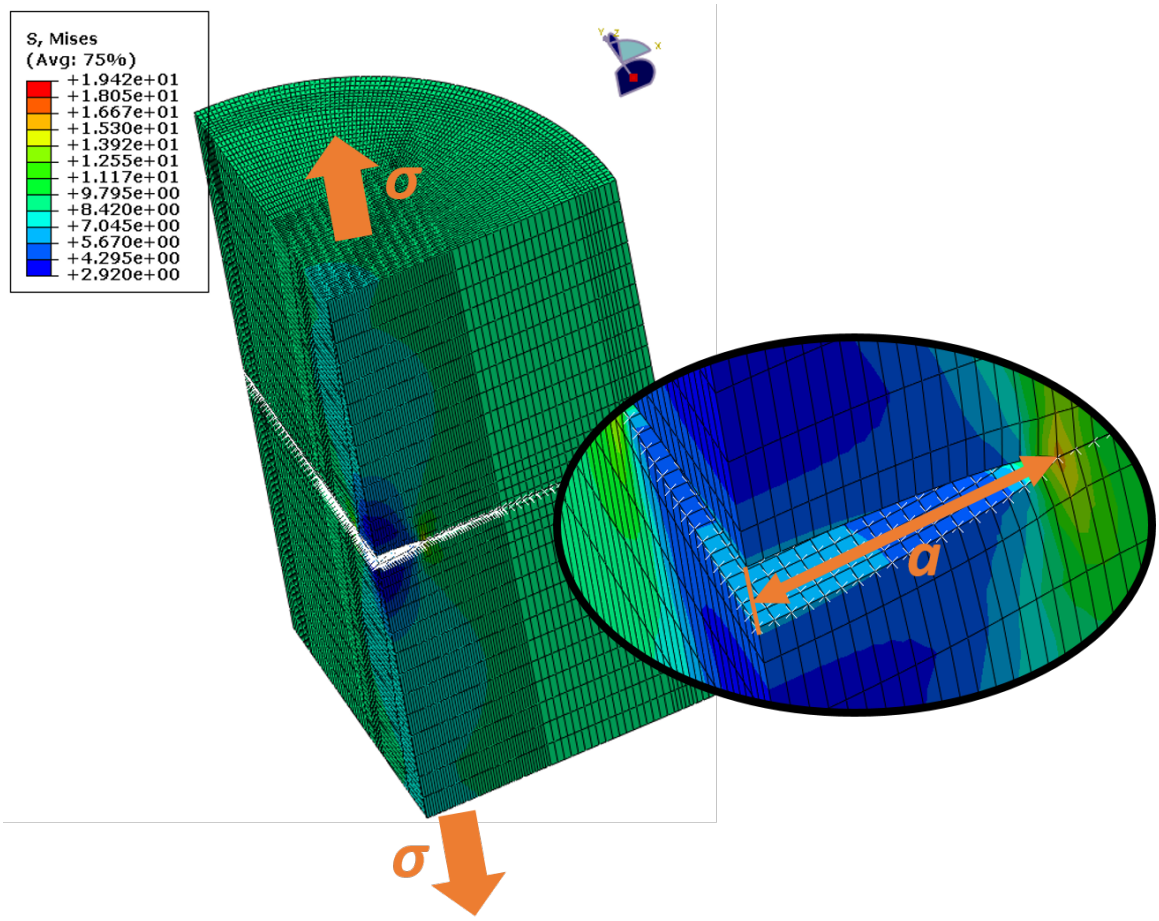

Fig. 4.1: The quarter cylinder model used to evaluate a circular and elliptical crack.

First, VCCT and spring models were compared using a geometry conforming mesh (see Fig. 4.2). This was expected to calculate G most accurately because the delamination is on nodes of the mesh, where G is calculated. Fig. 4.2 shows that calculated G values are constant around the delamination. The VCCT analysis time was ∼2300 seconds and the spring model analysis time was ∼400 seconds.

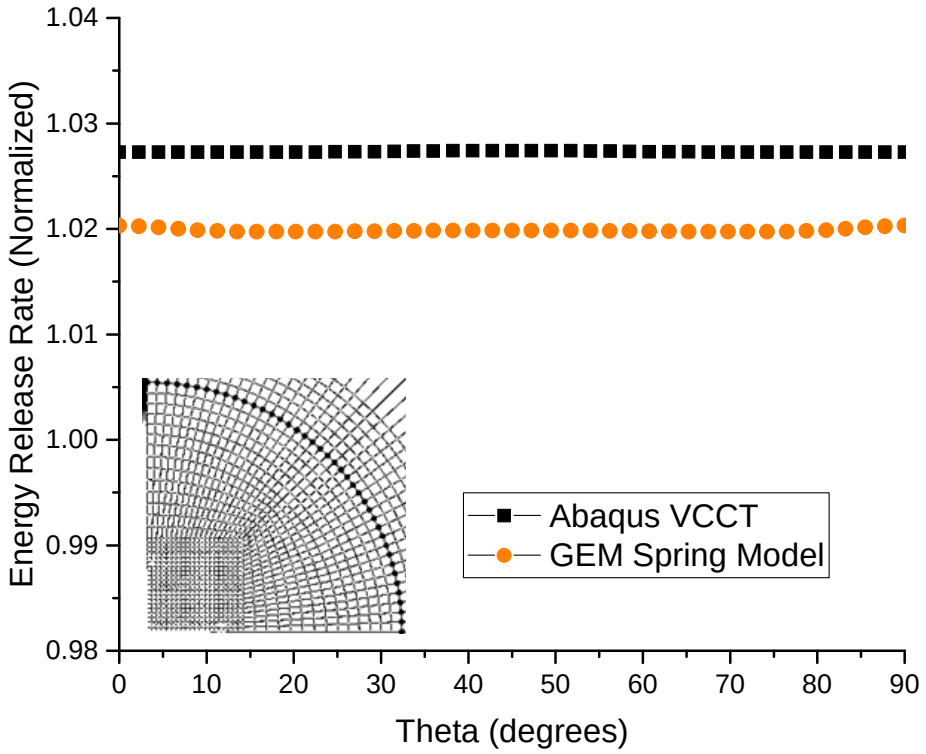

Fig. 4.2: The GEM spring model is compared with Abaqus VCCT for a circle crack with a geometry conforming mesh (normalized with the analytical solution Eq. 4.1).

The second model used for comparison had a square mesh with a circular delamination that was approximated by close nodes. Fig. 4.3 reveals that a square mesh has less accurate calculations than the geometry conforming mesh and Abaqus VCCT still requires more computation time (VCCT ∼22,000 seconds vs. spring model ∼1500 seconds). The GEM spring model uses a smoothing function to improve the energy release rate prediction, and it is unknown if Abaqus VCCT has a smoothing function. Negative (compressive) forces do not open cracks, and they occur in front of an opening crack to counteract the high tensile forces at the crack front. In the GEM model, negative values of G are set to zero.

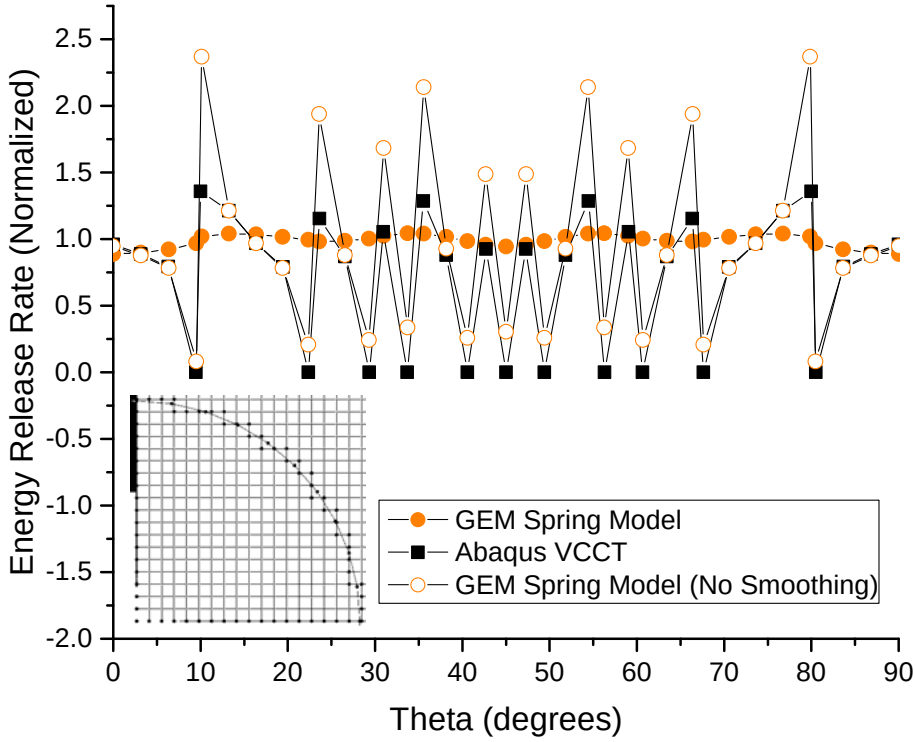

Fig. 4.3: The GEM spring model is compared with Abaqus VCCT for a circle crack with a square mesh (normalized with the analytical solution Eq. 4.1).

4.1.2 Elliptical Crack in Infinite Solid (Mode I Crack)

An elliptical crack solution was normalized with a circular crack solution as in [1]. The analytical solution of G is given in Eq. 4.2 where a and c are minor and major radii, respectively [49]. The model was the same as described in Sec. 4.1.1, but the major radii was changed ($c = 16\ mm$) while the minor radius remained the same ($a = 8\ mm$). In [1], a solution was proposed to improve the mesh-independent energy release rate calculation. This zig-zag solution was compared with the GEM spring model solution (see Fig. 4.4 where all values are normalized with the circular crack solution). The zig-zag method, in [1], does not place any springs behind the delamination front. This is unlike the GEM spring model

that places springs based on the area traversed by the delamination. This zig-zag method only calculates G where the delamination front is very near a node. Although this may improve the G values along the front, this method is less mesh-independent.

$$G_{I0} = \frac{\pi(1-\nu^2)\sigma^2}{E\Phi^2}\left(\frac{a}{c}\right)(a^2 cos^2\theta + c^2 sin^2\theta)^{\frac{1}{2}}$$
$$\Phi = \int_0^{\pi/2}\left(sin^2\theta + \left(\frac{a}{c}\right)^2 cos^2\theta\right)^{\frac{1}{2}} d\theta \tag{4.2}$$

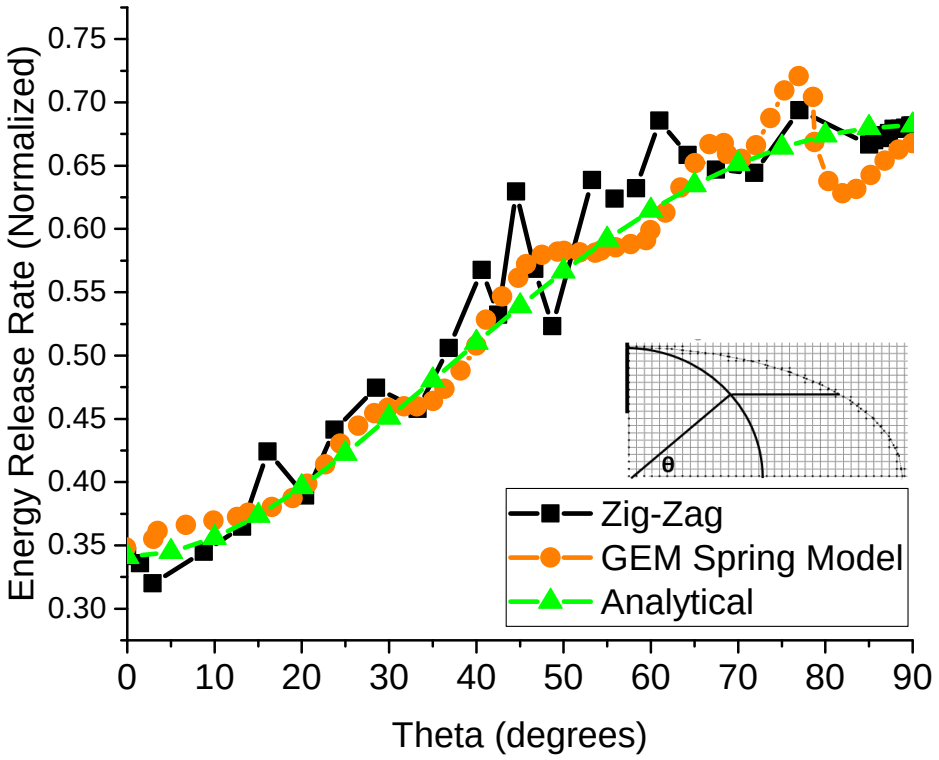

Fig. 4.4: The GEM spring model is compared with the analytical solution and a zig-zag method from [1] (all are normalized with the circular crack solution [1]).

4.1.3 Mode II Crack

The mode II (sliding shear) energy release rate for a triple-cantilever beam (TCB) was compared with J-integral and the GEM spring model (Fig. 4.5). Cantilever beams are often associated with bending forces, but there are no bending forces in this beam. J-integral is an integral that is used to calculate energy per unit of new surface area (energy release rate) [13]. The mesh can be complex for J-integral because it converges better when there is a circular path around the crack front. The analytical plane strain solution (Eq. 4.3) was derived by using the compliance form of the constitutive equation; it is not a function of the crack length c. For comparison, the load was $P = 10,000\ N$, the thickness was $B = 25\ mm$, the height was $H = 24\ mm$, and the modulus of elasticity was $161,000\ N/mm^2$. The computed analytical solution of G was $G_{II0} = 0.0207\ N/mm$. The TCB had a crack length of $c = 25\ mm$ and a beam length of $L = 100\ mm$.

$$G_{II0} = \frac{P^2}{2B^2HE} \tag{4.3}$$

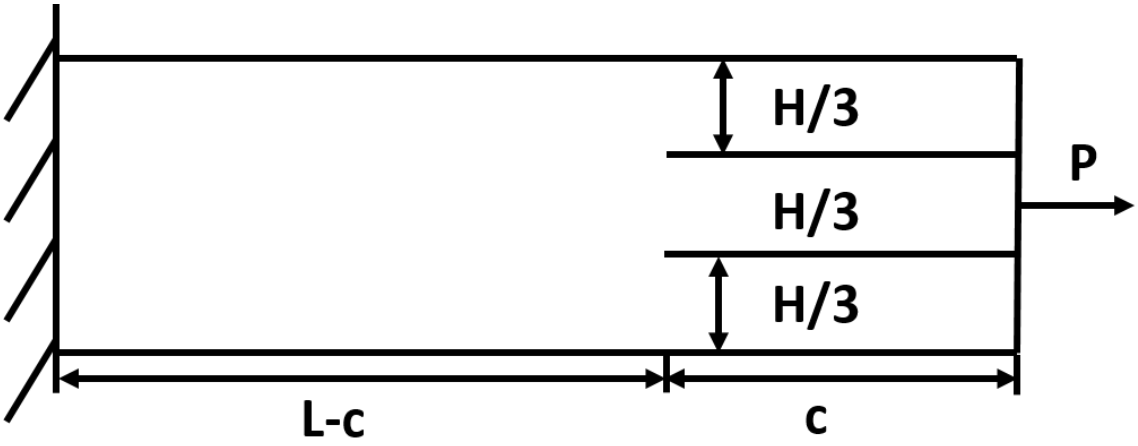

Fig. 4.5: Triple-cantilever beam (TCB) is used for mode II energy release rate calculation and comparison.

The mode II energy release rate is predicted well (Fig. 4.6); in this example, the mesh

conforms to the geometry of the delamination. Fig. 4.6 shows that energy release rate (G) calculation diverges from the analytical solution near the edges of the beam where there is no longer plane strain. The GEM spring model has a smoothing function that makes its G calculation less accurate than the J-integral at the edges.

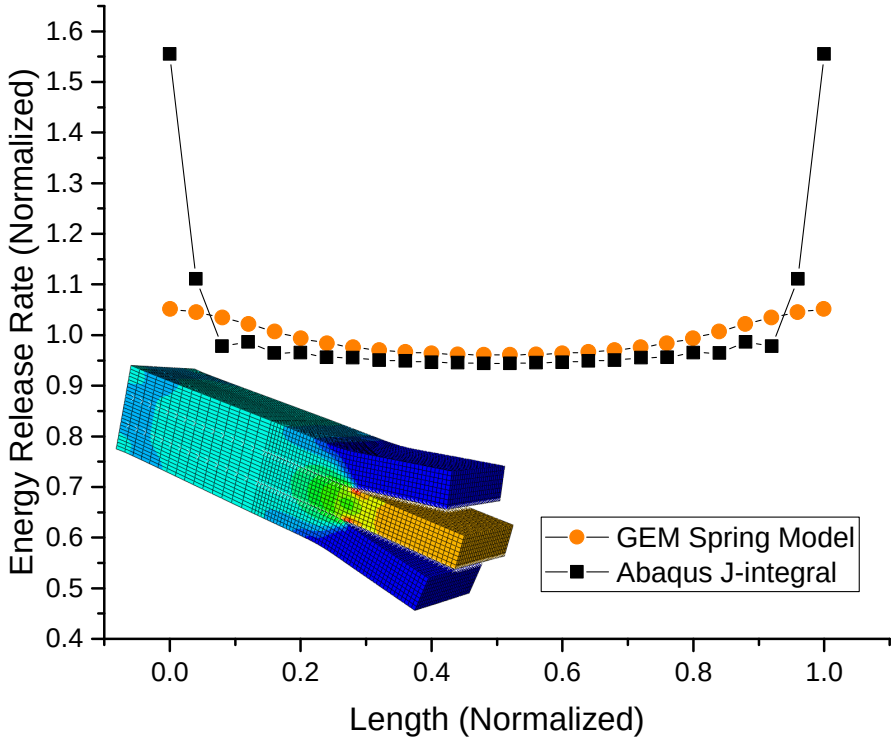

Fig. 4.6: The GEM spring model is compared with J-integral using a triple-cantilever beam (TCB) to analyze mode II energy release rate (G_{II}). The energy release rate is normalized by Eq. 4.3, and the length of the delamination is normalized by the width of the beam.

4.2 Evaluation of New Spring Model (New Approach)

4.2.1 One-Dimensional Crack

A double-cantilever beam (DCB) was used to view the VCCT calculation as a crack

front moves through an element (Fig. 4.7). The DCB has a 1-dimensional crack, and the calculated mode I energy release rate (G_I) was verified by the analytical solution (Eq. 4.4) [13]. This model (Fig. 4.7) has square elements with lengths of 1 mm (height $h = 5$ mm, width $b = 1$ mm, applied force $P = 200$ N, modulus $E = 161,000$ N/mm^2). The overall length of the beam was $L = 100$ mm and the crack length a was varied.

$$G_{I0} = \frac{12P^2a^2}{b^2h^3E} \tag{4.4}$$

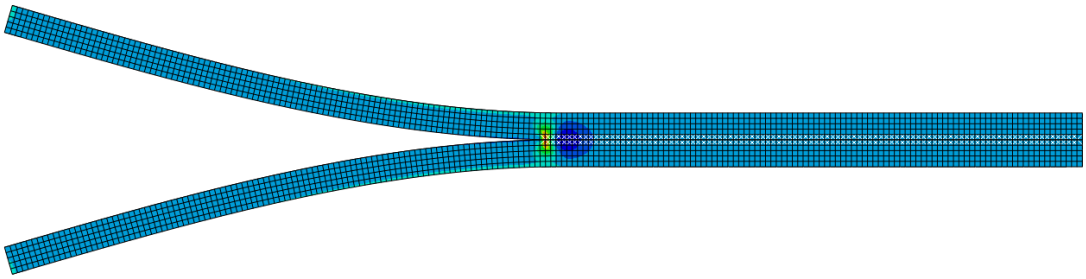

Fig. 4.7: DCB model was used to verify the new spring model with a 1-dimensional crack.

In the DCB model, the delamination front was moved at increments of 0.1 mm. Fig. 4.8 shows force and displacements, used in the G calculation, for various locations of the front among two elements. The variation in these plots is not physical, but it is an artifact of the model. When force and displacement are multiplied, most of the error balances out and the outcome for G is reliable. In the analytical solution (Eq. 4.4), the crack length a is in units of millimeters. As the front moves closer to the bonded node, the force changes non-linearly (the force is based on the displacement at that point ($F = kx$)). The displacement used in the G calculation varies almost linearly.

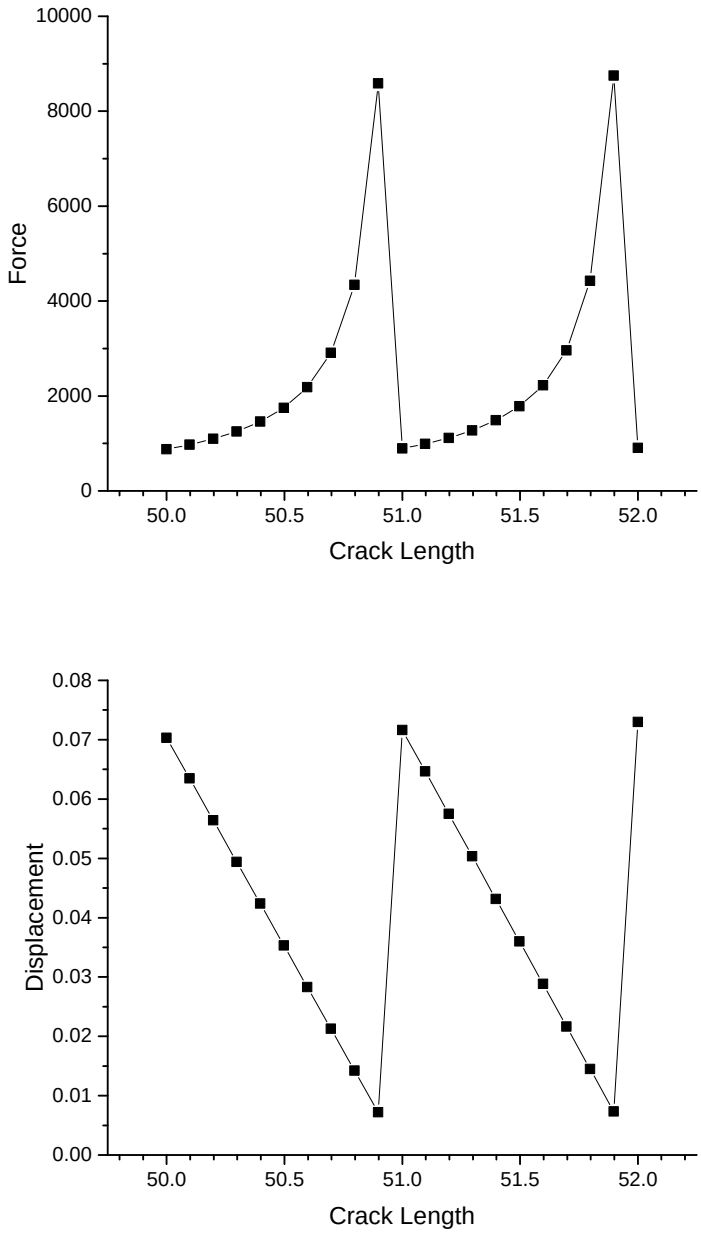

Fig. 4.8: Force increased non-linearly as crack length increased through an element (top). Displacement was taken from an element length behind the crack front and its variation is very near linear (bottom). The variation in these plots is an artifact of the model.

The energy release rate (G), for the DCB model discussed above, is shown in Fig. 4.9.

The variation of G is within 3.5% of the analytical value, but it varies almost linearly as the front moves through each element. When the front was within about one percent of the element length away from the node, the G value approached infinity (this is not shown in the plots). A fix for this discontinuity was to move the front (internal spring) to the node if it was within two-percent of the element length away from the node.

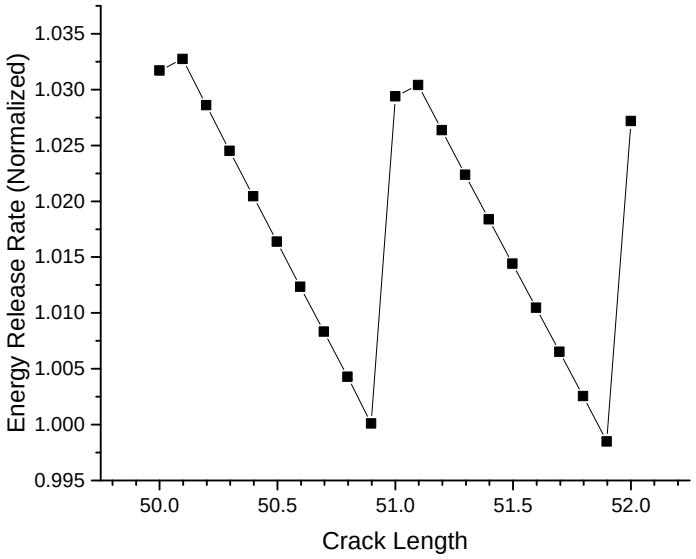

Fig. 4.9: G varies almost linearly, and the variation is within 3.5% of the analytical solution.

4.2.2 Two-Dimensional Crack (Delamination)

It is readily seen that this new spring model requires some sort of smoothing when used with a two-dimensional circular delamination (Fig. 4.10). There are some values that are outliers and they generally occur in pairs. The mesh is shown in Fig. 4.11.

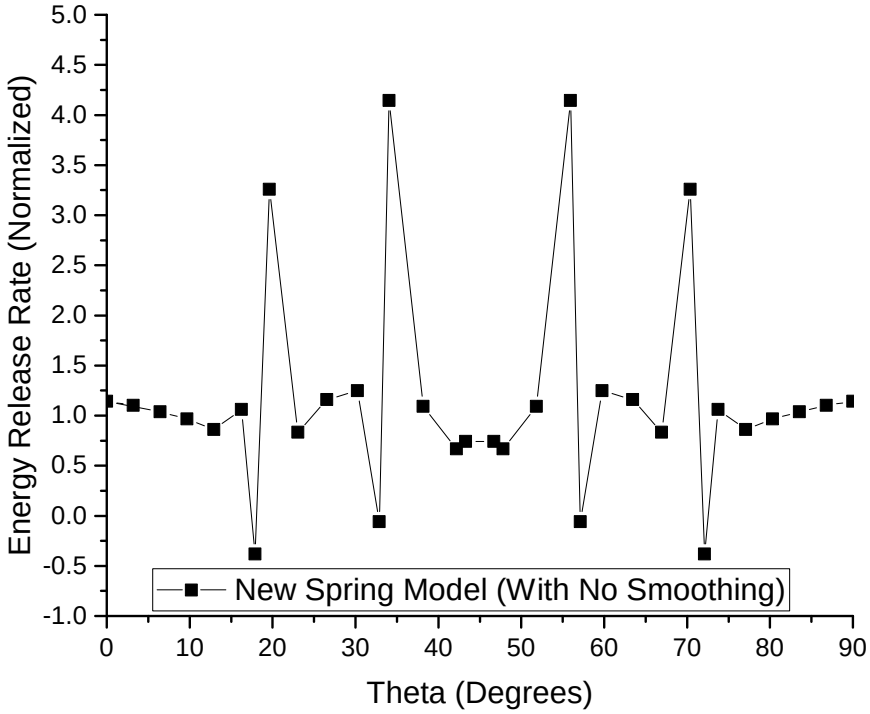

Fig. 4.10: The results for G are not ideal and smoothing will show that they are comparable to those of the GEM spring model.

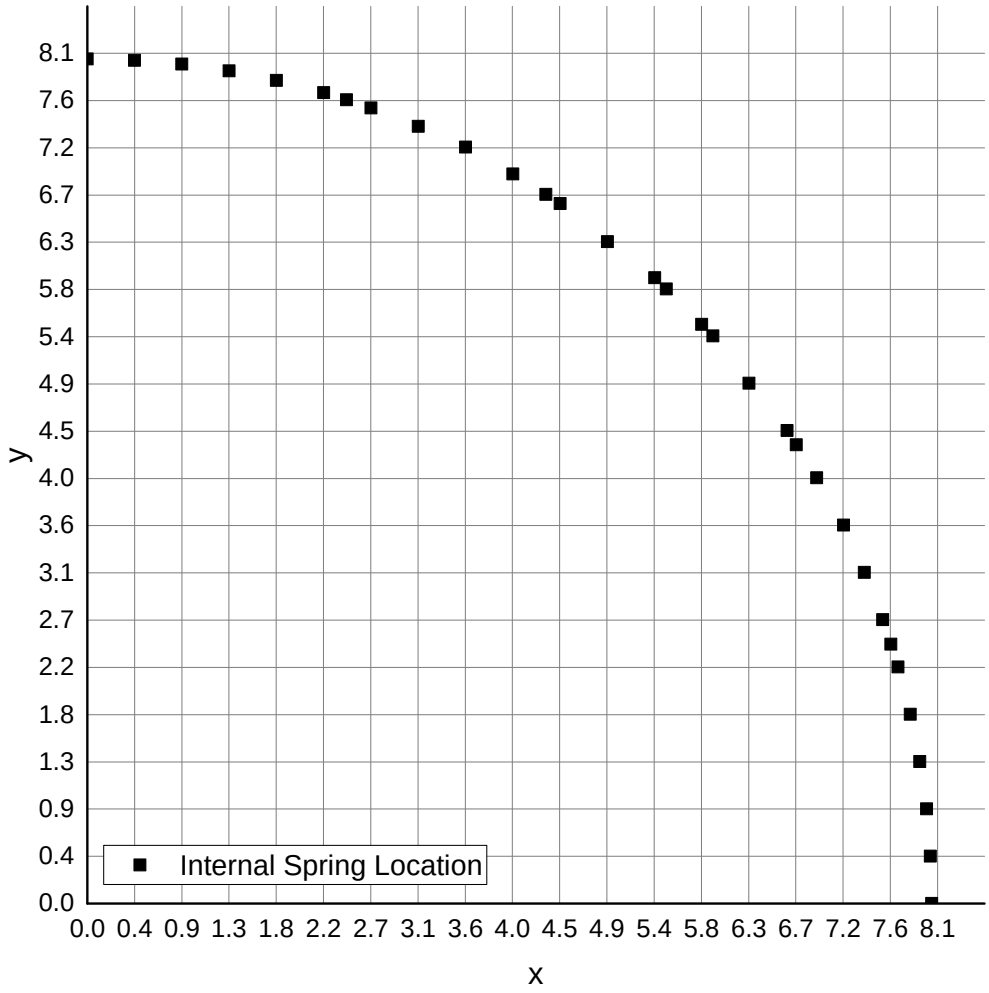

Fig. 4.11: Internal springs that represent the front are shown inside a square mesh (results for this circular delamination are shown in Fig. 4.10).

A smoothing function was created to remove outlying points and to average G values around the circular delamination (Fig. 4.12). In the figure, G values were averaged across 5 elements on either side of where G was calculated. The results were averaged twice. This simple smoothing algorithm is in Sec. B.4. The algorithm loops through all the G values on the delamination front, and if the location of that G value is within a specified distance of the node ($igsmooth \times ElementLength$) then that value is included in the average. The

algorithm can loop through the new smoothed G values by specifying an integer value greater than one for *numSmooth*.

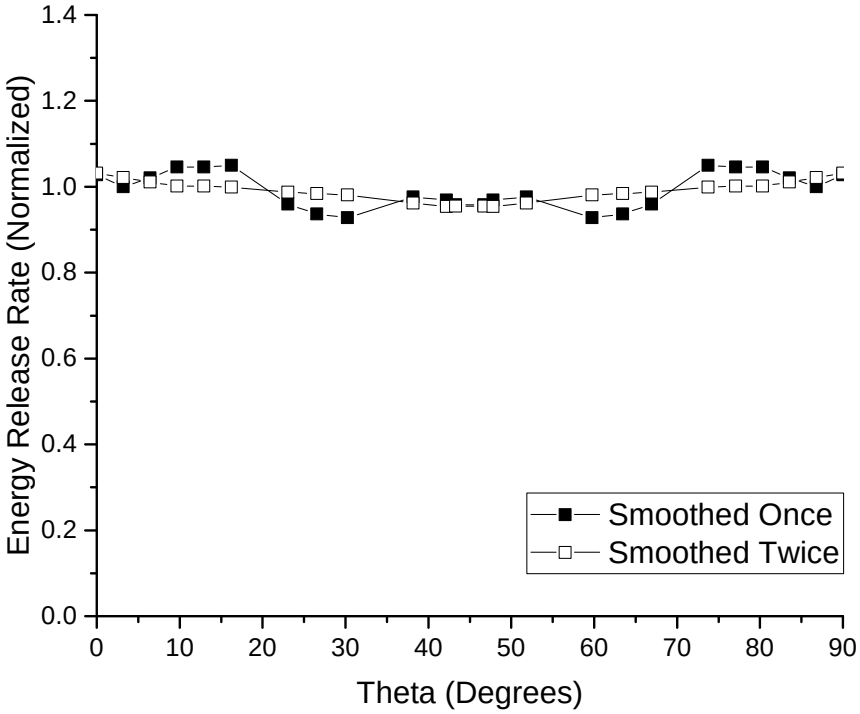

Fig. 4.12: The G values, from the circular crack analysis in the new spring model, were smoothed after removing outlying points (see Fig. 4.10).

4.3 Comparison (GEM Spring Model and New Spring Model)

It is shown that the computed values (of G) for the GEM spring model and the new spring model are both inaccurate (Fig. 4.13). The G values from the new spring model are more concentrated around the analytical solution but have a wider band. The GEM spring model has a smaller band, but the values are less concentrated. Even with smoothing, points located close together tend to skew the results.

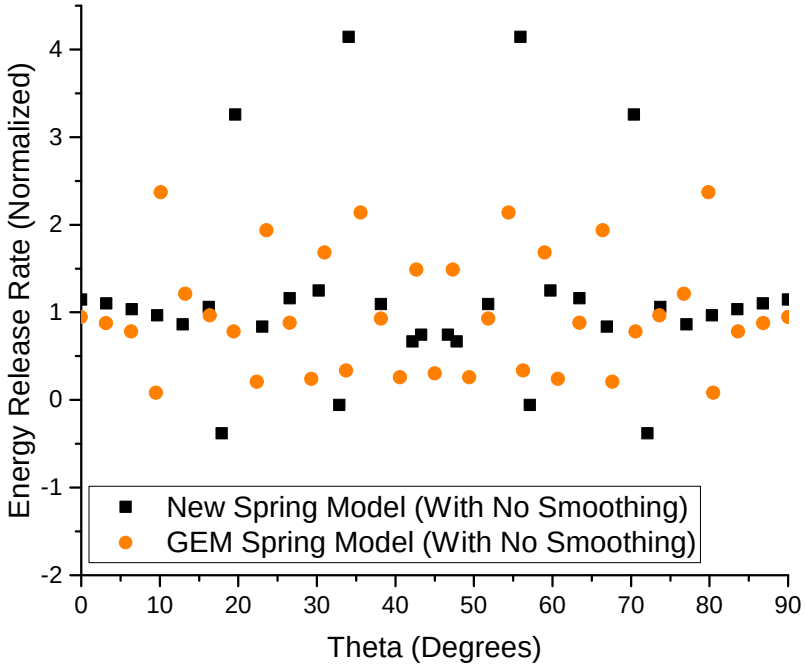

Fig. 4.13: The new spring model and GEM spring model are both inaccurate (values are not smoothed).

Smoothing was done on both the GEM spring model and new spring models (Fig. 4.14). The smoothing algorithm is described in Sec. B.4 and Sec. 4.2.2. The new spring model is better smoothed when outlying points are removed. The GEM spring model gets the best results when all points are included. The smoothing shown here was averaging across about five elements just once. The smoothing shown in Fig. 4.3 and Fig. 4.4 is a more complex smoothing function that weights the average with the node area and decreases the number of smoothing points if the differences in G are too great. It also weights the average differently if a G value at a point is 10× greater than at an adjacent point. A simpler algorithm made it easier to compare results as shown in Fig. 4.14.

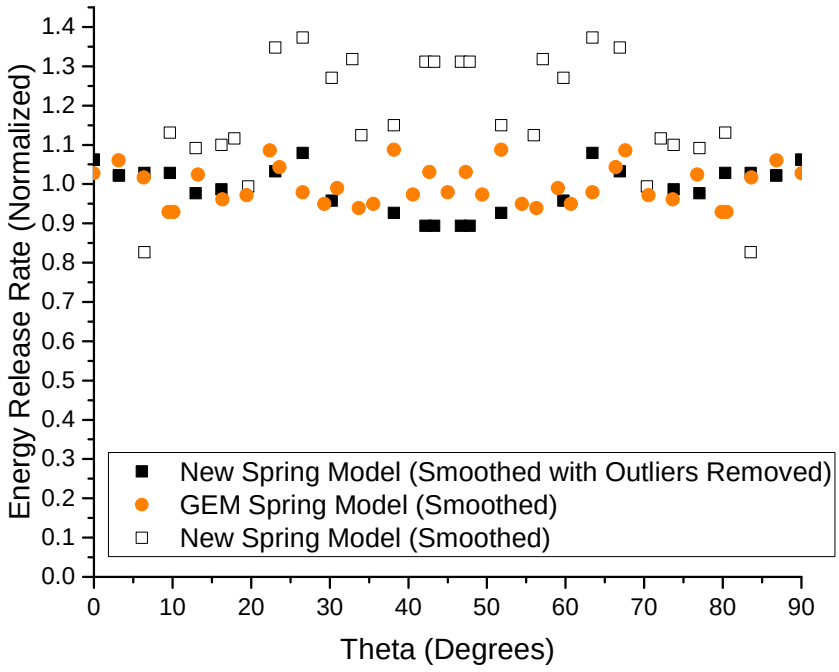

Fig. 4.14: Values were smoothed for the GEM spring model and new spring model.

From the DCB example in Sec. 4.2.1, the force was shown to not be constant, but the energy release rate was still accurate (to about 3.5%). The displacement (in the G calculation) decreased as the force increased. Fig. 4.15 displays a case where a displacement point is very close to a node that has stiffness distributed to it. If a node has any stiffness, the displacement is predicted to be very low. This is a possible source of error.

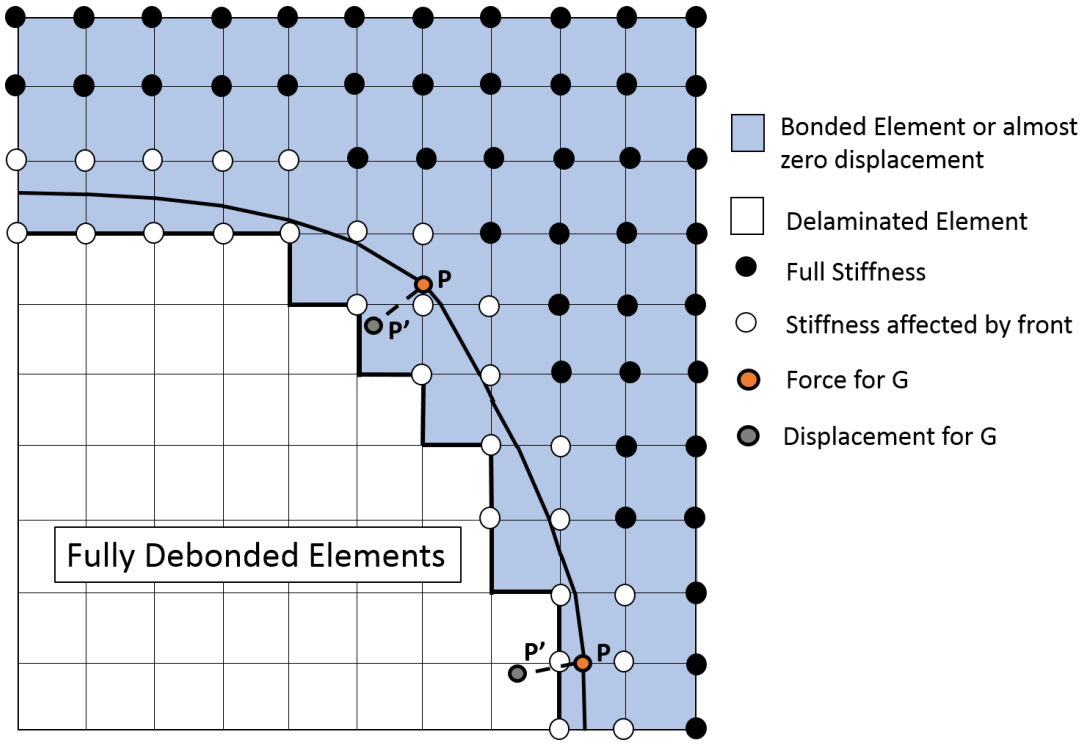

Fig. 4.15: The displacement point is affected by how stiffness is distributed.

4.4 Demonstration of Fatigue Crack Propagation

A demonstration of mesh-independent fatigue analysis verifies that fatigue life can be predicted well but needs improvement. The goal of the new spring model was to improve the fatigue calculation of the GEM spring model. The comparison of fatigue results (for the new spring model vs. the GEM spring model) is not necessary until further work is done, because the static energy release rate calculation of the new method is not an improvement to the GEM method. The fatigue model will be included in future work. The fatigue prediction in the GEM model is affected by how well G is smoothed across the delamination front. The G calculation in the new model varies depending on how close the nodes are to the real delamination front and requires smoothing to get results like those from the GEM model.

4.4.1 Paris Law for Circular Crack in Infinite Solid (Mode I Propagation)

The Paris law is integrated into the GEM spring model; the GEM spring model was

compared with an analytical solution and Abaqus VCCT. An analytical solution, for a circular crack, is shown in Eq. 4.5. This equation was derived from Eqs. 4.1 and 1.4 , and it is plotted in Fig. 4.16. Parameters were $c_1 = 3.52\ mm/cycle$, $m_1 = 5.0$, $G_{IC} = 0.05\ N/mm$, $R = 0.1$, $\sigma = 8.5\ N/mm^2$, $\nu = 0.3$, $E = 161,000\ N/mm^2$, and $a_0 = 8\ mm$. The model used, for studying the effect of mesh-independent energy release rate, was the same as described in Sec. 4.1.1. In Fig. 4.16, life prediction of the GEM spring model is too high. It is better than the Abaqus VCCT prediction and the analysis time was about 3.5 hours (Abaqus VCCT analysis time was about 300 hours). The average radius was used for the Abaqus VCCT model because the delamination didn't grow at a constant rate along the circumference of the delamination.

$$N_{total} = c_1^{-1} \left(\frac{1 - R^2}{G_{IC}} \times \frac{4\sigma^2(1 - \nu^2)}{\pi E} \right)^{-m_1} \left(\frac{a^{(1-m_1)} - a_0^{(1-m_1)}}{1 - m_1} \right) \tag{4.5}$$

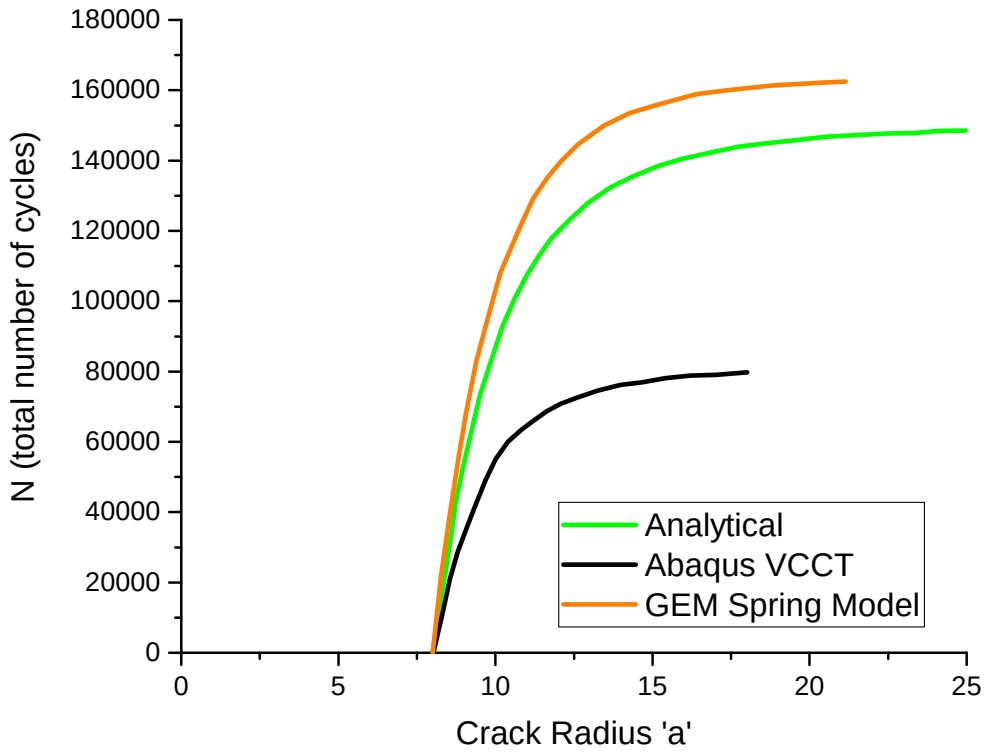

Fig. 4.16: The GEM spring model is compared with Abaqus VCCT and an analytical solution for a circular crack in fatigue.

Abaqus VCCT does not predict crack growth well in models with non-geometry conforming meshes. Fig. 4.17 shows that the VCCT analysis incorrectly predicted the shape of the circle crack propagation. The crack should remain circular as it grows because all stress conditions are the same at each point along the crack. The GEM spring model predicts that the circular crack remains circular, as it should.

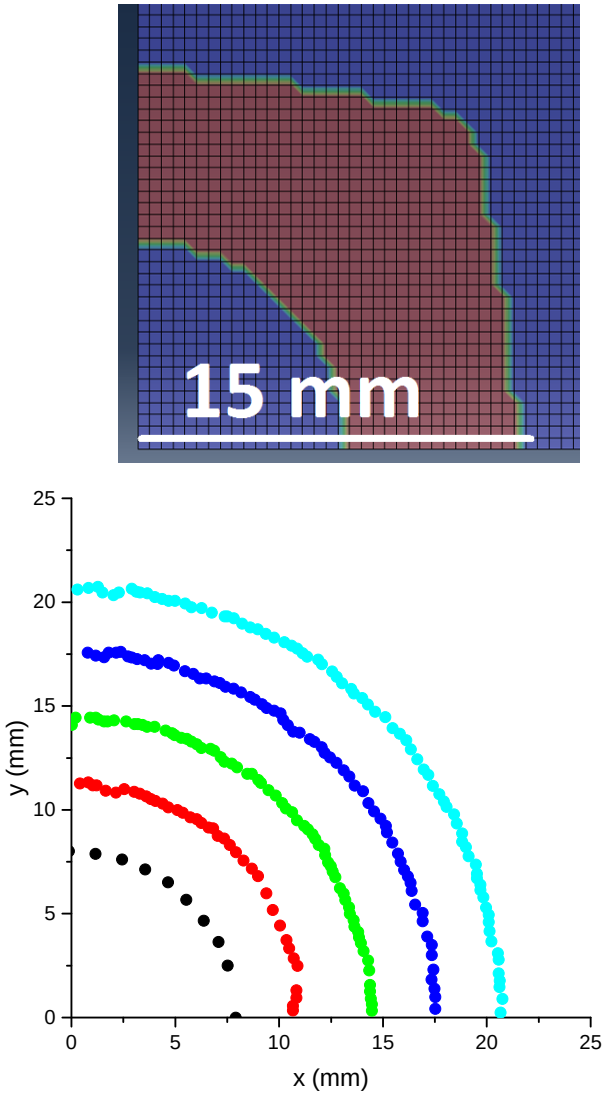

Fig. 4.17: Abaqus VCCT delamination growth (top) is compared with the GEM spring model delamination growth (bottom).

CHAPTER 5
CONCLUSIONS AND PLANS FOR FUTURE STUDIES

5.1 Conclusions

The efficiency of the new spring model is improved because of the global to natural coordinate conversion that was derived. In the GEM spring model, an iterative method was used to find the natural coordinates of the displacement point in the G calculation. The new spring model calculates the exact values of the natural coordinates directly.

The approach of the new spring model is more straightforward than that of the GEM spring model. It is conceptually better to calculate G at the real delamination front than at nodes that approximate the front. G does not have to be projected to the front from a nearby node.

There is close accuracy of G, but it is not a significant improvement to the GEM spring model. The G values for the new spring model were usually under-predicted when the cut elements were 3-point or 5-point cut elements. The stiffness distribution is different in both the GEM spring model and the new spring model; neither model has a stiffness distribution that is completely physical.

5.2 Plans for Future Studies

More study will be done to assign appropriate stiffness to springs in cut interface elements. It is difficult to say what amount of crack surface area is under the influence of each spring (a square mesh is often used). A better understanding of that would allow for a more accurate area to be used in the G calculation and stiffness assignment.

Study into where springs are placed (and where G is calculated) may improve stiffness distribution. Internal springs, representing the front in the new spring model, were added to the edges of elements because stiffness could easily be distributed without changing the

overall stiffness of the model. As a possible improvement, a single spring could be placed inside the element (not on an edge). With this formulation, springs can be anywhere in the element and be distributed to the nodes. By only adding internal springs to select elements instead of every element that lies on the delamination front, the model would be less mesh-independent, but it could improve the results of G if there are enough internal springs to model the front.

Research could be done to improve the kinematic relationship of the node pair behind the delamination front. The method described in [47], is very similar to what was done for this thesis. By some kinematic relationship, the force was decreased at the front node as the delamination progressed through the element. The energy release rate was still calculated at the same points. The force was taken from the node and the displacement point was taken from an element distance behind that node.

Smoothing and higher-order shape functions will be studied. In [47], a Gaussian smoothing function was used. Considerations to be made are whether outlying calculated values should be removed before smoothing, and whether smoothing should only involve adjacent elements or multiple elements. Smoothing is more efficient than refining mesh or using smaller crack growth increments in fatigue; it is used because the results for G are not accurate without smoothing.

Future work should include a fatigue growth method because the goal of this work is to improve fatigue life predictions. This method would need to include a calculation of a normal vector and the ability for the delamination to grow in a non-self-similar way. Future work should investigate mixed-mode fatigue as well.

REFERENCES

[1] Xie, D. and Biggers, S. B., "Strain energy release rate calculation for a moving delamination front of arbitrary shape based on the virtual crack closure technique. Part I: formulation and validation," *Engineering fracture mechanics*, Vol. 73, No. 6, 2006, pp. 771–785.

[2] Baptista, R., Almeida, A., and Infante, V., "Micro-crack propagation on a biomimetic bone like composite material studied with the extended finite element method," *Procedia Structural Integrity*, Vol. 1, 2016, pp. 18–25.

[3] Hyer, M. W., *Stress analysis of fiber-reinforced composite materials*, DEStech Publications, Inc, 2009.

[4] Hsieh, T., Kinloch, A., Masania, K., Lee, J. S., Taylor, A., and Sprenger, S., "The toughness of epoxy polymers and fibre composites modified with rubber microparticles and silica nanoparticles," *Journal of materials science*, Vol. 45, No. 5, 2010, pp. 1193–1210.

[5] Botelho, E. C., Silva, R. A., Pardini, L. C., and Rezende, M. C., "A review on the development and properties of continuous fiber/epoxy/aluminum hybrid composites for aircraft structures," *Materials Research*, Vol. 9, No. 3, 2006, pp. 247–256.

[6] Notta-Cuvier, D., Lauro, F., Bennani, B., and Balieu, R., "Damage of short-fibre reinforced materials with anisotropy induced by complex fibres orientations," *Mechanics of materials*, Vol. 68, 2014, pp. 193–206.

[7] Callister, W. D. and Rethwisch, D. G., *Materials science and engineering*, Vol. 5, John Wiley & Sons NY, 2011.

[8] Asmatulu, E., Twomey, J., and Overcash, M., "Recycling of fiber-reinforced composites and direct structural composite recycling concept," *Journal of Composite Materials*, Vol. 48, No. 5, 2014, pp. 593–608.

[9] Mauldin, T. C. and Kessler, M., "Self-healing polymers and composites," *International Materials Reviews*, Vol. 55, No. 6, 2010, pp. 317–346.

[10] Yi, X.-S. and An, X., "Developments of high-performance composites by innovative ex situ concept for aerospace application," *Journal of Thermoplastic composite materials*, Vol. 22, No. 1, 2009, pp. 29–49.

[11] McCarty, J. E., "Commercial transport aircraft composite structures," 1983.

[12] Braden, P. F., "F-16 Wing Structure Lifecycle," *Procedia Structural Integrity*, Vol. 1, 2016, pp. 106–109.

[13] Hertzberg, R. W., Vinci, R. P., and Hertzberg, J. L., *Deformation and Fracture Mechanics of Engineering Materials*, Vol. 5, John Wiley & Sons NY, 2013.

[14] Tsai, S. W. and Wu, E. M., "A general theory of strength for anisotropic materials," *Journal of composite materials*, Vol. 5, No. 1, 1971, pp. 58–80.

[15] Bowkett, M. and Thanapalan, K., "Comparative analysis of failure detection methods of composites materials systems," *Systems Science & Control Engineering*, Vol. 5, No. 1, 2017, pp. 168–177.

[16] Dassios, K. G., "A review of the pull-out mechanism in the fracture of brittle-matrix fibre-reinforced composites," *Advanced Composites Letters*, Vol. 16, No. 1, 2007, pp. 17–24.

[17] Tan, K. T., Watanabe, N., and Iwahori, Y., "Impact damage resistance, response, and mechanisms of laminated composites reinforced by through-thickness stitching," *International Journal of Damage Mechanics*, Vol. 21, No. 1, 2012, pp. 51–80.

[18] Talreja, R., "Fatigue of composite materials: damage mechanisms and fatigue-life diagrams," *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, Vol. 378, The Royal Society, 1981, pp. 461–475.

[19] Hutchinson, J. W. and Jensen, H. M., "Models of fiber debonding and pullout in brittle composites with friction," *Mechanics of materials*, Vol. 9, No. 2, 1990, pp. 139–163.

[20] De Freitas, M. and Reis, L., "Failure mechanisms on composite specimens subjected to compression after impact," *Composite Structures*, Vol. 42, No. 4, 1998, pp. 365–373.

[21] Anes, V., Pedro, R., Henriques, E., Freitas, M., and Reis, L., "Galvanic corrosion of aircraft bonded joints as a result of adhesive microcracks," *Procedia Structural Integrity*, Vol. 1, 2016, pp. 218–225.

[22] Banea, M. and da Silva, L. F., "Adhesively bonded joints in composite materials: an overview," *Proceedings of the Institution of Mechanical Engineers, Part L: Journal of Materials: Design and Applications*, Vol. 223, No. 1, 2009, pp. 1–18.

[23] Kaveh, A., "Determining the crack propagation in DCB composite specimens by fiber bridging modeling," *Revista Tecnica de la Facultad de Ingenieria Universidad del Zulia*, Vol. 37, 2014, pp. 1–9.

[24] Bowkett, M. and Thanapalan, K., "Damage Detection and Critical Failure Prevention of Composites," *Failure Analysis and Prevention*, InTech, 2017.

[25] Gholizadeh, S., "A review of non-destructive testing methods of composite materials," *Procedia Structural Integrity*, Vol. 1, 2016, pp. 50–57.

[26] Tian, Z., Yu, L., and Leckey, C., "Delamination detection and quantification on laminated composite structures with Lamb waves and wavenumber analysis," *Journal of Intelligent Material Systems and Structures*, Vol. 26, No. 13, 2015, pp. 1723–1738.

[27] Katunin, A. and Wronkowicz, A., "Characterization of failure mechanisms of composite structures subjected to fatigue dominated by the self-heating effect," *Composite Structures*, Vol. 180, 2017, pp. 1–8.

[28] Nikishkov, Y., Makeev, A., and Seon, G., "Progressive fatigue damage simulation method for composites," *International Journal of Fatigue*, Vol. 48, 2013, pp. 266–279.

[29] Martins, R., Reis, L., and Marat-Mendes, R., "Finite element prediction of stress-strain fields on sandwich composites," *Procedia Structural Integrity*, Vol. 1, 2016, pp. 66–73.

[30] Rabbolini, S., Beretta, S., and Foletti, S., "Fatigue crack growth in low cycle fatigue: an analysis of crack closure based on image correlation," *Procedia Structural Integrity*, Vol. 1, 2016, pp. 158–165.

[31] Ishbir, C., Banks-Sills, L., Fourman, V., and Eliasi, R., "Delamination propagation in a multi-directional woven composite DCB specimen subjected to fatigue loading," *Composites Part B: Engineering*, Vol. 66, 2014, pp. 180–189.

[32] Ducept, F., Davies, P., and Gamby, D., "An experimental study to validate tests used to determine mixed mode failure criteria of glass/epoxy composites," *Composites Part A: Applied Science and Manufacturing*, Vol. 28, No. 8, 1997, pp. 719–729.

[33] Floros, I., Tserpes, K., and Löbel, T., "Mode-I, mode-II and mixed-mode I+ II fracture behavior of composite bonded joints: Experimental characterization and numerical simulation," *Composites Part B: Engineering*, Vol. 78, 2015, pp. 459–468.

[34] Blanco, N., Gamstedt, E. K., Asp, L., and Costa, J., "Mixed-mode delamination growth in carbon–fibre composite laminates under cyclic loading," *International journal of solids and structures*, Vol. 41, No. 15, 2004, pp. 4219–4235.

[35] Turon, A., Costa, J., Camanho, P., and Dávila, C., "Simulation of delamination in composites under high-cycle fatigue," *Composites Part A: applied science and manufacturing*, Vol. 38, No. 11, 2007, pp. 2270–2282.

[36] Davis, B. R., Wawrzynek, P. A., and Ingraffea, A. R., "Simulation of Arbitrary Delamination Growth in Composite Structures Using the Virtual Crack Extension Method," *27th Technical Conference of the American Society for Composites*, 2012.

[37] Skvortsov, Y. V., Chernyakin, S., Glushkov, S., and Perov, S., "Simulation of fatigue delamination growth in composite laminates under mode I loading," *Applied Mathematical Modelling*, Vol. 40, No. 15, 2016, pp. 7216–7224.

[38] Krueger, R., "Virtual crack closure technique: history, approach, and applications," *Applied Mechanics Reviews*, Vol. 57, No. 2, 2004, pp. 109–143.

[39] Wimmer, G., Schuecker, C., and Pettermann, H. E., *Numerical simulation of delamination onset and growth in laminated composites*, na, 2006.

[40] Pirondi, A., Giuliese, G., Moroni, F., Bernasconi, A., and Jamil, A., "Comparative study of cohesive zone and virtual crack closure techniques for three-dimensional fatigue debonding," *The Journal of Adhesion*, Vol. 90, No. 5-6, 2014, pp. 457–481.

[41] Zhao, L., Zhi, J., Zhang, J., Liu, Z., and Hu, N., "XFEM simulation of delamination in composite laminates," *Composites Part A: Applied Science and Manufacturing*, Vol. 80, 2016, pp. 61–71.

[42] Chen, B., Tay, T., Pinho, S., and Tan, V., "Modelling the tensile failure of composites with the floating node method," *Computer Methods in Applied Mechanics and Engineering*, Vol. 308, 2016, pp. 414–442.

[43] Cook, R. D. et al., *Concepts and applications of finite element analysis*, John Wiley & Sons, 2007.

[44] Naderi, M. and Iyyer, N., "3D modeling of arbitrary cracking in solids using augmented finite element method," *Composite Structures*, Vol. 160, 2017, pp. 220–231.

[45] Van der Meer, F., Sluys, L., Hallett, S., and Wisnom, M., "Computational modeling of complex failure mechanisms in laminates," *Journal of Composite Materials*, Vol. 46, No. 5, 2012, pp. 603–623.

[46] Latifi, M. and Sluijs, L. J., "Computational modeling of failure in composites under fatigue loading conditions," 2017.

[47] Vieira De Carvalho, N., Krueger, R., Mabson, G. E., and Deobald, L., "Combining Progressive Nodal Release with the Virtual Crack Closure Technique to Model Fatigue Delamination Growth Without Re-meshing," *2018 AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, 2018, p. 1468.

[48] Turon, A., Davila, C. G., Camanho, P. P., and Costa, J., "An engineering solution for mesh size effects in the simulation of delamination using cohesive zone models," *Engineering fracture mechanics*, Vol. 74, No. 10, 2007, pp. 1665–1682.

[49] Irwin, G. R., "Crack-extension force for a part-through crack in a plate," *Journal of Applied Mechanics*, Vol. 29, No. 4, 1962, pp. 651–654.

APPENDICES

APPENDIX A
FORMULATION OF UEL

A.1 UEL Subroutine Outline

1. Pre-process
   (a) Read in nodes and elements (from mesh.txt)
   (b) Determine if node is inside delamination
   (c) Find cut sides of elements, Force Coord., Disp. Coord., and spring areas/stiffness (and move to node if needed)
2. Find natural/local coordinates of internal springs on the delamination front
3. Find natural coordinates of displacement point
4. Nodal Stiffness Matrix
5. Internal Spring Stiffness Matrix
6. Shape Function Matrix (and transpose)
7. AMATRX and RHS
8. Get force for internal spring at the real front
9. Post-process (called when last UEL is called)
   (a) Combine forces that are at the same locations (change size of array)
   (b) Calculate G
   (c) Smooth function

A.2 Q9 Element Shape Functions

A Q9 UEL was created because the displacements are more accurate, but the results for G values did not improve. A Q9 element has shape functions that do not create straight lines when a natural coordinate is held constant (see Fig. A.1 and Eq. A.1). Therefore, when these shape functions are used, an optimization routine is used. The steepest-descent method is used to locate the natural coordinates from the input global coordinates (see Sec. A.3).

$$
\begin{aligned}
N_1 &= \frac{1}{4}(1-\xi)(1-\eta) - \frac{1}{2}(N_5+N_8) - \frac{1}{4}N_9 \\
N_2 &= \frac{1}{4}(1+\xi)(1-\eta) - \frac{1}{2}(N_5+N_6) - \frac{1}{4}N_9 \\
N_3 &= \frac{1}{4}(1+\xi)(1+\eta) - \frac{1}{2}(N_6+N_7) - \frac{1}{4}N_9 \\
N_4 &= \frac{1}{4}(1-\xi)(1+\eta) - \frac{1}{2}(N_7+N_8) - \frac{1}{4}N_9 \\
N_5 &= \frac{1}{2}(1-\xi^2)(1-\eta) - \frac{1}{2}N_9 \\
N_6 &= \frac{1}{2}(1+\xi)(1-\eta^2) - \frac{1}{2}N_9 \\
N_7 &= \frac{1}{2}(1-\xi^2)(1+\eta) - \frac{1}{2}N_9 \\
N_8 &= \frac{1}{2}(1-\xi)(1-\eta^2) - \frac{1}{2}N_9 \\
N_9 &= (1-\xi^2)(1-\eta^2)
\end{aligned}
\tag{A.1}
$$

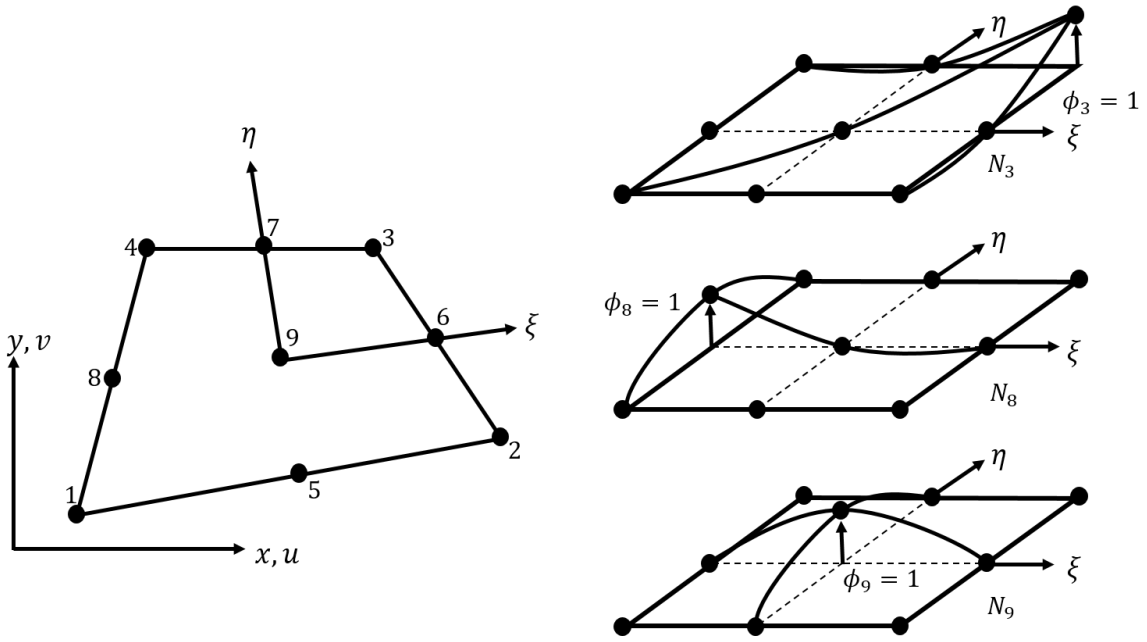

Fig. A.1: Q9 Shape functions predict displacement more accurately than linear shape functions.

A.3 Global to Natural (Q9 Element)

Changing global coordinates to natural coordinates for a Q9 element requires optimization because slopes between points are not constant when one natural coordinate is held constant. Optimization was performed using the steepest descent method (Eq. A.2). The optimization function was verified in MATLAB (Fig. A.2). For odd shapes, the iteration parameter needs to be changed, and the number of iterations must be increased.

$$
\begin{aligned}
f(\xi,\eta) &= x_0 - \sum N_i x_i = 0 \\
g(\xi,\eta) &= y_0 - \sum N_i y_i = 0 \\
F(\xi,\eta) &= g^2 + f^2 = 0 \\
\frac{\partial F}{\partial \xi} &= \frac{\partial g^2}{\partial \xi} + \frac{\partial f^2}{\partial \xi} = 0 \\
\frac{\partial F}{\partial \eta} &= \frac{\partial g^2}{\partial \eta} + \frac{\partial f^2}{\partial \eta} = 0
\end{aligned}
\tag{A.2}
$$

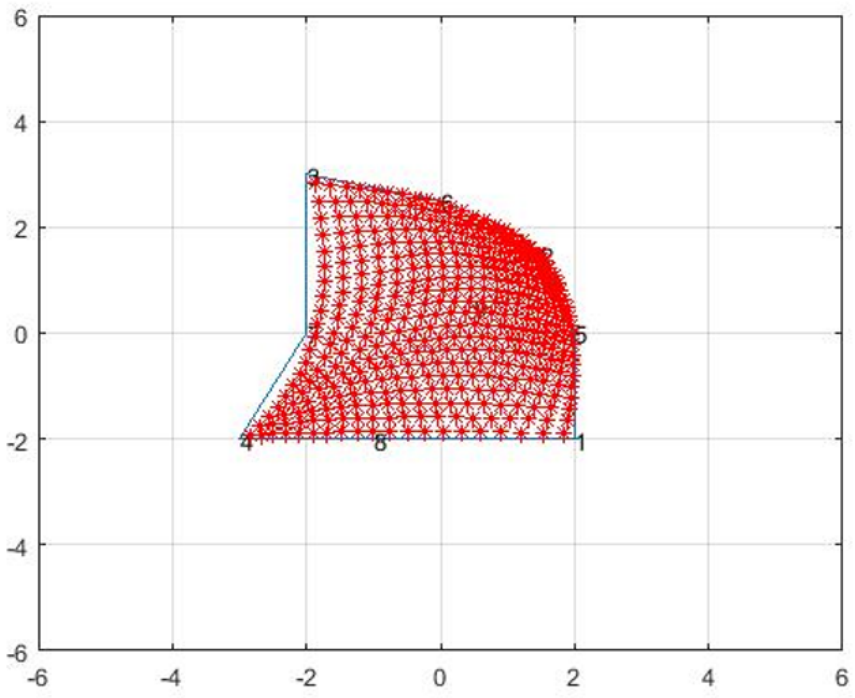

Fig. A.2: A MATLAB verification shows a sweep through natural coordinates which were changed to global coordinates and changed back to the natural coordinates.

MATLAB Code

```
% steepest descent function

n=0;
eps=2;
a=0.09;                 % iteration parameter
xieta=[1;1];            % initial guess for xi and eta

while eps>1e-10 && n<10000
    gradf=[dFdXi;dFdEta];
    eps=abs(gradf(1))+abs(gradf(2));
```

```
11        iterate=xieta-a*gradf;
12        xieta=iterate;
13        n=n+1;
14  end
```

APPENDIX B

UEL FORTRAN CODE SAMPLES

B.1 Shape Function Matrix Code Sample

```
j=0
do i = 1, 3*NNewSpring,3
! NNewSpring is the Number of internal springs in the UEL
 j=j+1
 SHAPE1(i) = 0.25D0*(1.-xi_spring(j))*(1.-eta_spring(j))
 SHAPE2(i) = 0.25D0*(1.+xi_spring(j))*(1.-eta_spring(j))
 SHAPE3(i) = 0.25D0*(1.+xi_spring(j))*(1.+eta_spring(j))
 SHAPE4(i) = 0.25D0*(1.-xi_spring(j))*(1.+eta_spring(j))
 SHAPE1(i+1) = 0.25D0*(1.-xi_spring(j))*(1.-eta_spring(j))
 SHAPE2(i+1) = 0.25D0*(1.+xi_spring(j))*(1.-eta_spring(j))
 SHAPE3(i+1) = 0.25D0*(1.+xi_spring(j))*(1.+eta_spring(j))
 SHAPE4(i+1) = 0.25D0*(1.-xi_spring(j))*(1.+eta_spring(j))
 SHAPE1(i+2) = 0.25D0*(1.-xi_spring(j))*(1.-eta_spring(j))
 SHAPE2(i+2) = 0.25D0*(1.+xi_spring(j))*(1.-eta_spring(j))
 SHAPE3(i+2) = 0.25D0*(1.+xi_spring(j))*(1.+eta_spring(j))
 SHAPE4(i+2) = 0.25D0*(1.-xi_spring(j))*(1.+eta_spring(j))
end do

N = 0
Ntrans = 0
do i = 1, 3*NNewSpring
 do j = 1, 3
```

```
23    do k = 1, NNewSpring
24     if ( i.eq.(3*(k-1)+j) ) then
25      N(i,j) =  SHAPE1(i)
26      Ntrans(j,i) = SHAPE1(i)
27      N(i+3*NNewSpring,j+12) = N(i,j)
28      Ntrans(j+12,i+3*NNewSpring) = SHAPE1(i)
29     end if
30    end do
31   end do
32   do j = 4, 6
33    do k = 1, NNewSpring
34     if ( i.eq.(3*(k-2)+j) ) then
35      N(i,j) =  SHAPE2(i)
36      Ntrans(j,i) = SHAPE2(i)
37      N(i+3*NNewSpring,j+12) = N(i,j)
38      Ntrans(j+12,i+3*NNewSpring) = SHAPE2(i)
39     end if
40    end do
41   end do
42   do j = 7, 9
43    do k = 1, NNewSpring
44     if ( i.eq.(3*(k-3)+j) ) then
45      N(i,j) =  SHAPE3(i)
46      Ntrans(j,i) = SHAPE3(i)
47      N(i+3*NNewSpring,j+12) = N(i,j)
48      Ntrans(j+12,i+3*NNewSpring) = SHAPE3(i)
49     end if
50    end do
```

```
51  end do
52  do j = 10, 12
53   do k = 1, NNewSpring
54    if ( i.eq.(3*(k-4)+j) ) then
55     N(i,j) =  SHAPE4(i)
56     Ntrans(j,i) =  SHAPE4(i)
57     N(i+3*NNewSpring,j+12) = N(i,j)
58     Ntrans(j+12,i+3*NNewSpring) =  SHAPE4(i)
59    end if
60   end do
61  end do
62 end do
```

B.2 Stiffness Matrix Code Sample

```
1  do i = 1, NDOFEL/8 ! three
2   do j = 1, NDOFEL/8
3    if (i.eq.j) then
4     Knodal(i,j) = nStiff(1)
5     Knodal(i+3,j+3) = nStiff(2)
6     Knodal(i+6,j+6) = nStiff(3)
7     Knodal(i+9,j+9) = nStiff(4)
8
9     Knodal(i+12,j+12) = nStiff(1)
10    Knodal(i+15,j+15) = nStiff(2)
11    Knodal(i+18,j+18) = nStiff(3)
12    Knodal(i+21,j+21) = nStiff(4)
13
14    Knodal(i+12,j)   = -nStiff(1)
```

```
15      Knodal(i+15,j+3) = -nStiff(2)
16      Knodal(i+18,j+6) = -nStiff(3)
17      Knodal(i+21,j+9) = -nStiff(4)
18
19      Knodal(i,j+12)   = -nStiff(1)
20      Knodal(i+3,j+15) = -nStiff(2)
21      Knodal(i+6,j+18) = -nStiff(3)
22      Knodal(i+9,j+21) = -nStiff(4)
23     end if
24    end do
25   end do
26
27   Kfront = 0
28   do i = 1, 6*NNewSpring/4 ! three
29    do j = 1, 6*NNewSpring/4
30     if (i.eq.j) then
31      Kfront(i,j)     = springStiff(1,JELEM-numElem)
32      Kfront(i+3,j+3) = springStiff(2,JELEM-numElem)
33
34      Kfront(i+6,j+6) = springStiff(1,JELEM-numElem)
35      Kfront(i+9,j+9) = springStiff(2,JELEM-numElem)
36
37      Kfront(i+6,j)   = -springStiff(1,JELEM-numElem)
38      Kfront(i+9,j+3) = -springStiff(2,JELEM-numElem)
39
40      Kfront(i,j+6)   = -springStiff(1,JELEM-numElem)
41      Kfront(i+3,j+9) = -springStiff(2,JELEM-numElem)
42     end if
```

```
 end do
end do

i=NNewSpring*6
j=NDOFEL
KfrontN(1:i,1:j) = matmul(Kfront(1:i,1:i),N(1:i,1:j))
NTKfrontN(1:j,1:j) = matmul(Ntrans(1:j,1:i),KfrontN(1:i,1:j))

AMATRX(1:24,1:24) = (Knodal(1:24,1:24) + NTKfrontN(1:24,1:24))

do i = 1, NDOFEL
 RHS(i, 1) = 0.
 do k = 1, NDOFEL
  RHS(i, 1) = RHS(i, 1) - AMATRX(i, k)*U(k)
 end do
end do
```

B.3 Global to Natural Code Sample

```
SUBROUTINE kbilinear(xi,eta,xbar,ybar,x,y)
real*8, intent(in) :: xbar,ybar
real*8, intent(in) :: x(4),y(4)
real*8, intent(out) :: xi, eta
real*8 :: c1,c2,c3,c4,k1,k2,k3,k4
real*8 :: Ax1,Ax2,Ay1,Ay2,Bx1,Bx2,By1,By2
real*8 :: mAC,mCB,C,B,A,xi1,xi2
integer :: i,j,k

! constants for solving second natural/local coordinate
```

```
11 c1 = 4*xbar-(x(1)+x(2)+x(3)+x(4))
12 c2 = -x(1)-x(2)+x(3)+x(4)
13 c3 = -x(1)+x(2)+x(3)-x(4)
14 c4 = +x(1)-x(2)+x(3)-x(4)
15 k1 = 4*ybar-(y(1)+y(2)+y(3)+y(4))
16 k2 = -y(1)-y(2)+y(3)+y(4)
17 k3 = -y(1)+y(2)+y(3)-y(4)
18 k4 = +y(1)-y(2)+y(3)-y(4)
19
20 ! check slopes
21 if ((x(3)-x(2)).eq.0.and.(x(4)-x(1)).eq.0) then
22  ! print *,'lines 23 and 14 are aligned with y-axis'
23  xi = 2*(xbar-x(1))/(x(2)-x(1))-1
24 else
25  ! print *,'lines 23 and 14 are not vertical'
26  ! x coordinate of point A (Ax)
27  Ax1 = (x(2)+x(1))/2.
28  Ax2 = (x(2)-x(1))/2.
29  !  x coordinate of point B (Bx)
30  Bx1 = (x(3)+x(4))/2.
31  Bx2 = (x(3)-x(4))/2.
32  ! y coordinate of point A (Ay)
33  Ay1 = (y(2)+y(1))/2.
34  Ay2 = (y(2)-y(1))/2.
35  ! y coordinate of point B (By)
36  By1 = (y(3)+y(4))/2.
37  By2 = (y(3)-y(4))/2.
38  ! slopes from input xi
```

```
39  mAC = (ybar-Ay)/(xbar-Ax)
40  mCB = (By-ybar)/(Bx-xbar)
41  ! quadratic coefficients
42  C = (Bx1-Ax1)*ybar+(Ay1-By1)*xbar+By1*Ax1-Ay1*Bx1
43  B = (Bx2-Ax2)*ybar+(Ay2-By2)*xbar+By1*Ax2+By2*Ax1-Ay2*Bx1-Ay1
       *Bx2
44  A = Ax2*By2-Ay2*Bx2
45  ! Calculate xi (local coordinate)
46  if (A.eq.0) then
47   xi1 = -C/B
48   xi2 = -C/B
49  else
50   xi1 = (-B+sqrt(B**2-4*A*C))/(2*A)
51   xi2 = (-B-sqrt(B**2-4*A*C))/(2*A)
52   if (B**2-4*A*C.lt.0) then
53    print *,'!!!error discriminant less than zero'
54    xi1 = 2 ! outside element
55   end if
56   if (abs(xi1).le.1) then
57    if ( abs(xi2).le.1 ) then
58     print *,'!!!Warning, both are acceptable roots'
59    end if
60   end if
61  end if
62  if (abs(xi1).le.1) then
63   xi = xi1
64  else if (abs(xi2).le.1) then
65   xi = xi2
```

```
66   else ! both roots are acceptable or not acceptable
67    xi = xi1
68   end if
69  end if
70
71  ! local coordinate eta
72  eta = (c1-c3*xi)/(c2+c4*xi)
73  if ((c2+c4*xi).eq.0) then
74   eta = (k1-k3*xi)/(k2+k4*xi)
75  end if
76  end subroutine kbilinear
```

B.4 Smoothing Function

```
1   igsmooth=5.D0
2   numSmooth=3
3   do jj=1,numSmooth
4    write(300,*),'----------------------------'
5    do i=1,k ! k is the number of Points
6     sumavg=0.
7     counter=0
8     if (gtotal(i).gt.0.1*G_ana(i) .and.
9   1   gtotal(i).lt.1.9*G_ana(i)) then
10     do j=1,k
11      if (abs(ForceCoordComb(i,1)-ForceCoordComb(j,1)).lt.
12  1    igsmooth*dist0 .and. abs(ForceCoordComb(i,2)-
13  2    ForceCoordComb(j,2)).lt.igsmooth*dist0) then
14       if (gtotal(j).gt.0.1*G_ana(j) .and.
15  1     gtotal(j).lt.1.9*G_ana(j)) then
```

```
16          counter=counter+1
17          if (jj.eq.1) then
18           sumavg=sumavg+Gtotal(j)
19          else
20           sumavg=sumavg+GsmoothOld(j)
21          end if
22         end if
23        end if
24       end do
25      end if
26
27      if (counter.gt.0) then
28       Gsmooth(i)=sumavg/counter
29       write(300,207),datan2d(ForceCoordComb(i,2),
30     1    ForceCoordComb(i,1)),Gtotal(i),Gsmooth(i)/G_ana(i),
31     2    ForceCoordComb(i,1:2)
32      end if
33     end do
34     GsmoothOld(1:k)=Gsmooth(1:k)
35    end do
```