INTERNAL DAMAGE CHARACTERIZATION FOR COMPOSITE MATERIALS UNDER BIAXIAL LOADING CONFIGURATION

by

Jay David Smith

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

of

Masters of Science

in

Mechanical Engineering

MONTANA STATE UNIVERSITY
Bozeman, Montana

January 2007

APPROVAL

of a thesis submitted by

Jay David Smith

This thesis has been read by each member of the thesis committee and has been found to be satisfactory regarding content, English usage, format, citations, bibliographic style, and consistency, and is ready for submission to the Division of Graduate Education.

Dr. Douglas Cairns, Committee Chair

Approved for the Department of Mechanical and Industrial Engineering

Dr. Christopher H. M. Jenkins, Department Head

Approved for the Division of Graduate Education

Dr. Carl A. Fox, Vice Provost

STATEMENT OF PERMISSION TO USE

In presenting this thesis in partial fulfillment of the requirements of a master's degree at Montana State University, I agree that the library shall make it available to borrowers under rules of the library.

If I have indicated my intention to copyright this thesis by including a copyright notice page, copying is allowable only for scholarly purposes, consistent with "fair use" as described in the U.S. Copyright Law. Requests for permission for extended quotation from the reproduction of the thesis in whole or in parts may be granted only by the copyright holder.

Jay David Smith

January 2007

AKNOWLEDGEMENTS

I would like to thank the following individuals and offices for continued help on the project:

Office of Naval Research

Dr. Douglas Cairns

Dr. Jay Conant

Dr. Ladean MiKittrick

Montana State University Technical Services

All the members of the Montana State University Composites Group

TABLE OF CONTENTS

TABLE OF CONTENTS - CONTINUED

LIST OF TABLES

LIST OF FIGURES

LIST OF FIGURES-CONTINUED

LIST OF FIGURES

Figure Page

LIST OF FIGURES

Figure Page

ABSTRACT

This thesis contains the results of a composite material database developed for fiber glass laminates using test data from the in-plane loader (IPL). The IPL is a unique multi-axial test machine developed at Montana State University. The research was completed with the aim to improve the reliability of composite materials, namely fiber glass for use in wind turbine blades.

An energy method was used to characterize strain-induced damage in fiber glass coupons. The energy dissipated by internal failure mechanisms was employed as a metric of internal damage. Thus, by means of a deconvolution procedure data from the IPL was used to obtain a dissipated energy density function. The dissipated energy density function was utilized to characterize the behavior of fiber glass between the onset of damage through ultimate material failure.

Two cases studies were used to evaluate the current capabilities of the dissipated energy density function created from IPL data. Open-hole compression and bearing tension were selected as representative composite structure. The load-displacement response was predicted for both of the structures. By comparing the predictions with experimental results a determination was made concerning the transportable nature of the dissipated energy density function between composite structures of different geometry and laminate lay-up.

Under the current dissipated energy density paradigm a linear finite element model was used to determine the strain field for a post damaged coupon. The validity of this procedure is unclear. As a part of this work, the effects of material nonlinearity on damage induced strain redistribution was explored. A best fit bilinear in-situ material constitutive response was determined based on experimental data obtained from the IPL. The level of strain redistribution was determined by directly comparing the strain field computed from the nonlinear finite element model with the linear counterpart.

INTRODUCTION

Basics

Modern engineering structures have evolved over time to the point where traditional engineering materials don't always meet the performance requirements of the design. Therefore, a need for higher performance materials is inevitable. However, the superior performance of using more exotic materials does not come without a price. History has shaped the present understanding of materials. Failures throughout the last century have served as valuable lessons for engineers. Take for example the experience gained after the catastrophic failures of the De Havilland Comet 1 shown in Figure 1. The design requirement for higher than normal cabin operating pressure combined with thin gage skin panels and severe stress concentrations at a square cutouts resulted in explosive decompression of two Comet 1 aircraft during flight. Analysis revealed that the failure was a result of metal fatigue cracks.

Figure 1. The De Havilland Comet 1 flew the first commercial jet passenger flight.

Thus, fatigue cracks are considered in airplane design based in part on the failures of the Comet 1. History has provided present day engineers with a set of materials with a proven service record, as well mature methods of analysis and design. Fundamental analysis and design equations have been developed for traditional engineering materials. These equations are most often linear and are commonly used by engineers comfortably. The set of equations that govern more exotic material can be nonlinear and significantly more complicated relative to more traditional materials. Since more exotic materials have varied service history and require analysis techniques that deviate from traditional methods they are more difficult to work with in a design.

Composites are one of the materials that engineers have used in order to obtain greater material performance. Composite material generally have an excellent strength to weight and stiffness to weight ratio.(Figure 2) They look extremely attractive for application where parasitic weight must be a minimum. Composite materials also provide the unique ability to be tailored for a specific application. The material can be designed concurrently with the structure resulting in a more optimized design.

The current work will focus specifically on fiber reinforced polymer matrix composites. A significant amount of research at Montana State University (MSU) has been conducted on composite materials. Obtaining a better understanding of these materials, in terms of the constitutive behavior and the failure modes will increase the material reliability, therefore increasing the number of potential in service applications.

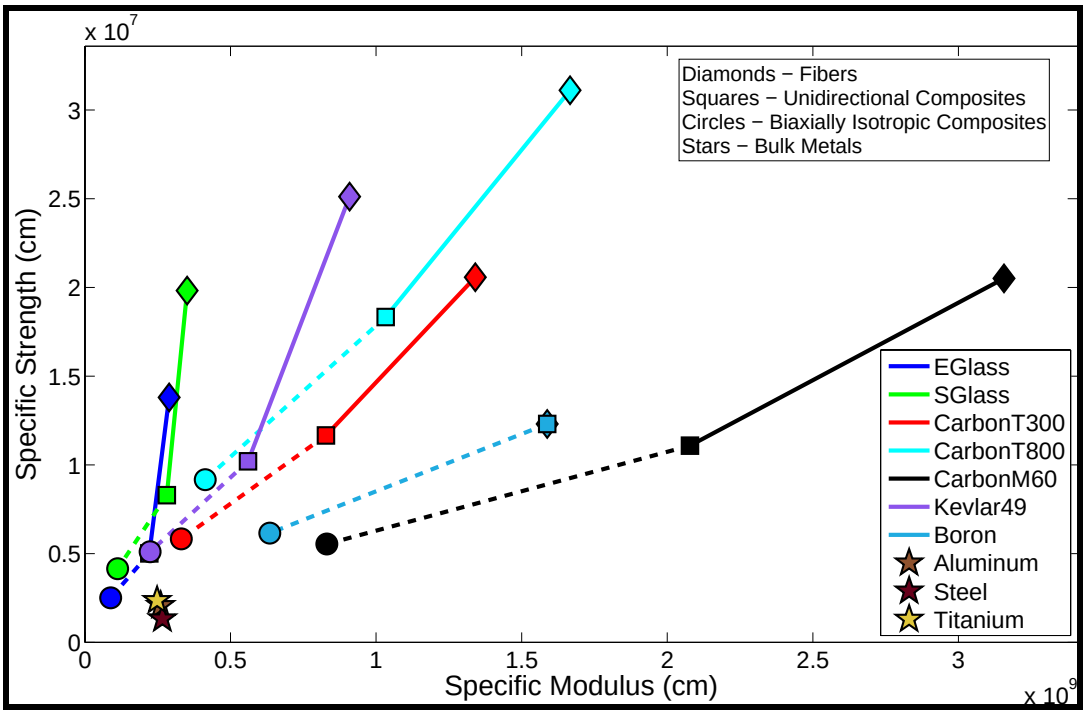


Figure 2. Specific modulus and strength properties of composite and bulk metals. [2][5]

Under previous research, investigators at MSU have built a machine to test composite material coupons. The machine is called the In-Plane Loader (IPL) and is used to put a combination of in-plane loads on a composite coupon [7][8]. The IPL has the capability to load a composite coupon in two orthogonal directions as well as applying a boundary rotation. Any arbitrary combination of these three displacement modes can be applied simultaneously to a coupon. The IPL was designed with the intent to be able to create a large material database testing composite materials over a large strain domain. The material database presented in the current work was constructed in such a way that it would be transportable between lay-up schedules and different geometries in order to be used as a universal analysis tool.

The intent of the current work was to further investigate the dissipated energy density (DED) method of characterizing the behavior of composite materials outlined in [8]. Using the most current material database produced from the third generation IPL provided new insight into the methods associated with using the DED method. The results presented in the current work represents the most current attempt by the MSU composites group to generate a DED database that can be applied to a composite structure. Success is a function of the ability of the material database to be transported to a different lay-up and geometry. Open-hole tension and bearing compression are the different test configurations that were used to evaluate the current methods. A direct comparison between prediction and experimental data will determine the quality of data being produced by the IPL under the current configuration.

Engineering methods inherently have simplifying assumption built in to make analysis more manageable. One of the assumptions used in [8], namely using linear FEA analysis to resolve the element strains in a damaged coupon was studied. Direct comparison of the results from a linear and a nonlinear finite element models provide information about damage induced strain redistribution within IPL coupons. The comparison assists in determining the validity of this assumption, thus effecting the future direction of DED database research at MSU.

FAILURE CRITERIA AND STRENGTH DESIGN

Strength Criteria for Fiber Reinforced Composites

A vast number of conditions can characterize failure in a structural component. The engineer of structural components must be apprised of all such conditions, and design components that have a low probability of failure. Ultimately, the failure occurs when a structural component can no longer perform its intended function. Many factors may contribute to the failure of a structural component and it is not necessarily true to assume that these factors act independently. Some of the common conditions that drive engineering designs are strength, stiffness, durability, temperature resistance, and corrosion. In theory, the engineer identifies the most critical conditions and the component is designed for those conditions.

Composite materials are unique because they can be tailored for a specific application. The stiffness of a composite can be increased or decreased through the selection of different fiber material and by varying the fiber orientation relative to the primary loading direction [15]. The temperature resistance can be increased or decreased as a result of varying the matrix material [2]. By taking advantage of the design degrees of freedom that are inherent to composite materials, almost any component can be optimized with respect to any of the aforementioned design parameters in order to reduce parasitic losses. As an example, consider a component optimized with respect to strength. The engineer must have two things, (1) the internal loads distribution and (2) a reliable strength criterion. If these conditions are met the engineer can design the

composite to possess only the minimum strength required for success reducing any excess material. If the designer feels confident in predicting the material response to applied loads up to and including ultimate failure, then this optimization process can provide extremely efficient and reliable components. The following chapters will address need for a reliable method to predict strength by reviewing the current techniques for predicting the material response of composite materials under multi-axial loading.

Mechanics of composites provide a method to resolve the applied loads on a given structure into internal stresses and strains. These tensor quantities can be rotated into material principal directions where they are compared with the appropriate strength criteria. There are two common approaches to predicting the damage response of a laminate to static or low frequency loading. The first is based on local fracture mechanisms that initiate and propagate in the laminate. The concepts of fracture mechanics are applied to the matrix material to describe the initiation and propagation of matrix-dominated cracks. The initiation and propagation of matrix cracks can occur both at the fiber-matrix interface and the ply-by-ply interface. In either case, the toughness of the matrix material dominates the ultimate strength of the laminate more than the ultimate strength of the material constituents. The laminate failure is determined when the strain energy release rate reaches a critical value such that the propagation of a matrix crack becomes unstable. Hence, matrix cracking is assumed to dominate the behavior of the laminate. The second approach to progressive failure of a composite laminate is first ply failure (FPF) based on traditional strengths of materials [2]. First ply failure can be use to predict the behavior of the laminate under the following assumptions:

- The response of the laminate can be determined from a ply-by-ply analysis.
- Fiber failure in tension or compression constitutes final failure.

This approach requires the engineer to have a reliable way of predicting the ply strength. Each of the ply principal stresses or strains in the laminate are checked against the strength criteria to determine the critical ply. Each of the plies are assumed to act independently, therefore no interaction between adjacent plies is accounted for. Once the critical ply is identified, the properties of that ply are degraded and the load is redistributed among the remaining plies. In polymer matrix composites the damage associated with the first critical ply is matrix damage. As damage accrues, the loads are redistributed ultimately causing a fiber failure usually coinciding with ultimate failure of the material. This approach to progressive damage works very well for unidirectional loading where the fibers are predominately in the loading direction. It also provides the framework for a method to evaluate composite materials under multi-axial loading assuming that a reliable ply-by-ply strength criterion can be employed.

Currently, many multi-axial ply strength theories have been proposed. Nearly all introductory composites text book will cover the most commonly used criteria including maximum stress theory, maximum strain theory, Tsai-Hill theory and Tsai-Wu tensor theory. These theories are popular because mechanically they are easy to use and they work reasonably well if certain conditions are met. Many other internationally recognize strength theories exist that appeared in a 'World-Wide Failure Exercise' conducted by P.D. Soden, M.J. Hinton and A.S. Kaddour [3]. The purpose of the failure exercise was to expose areas where the failure theories are believed to be relatively mature in their

predictive capability and also determine where the understanding of failure is deficient. Theoretically the exercise provided a tool that design engineers can use as a guide to the most appropriate strength criteria for a given analysis. A number of different conclusions were drawn from the failure exercise. However, with one exception the details will be left to the reader to independently discover. One of the interesting conclusions was that for a unidirectional lamina under a bi-axial state of stress there were no two theories that gave an identical prediction of the failure envelope. Furthermore, predictions within +/- 10% of experimental data were achieved by the best of the current theories on only 15% of the test features [4]. This result is of interest because it alludes to the complexity of materials failure modes. Is it possible that none of the criteria tested in the exercise completely capture and correctly represent the failure modes? Are the failure modes so complex that a unified failure theory is not attainable? Open questions such as these provide the motivation for the current work.

Multi-Axial Strength Theories

The most popular strength theories were listed without explanation in the previous section. The present section will consider each of the strength theories independently providing sufficient information to obtain a working understanding of each of them. Composite material failure theories are challenged with the task of integrating the various failure modes into a single criterion that adequately describes the material response. This is no easy task considering the interaction of multiple different modes of failure in a polymer matrix composite. Not to mention the interaction and transition between each

failure mode. Some of the common failure modes that composite material exhibit are outlined below [2].

- Fiber Breaking: Usually occurs during tensile loads and may initiate at stresses much less than the ultimate stress.
- Matrix Cracking: The brittle nature of the matrix material is suceptable to cracking caused by mechanical loads, curing induced residual stresses, thermal stresses, or aging.
- Fiber Debonding: The interface between the fiber and the polymer matrix fails.
- Delamination: The separation between plies in a laminate. Delaminations are caused by inter-laminar stresses that exceed the matrix material capability. Compression induced delaminations can be extremely catastrophic because of the deleterious effect on section properties once the material becomes discontinous.

Each of the failure theories attempt to create a failure envelope that represents termination of the linear elastic behavior under multi-axial stress state. Unfortunately, failure envelopes generally tend to mask the actual material phenomena that occurs on a local scale sometimes at lower than critical stress levels. Therefore, it is difficult to account for all microscopic failure mechanisms. An arbitrary failure surface in ply principle stress or strain axes is represented graphically in Figure 3. A curve such as this can be used to identify a critical ply within a laminate. If the stress or strain at a specific location in a structural component is within the bounds of the onset of damage envelope the material at that location is considered to be undamaged. If the stress or strain is outside of the bounds of the onset of damage envelope but within the failure envelope

then some amount of damage is assumed to have accrued at that location. Lastly, if the stress or strain at a point in the material is outside the failure envelope the material is considered to have lost all structural integrity at that location.

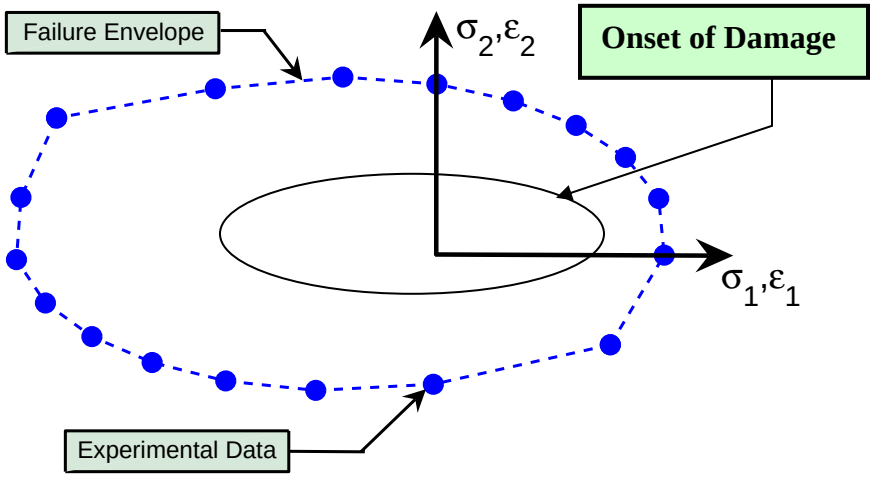


Figure 3. Arbitrary failure envelope in 2D stress or strain space.

Maximum Stress Theory. The failure envelope for the maximum stress theory is posed as a set of three inequalities displayed as equation (1.1). Each inequality corresponds to one of the three material principal stress components. To be definite assume that a composite evaluated with the maximum stress theory is loaded in such a way to produce significant stresses in the 1-2 plane, and negligible stress in the 2-3 and 1-3 planes. This configuration is given the name plane stress and will be referred to as such here after. The coordinate system in Figure 3 represents plane stress when the shear stress axis τ_{12} extends out of the page. The extension into third dimension enables all three in-plane stresses to be represented. The parameters X, Y and S in equation (1.1) represent the failure stresses along the 1, 2 and 1-2 axes respectively. The failure stresses in compression are written as negative quantities consistent with standard engineering

sign convention. The subscript t and c distinguish between tensile or compressive failure stresses. If any one of the inequalities are false then damage initiation is said to have occurred. It is important to note that there is no consideration for interaction between stresses, such that all modes of failure are considered to be independent. The failure surface is therefore represented in three dimensional stress space as a cuboid. Pointed out in [2], maximum stress theory may not be conservative for stress states that are not dominated by a single stress component.

$$\begin{aligned} \sigma_1 < X_t \ \ or \ \ \sigma_1 > X_c \\ \sigma_2 < Y_t \ \ or \ \ \sigma_2 > Y_c \\ \left|\tau_{12}\right| < S \end{aligned} \qquad (1.1)$$

Maximum Strain Theory. The maximum strain theory is similar to the maximum stress theory. Once again, the material is considered to be damaged if one of three inequality equations is false. The set of strain inequalities displayed as equation (1.2) are analogous to the set of stress equations used in the maximum stress theory.

$$\begin{aligned} \varepsilon_1 < X_{\varepsilon t} \ \ or \ \ \varepsilon_1 > X_{\varepsilon c} \\ \varepsilon_1 < X_{\varepsilon t} \ \ or \ \ \varepsilon_1 > X_{\varepsilon c} \\ \left|\varepsilon_{12}\right| < S_\varepsilon \end{aligned} \qquad (1.2)$$

The parameters X, Y and S represent the failure strains and are differentiated from the failure stresses in equation (1.1) by the subscript ε. The failure strains in compression are written as negative quantities consistent with standard engineering sign convention. The subtle difference between maximum stress theory and maximum strain

theory is that the later of the two includes Poisson's effects. A detailed example illustrating the difference can be seen in Jones [1]. Both the maximum stress theory and the maximum strain theory have the advantage that the mode of failure can be somewhat identified through the use of three independent inequality equations.

Tsai-Hill. The development of the Tsai-Hill equation is an extension of Von Mises' isotropic yield criterion. The Von Mises' yield criteria also referred to as the distortion energy theory that relates the stresses to the amount of energy required for material distortion excluding the energy of volumetric dilatation. Assuming that plane stress applies, the Von Mises' stress written in terms of the principal stresses is given as equation (1.3).

$$S_y = \sqrt{\sigma_1^{\,2} + \sigma_2^{\,2} - \sigma_1\sigma_2} \qquad (1.3)$$

A graphical interpretation of Von Mises' yield criterion can be realized by rewriting equation (1.3) as

$$S_y = \frac{(\sigma_1 + \sigma_2)^2}{4} + \frac{(\sigma_1 - \sigma_2)^2}{4/3} \qquad (1.4)$$

and upon considering the stress axes as in Figure 3 is recognizable as an ellipse. For isotropic materials there are no material couplings (i.e. shear-extension), thus the distortion and dilatation are independent. For the general case of an orthotropic material, the distortion can not be separated from the dilatation because of material couplings, thus energy required for material distortion is inseparable from the energy required for volumetric dilatation. The Tsai-Hill criterion, equation (1.5), written for plane stress acting on an orthotropic material is not directly related to the energy of distortion. The

same lamina strengths used in Maximum Stress Theory are used in the Tsai-Hill equation.

$$\frac{\sigma_1^{\,2}}{X^2}-\frac{\sigma_1\sigma_2}{X^2}+\frac{\sigma_2^{\,2}}{Y^2}+\frac{\tau_{12}^{\,2}}{S^2}=1 \qquad (1.5)$$

The Tsai-Hill criteria have some advantages over the two previous theories. Namely, an interaction between failure strengths is taken into account, therefore the modes of failure are not considered independent. It is also pleasing that under consideration of an isotropic material the Tsai-Hill equation reduces to the Von Mises' criteria which correctly predicts the behavior of quite a few ductile materials. The later of the advantages is not surprising knowing the origin of the Tsai-Hill equation, however it is listed as an advantage because it has been extrapolated from a mature existing theory.

The disadvantages of the Tsai-Hill criteria lie in the fact that the energy of distortion and dilatation can not be separated for an orthotropic material, thus there is no distinction between tension and compression failure which can be significantly different in polymer matrix composites.

Tsai-Wu. Tsai-Wu is the most popular failure criteria for polymer matrix composites [1]. Tsai-Wu represents improvement over all other previously discussed criterion [15]. First, by including more terms in the predicting equation the result is a better fit to experimental data. Secondly, failure in tension and compression are distinguishable. Lastly, strengths are represented as tensor quantities that are well defined mathematical entities. The criteria written for full three dimensional stresses in contracted tensor notation is

$$F_i\sigma_i + F_{ij}\sigma_i\sigma_j = 1 \quad \textit{where} \quad i, j = 1,\ldots,6 \tag{1.6}$$

where F_i and F_{ij} are strength tensors of second and fourth rank respectively. Definition of contracted tensor notation can be seen in [1]. For consistency with the previous strength criteria the Tsai-Wu equation will be written for an orthotropic material under plane stress. Writing it as such provides not only a direct comparison with the previously suggested criteria, but also allows for dissection of the equation into three distinct parts convenient for gaining a physical understanding of the criteria.

$$F_1\sigma_1 + F_2\sigma_2 + F_6\sigma_6 + F_{11}\sigma_1^{\,2} + F_{22}\sigma_2^{\,2} + F_{66}\sigma_6^{\,2} + 2F_{12}\sigma_1\sigma_2 = 1 \tag{1.7}$$

The first three terms reading from the left are linear in stress and represent the normal strengths in tension/compression and shear. Each of these terms contain one of the tensor strength components F_1, F_2 and F_6 which are analogous to X, Y and S in the preceding failure criteria. The next three terms are second order in stress and represent the ellipsoidal shape of the Tsai-Wu in stress space. The last term is also second order in stress and represents an interaction between the normal stress in the one direction and the normal stress in the two direction. This interaction is somewhat unconventional due to the fact that it is not through shear stress. The failure strength F_{12} represents the failure interaction between two normal stresses and therefore cannot be determined by a single uni-axial test. A bi-axial test is required in order to determine the strength component F_{12}. The Tsai-Wu criterion has the following advantages:

- Provides good agreement with experimental data [1].
- Invariant under rotation of coordinates.

- Strength parameters are defined as tensor quantities that can be manipulated with well known mathematical operations.
- Strength tensors have symmetric properties parallel to the properties of the stiffness and compliance tensors [1].

The disadvantage of Tsai-Wu is in the determination of the strength component F_{12}. Uni-axial tests are much easier to run with test machines that are common in any materials research laboratory. An example of a bi-axial testing machine can be seen in reference [6].

MSU IN-PLANE LOADER (IPL)

Isotropic vs. Anisotropic Material Testing

Characterization of composite materials presents some challenges over traditional engineering materials. Traditional engineering materials are generally isotropic and can commonly be represented using linear mathematical models. Material characterization of isotropic materials can be done by completing single axis tests on any standard uni-axial test machine such as those made by the Instron Company. Since isotropic materials have infinitely many planes of material symmetry, a single uni-axial test can be used in infer the behavior of the material in any desired direction. Failure of these materials generally falls into one of two categories, ductile failure or brittle failure. Each of these failure modes have relatively mature engineering analysis techniques developed in order to design acceptable components when either ductile or brittle failure is a concern. Composites on the other hand are generally orthotropic and often exhibit nonlinear behavior with interacting failure modes much more complex when compared to isotropic materials. Consequently, there are difficulties associated with characterizing composite materials under general bi-axial loading [19]. Uni-axial tests similar to those applicable to isotropic materials are no longer representative of a composite material. Material couplings and complex failure modes are the challenges that composite materials represent for characterization under general bi-axial loading. From the review of generally accepted strength theories in the previous chapter it is clear that with the exception of Tsai-Wu the strength methods rely completely on the results from uni-

axially loaded test coupons. For these strength theories there is no need for a complex test machine or test procedures when uni-axial test procedures have already reached a significant level of maturity for isotropic material and can be borrowed for testing composites with reasonable effort. Bi-axial failure theories survive on test data primarily obtained on a uni-axial test machine using coupons with different material directions.

Composites research at Montana State University (MSU) under the direction of Dr. Doug Cairns has taken a slightly different approach to characterizing composite materials from those described in the previous chapter. Current efforts have focused on using dissipated energy density (DED) as a metric to modeling progressive damage in composite materials. The method is semi-empirical and requires test data from composite coupons loaded in such a manor to produce bi-axial strains within the test coupon. Thus, defining one important application of the MSU In-Plane Loader developed to test composite material under complex loading. The details of the DED method will be explained completely in the next chapters.

Explanation of IPL

The In-Plane Loader (IPL) was designed and built as a senior design project at MSU. After the initial design was completed a substantial number of modifications to the test machine have been completed by other undergraduate students and graduate students. The most recent modifications and qualification of the IPL was completed by Aaron Collett to earn a masters degree completed in May 2006. A detailed explanation

of the IPL will be left to the reader to discover in references [7] and [8] however a general explanation will be given in the current work for the sake of completeness.

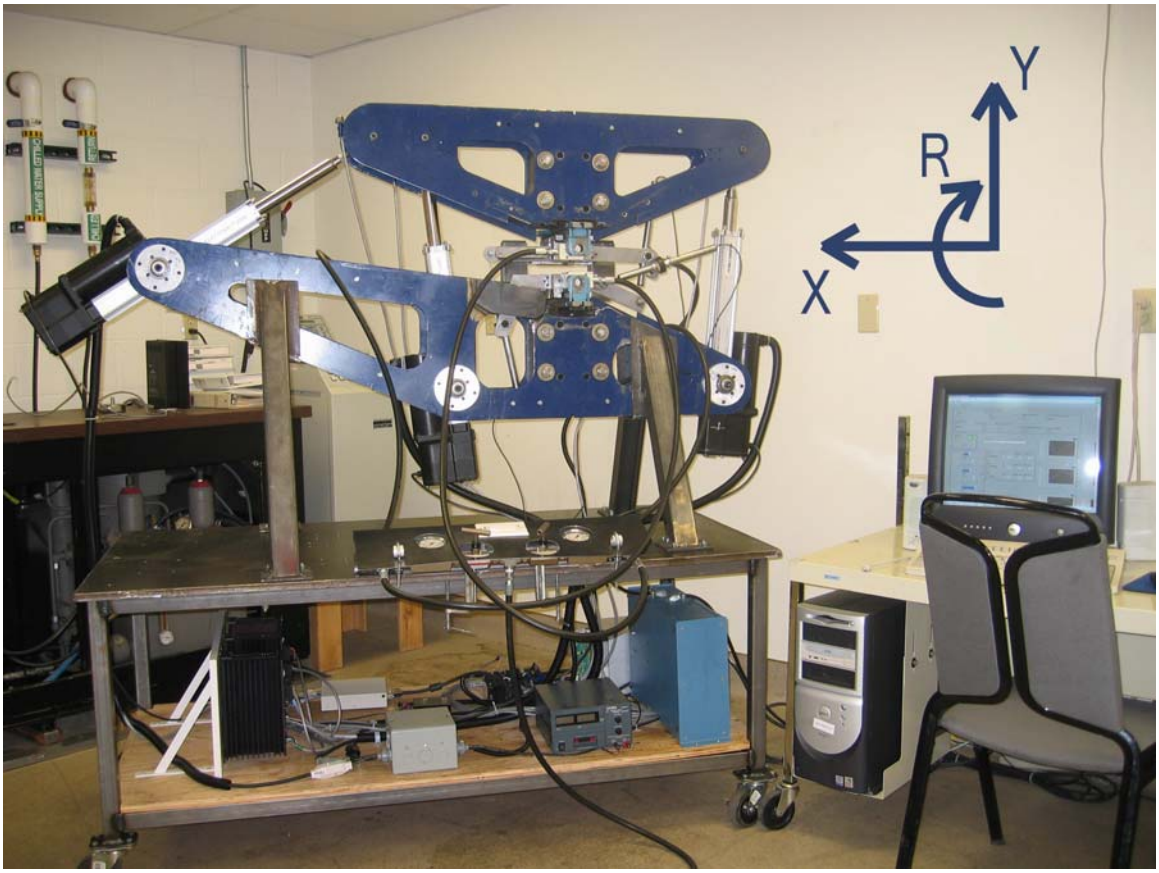


Figure 4. In-Plane Loader (IPL) at Montana State University.

The need at MSU for a bi-axial test machine is currently being filled with the IPL test machine seen in Figure 4. The IPL consists of four major components (1) the frame, (2) actuators, (3) displacement acquisition system and (4) load acquisition system. The upper portion of the frame moves independently from the bottom portion of the frame that is fixed to the table. The upper frame is positioned in space by three actuators controlled such that the upper frame is capable of any planar displacement as well as rotation. The test coupon is clamped between the upper and lower portion of the frame

such that loads are transferred to the coupon as the upper frame moves. The coordinate system displayed in Figure 4 provides a reference for all subsequent IPL explanations and test results.

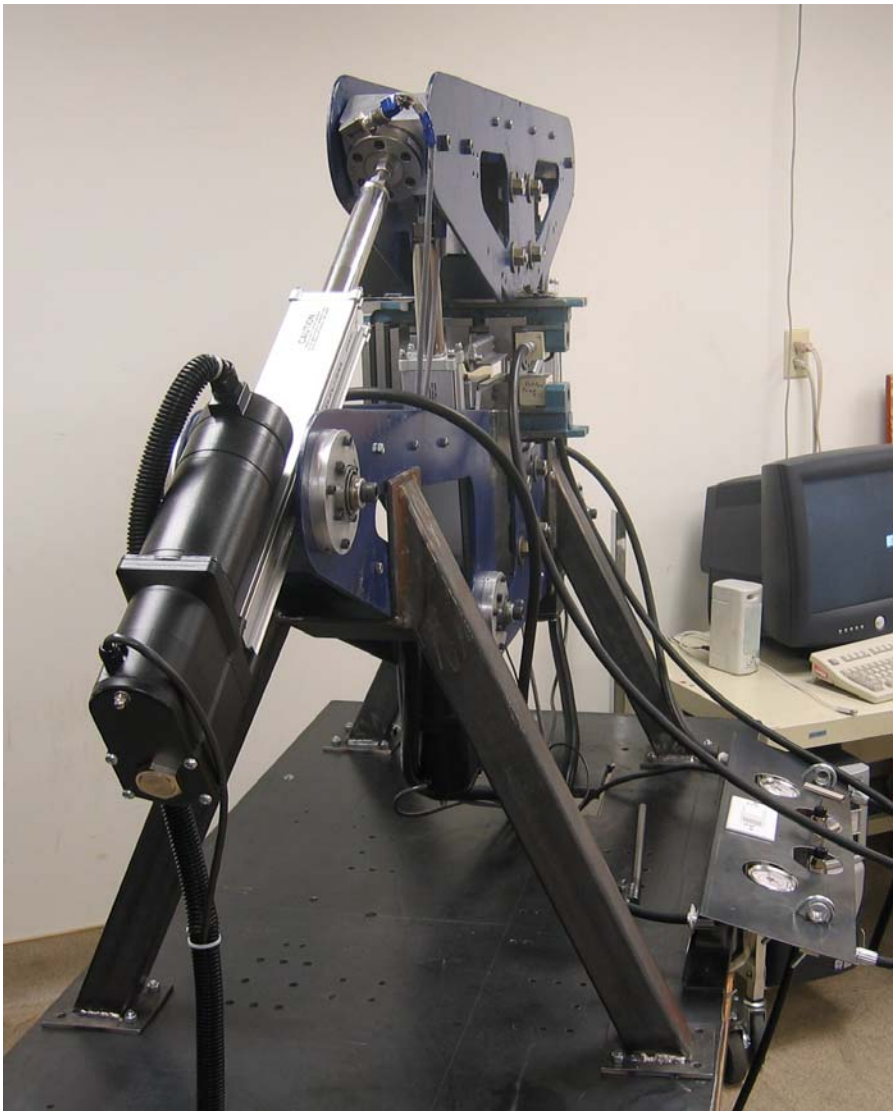


Figure 5. End view of IPL multi-axial test machine.

The sole purpose of the frame is to transfer actuator displacements to the test coupon. The bulk of the frame is composed of two 6.35 mm steel plates held approximately 10 cm apart by strategically spaced steel spacers. An end view of the IPL

can be seen in Figure 5. The upper and lower frame are held aligned in a plane by an in-plane constraining device [7]. The frame of the IPL was designed sufficiently for strength but lacks the desired stiffness of a frame used in a material test machine at high loads. As a result a significant amount of strain energy may be stored in the frame during a test, depending on the magnitude of the loads. The machine compliance is most deleterious from a controls point of view and in theory is independent of the results of a test.

There are three stepper motor driven ball screw actuators that move the upper frame relative to the lower frame. Each of the actuators is rated to a maximum load of 8900 N. Therefore, the load capacity in the X and Y direction is approximately 17900 N and 8900 N respectively. The actuators are attached to the upper and lower frame in strategic positions such that pure coupon displacements require coupled input from all three actuators. Each of the actuators has a manufacture defined position control accuracy of $0.5\,\mu m$.

The displacement acquisition system contains linear variable differential transformers (LVDT's) that determine the displacement of the upper frame relative to the lower frame. The LVDT's as well as the actuator motor drivers are coupled to a personal computer (PC) through a Labview data acquisition unit. Displacement control is executed through a feedback loop which continues until the actual position of the upper frame relative to the lower frame matches the desired relative position within a specified convergence tolerance. The system kinematics are resolved using vector loop equations solved simultaneously using Matlab for each convergence step. Once position

convergence is met for each discrete displacement step the corresponding load and relative frame position data is written to a file and stored on the PC. An arbitrary test path is thus prescribed by the engineer by specifying a number of discrete coupon boundary displacement steps. Displacement control has been developed in this manor to insure accurate motion of the upper frame.

Load readings are acquired from pancake load cells mounted in-line with each actuator. The loads are resolved into X, Y and R directions using the systems kinematic position equations. Each individual load cell requires signal conditioning in the form of amplification to be accurately read by the Labview acquisition unit . The load cells on the IPL were calibrated against an Instron load cell and are known to be accurate up to 22.2 kN.

Third Generation IPL

The third generation IPL has a couple of slight modifications making it differ from the second generation IPL which is described in [7]. This section provides a mere overview of the current IPL and does not contain any data to substantiate the modifications to the IPL. The following modifications have been realized in the third generation IPL.

1. Grips with improved abrasive gripping surface.
2. Upgraded hydraulic system for increased gripping pressure.

A new set of grips with an improved abrasive gripping surface have been implemented. The surface treatment is referred to as Carbinite coating. Carbinite is a carbide alloy

coating that is metallurgically bonded to metal components using a process called electrofusion [9]. The decision to experiment with Carbinite coatings was a result of a telephone interview with Dr. Donald Adams, president of Wyoming Test Fixtures, Inc., a company specializing in the design and fabrication of mechanical test fixtures for composite materials [10]. The company that applied the abrasive grip surface, Carbinite Metal Coating, boasts a hardness above 70 Rockwell making the coating surface extremely wear resistant. Carbinite Metal Coating offers five different grades of coating, each grade representing a different grit. Based on [10] both 60 and 100 grit surfaces have been purchased by the MSU composites group for the IPL. The five discrete grits can be seen in Figure 6 together with a grip used with the IPL. The potential to reduce the relative slip between the grips and the composite coupon as well as provide a consistent gripping surface is likely to be obtained with the implementation of the new grips.

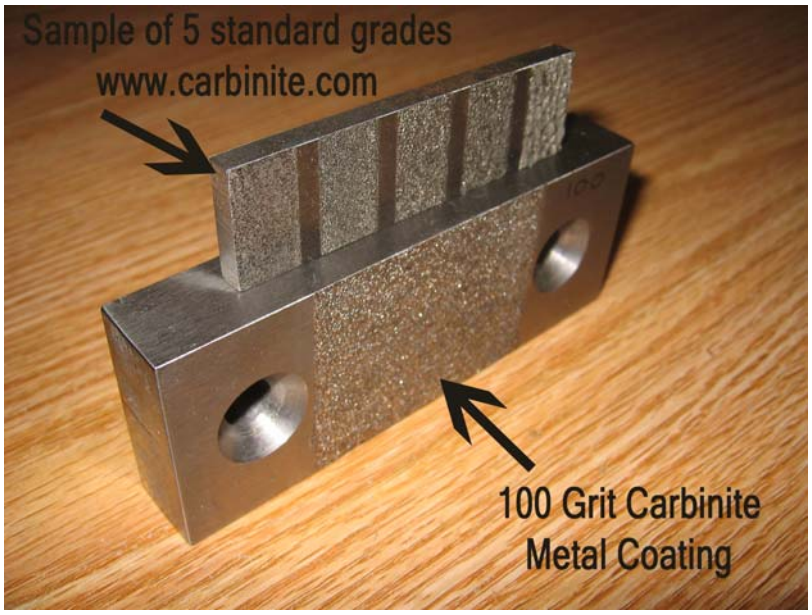


Figure 6. Carbinite coating provides an abrasive gripping surface for composite IPL test coupons.

In addition to new grips, the IPL was fitted with a new hydraulic pump (Enerpac PA133) and associated hydraulic components to increase the gripping pressure on test coupons. The previous hydraulic system had a maximum line pressure of 20.7 MPa resulting in a gripping pressure at the coupon of approximately 23.5 MPa. The new hydraulic system provides 3.3 times the available gripping pressure at the coupon with a maximum line pressure rating of 69.0 MPa. In addition to increased gripping pressure, the new hydraulic pump has the advantage of being pneumatically operated opposed to the hand operation of the old hydraulic pump. The pneumatically operated hydraulic pump is actuated by foot leaving the operator with both hands free to ensure proper test coupon alignment. This is a significant advantage when considering the number of tests that are required to generate a material database.

Test Coupon

Fiberglass has emerged as a material worthy of consideration in many engineering designs such as wind turbine blades, aircraft, marine vessels, and sporting goods to name a few. Therefore, a demand exists for methods to increase the reliability of fiberglass components. Materials become reliable for use in a design through a pyramid testing scheme starting with coupons of simple geometry. Each subsequent level of testing uses increased coupon size and complexity until the test of a full scale production part test. This methodology has been used primarily in the aircraft industry where financial constraints are much less of a concern when compared to some of the other industries with potential for applications of fiber glass. Thus, the IPL was designed to meet this

demand to characterize composite material increasing the reliability of the material eliminating expensive large scale tests. Accurate material characterization is obtained by testing many composite coupons over a large range of in-plane strains representative of those experienced by the component in service. The IPL provides a data rich environment for which a designer can use as the basis for making design decisions. Hence, E-glass/epoxy was selected as the material used in the current work based on the availability at MSU with the focus on increasing the reliability of the material for use in the wind turbine industry.

Manufacturing

All of the coupons used in the current work were manufactured using CDB200 triax fabric and Epon resin 826 EPI-CURE 3234 curing agent was mixed 7:1 with the Epon resin to initiate cross-linking in the matrix. Hand lay-up was used exclusively to build up the thickness of the laminate. The coupons used for IPL tests consisted of 2 fabric plies placed such that a lay-up of $(0/45/-45)_s$ was obtained. The zero direction fiber was placed away from the neutral axis to increase the bending stiffness of the laminate and reduce the potential for local forced crippling in the test coupons. The open-hole compression (OHC) and bearing tension (BT) coupons were constructed using the same material constituents as the IPL coupons, however, four times as many fabric plies were included in the laminate to obtain $(0/45/-45)_{4s}$. Using an empirical formula (1.8) developed by the MSU composites group the fiber volume fraction was found for the test coupons.

$$V_f = \frac{0.274mm}{t_{ply}} \qquad (1.8)$$

The fiber volume fraction V_f of 36 percent was calculated based on a fabric ply thickness t_{ply} = 0.762mm. Equation (1.8) is valid for the current material system using hand lay-up techniques.

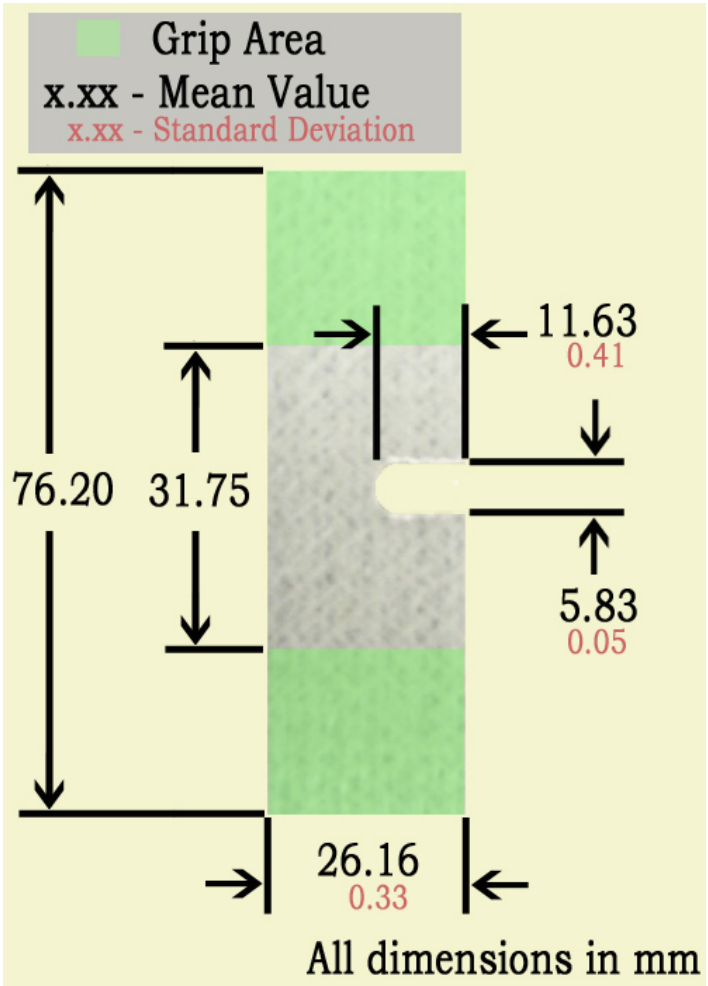


Figure 7. IPL coupon geometry used to develop the material database. The lay-up was $(0/45/-45)_s$.

Geometry

At first glance, the notched coupon geometry selected for the IPL is unusual. However, the notched coupon geometry does have a logical basis which requires some explanation for clarity. The coupon geometry roughly resembles that of a more traditional compact tension sample commonly used to determine the fracture toughness of a material. This impression can mislead an engineer into thinking that the IPL is used to get the fracture toughness of the composite material. The IPL coupon geometry seen in Figure 7 was used for two specific purposes, (1) minimize the effects of gripping the test coupon and (2) under a prescribed boundary displacement provide a significant strain gradient within the test coupon.

The necessity of the first consideration is a result of the configuration of the displacement acquisition system. The IPL has many unique features differentiating it from other material testing machines. One of which is the way the gage section or test section of the coupon is defined. The gage section on the IPL is defined by the distance between the upper and lower grip edges. This differs from traditional test coupons where it is desirable to locate the test section away from the location of displacement application. The grip edges provide a region of the coupon for which test boundary displacements are determined under the assumption that displacement continuity exists between the grip edges and the coupon material. The validity of this assumption is realized only if a region of increased compliance exists within the test coupon. Information about the interface between the grips and the test coupon as well as the required compliance of IPL coupons can be found in [7]. The notch geometry effectively

provides a region of increased compliance essentially a strain concentration. Since the strains are assumed to be much larger at the notch than they are at the grip edges the three dimensional effects of gripping the coupon are neglected in the analysis.

The second consideration is related to development of the DED function and will become clear after reviewing chapter four. The strain concentration around the notch creates a significant strain gradient in the coupon. Hence, in-plane strains that are much greater than they would be if the notch were absent and the coupon was rectangular for example. Experimental tests containing larger magnitude strain values increase the applicability of the derived DED function reducing the need to extrapolate outside of the experimental strain values.

Linear FEA Model

The IPL coupon geometry and the boundary displacements are somewhat more complicated than most uni-axial material coupon tests. For this reason higher order techniques are required to analyze IPL data. The complexities aforementioned eliminate any chance of simply resolving the strains in the test coupon by assuming a distribution of strain and using a calculation such as the quotient of the change in length over the original length. Hence, finite element analysis is used to resolve the strain distribution within the coupon.

ANSYS was the preferred finite element code for the current work based on its accessibility at MSU. The general procedure was to build three independent linear models differentiated by only the applied boundary displacements X, Y and R respectively. Using linear finite element models enabled the superposition of strain fields

from each of the models to create the strain field for an arbitrary combination of X, Y and R boundary displacements. The validity of a linear finite element model representing strain fields within a coupon that has undergone obvious damage is no doubt questionable. Contrariwise, the reader must be apprised of a potential underlying paradox. Figure 8 contains photographs of the damage zone within an IPL test coupon. A linear model is undoubtedly valid for the material in the absence of the red damage zone. The question lies in the deviation from linear behavior caused by the material within the red damage zone. It is important to keep in mind that all the IPL test are conducted under displacement control and the finite element modeling is only necessary to resolve the strain distribution due to the changes in the coupon geometry. Material continuity exists at the boundary of the damage zone independent of the damage progression. Since the nature of damage is primarily matrix cracking and ply delamination, practically speaking the strain field may or may not deviate significantly from the undamaged material. The point being, it may not be as far fetched using a linear elastic strain field to develop the DED function as originally perceived. As unpleasing as it may be, in the interest of practicality, the assumption that linear finite element models produce strain fields that are "good enough" has been made. The linear paradigm exists under the premise that the resulting DED function be verified through the results of case studies including open-hole compression and bearing tension.

Three linear IPL coupon models were written in ANSYS finite element code. Each of the models was two dimensional models under the assumption of plane stress. Each one of the models were built in ANSYS. There are three primary element types

designed to model a layered composite material, Shell99, Shell91 and Shell181. A brief description of each element will be given as justification for element selection for the current work.

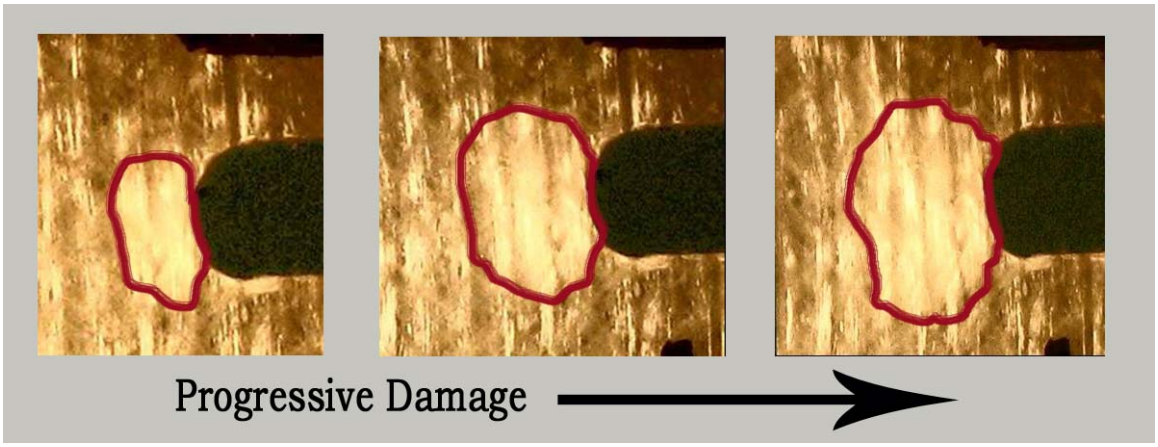


Figure 8. Contrasted photograph of E-glass/epoxy test coupon to illustrate the damage zone.

1. Shell91 - is and 8-node, 3D shell element with 6 degrees of freedom at each node. This element allows only 100 layers maximum that can have differential thickness varying bi-linearly over the area of the layer. Shell91 supports plasticity, large-strains and has an option for modeling sandwich construction composites. It is much more robust compared to Shell99 for modeling large deflections.

2. Shell99 - is an 8-node, 3D shell element with 6 degrees of freedom at each node. It is designed to model moderately thick plate and shell structures. The element allows for a maximum of 125 layers with thicknesses that vary bi-linearly over the area of the layer. Shell99 uses less computational time compared to Shell91 when running a linear model with more than 3 layers. Shell 99 has nonlinear capabilities that are slightly inferior to Shell91.

3. Shell181 - is a 4-node, 3D shell element with 6 degrees of freedom at each node. A maximum of 250 layers are available for modeling somewhat thicker plate and shell structures. Shell181 has full nonlinear capabilities.

The decision was made to use Shell99 elements to take advantage of the reduced computational time running linear models in the absence of nonlinear effects. Shell99 is a 3D element with capabilities to calculate third dimension strains using full orthotropic material constitutive parameters, however, none of the out-of-plane capabilities were necessary for the current work and hence the models will be referred to as 2D from this point forward.

In order to maximize computational efficiency while determining the DED function, it was important to minimize the number of finite elements used to model the IPL composite test coupon. Using the same element mesh for each of the three linear models differentiated by each of the three pure displacement modes was equally important for superposition. Thus, a linearly graded mesh density was used to minimize the number of elements. The element density was a maximum at the notch tip and decreased as the radial distance from the notch tip increased. Mesh convergence studies were preformed on all of the models to ensure appropriate element density where large gradients in strain existed within the test coupon. Based on the procedures defined in previous work [8], finite element strains resulting from any arbitrary combination of boundary displacements were created through linear superposition of the three pure models X, Y and R. However, a study conducted as a part of the current work revealed that linear superposition of the rotational displacement component lead to errors in strain

values in excess of 10%. As a result, separate rotational displacement models were ran for an array of displacements sufficient for all of the IPL test paths. Linear interpolation between separate rotational displacement models was used to obtain the strains for an arbitrary rotational displacement. Thus, the strain field under combined boundary displacement was calculated using superposition of the strains from the linearly scaled X and Y component models and the linearly interpolated R component models. Representative strain contour plots obtained for X, Y and R boundary displacements are presented in Figures 9, 10 and 11 respectively.

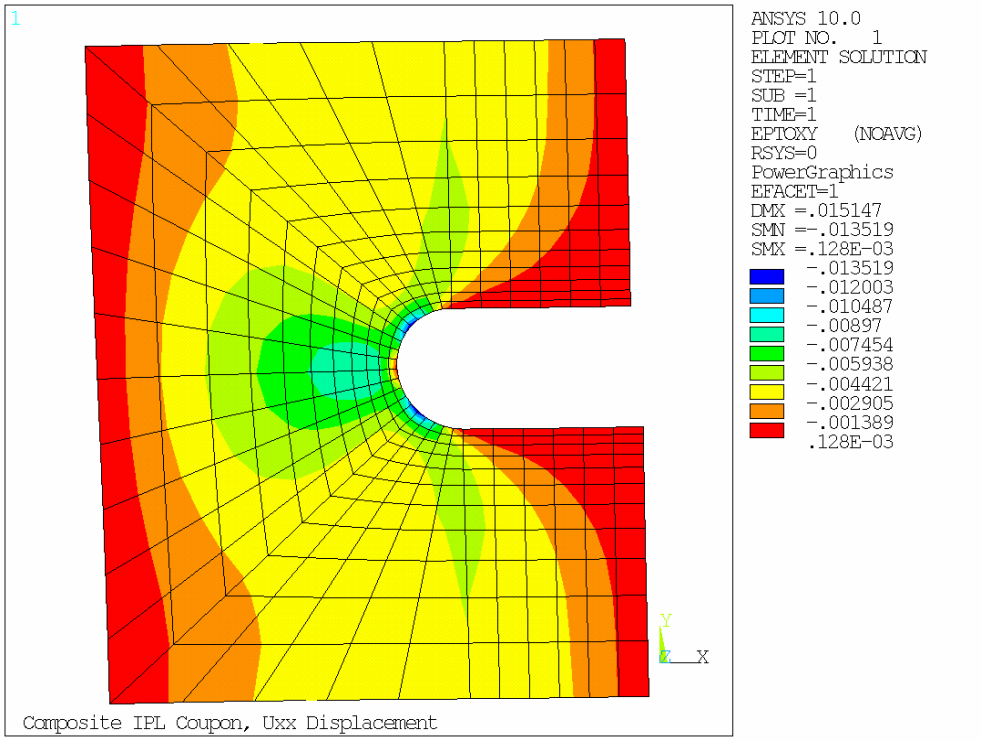


Figure 9. IPL coupon XY shear strain contour under pure negative X boundary displacement.

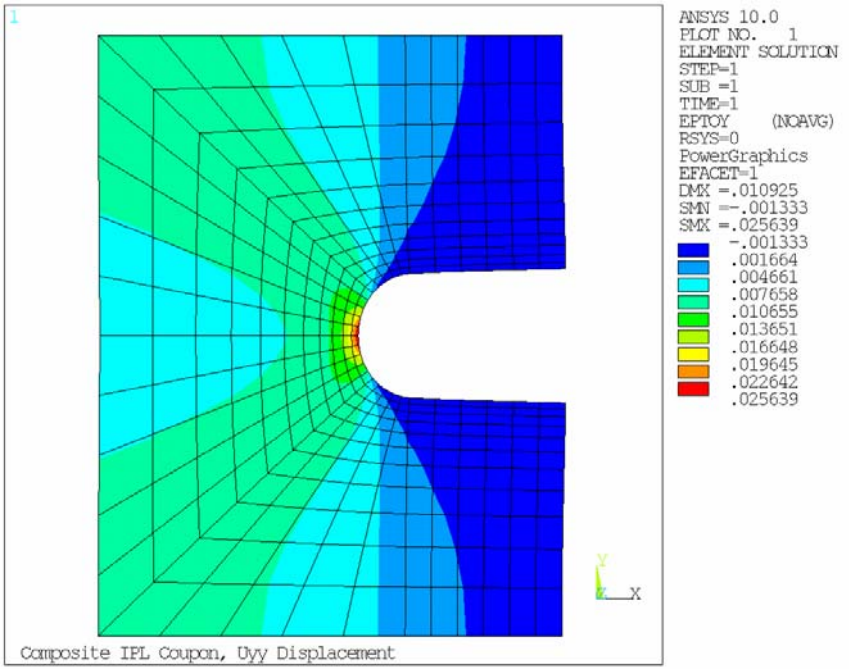


Figure 10. IPL coupon Y-direction normal strain contour under pure Y boundary displacement.

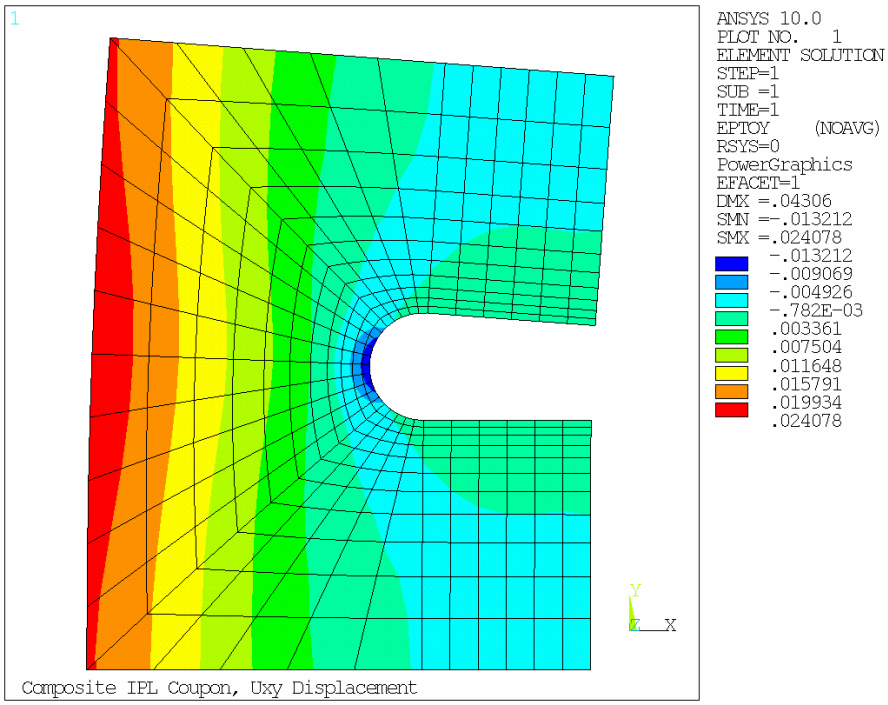


Figure 11. IPL coupon Y-direction normal strain under pure negative rotational (R) boundary displacement.

Material Database

Accurate determination of the DED function required a significant amount of material test data. The IPL was originally designed for this purpose and therefore was used exclusively to derive the necessary data for the current work. The initial step in creating the material database was to define a series of load paths. Each load path represented a unique combination of the three pure displacements X, Y and R. The load paths used to generate the material database are represented graphically in Figure 13 and can be seen in tabular form in Table 1. The axes in Figure 13 represent the three pure displacements directions and the points on the unit displacement sphere represent a normalized combination of pure displacement components used to build the material database. The material database consists of a collection of load-displacement data for a single material system. A representative set of load-displacement curves are shown in

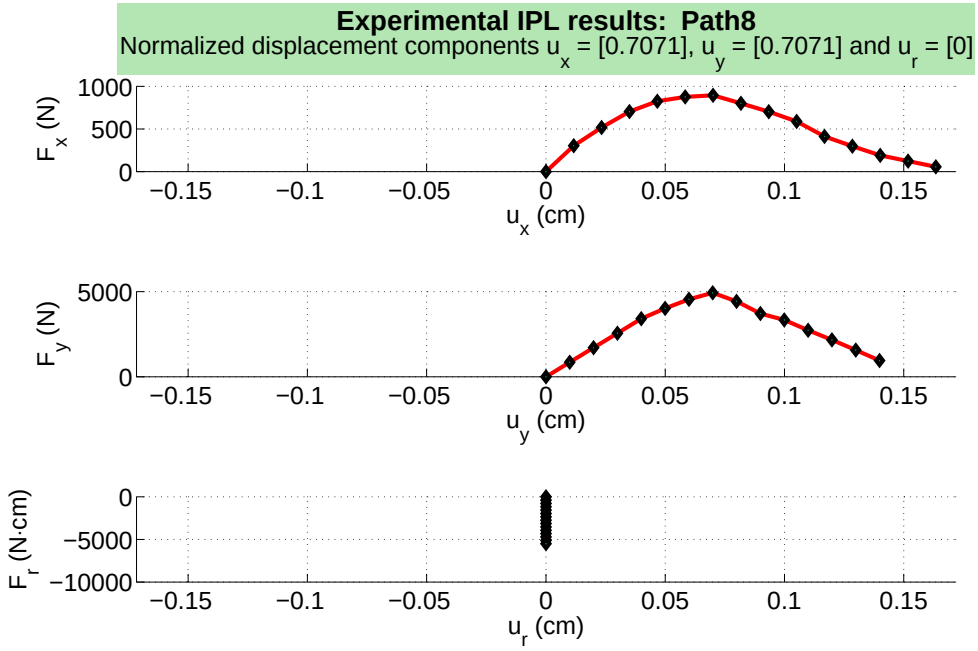


Figure 12. IPL test coupon under combined X and Y boundary displacement.

Figure 12 for load path 8 representing equal X and Y boundary displacements. Each load path within the database produces a set of load-displacement curves, one curve for each of the three different boundary displacements. The complete set of experimental load displacement curves for load paths 1-17 can be seen in Appendix A.

Coupon tests for each load path were repeated only one time for a total of two tests per load path. The decision to do so was based on the lengthy duration of each test. The configuration of the IPL at the time of this work was not optimized around repeated testing and thus the cycle time for each test was approximately thirty minutes. The material database was developed over a period of approximately seventeen hours. Fortunately, after the testing concluded the reduction of the test data revealed that the load path data from the two tests was sufficiently precise to use the average as a representative data set. Statistical analysis beyond the scope of the current work would be necessary to show that the average data was representative of the population.

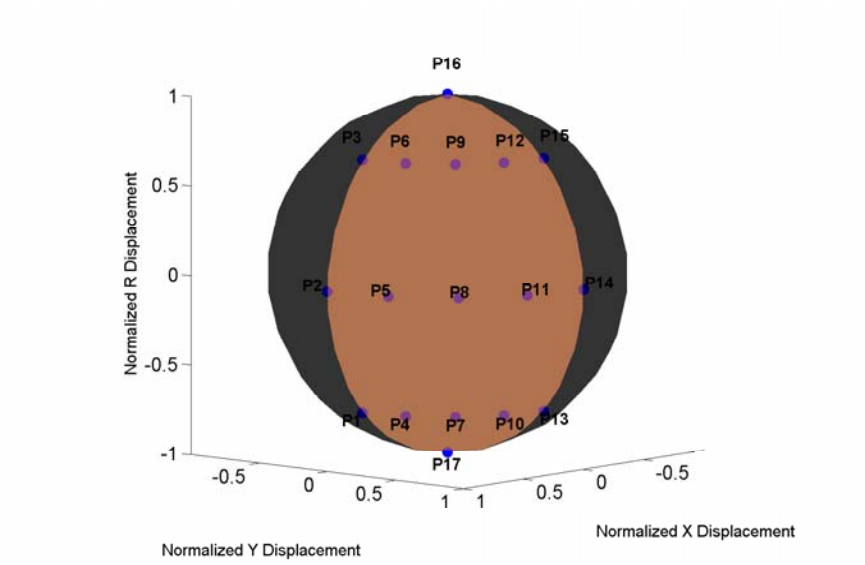


Figure 13. Graphical representation of 17 different load paths in normalized IPL displacement space.

Table 1. Normalized IPL displacement components.

Load Path	ND-X	ND-Y	ND-R
1	0.7071	0	-0.7071
2	1	0	0
3	0.7071	0	0.7071
4	0.6533	0.2706	-0.7071
5	0.9239	0.3827	0.7071
6	0.6533	0.2706	0.7071
7	0.5	0.5	-0.7071
8	0.7071	0.7071	0
9	0.5	0.5	0.7071
10	0.2706	0.6533	-0.7071
11	0.3827	0.9239	0
12	0.2706	0.6533	-0.7071
13	0	0.7071	-0.7071
14	0	1	0
15	0	0.7071	0.7071
16	0	0	1
17	0	0	-1

EXPERIMENTAL DISSIPATED ENERGY FUNCTION

DED Numerical Model

The general procedure to find the dissipated energy density function after obtaining the required test data is conceptually not that difficult. The complexity arises in the obvious bookkeeping required with such an abundance of test data. Dissecting the current method of determining the DED function reveals nothing but an obscure curve fitting technique. The fact that the current method is based on actual multi-axial test data distances it from many other composite damage models. The inherent complexities of composite materials contained in both the modes of failure and the constitutive material response results in numerical models as a favorable approach compared with analytical methods. The following section outlines the procedure required to obtain the DED function and provides an explanation of the computations that are included as a part of the current work.

General Approach

The following paragraphs will describe the solution techniques used in the current work. The goal of this method was to use data from the IPL in its current configuration to experimentally determine the DED function for a material system. At the onset, the DED function was assumed to have the form displayed as equation (1.9).

$$\phi(\bar{\varepsilon}) = \sum_i C_i N_i(\bar{\varepsilon}) \qquad (1.9)$$

In equation (1.9), the strain vector $\bar{\varepsilon}$ is composed of the three components of in-plane strain, namely, (ε_{11}, ε_{22}, ε_{12}). The notation used to display the strain components implies that principal material strains are being considered. Thus, the DED database is generated independently of a specific laminate. The other parameters in equation (1.9) are the scalar quantity ϕ representing the dissipated energy density, N_i representing a set of basis functions, and C_i a set of unknown coefficient to be determined from the experimental data. Equation (1.9) is valid over a domain S that will here after be synonymous with the "solution domain." The domain is defined by the extreme values of experimentally tested strain values. There are distinct similarities between the method being described in the current work and what is described in reference [11]. The approach in the current work can be differentiated by the method for which the unknown coefficients C_i are calculated. At this point, it should be clear that the determination of the DED function has now been reduced to determining a set of unknown coefficients C_i. The remainder of the section will outline a logical procedure for obtaining these unknown coefficients using the material database generated with the IPL.

The DED function form is a series representation and similar to other series representations the difficulty lies in the determination of unknown coefficients. The aforementioned determination can only begin once a set of basis functions has been selected. For the current work, two sets of basis functions were considered, (1) linear interpolation functions and (2) quadratic interpolation functions. These sets of interpolation functions are identical to those commonly used in finite element modeling. The consideration of the two different interpolation function sets resulted from

information contained within reference [8]. Based on the findings of a preliminary study comparing the two different sets of basis functions the later of the two sets was abandon based the increased complexity without significant foreseeable benefits. Therefore, linear interpolation functions were used as the basis functions in equation (1.9).

Piecewise Interpolation of DED

The functional relationship between $\bar{\varepsilon}$ and DED was unknown at the onset of the current work. One of the advantages of assuming a series representation was the inherent generality of this form. This was considered an asset to the approach that had to be preserved. As a result, the approach was to discritized the solution domain into multiple elements. At any location within an element other than a nodal location, the DED was defined by the linear interpolation of the nodal values over the element. Consequently, the DED was completely defined over the solution domain by piecewise three dimensional linear interpolation. The analog in a single dimensional space would be a piecewise linear fit of an arbitrary continuous function. As previously mentioned, the number of strain components considered in the current work was limited to three corresponding to the total number of in-plane strains and therefore a three dimensional strain space was used. Each element in the solution space was defined by an eight node cuboid. The element density was variable thus allowing for the analog effect of mesh convergence when comparing with finite element modeling. Figure 14 represents an example of a mesh configuration described in the current work as [4 4 5] representing the number of nodes per side for each dimension respectively. Two distinct mesh configurations were explored namely, [5 5 7] and [9 9 12]. In each case a greater number

of nodes were placed along the third dimension corresponding to ε_{12} compared to the other two dimensions ε_{11} and ε_{22}. This was strategically done to maintain element aspect ratios of approximately one. There were three specific requirements for the mesh; (1) the mesh must span the entire solution space, (2) the mesh density must be large enough to reasonably represent an arbitrary function, but not so dense that the total number of nodes becomes excessively large and (3) there must be a node located at

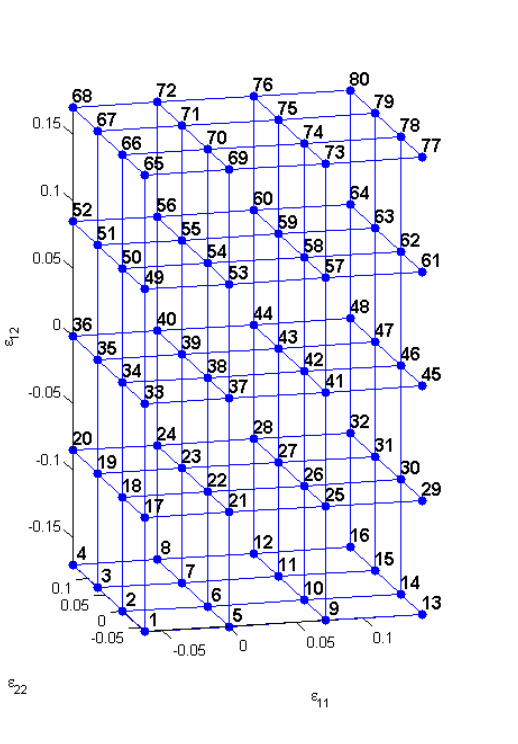


Figure 14. Four node linear interpolation elements used to represent the DED over the solution domain.

$\bar{\varepsilon} = \bar{0}$. The first requirement is somewhat natural consequence of the solution domain. Elements must span the solution space in order to accurately determine the DED coefficients from the experimental data. Justification for the second requirement will

become clear in the next sections. The third requirement is necessary for the following physically realistic condition to be imposed on the DED function, videlicet, $\phi(\bar{\varepsilon} = \bar{0}) = 0$.

Constrained Linear Least Squares

Using the element by element interpolation previously mentioned, a linear system of equations was constructed in terms of the unknown coefficients, C_i. The index i on the set of unknown coefficients represents the total number of nodes within the solution domain. As an example, the total number of nodes contained within the solution domain in Figure 14 is eighty or the largest global node index. The node numbers in Figure 14 represent the global node numbering scheme and will hereafter be referred to as such. The remainder of the current section contains the procedures required to generate the system of equations, including both equality and inequality equations so that the end result was a correctly posed system of equations for a constrained linear least squares solution.

The procedure used to develop each equation in the system of equations is identical, thus a detailed explanation of the procedure will only be completed for a single arbitrary experimental point on an arbitrary experimental load path. Hence, the number of experimental data points integrated over all of the test paths is equal to the total number of equations in the system. Load path 8 was arbitrarily selected to be used in the development and can be seen in Figure 15. The recoverable energy (RE) and the dissipated energy (DE) are represented graphically. An arbitrary experimental data point has been selected and is circled. The arithmetic summation of the dissipated energies calculated from each component gives the total dissipated energy (TDE). Once again a

graphical representation is provided as Figure 16. The curves plotted in red represent the component dissipated energies and the curve plotted in blue represents the total dissipated energy. The TDE point that is circled corresponds to the experimental data points circled in Figure 15. In Figure 16 the horizontal axis is representative of a data point index opposed to a physical displacement. The reason for the plot being constructed that way was a result of the TDE being a scalar quantity while the test coupon boundary displacement is a vector quantity. Be that as it may, the combined information

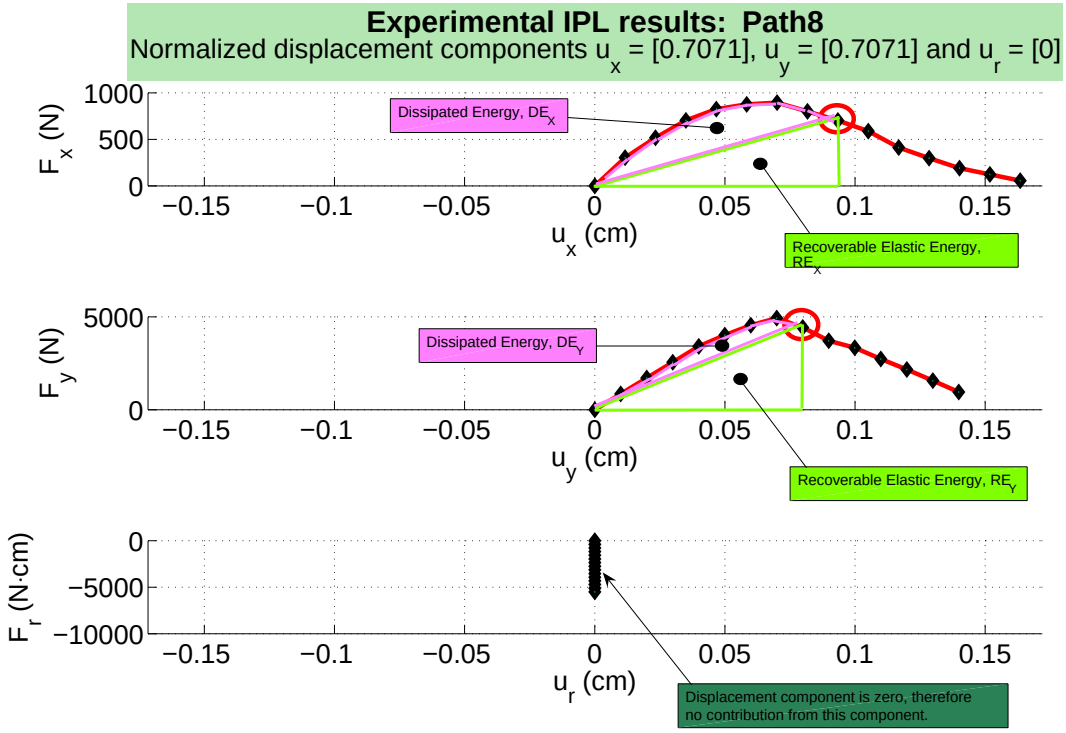


Figure 15. Recoverable energy (RE) and dissipated energy (DE) for an arbitrary experimental point circled in red.

from the two figures is enough to experimentally determined TDE for the arbitrary point of interest. For the sake of clarity some notation must be introduced.

- i - global node index
- j - local node index
- k - index representing an arbitrary experimental data point
- e - FEA element index
- p - ply orientation volume fraction index

Summation of equation (1.9) over the coupon volume results in the TDE for an arbitrary experimental data point. Thus, the TDE written in terms of the DED is

$$TDE_k = \sum_e \phi_e V_e \qquad (1.10)$$

where V_e represents the FEA coupon element volumes. Replacing ϕ_e in Equation (1.10) yields an expression for the TDE of an arbitrary k^{th} experimental data point.

$$TDE_k = \sum_e \sum_p \sum_i C_i \left[N_i(\bar{\varepsilon}_{p,e}) \right]_k V_e \lambda_p \qquad (1.11)$$

The ply orientation volume fraction is introduced in Equation (1.11) as λ_p. The definition for this term is implied by the name. The ply orientation volume fraction is the portion of the volume in the laminate that is occupied by multiple plies with a single ply orientation. By merely taking advantage of the linear nature of summation and exchanging the order of the summations along with the conversion between index and matrix notation should make the procedure for building the linear system of equations reasonable clear.

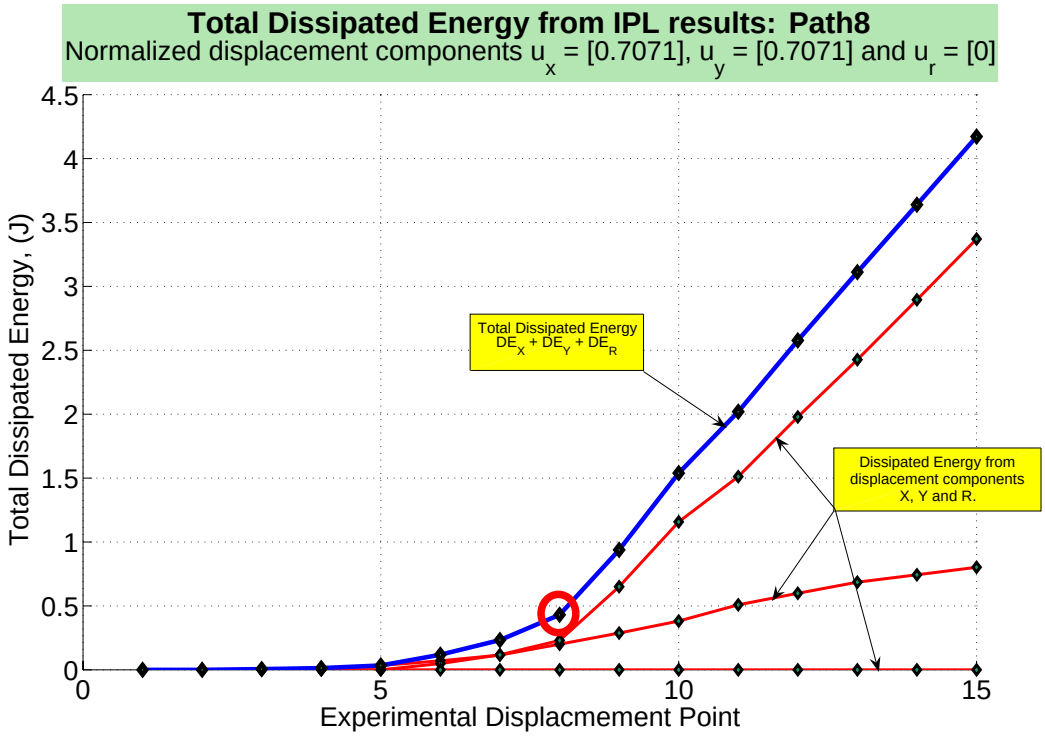


Figure 16. Total dissipated energy (TDE) plotted in blue for load path 8.

Starting with equation (1.11), the order of the triple summation is changed to obtain,

$$TDE_k = \sum_i C_i \left[\sum_e \sum_p N_i(\bar{\varepsilon}_{p,e}) V_e \lambda_p \right]_k \qquad (1.12)$$

where the summation over the FEA elements and the ply volume fraction have been moved inside the summation over the global DED nodal indices. For simplicity, define the quantity $a_{i,k}$ as,

$$a_{i,k} \equiv \left[\sum \sum N_i(\bar{\varepsilon}_{p,e}) V_e \lambda_p \right]_k \qquad (1.13)$$

so that substitution of $a_{i,k}$ in equation (1.12), yields,

$$TDE_k = \sum_i C_i a_{i,k} \ . \qquad (1.14)$$

Equation (1.14) can be further simplified by changing from index notation to matrix notation. Thus,

$$\{TDE\}_k = \{C_i\}\{a_i\}_k^{\mathrm{T}} \qquad (1.15)$$

is identically equal the equation (1.14) written in matrix notation. The next operation uses the identity of transposing of a scalar quantity with itself, namely,

$$\{s\} = \{s\}^{\mathrm{T}} \qquad (1.16)$$

where s is any arbitrary scalar quantity. Recalling that TDE is a scalar quantity, the transpose of equation (1.15) is used and the aforementioned identity is applied. The bracket notation is hereafter omitted on scalar quantities. Therefore, the final step is using the transpose operation to get

$$TDE_k = \left\{\{C_i\}\{a_i\}_k^{\mathrm{T}}\right\}^{\mathrm{T}} = \{a_i\}_k \{C_i\}^{\mathrm{T}} \qquad (1.17)$$

where $\{a_i\}_k$ is now a row vector and $\{C_i\}^{\mathrm{T}}$ is a column vector. Equation (1.17) represents a single equation written for an arbitrary experimental data point. When multiple experimental data points are considered the result is a system of equations that take on the form,

$$\begin{Bmatrix} TDE_1 \\ TDE_2 \\ TDE_3 \\ \vdots \\ TDE_{k_N} \end{Bmatrix} = \begin{bmatrix} \{a_i\}_1 \\ \{a_i\}_2 \\ \{a_i\}_3 \\ \vdots \\ \{a_i\}_{k_N} \end{bmatrix} \begin{Bmatrix} C_1 \\ C_2 \\ C_3 \\ \vdots \\ C_{k_N} \end{Bmatrix} \Rightarrow \{b\} = [A]\{x\} \qquad (1.18)$$

where $\{b\}$ is the vector of known values with dimensions k_N by 1, $[A]$ is the coefficient matrix with dimensions k_N by i_N and $\{x\}$ is the vector of unknowns with dimensions i_N by 1. This is the general procedure to build the system of equality equations necessary to find the set of unknown DED coefficients C_i. With the system of equations cast in this form it is easy to see that if $k_N < i_N$ the solution set will be an infinite set, if $k_N = i_N$ the solution set will be a finite set, and if $k_N > i_N$ the solution set will be an empty set. Because k_N represents the total number of experimental data points in the material database, it is a parameter that is determined by the engineer. The minimum number of experimental points, k_N, is determined by the number of unknown coefficients i_N, or in terms of a more physically meaningful quantity the number of global DED nodes in the solution domain. Consider the case where $k_N = i_N$. For the present example lets say there are n sets of experimental data. Each data set is denoted by the capital letter epsilon, Υ_n and contain k_N experimental data points. There is no reason why equation (1.18) can not be solved using each of the data sets independently, however, in general the resulting solutions sets will be different. Hence, the solution set is a finite set. This situation is undesirable because the "best" solution set must be chosen from many possible solution sets. The most desirable situation combines all the experimental data into a method that results in a single solution set that is defined to be the "best". The definition of which solution is "best" is arguable, but in the current work the best solution was defined as the one that minimizes the squared difference between

the predicted solution set and the experimental data, also known as the least squares solution. The least squares method is attractive when the set of experimental data is greater than the number of unknown nodal values. (i.e. $k_N > i_N$)

The solution that minimizes the squared difference between the solution set and the experimental data is not enough to get the "best" solution. Further constraints were necessary to make sure that the solution set produced physically realistic results, videlicet, the minimum value be no less than zero and the maximum value be no greater than the available strain energy density. The constraints bounded the solution set between specific limits. The Matlab built in subroutine "LSQLIN" was used to solve the constrained linear least squares problem.

Matlab Program

Matlab was the computer program that was selected to complete the numerical computations. The software selection was based primarily on the availability at MSU. The Matlab code was written to execute the required computations to obtain the DED coefficients, C_i, based on the aforementioned methodology. Many different sub routines were used as a part of the programming to make the Matlab code more flexible and to minimize programming mistakes. Sound programming techniques were used as frequently as possible to make the Matlab code usable for future research. The Matlab code can be divided into two groups; (1) script files and (2) function files. For reference the most important subroutines used in the program will be listed with a brief description.

1. Matlab Scripts

- DED_func_eff.m - This script is the back bone of the program. Many important parameters can be changed within this script. Run this script after setting all the parameters to obtain the unknown DED coefficients.
- solve_full.m - This script is called by DED_func_eff.m to execute the solution phase of the program after the system of equation and the constraint equations have been defined. The Matlab subroutine "LSQLIN" is called from this script.

2. Matlab Functions

- ANSYS_test_ixy.m - This function obtains the global coupon coordinate strains and volumes for each element in the FEA model as well as provides the experimentally determined TDE. Within this function data files from ANSYS finite element models are accessed.
- DE_total_eff.m - This function accesses data files created from IPL experimental data and obtains the experimental TDE curve as well as the corresponding displacement components.
- data_fit.m - This function takes the experimental data and uses linear interpolation to add artificial data points to the data set.
- secant_fit.m - This function computes a secant modulus for all the experimental data. The secant modulus is computed when the non linearity in the load-displacement curve reaches 3% of the elastic recoverable energy.

- diss_E_calc.m - This function computes the TDE given an experimental load-displacement curve.
- Ply_Principal_Strain.m - This function preforms the strain tensor transform into material principal coordinates.
- finite_diff_bad_implicit.m - This function computes nodal values for elements that lack any experimental data. As implied by the name, the method uses an implicit finite difference method to calculate the nodal values of bad elements based on the nodal values of good elements. The definition of a bad element is one doesn't contain any experimental data points. A good element is one that contains at least one experimental data point.
- DED_plot.m - This function plots the experimental data against the results of the linear least squares calculation.

It is important to remember that the list provided above does not include all of the subroutines required to run the Matlab program. An entire list of the Matlab subroutines required to run the program is included in the Appendix B.

Numerical Matlab Program

The Matlab program described previously were written in such a way that the engineer has some flexibility in terms of the experimental data included in developing the solution. The reason for the flexibility was based on the suspicion that the data paths without a rotation component were more reliable. Justification for this lies in a comparison between the experimental load-displacement curves and the predicted results

using an ANSYS FEA model. Figure 17 and Figure 18 represent the two extreme cases seen in the database. The first case, Figure 17, is an example insufficient correlation between the experimental data and the FEA data. The reason for this discrepancy is unknown. However, it should be investigated before any future IPL testing. Figure 18 represents an example of sufficient correlation between the experimental data and the FEA data. Load paths without a rotation component commonly exhibited this level of correlation. Based on the aforementioned comparison, two distinct sets of experimental data were analyzed in the current work, namely, a set of experimental data that contained all the load paths and a truncated set of experimental data that included four load paths without a rotation component. The disadvantage of including fewer load paths is the potential decrease in the unique combinations of in-plane strains present in the database. A graphical review of the density of strain data seen in Figure 19 compared with the same plot for the database with only four load paths provided evidence that the distribution of strain points for each database is similar. In the strain density plot of Figure 19, the color of each data point distinguishes the association the strain point has with the different interpolation elements.

The distribution of the data in the solution domain is clearly most concentrated around the origin. Considering IPL coupons under displacement control the strain distribution is determined primarily by the geometry of the test specimen. The strain gradient effects of the notched coupon result in an increased strain data density at the boundary of the solution domain when compared to the results for an un-notched coupon.

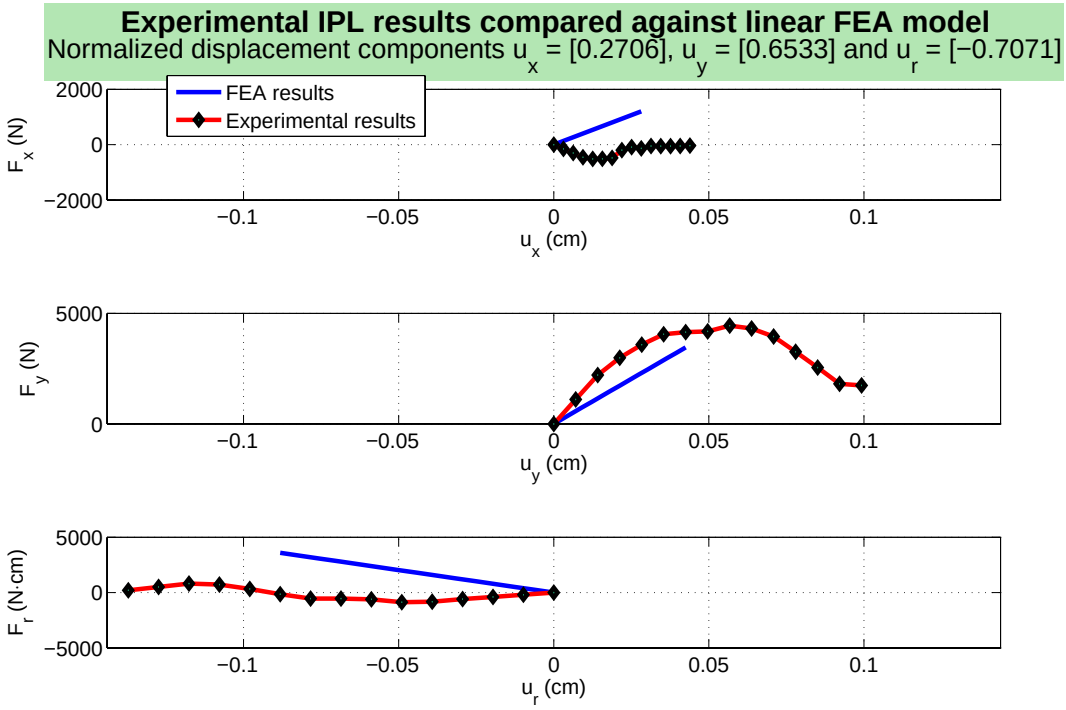


Figure 17. FEA prediction of load-displacement curves for load path 10.

The DED function becomes more reliable as the density of strain data at the extremes of the solution domain increases. Thus, it is advantageous to have a notched coupon with a strong strain gradient to produce larger more unique combinations of strains. A frequency distribution plot representing the experimental strain data points as they appear in different interpolation elements is represented in Figure 20. The distribution of strain data is severely skewed towards the elements located around the strain origin. Although a strain gradient exists the number of elements with low strain values seriously outweigh the number elements effected by the strain concentration.

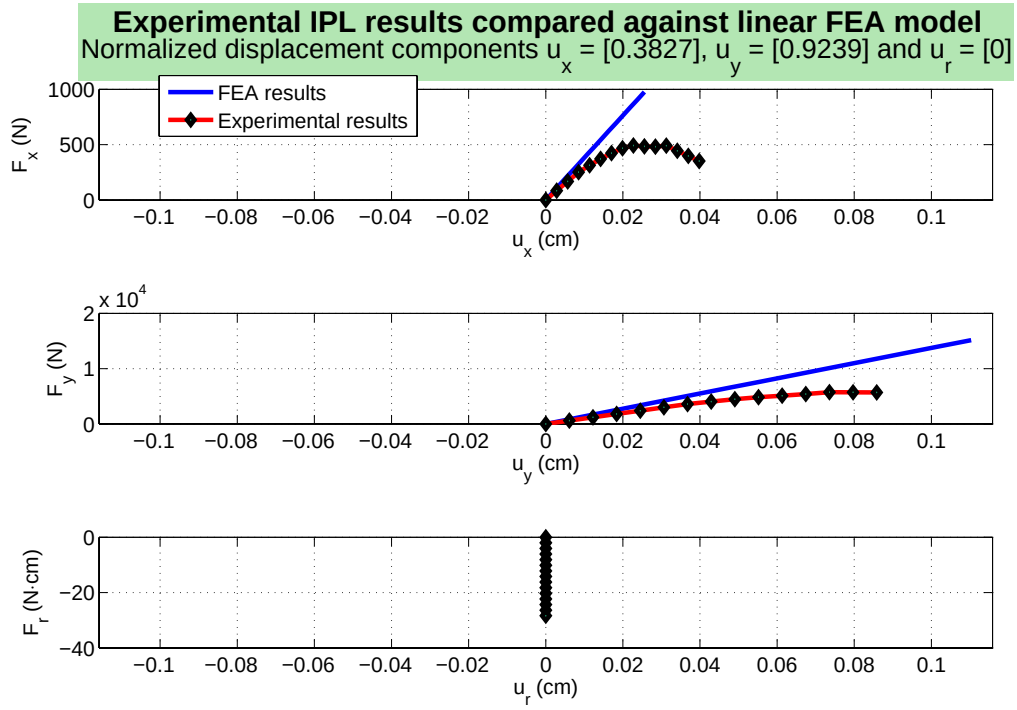


Figure 18. FEA prediction of load-displacement curves for load path 11.

It should also be noted that the number of interpolation elements with little or no data are significant. Elements that don't contain any experimental data points or that are only sparsely populated disturb the reliability of the DED function. The inherent nature of the interpolation functions handle these elements quite well as long as the data less element is surrounded by adjacent elements with sufficient data density. Unfortunately, this is not always the case and the adjacent elements lack sufficient data density. This case most frequently occurs towards the extreme boundaries of the solution domain. The worst case is when the adjacent elements have no data. Thus, there are no experimental data points to write nodal DED equations. In terms of the mathematics this represents a condition where a column of zeros is present in the coefficient matrix $[A]$.

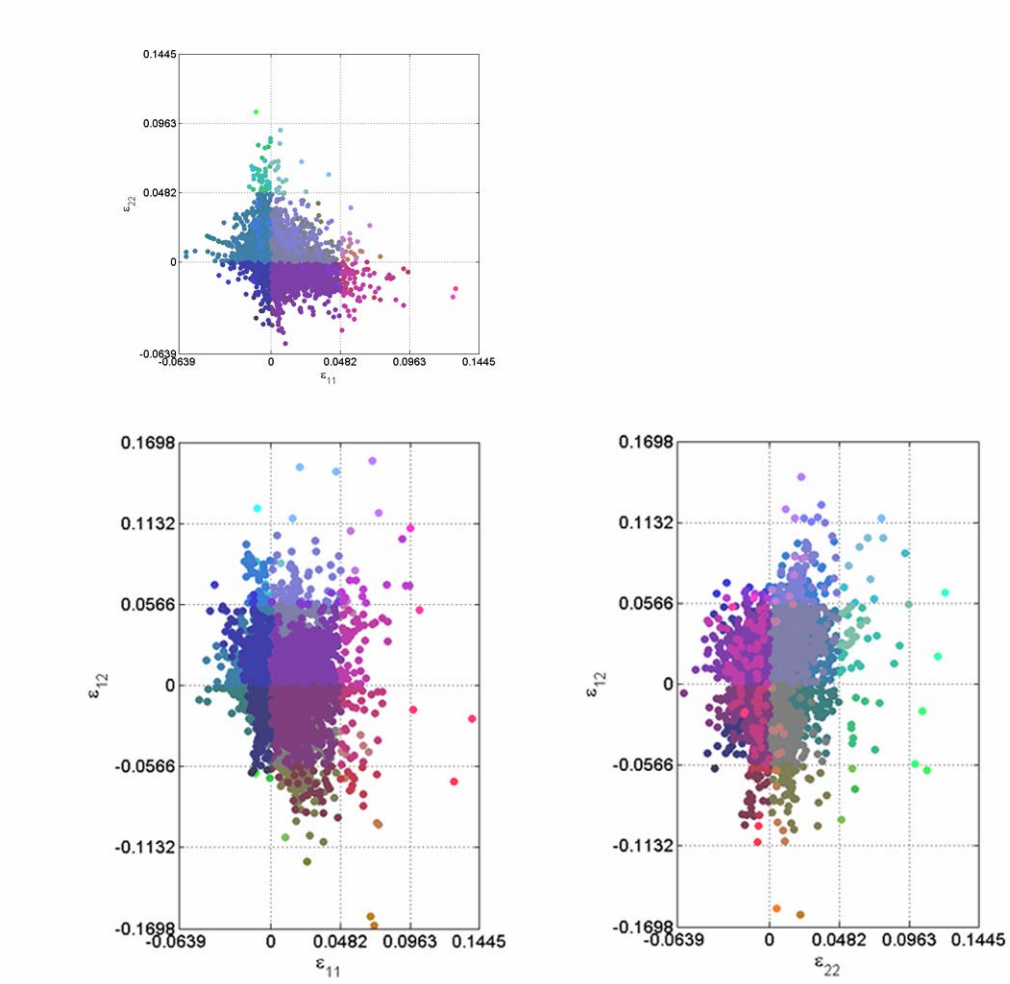


Figure 19. Strain data point density for the database created using all the load paths.

Subsequently, any nodal value will exactly satisfy the equation constructed for a node common to adjacent elements without any experimental data. To mitigate this issue, an implicit finite difference method was implemented. Under this method, nodes associated with null populated elements were assigned the average value of the nearest nodes associated with good elements. Entire regions of null populated elements were handled without difficulty using this method.

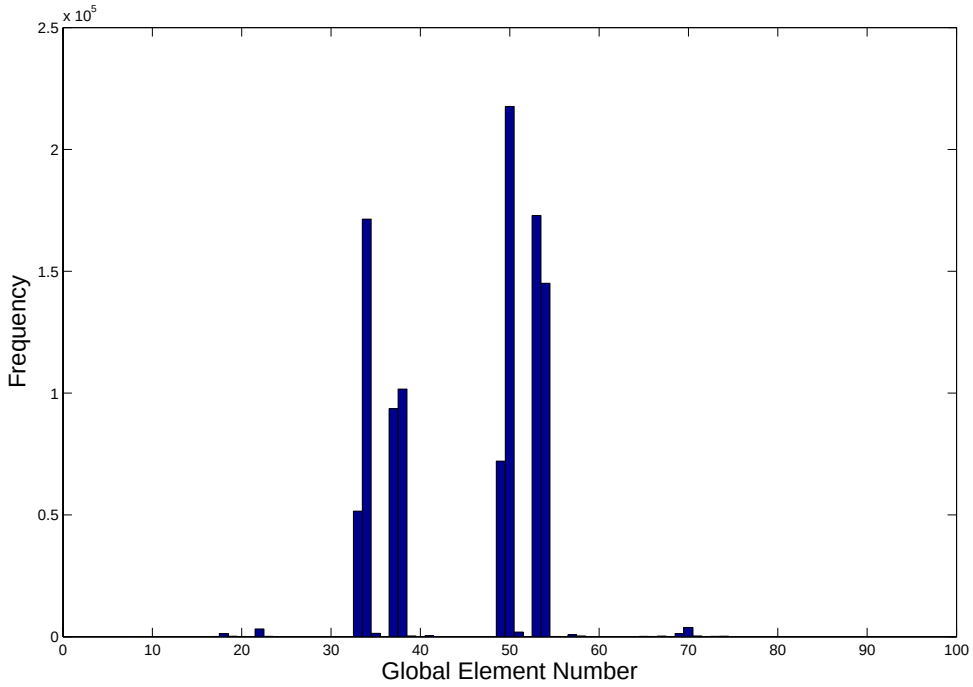


Figure 20. The frequency distribution of data for a [5 5 7] mesh.

The Matlab subroutine written to execute the finite difference solution is identified as finite_diff_bad_implicit.m, previously referenced in section on the Matlab program.

For the sake of clarity, some notation is required before presenting the experimental results. As a part of the current work, a couple of different combinations of DED solutions are presented. The solution vectors and the naming convention for each of them are outlined in Table 2. Here after, the naming convention defined in Table 2 will be used to determine the relevant DED solution vector.

Table 2. Solution set combination matrix.

	All Load Paths	Select Load Paths
Solution Vector	BAP	BSP

It should be noted that the unbounded linear least squares solution provided a better fit of the experimental data but was excluded based because the physical likelihood of such a solution is not possible. This is true because it includes the possibility for the TDE to be greater than the available energy. Figure 21 is composed of a plot of the DED function over the solution domain for the case of an unbounded solution. This 4D plot of the DED function over the strain domain should be interpreted analogous to the representation of a temperature distribution over a spatial domain. The lack of contour lines is a consequence of local singularities in DED. The singularities by definition produce high gradient regions eliminating DED contours over the bulk of the solution space. As a consequence, the DED function appears to be constant over the bulk of the strain domain. In addition to the low probability of the unbounded solution being physically realistic, if the DED function truly contains near singular points, the method of linear interpolation over elements within the solution domain would need to be seriously revised. The approach described in the current work is not capable of representing an arbitrary DED field that contains singular points with the accuracy required in material characterization. The unbounded solution was presented to provide future researchers with insight into the behavior of the DED curve fit.

DED Database

BAP Solution. The results of the BAP solution vector will be presented first. This solution was the result of a material database that contained all seventeen load paths and used a bounded linear least squares solution. Hence, an upper and lower bound were imposed on the solution vector.

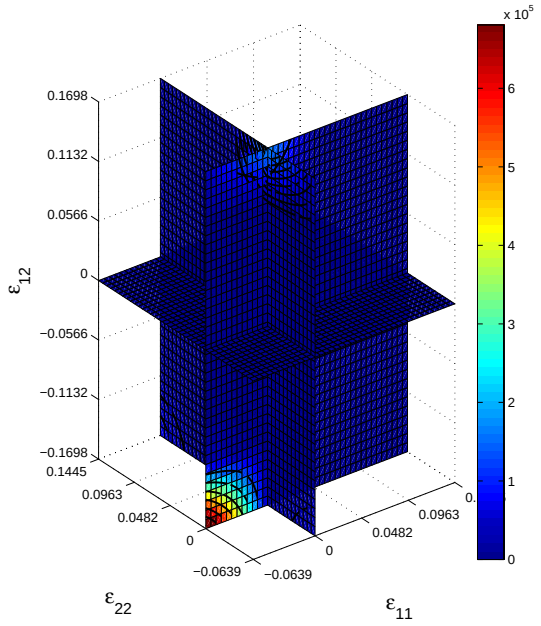


Figure 21. Contour plot of DED function using the unbounded solution vector. The DED is represented in units of J/cm^3.

The lower bound was set such that a nodal value less than zero were not allowed. Negative dissipated energy density has no physical significance. The upper bound was set to be no greater than the total amount of available energy if the material were to continue to be elastic. Figure 22 gives a graphical representation of the DED plotted over the solution domain using the BAP solution. Based on the contents of Figure 23 the BAP results fit the experimental data reasonably well. The BAP solution fits the experimental data with a calculated r-squared value equal to 0.92. There are two solutions plotted as a part of this figure, namely, the BAP solution plotted in green and the unbounded solution plotted in red. The greater r-squared value corresponding with the unbounded solution

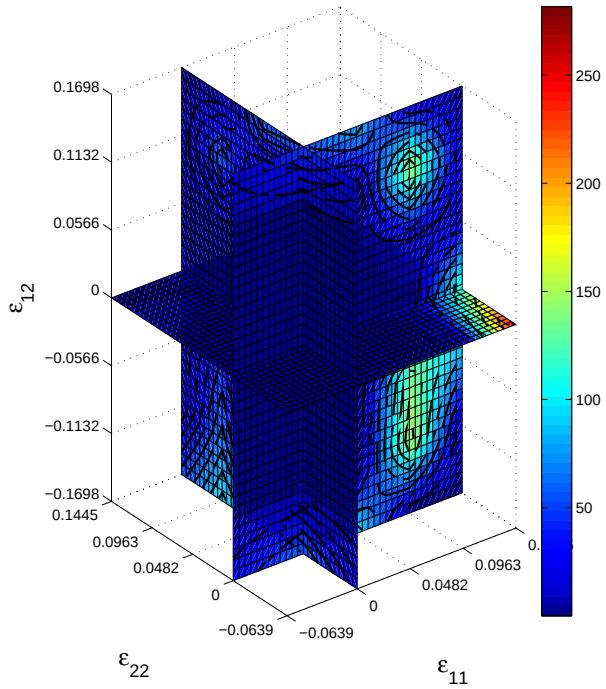


Figure 22. Contour plot of DED function using the BAP solution vector. The DED is represented in units of J/cm^3.

confirms that the unbounded solution fits the experimental data better as previously stated.

As a check, the DED function was used to predict the behavior of two IPL coupons with in the material database. Select load paths were limited to those with a single displacement component. Using paths with single displacement components made it possible to determine a unique load displacement curve for the coupon. Load path two and fourteen were selected for load- displacement curves to be generated based on the results of the DED function. Figure 24 is composed of the predicted and experimental

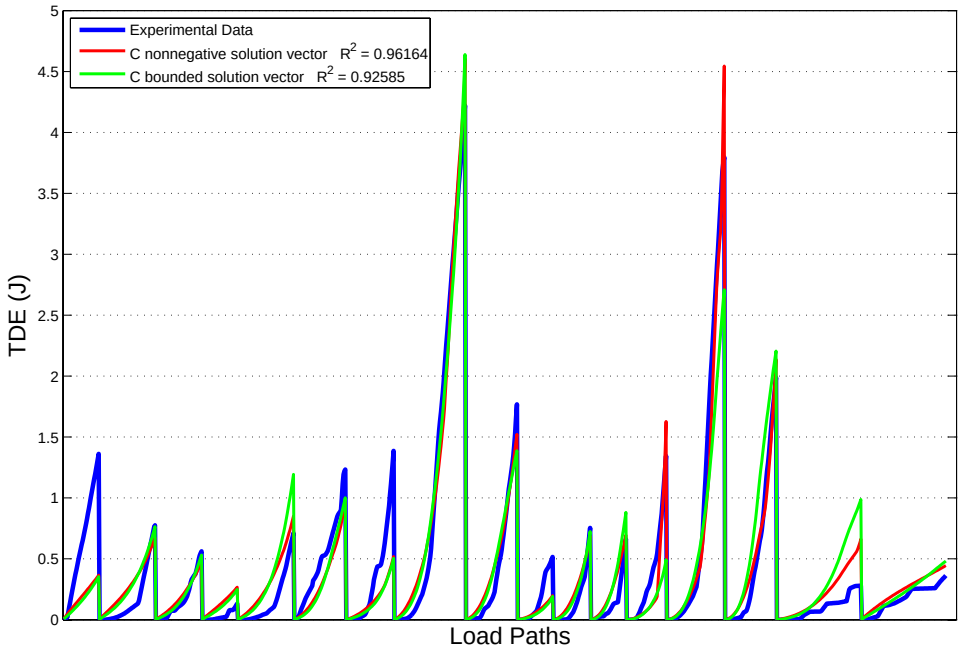


Figure 23. Experimental and predicted total dissipated energy for a solution including all load paths. (Table 1)

load-displacement curves for an IPL coupon under pure Y-direction boundary displacement. The actual behavior of the coupon is captured quite well when very little energy has been dissipated due to damage within the coupon. From an inspection of the results the predicted and the experimental curves become severely nonlinear at approximately the same displacement. However, towards the ultimate load carrying capability of the coupon it is clear that the predicted results are not accurately capturing all of the material behavior. Thus, the calculated r-squared value for the curve fit is 0.80.

Based on the experience gained from the current work, there are three factors that could contribute to produce such a result. First, the IPL does not handle coupons with significant reduction in residual strength very well. A nontrivial amount of strain energy

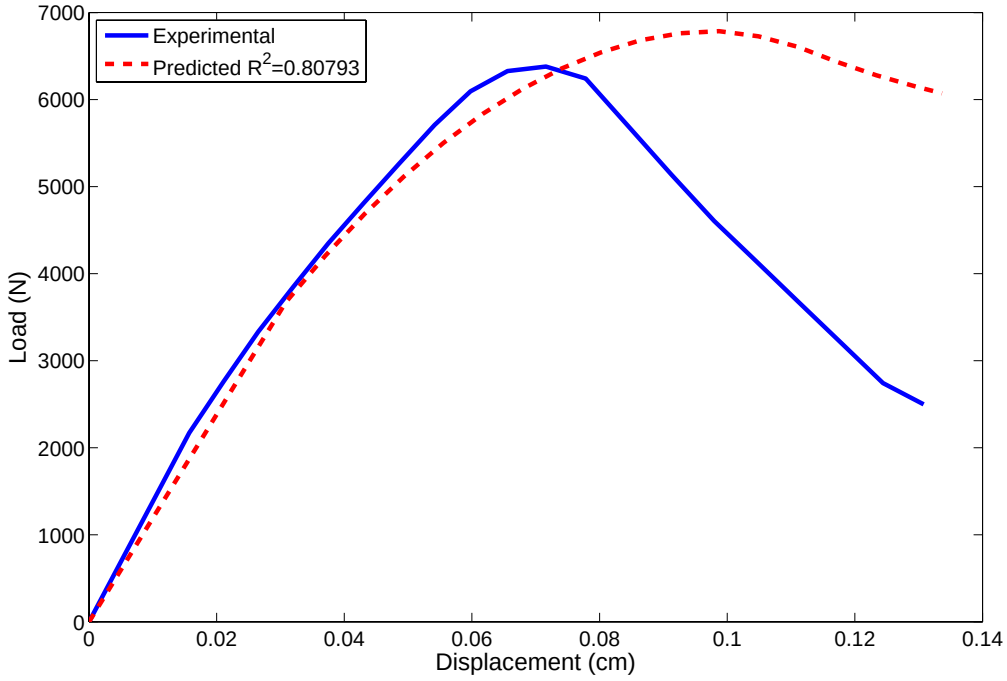


Figure 24. Predicted load-displacement curve for IPL coupon under a pure Y direction boundary displacement. This curve was derived using the BAP solution vector.

is stored in the frame of the IPL during testing. The strain energy stored in the frame is available for further damage propagation and the current instrumentation is not setup to capture this release into the coupon once the coupon softens due to damage accumulation. Also, during an IPL test the load-displacement data is recorded after displacement convergence criteria has been met. The iterative nature of the displacement convergence steps makes each displacement step relatively slow. The interaction of these two factors create an undesirable condition, being, when strain energy from the frame is transferred into the test coupon increasing the damage zone without data being recorded. As a result, the data for the material softening part of the curve may not contain all the material information required by the DED method for accurate characterization. The experimental

results shown in blue are not be the whole story. Second, the previous assumption concerning the strain field being linearly scalable during the evolution of damage within the coupon might not be correct. This assumption might prove to be an oversimplification of the problem and not valid. The fact that predicted results deviate from the experimental results at the point where nonlinear behavior becomes significant provides evidence for this conclusion. Lastly, there exists a discrepancy between the linear FEA load-displacement predictions and experimental results on a component by component basis. The comparison displayed in Figure 17 is representative of a significant number of load paths used to build the current DED database. The disconnect that exists between the FEA results and the experimental results could easily account for the deviation in the predicted curve.

Figure 25 is composed of the load-displacement data for an IPL coupon under X-direction boundary displacement. The predicted results for the X-direction are plagued with similar problems as those in the Y-direction. As the dissipated energy due to accrued damage increases the deviation between the predicted and experimental curves also increase. Thus the calculated r-squared value is 0.79.

An explanation for the deviation between the curves could be a consequence of any one of the previously mentioned factors. Unlike the Y-direction results, the X-direction prediction starts accruing damage at a much lower displacement. The reason for this outcome is difficult to diagnose. This behavior is consistent with over predicting the TDE curve at small displacements. One hypothesis that was investigated was that

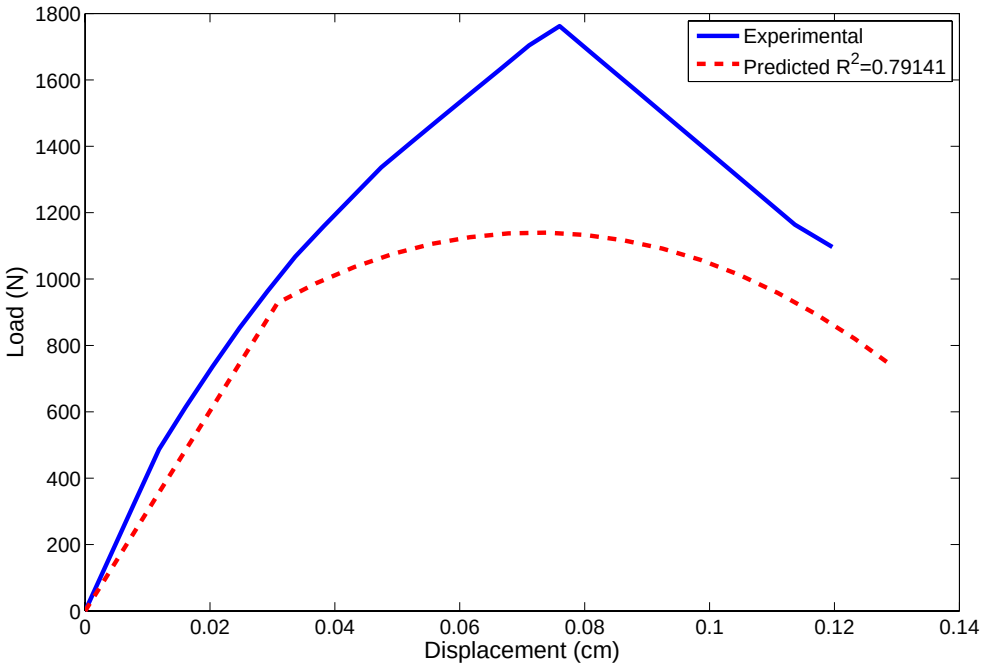


Figure 25. Predicted load-displacement curve for IPL coupon under a pure X-direction boundary displacement. The predicted curve was derived using the BAP solution vector.

"bad" load paths exist diluting the information of the "good" load paths and result is poor predicted results. Under this hypothesis the "bad" load paths were defined as load paths with a rotation component and "good" load paths were defined to be the set of load paths that do not contain a rotational component. The basis for the bimodal classification of load paths lie in the fact that some load paths match the FEA results and others differ significantly from the predictions of FEA results. Figure 17 and Figure 18 exemplify the fact that the FEA is not always consistent with IPL data. It should be noted that the results of the FEA analysis for a rectangular coupon geometry showed good correlation with Instron test data. In addition to what was previously mentioned, the coupling between rotation and Y-direction displacement it is difficult to interpret. Hence, the

engineer is left with no way to distinguish between accurate and erroneous results from IPL test that include complicated boundary displacements. An attempt to isolate only "good" load paths was made by using a truncated material database containing only four load paths. The BSP solution in

represents an attempt isolate only "good" load paths. The results of the truncated material database are included in the following paragraphs.

BSP Solution. The BSP solution vector was created based on a truncated material database containing only four load paths. The primary reasoning for including this material database was an effort to isolate reliable load path data sets. The goal was to determine if erroneous load path data was present in the material database used to obtain the BAP solution. Direct comparison between results of the two different databases would hopefully produce one of two conclusions: (1) the two solutions are not significantly different therefore any arbitrary load path is likely to be as accurate as any other load path and (2) the two solutions are significantly different and therefore select load paths are likely more reliable than other load paths.

Figure 26 is a contour plot of the DED function over the solution domain for the BSP solution. Overall, the DED function for the BSP solution is similar to the BAP solution. The BSP contour plot still contains local maximums similar to the BAP solution. The boundaries that define the strain domain for each of the two solutions are almost the same. Thus, the overall strain envelope for which the DED function is valid does not change significantly between the two solutions.

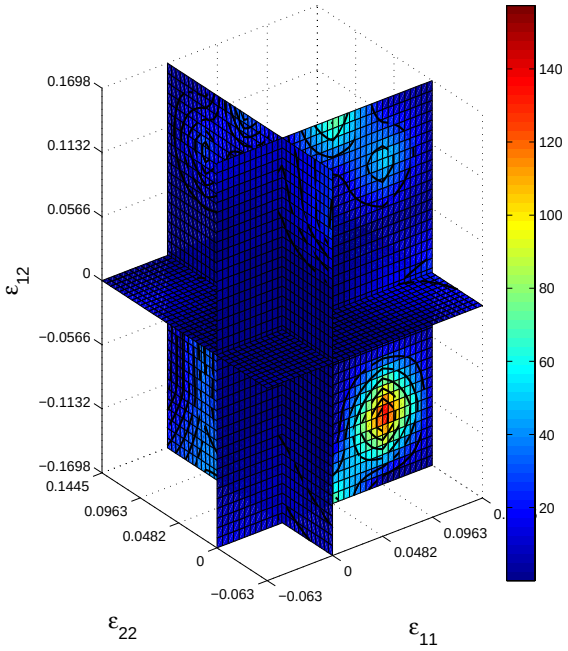


Figure 26. Contour plot of DED function using the BSP solution vector. The DED is represented in units of J/cm^3.

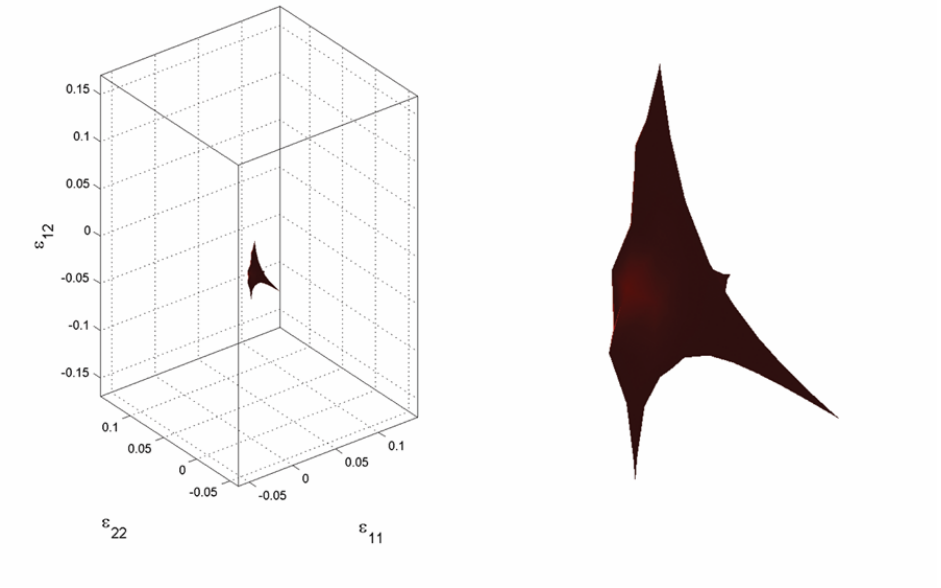


Figure 27. Three dimensional damage initiation envelope for BSP solution vector.

An onset of damage envelope was plotted for the BSP solution and presented as Figure 27. The plot on the left represents the orientation of the damage surface relative to the three strain axes, while the plot on the right represents an enlarged view of the damage initiation surface shape. The surface was defined by the transition between a purely elastic coupon and the point in strain space where DED becomes nonzero. The damage surface presented as Figure 27 is quite different in comparison to the criterion covered in chapter two. An explanation for the distinct damage initiation surface shown for the BSP solution is unclear.

Based on the r-squared calculation displayed in the legend of Figure 28 the BSP solution fits the data very well. The calculated r-squared value for the BSP solution is

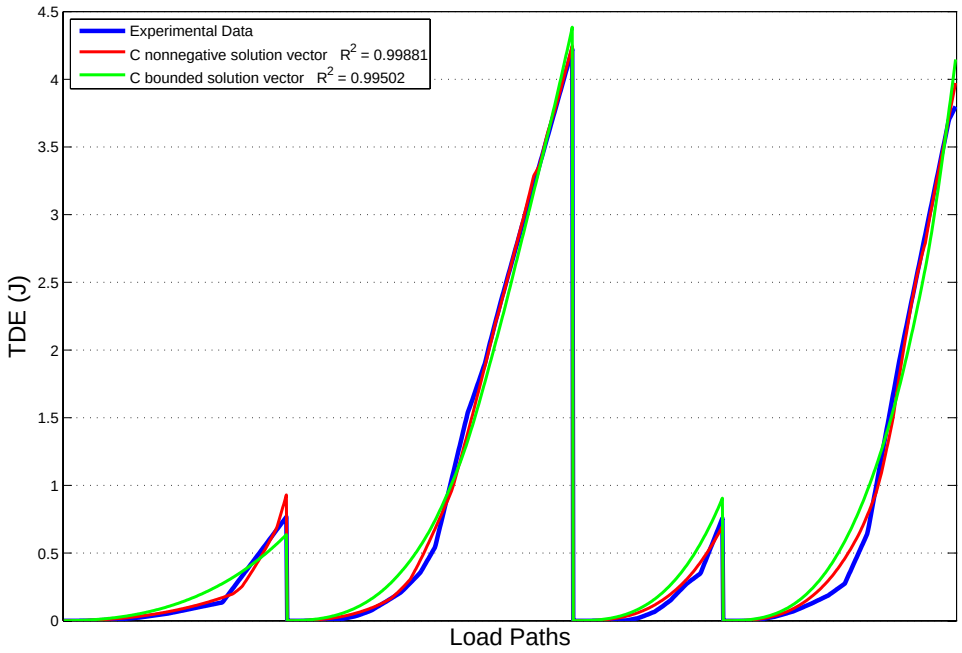


Figure 28. Comparison between experimental and predicted TDE for a solution resulting from a truncated material database.

0.99 compared to 0.92 for the BAP solution. A visual comparison of the experimental and predicted TDE yields some interesting points about the fit of the data. First, the TDE is always over predicted at the base of the curve when the TDE starts to increase. Further examination of the technique that is used to fit the data might reveal that a weighted fit is required. Next, the unbounded solution plotted in red fit the experimental data much better than the BSP solution however the unbounded solution is not physically realistic. Lastly, the predicted TDE curve doesn't seem to be capturing all the material behavior. This point becomes most apparent after examination of the curve to the far left in Figure 28.

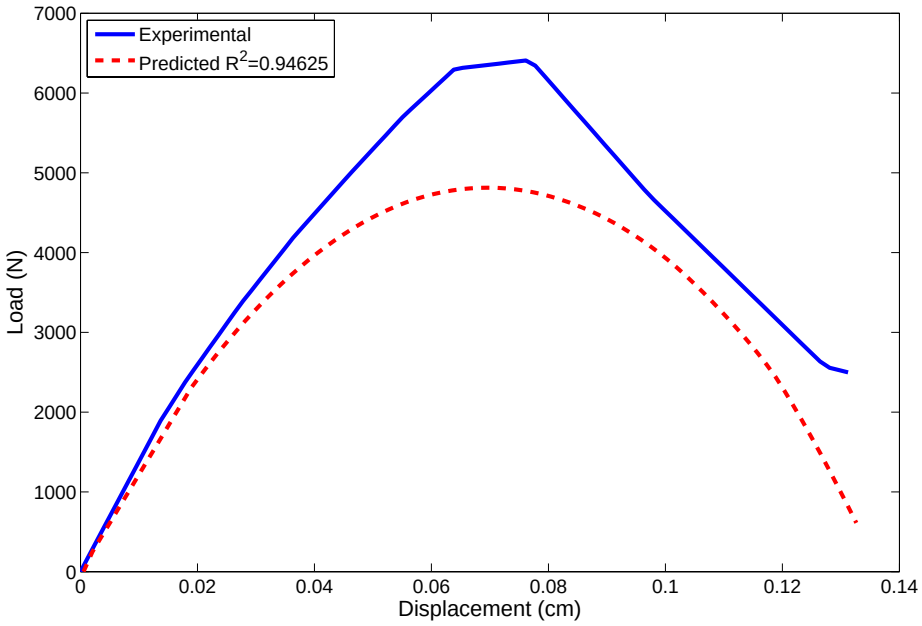


Figure 29. Predicted load-displacement curve for IPL coupon under a pure Y-direction boundary displacement. The predicted curve was derived using the BSP solution vector.

A consistent set of predictions were made using BSP solution to compare with the predictions of the BAP solution. Hence, load paths two and fourteen representing pure X and Y boundary displacement respectively were once again examined.

Figure 29 represents the results for the Y-displacement of an IPL coupon. The curve plotted in blue represents the experimental results which are identical to the experimental results displayed previously in Figure 24. The shape of the experimental load-displacement curve is predicted quite well using the BSP solution, thus producing r-squared value of 0.94, greater than 0.80 calculated for the BAP solution. Despite the increased r-squared value, the predicted curve doesn't appear to be a better fit of the experimental data. This is largely due to the over prediction in the TDE. The error as a result of over predicting the TDE curve is magnified in the load-displacement curve, thus the relatively large discrepancy between predicted and experimental results. This problem would be eliminated using a weighted fit of the data such that the TDE curve correlates better with experimental data when the TDE values are still relatively small. A weighted fit should be investigated for future work.

The results of the X-direction boundary displacement display similar characteristics as the Y-direction boundary displacement. Comparison of the r-squared values for the BAP and the BSP solution yield 0.70 and 0.89 respectively. Based on the r-squared value the BSP solution does a better job of reproducing the load-displacement data. The discrepancy between the two curves is once again attributed to the over prediction of the TDE curve. The fact that material softening occurs in the predicted results well before the experimental results is a direct consequence of the over prediction

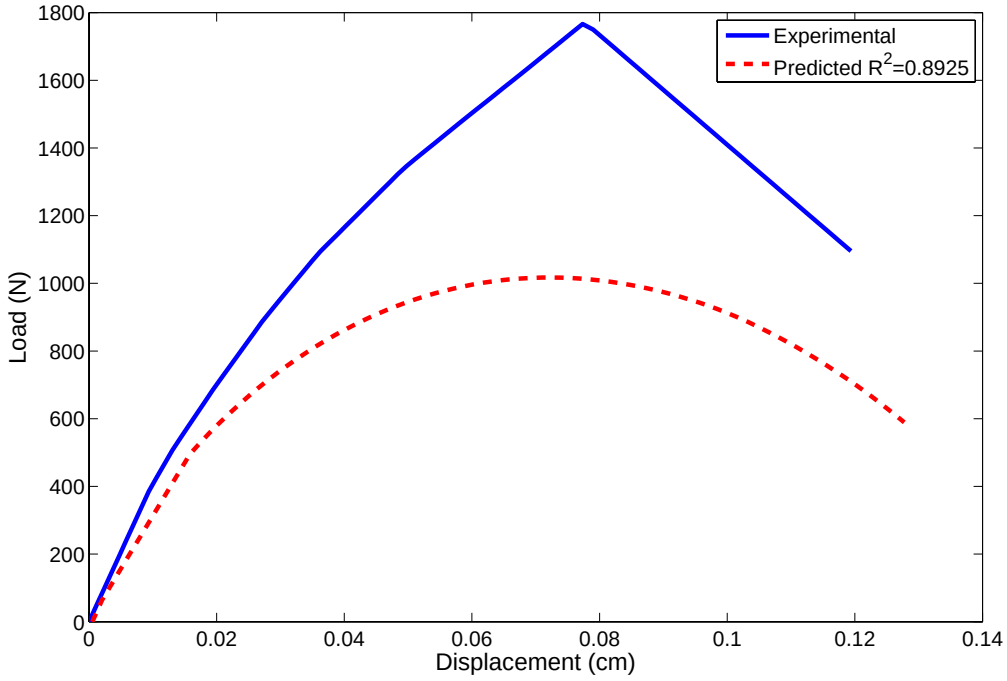


Figure 30. Predicted load-displacement curve for IPL coupon under a pure X-direction boundary displacement. The predicted curve was derived using the BSP solution vector.

of the TDE curve. A characteristic common to the X and Y boundary load-displacement predicted curves in Figure 24 and Figure 25 is the misrepresentation of the detailed material behavior. This could be an artifact of the current method, but whatever the source the fact remains that this level of accuracy is not acceptable.

Based on the results of the BSP solution it is difficult to determine if inaccurate load path data was deleterious to the BAP solution. At first glance, the load-displacement curves don't appear to be as good using the BSP compared to the BAP solution. However, based on the r-squared statistic summarized in Table 3 for both of the solutions the BSP solution is fitting the data better. Based on the r-squared statistic it is possible that inaccurate load path data is present in the BAP solution. To reach a final resolve the

IPL as a test machine would have to undergo further characterization specifically for load paths that contain complicated combinations of boundary displacements including rotation.

Table 3. Comparison of the BAP and BSP solutions using the r-squared statistic.

	BAP	BSP
Global r-squared statistic	0.92	0.99
Y-direction r-squared statistic	0.8	0.94
X-direction r-squared statistic	0.7	0.89

APPLICATIONS IN COMPOSITE MATERIALS AND STRUCTURES

General Procedure

One of the goals in the current work was to determine the present capabilities of the IPL. Since the IPL is such a unique testing machine it was difficult to determine if the experimental data from the IPL was good data. Thus, the predictive capabilities of the DED database were challenged with open-hole compression and bearing tension. These two case studies were used to evaluate the ability to transport the DED database to other geometric configuration using the same material system. The results presented in the current work represent the first time a DED database developed from testing on the IPL has been used to predict the behavior of a composite structure tested on an independent machine.

Open-hole compression (OHC) was selected as one of the test configurations because theoretically the results of the DED database should be very applicable and well within the predictive capabilities of the current method. In addition to open-hole compression a second test configuration was selected, namely bearing tension. Bearing tension (BT) tests are complicated by the number of contributing factors that cause failure. Thus, bearing tension was selected to push the envelope of the predictive capabilities.

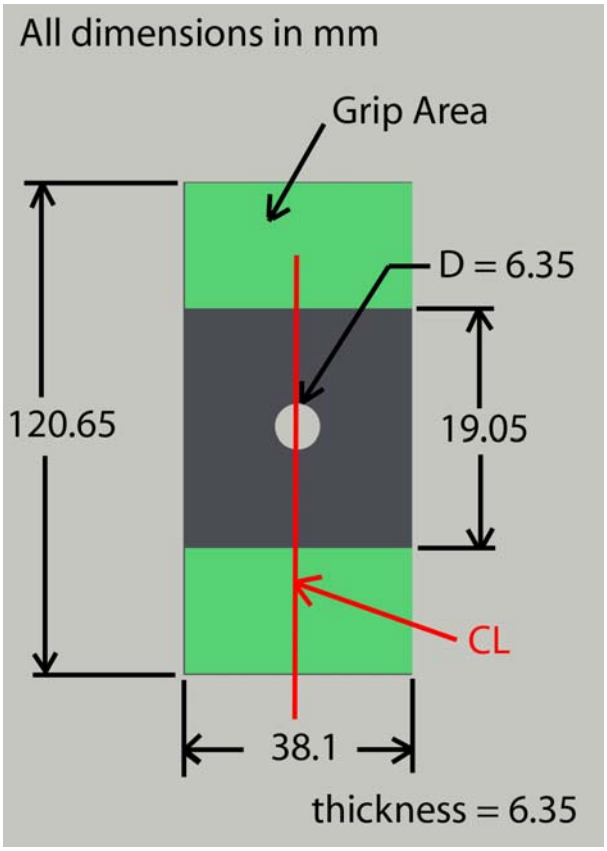


Figure 31. Open-hole compression (OHC) coupon geometry.

Open-Hole Compression

The open-hole compression test coupons were made using the same E-glass fiber and epoxy matrix material system that was used for the IPL test coupons. The lay-up for the test coupons was $(0/45/-45)_{4s}$. The compression coupons were four times the number of plies compared with the IPL coupons for additional bending stiffness to increase compression stability. The geometry of the test coupon, Figure 31, was no more complicated than a rectangular coupon with a centrally located hole.

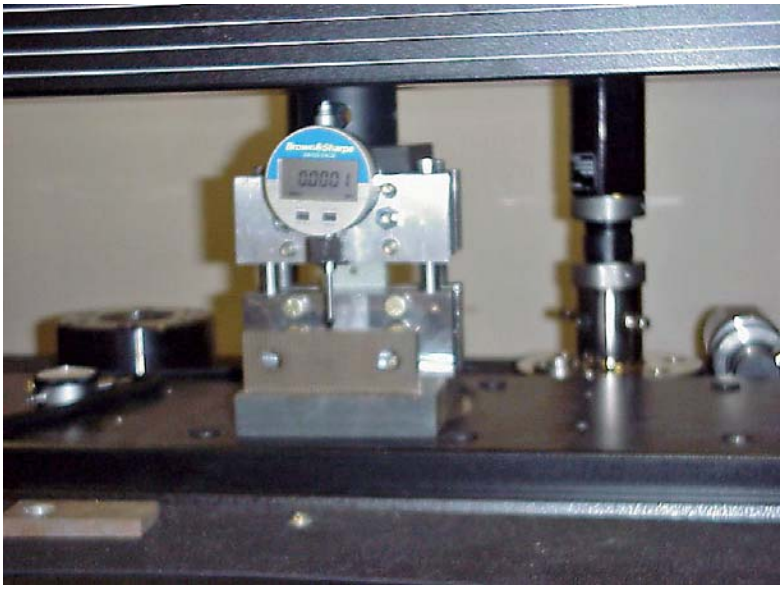


Figure 32. Open-hole compression (OHC) test setup.

The coupon was mounted in a fixture specifically designed for compression tests. The fixture was composed of three components, upper grips, lower grips, and cylindrical stabilizing rods. Each of the three components was made of heavy gage steel, thus providing a fixture that was laterally quite stiff. The test fixture in Figure 32 was used for two reasons, (1) to consistently grip the composite coupon so that grip displacements were reasonably accurate and (2) to react any accidentally applied moments so that the test remain planar. ASTM test standards were used as a guideline for the test procedures [12].

A total of five open-hole compression coupons were manufactured in the facilities operated by the Composites Group at MSU. Wet hand lay-up was the preferred manufacturing method. Caution was exercised during coupon manufacturing in order to achieve the highest quality test coupon. Hole preparation was critical because of the dominating presence of the notch. The central hole was drilled with adequate backing

support to minimize drill breakout delaminations. Any damage caused during the preparation of the bearing hole would blunt the stress concentration affecting the load carrying capacity of the coupon.

The OHC tests were run with a uni-axial Instron test machine under displacement control. The Instron test machine was programmed to move the cross-head at a speed consistent with the assumptions of a quasi-static test. The displacement rate was approximately 0.1 cm/min. Data points were manually acquired using a Brown and Sharpe digital indicator for displacements and the Instron load cell display for the loads.

Experimental Results.

There are three acceptable failure modes presented in the ASTM standard, reference [13] that was used as a guideline for the OHC tests. The first failure mode is a global laminate compressive failure. Under this mode, damage runs laterally across the center of the hole and buckling or kinking of the 0 degree fibers are primarily responsible for the failure. The second acceptable failure mode is initiated by local compressive failure at the hole, but the damage progression is primarily in the angle plies. Angle ply fibers generally cross the holes lateral center line. This is considered a matrix dominated failure. The last acceptable failure mode is a combination of the first two. The laminate initially fails in compression at the hole. The resulting load transfer is balanced between the angle plies and the axial plies in such a way that exhibits multiple modes.

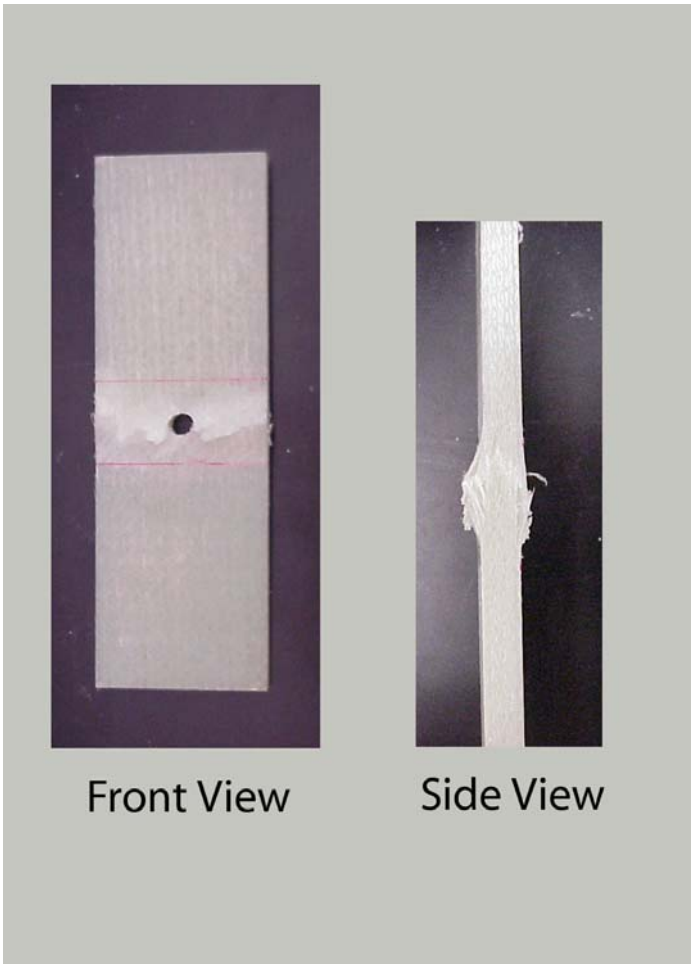


Figure 33. Open-hole compression coupon after ultimate failure.

The picture in Figure 33 is representative of all of the OHC coupons failure. The failure mode is most likely global laminate compressive failure and can be recognized best from the side view in Figure 33. The quantitative test results are displayed graphically in terms of load-displacement and TDE-displacement curves displayed in Figure 34 and Figure 35 respectively.

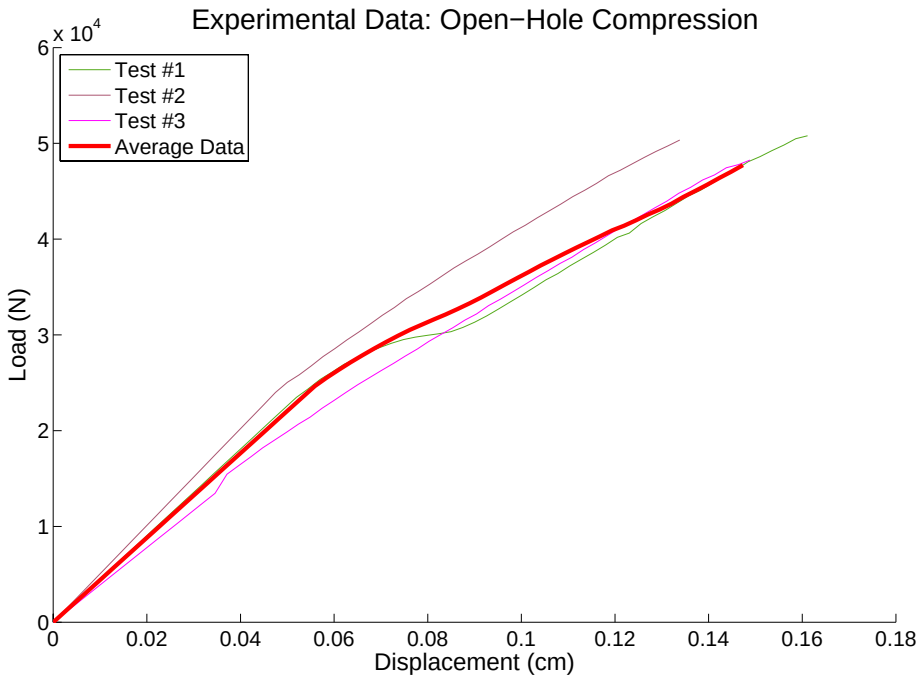


Figure 34. Experimental load-displacement data from OHC test.

The bold red line represents the average of three coupon tests. The results from two open-hole compression test coupons were excluded from the data set as a consequence of procedural errors that occurred during the testing. The load-displacement curve for test number one displays an abrupt softening starting at a displacement of approximately 0.07 cm. The load carrying capability of the coupon partially returns at a displacement of 0.09 cm. Behavior such as this is most likely related to stability opposed to accrued damage in the test coupon. For the purposes of comparing the experimental results with the prediction using DED function the average data curve will be used.

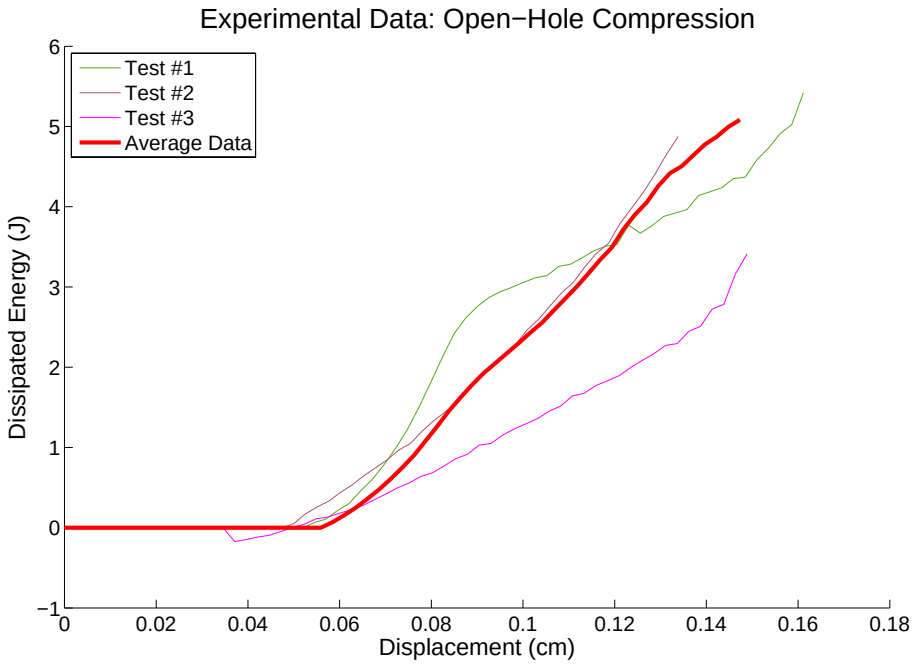


Figure 35. Experimental TDE-displacement data from OHC test.

FEA Modeling.

In order to predict the load-displacement curve using the DED function the strain distribution in the OHC coupon was required. To determine the strain distribution an FEA model was created. The model was created in ANSYS using the same approach outlined for the IPL coupons in section about linear FEA modeling. To remain consistent with the assumption included in the formulation of the DED function the resulting strain distribution from a linear FEA model was used to represent strains in the OHC coupon over the entire test. The model geometry was defined as the section of the test coupon that was gripped by the compression fixture. Therefore, the boundary conditions used in the FEA model were a uniform compressive nodal displacement in the Y-direction and unconstrained nodal displacement in the X-direction. The effective gripping constraint

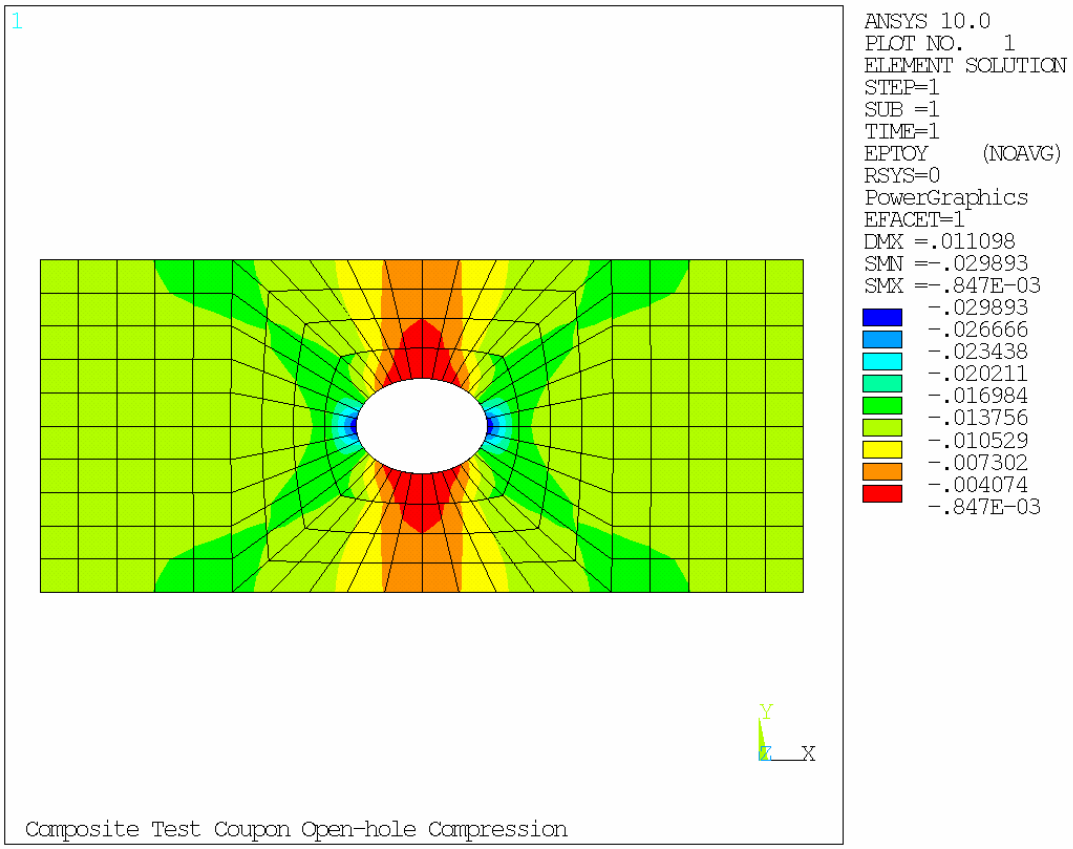


Figure 36. Y-direction normal strain distribution for an OHC coupon.

on coupon in the X-direction was neglected such that no Poisson induced stress developed. A contour plot of the Y direction normal strain distribution in the OHC coupon is included as Figure 36. The strain values in the FEA model were scaled linearly with the Y-direction displacements to obtain the coupon strains for different boundary displacements. The scaled strain values were subsequently used during the spatial integration of the DED function over the coupon to make a prediction of the TDE curve as a function of the Y boundary displacement. The prediction of the load-displacement was calculated from the TDE-displacement curve.

Comparison with DED Database

The goal for both the open-hole compression and the bearing tension tests was to provide a baseline to determine the generality of the DED database and to assess to what level the database could be transported to a structure with a different stacking sequence and different coupon geometry. The predicted results presented in this section are generated using the previously defined BAP solution. Figure 37 represents the TDE curve including both the experimental data and the predicted data. Based on a visual inspection of the curves, there is not conclusive evidence that the prediction is capturing the correct behavior of the OHC coupon. The fact that the prediction of TDE is nonzero at very low

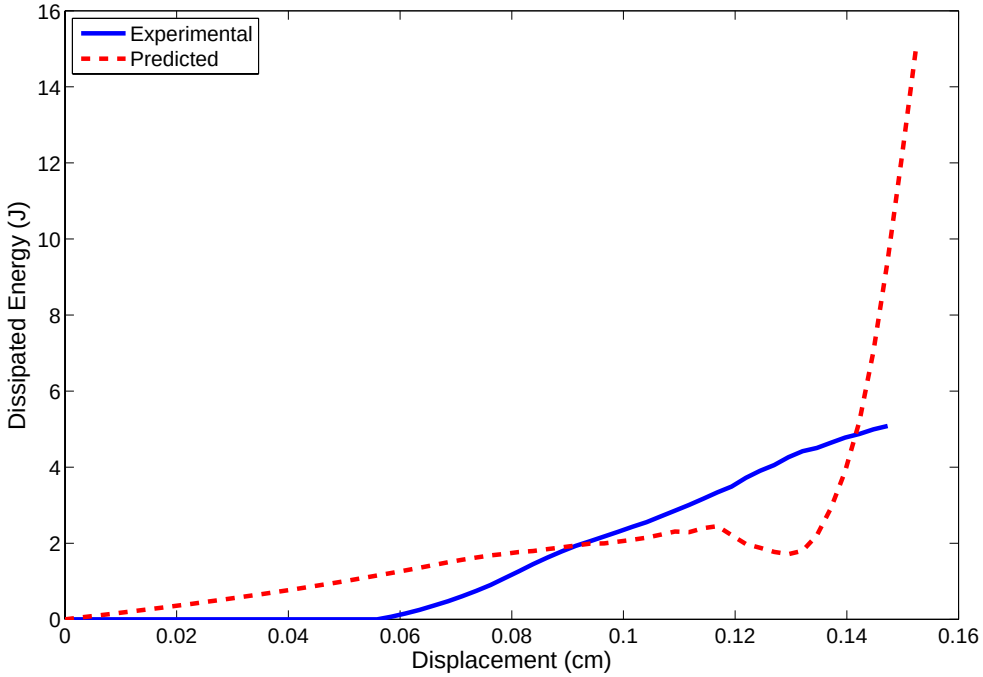


Figure 37. Predicted TDE-displacement for the OHC coupon.

displacement values isn't consistent with experimental data. The proportional limit for the predicted results in Figure 38 corresponding with a displacement of approximately

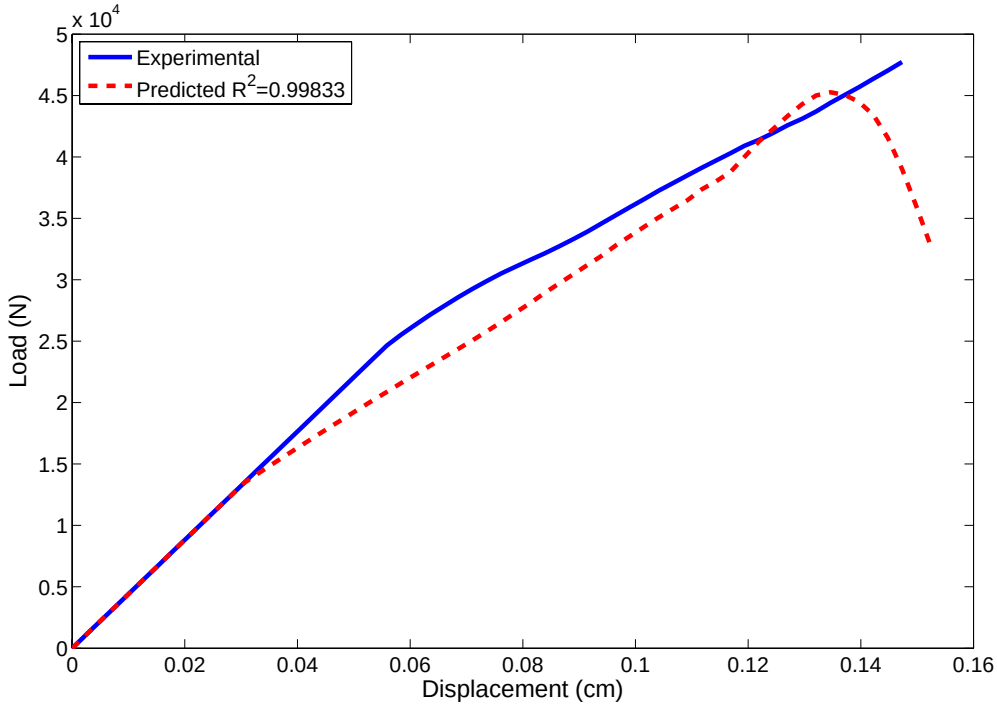


Figure 38. Predicted load-displacement for the OHC coupon.

0.035 cm is not consistent with the TDE curve in Figure 37. The total dissipated energy curve shows that at a displacement of 0.035 cm a reasonable amount of energy should have been dissipated. Over predicting the total dissipated energy curve in Figure 37 makes for a misbehaved load-displacement curve. The reason for the dip in the predicted TDE curve starting at approximately 0.12 cm is a suspicious curiosity. The load-displacement curve generated from the TDE curve is represented in Figure 38. This representation of the prediction of the DED database is not so bad. The calculated r-squared statistic for the fit is 0.99. Unfortunately, calculating a r-squared statistic close to one is not the whole picture. The mysterious dip in the TDE curve has carried through to the load-displacement curve to create a sudden decrease in load carrying ability at large displacements. One explanation for the dip in the TDE curve is presented in Figure 39.

This graphic presents the fraction of strain values that are characterized by one of two undesirable conditions. The first condition being that a reasonable fraction of strains lie outside the solution domain. Thus, extrapolation outside of the solution domain would be required to obtain DED data. Second, a fraction of strain values are in sub domain elements or elements that are lacking experimental data. Not surprisingly the fraction of strain points that lie outside the solution domain increase dramatically at a displacement of approximately 12 cm which corresponds to the dip in the TDE curve. More IPL testing is required in order to expand the solution domain, thus providing a better prediction of the OHC data presented in the current work.

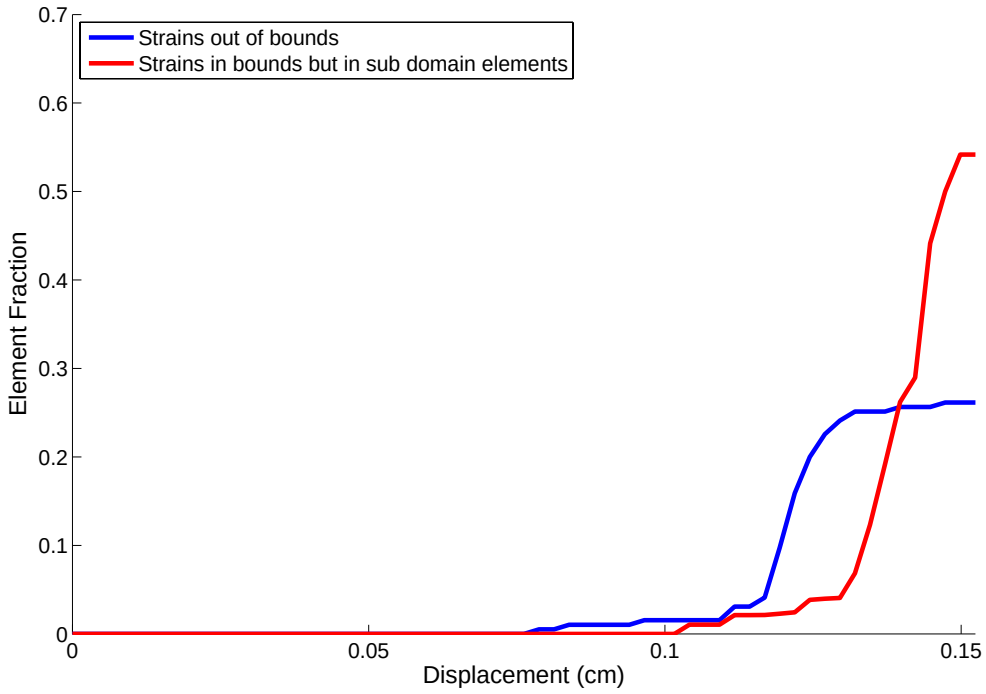


Figure 39. Fraction of sub domain elements as a function of OHC boundary displacement.

Bearing Tension

The bearing tension (BT) coupons and the OHC coupons were manufactured from the same E-glass/epoxy composite plate. The coupon geometry was slightly changed in order to accommodate the BT test fixture. The geometry of the BT coupon is included as Figure 41. The location of the centrally located hole was moved relative to the OHC coupon towards the end of the coupon. A minimum edge distance of 3D was preserved during the relocation of the hole.

Figure 40. Bearing tension (BT) coupon test setup loaded in the grips of a uni-axial Instron test machine.

The test fixture seen in Figure 40 was reasonable consistent with the recommendations of the ASTM standard [13] for bearing tension tests. The bearing pin diameter was nominally 0.635 cm and was made of hardened steel. A clearance fit between the bearing pin and the test coupon was used for the current work. It should be noted that the significance of the fit between the bearing pin and the hole in the test coupon may have been overlooked in the current work. Testing beyond the scope of the current work is required to determine the effect that the fit between the bearing pin and the coupon has on the results.

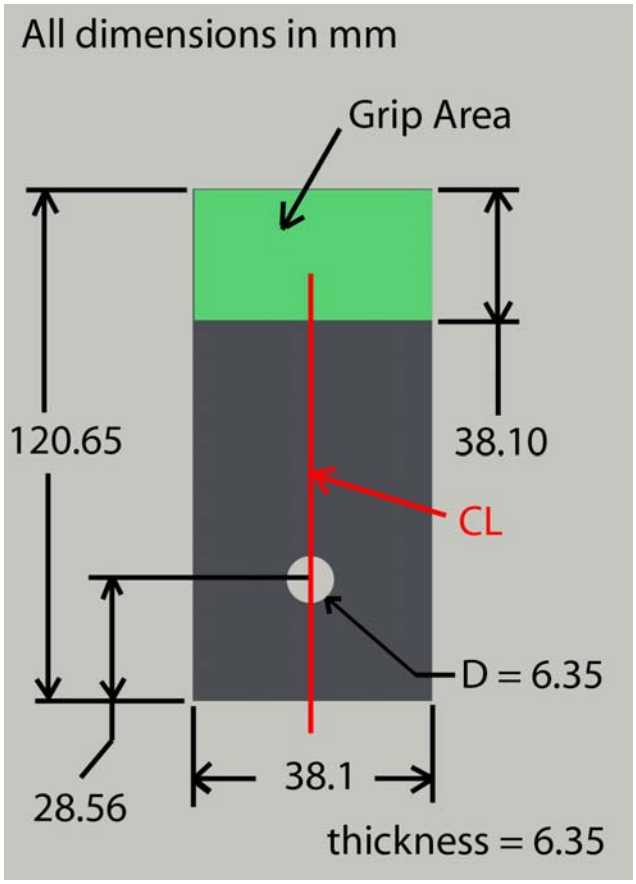


Figure 41. Bearing tension (BT) test coupon geometry.

The BT tests were run with a uni-axial Instron test machine under displacement control. The Instron test machine was programmed to move the cross-head at a speed consistent with the assumptions of a quasi-static test. Data points were manually acquired using a Brown and Sharpe digital indicator for displacements and the Instron load cell display for the loads.

Experimental Results

The results of the BT tests are presented graphically in Figure 42 and Figure 43. The load-displacement curve is somewhat different from a traditional composites curve. The curve for the BT coupon exhibits similar characteristics that a plastic yield curve for a metal would. The reason for this was a result of the BT coupon undergoing excessive deformation. Tearout was the predominant failure among all the coupons tested.

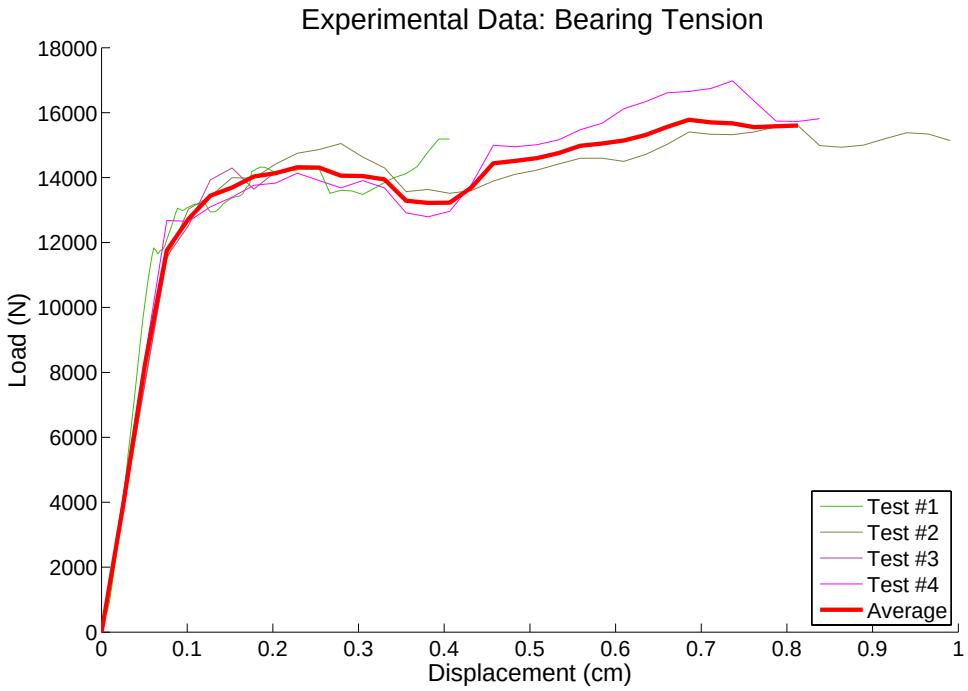


Figure 42. Experimental load-displacement data from bearing tension test.

As a result, the bearing pin displacements were relatively large, up to 1 cm for test number one. Figure 44 contains a picture of a BT coupon after it was removed from the test fixture.

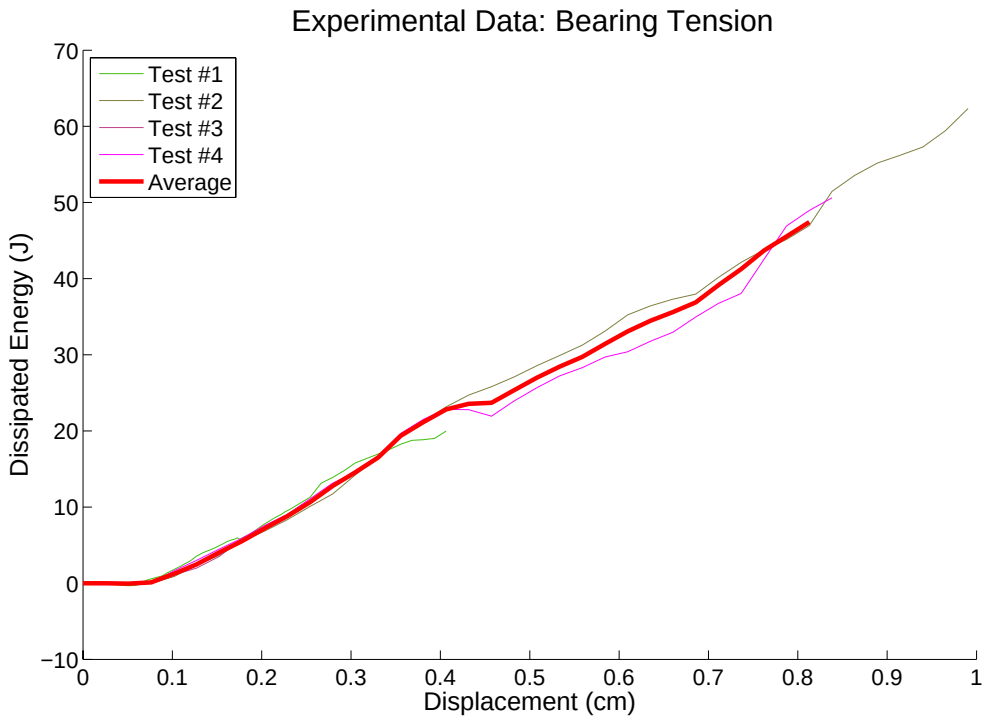


Figure 43. Experimental TDE-displacement data from bearing tension test.

At a displacement of approximately 0.4 cm, the load-displacement curve in Figure 42 momentarily softens before regaining some load carrying capability. Since many of the test specimens exhibited this behavior the chances of an anomaly are low. The explanation for this behavior is probably intertwined with the nature of the tearout failure. Initially, the transition from a pristine material to the onset of damage is most likely predominately bearing failure. Matrix cracking as well as ply delamination near the bearing surface initiate material softening. Local failure tends to decrease the notch

sensitivity as the damage becomes more widespread and transitions from a bearing failure to a tearout failure. During the tearout failure the coupon regains some stiffness due to material build-up in front of the bearing pin.

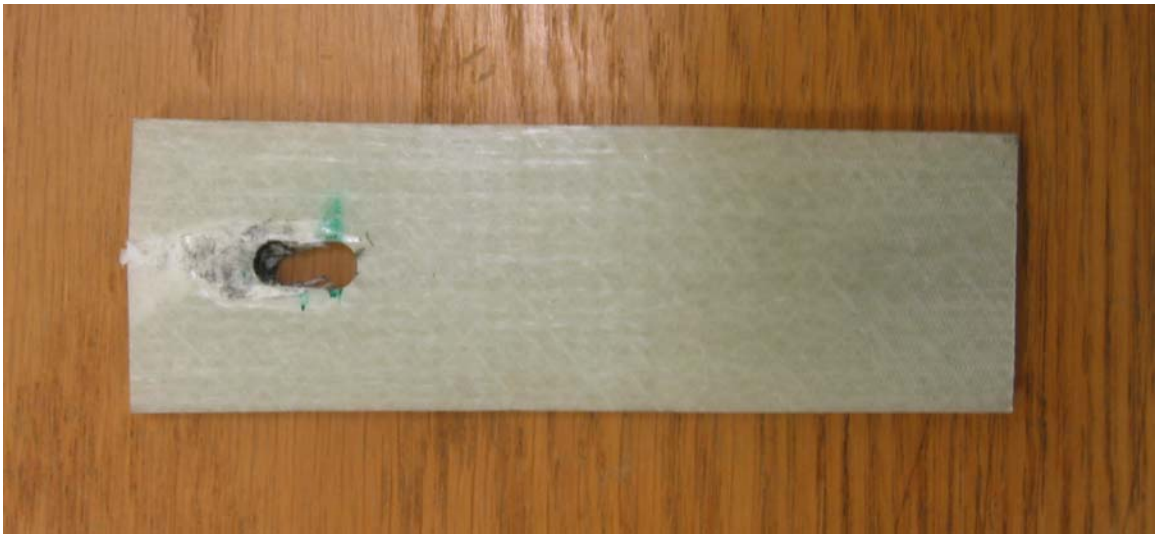


Figure 44. E-glass/epoxy BT coupon post failure. Tearout failure resulted in large bearing pin displacement.

The TDE for the BT coupon is plotted in Figure 43. The TDE curve is nearly bi-linear, hence the rate for which energy is dissipated in the coupon is approximately constant. This is different than the TDE results previously presented which exhibit an exponential increase in the rate for which energy is dissipated in the coupon. The energy dissipation rate is closely related to characterization of the failure of the material, however the details of these effects will not be considered in the current work. The reason for the difference in the TDE curve is in the details of the damage progression.

FEA Modeling

The procedure involved in making a prediction of the TDE-displacement and the load-displacement curves is the same for BT as it was for OHC. The prediction hinges

on resolving the BT coupon strains using a FEA model. A linear FEA model was used to resolve the strains over the coupon geometry consistent with previous assumptions of the current method. Thus, no strain redistribution due to damage is accounted for in the current paradigm.

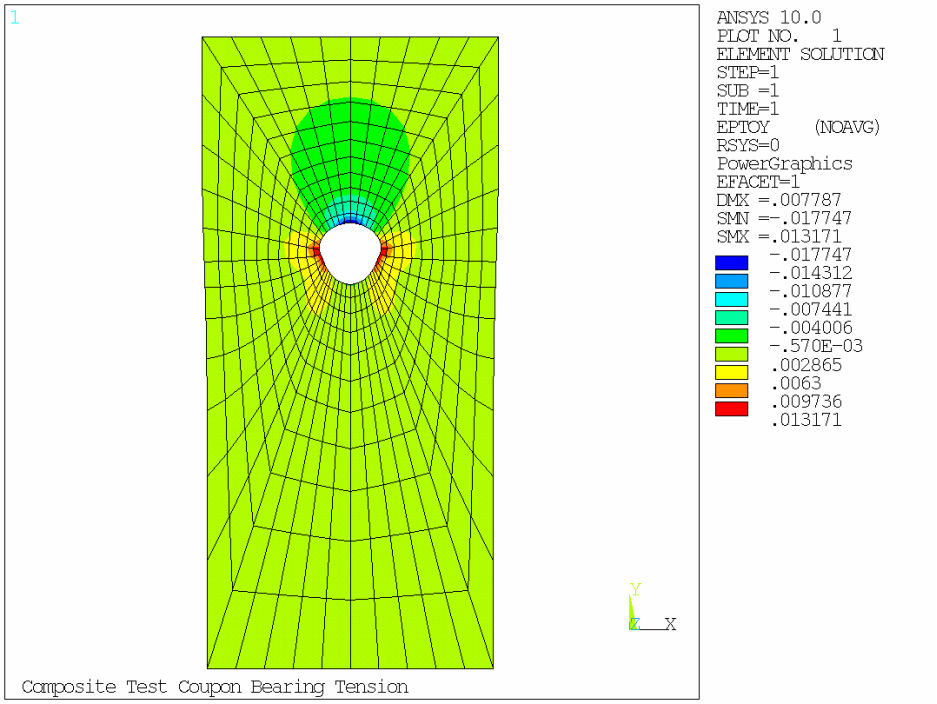


Figure 45. Contour plot of Y direction normal strain in the BT coupon.

The coupon geometry used in the FEA model can be seen in Figure 45. The boundary conditions imposed on the model were based on the theoretical boundary condition on the BT coupon during testing. The lower section of the coupon was securely clamped in the grips of the Instron testing machine, thus imposing a zero Y-direction displacement at the interface between face of the grips and the coupon. The load distribution caused by the bearing stress was modeled using a sine distribution of load over the inner surface of the hole. It is common engineering practice to use a sine distribution of load to model the

effects of a bearing load. The strain field produced by the FEA model was scaled linearly with the displacement of the bearing pin in the Y direction to obtain the strain field over a range of displacement values. The scaled strain values were subsequently used during the spatial integration of the DED function over the BT coupon to make a prediction of the TDE curve as a function of the Y boundary displacement.

Comparison with DED Database

Bearing failures are difficult to predict because of the vast numbers of contributing factors including, pin tolerance, friction and stacking sequence to name a few. It is clear from the results of the prediction that the current method struggled to capture all the contributing factors of the BT failure. The TDE results are presented in Figure 46.

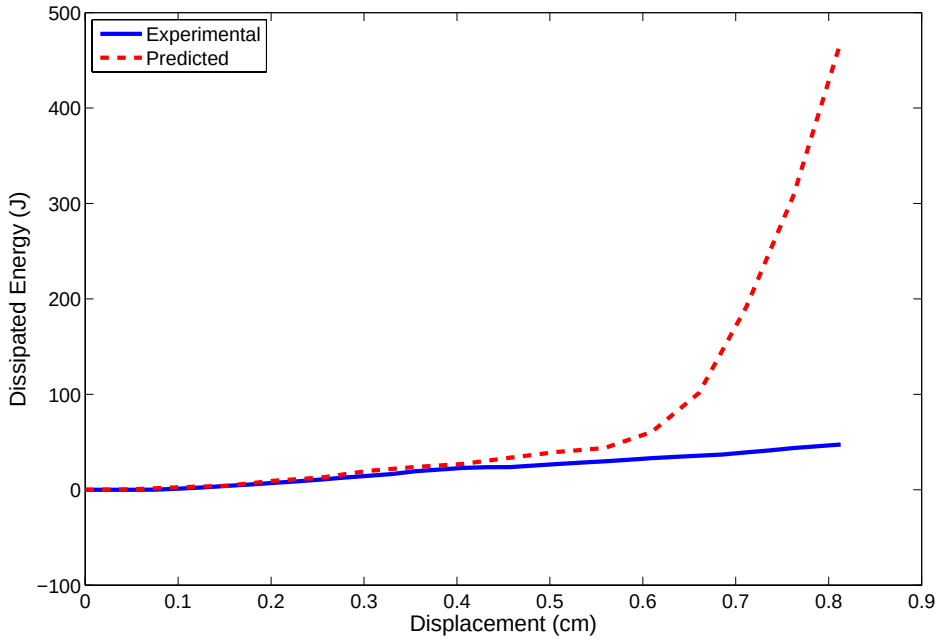


Figure 46. Predicted TDE-displacement for the BT coupon.

The first part of the TDE curve corresponds very well with the results of the experiment. However, the prediction becomes grossly large as the pin displacements increase. The reason for the wild prediction is probably the result of a failure mode that was absent in the IPL coupons responsible for the data in the material database. The remedy for the over prediction would be doing additional IPL testing on coupons with similar failure modes, namely a tearout failure. The data would be inserted as an additional load path and the material database would be redeveloped and applied in exactly the same manor presented in the current work.

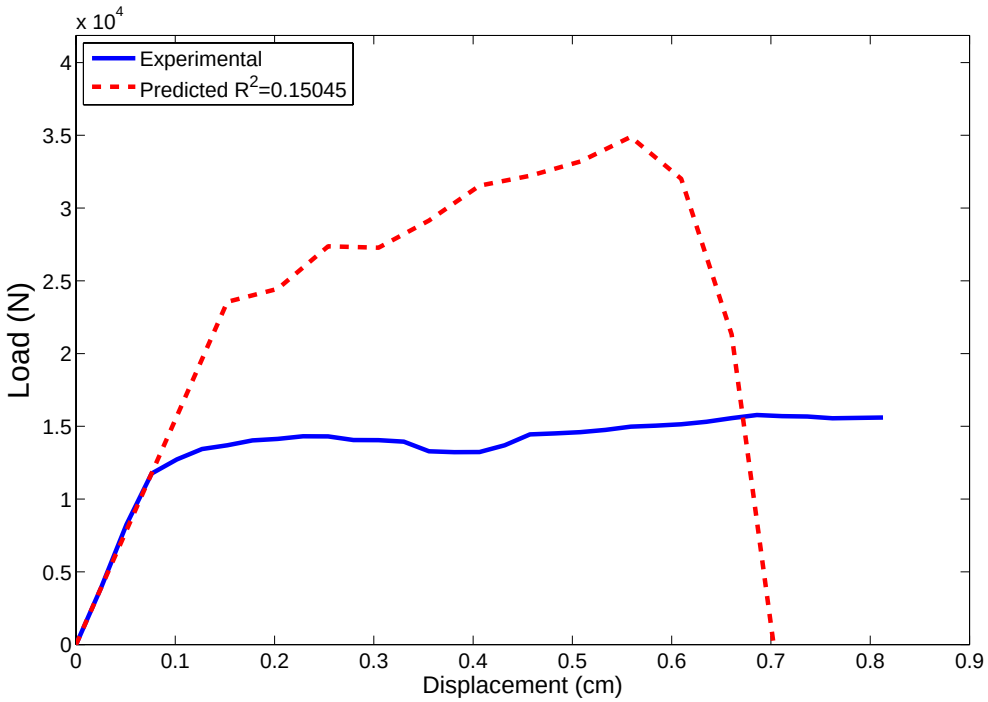


Figure 47. Predicted load-displacement for the BT coupon.

The predicted load displacement curve in Figure 47 represents a large deviation from experimental results and a r-squared statistic of 0.15. One of the factors that

contribute to the large deviation is the number of strain data points that lie out of bounds or in sub domain elements. Figure 48 contains the sub domain element fraction similar to Figure 39 plotted for the OHC results. Refer to the OHC section for a more detailed explanation of the labels in Figure 48. It is clear from the blue curve that the number of strain points that are outside the solution domain become significant at relatively small displacements. It is interesting to note that the number of strain points within sub domain elements is less for the BT compared to the OHC. Most likely this is a result of the IPL material database lacking compression coupons while containing many tension coupons. Subsequently there was more experimental data that exists for tension dominated applications than for compression dominated applications.

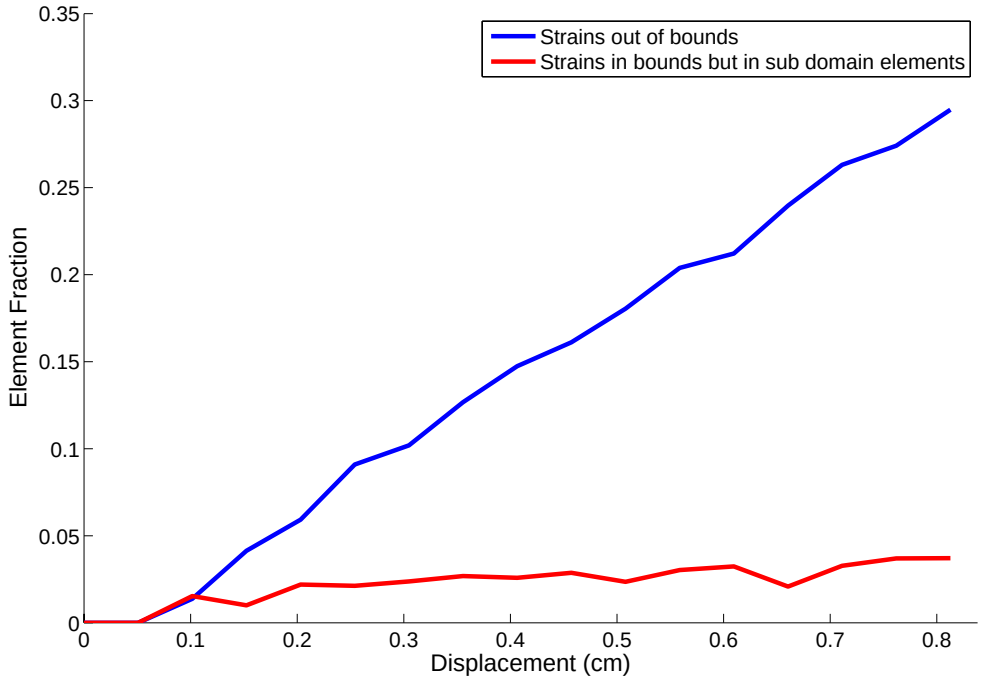


Figure 48. The fraction of sub domain elements as a function of the BT boundary displacement.

NONLINEAR MATERIAL MODEL

Material Nonlinearity

Based on the results presented previously for the OHC and the BT tests an examination of the assumptions of the DED database was warranted. One of the previously stated assumptions related to using linear FEA to obtain the strain distribution in the coupon. In the current development the results considering the pristine material were scaled linearly to provide a reasonable approximation of the strain distribution during damage accumulation. The remainder of this section will address the validity of this assumption.

In order to determine the validity of using a linear model it was necessary to construct a nonlinear model to compare with. Thus, it was necessary to determine the nonlinear material response. The approach was to determine an optimum set of in-situ material constitutive parameters. The number of parameters used to describe the constitutive response was limited to six under the assumption that the shape of the constitutive curve was bi-linear for all three planar stress-strain components. The best set of parameters for the bi-linear constitutive response was determined based on a nonlinear curve fit of the experimental data from the IPL material database.

A nonlinear ANSYS model was created using the geometry from an IPL coupon. The nonlinearity in the model was a result of the assumption of a bi-linear constitutive behavior. The notion of a bi-linear constitutive response for composites implies that there exists a linear material softening curve. Currently, the FEA software ANSYS has

reasonably good nonlinear capabilities. However, the minimum tangent slope of a material constitutive equation is zero corresponding with the response of a perfectly plastic material. This posed modeling problems for the bi-linear constitutive response that was desired for the composite material. To get around the limitation of the ANSYS software series of linear models were run based on a flow diagram displayed in Figure 49. Thus, for any bi-linear constitutive behavior the ANSYS model produced a set of boundary displacement curves that were compared with the curves produced from experimental data obtained from the IPL.

The two primary objectives for the nonlinear model were, (1) determine the best set of bi-linear constitutive parameters and (2) determine whether the strain distribution changes significantly during material softening compared to the strain distribution produced from the pristine material. If the strain distribution based on the nonlinear ANSYS model is significantly different than the linear counter part, the validity of the aforementioned linearly scalable strain field assumption is doubtful. Conversely, if the strain distribution in the nonlinear ANSYS model matches the linear model quite well, then the validity of the aforementioned assumption is no longer in question.

In-situ Material Constitutive Behavior

The material constitutive behavior for composite materials is significantly different from metal materials after the strain levels exceed the proportional limit. Yielding in metals has been studied for some time and plasticity theory is quite mature [16]. Unfortunately, the knowledge gained from metals yielding is not relevant for composite materials which

do not exhibit yielding behavior. As a consequence, composite materials lack an efficient mechanism for load redistribution and tend to retain load carrying capability until a sudden global failure occurs. Material softening in composites is a result of primarily matrix cracking and delamination. As a first cut at capturing the material softening in E-glass/epoxy coupons tested in the current work a linear softening curve was assumed. To completely define three bi-linear curves requires definition of six parameters. There are three strain parameters that define the proportional limit and three slopes that define the linear softening curve for each of the three in-plane strain components. The independence of these parameters is subject for debate. In fact, these parameters are most likely not independent. Consider a simplified example where a laminate develops transverse matrix cracking under a tension load in the primary fiber or axial direction. Naturally, the transverse crack decreases the stiffness in the axial direction. It is reasonable to assume that the shear stiffness would also be affected by the transverse cracking. Hence, there is an interaction between the axial direction and the shear. Transverse stiffness would most likely be unaffected by transverse cracking. The interaction between these parameters was not addressed in the current work. However, future work should address the interaction between these parameters. To determine the best set of six constitutive parameters required some optimization. Solving the inverse problem to obtain the optimum parameters would be difficult with a low potential of obtaining good results based on the complexity of the system. Thus, the approach was to solve the path forward problem multiple times to obtain the optimum set of parameters. Under this paradigm, the following steps were required to obtain results; (1) An initial

guess at the set of constitutive parameters was made. (2) The nonlinear ANSYS model was run as defined in the flow diagram of Figure 49. (3) The theoretical results were compared against the experimental results of the IPL material database. (4) Based on the difference between the theoretical and experimental results combined with engineering judgment of the behavior of physical system a more refined guess of constitutive parameters was made. Steps (1) - (4) were repeated until a satisfactory set of parameters were obtained. Hence, the convergence criteria were based on the comparison of the theoretical and experimental results. The complexity of the original path forward optimization problem was transformed into a multivariate minimization problem with fairly robust procedures outlined to obtain the information required for a solution.

In most cases the solution would be obtained using a traditional multivariate minimization routine such as conjugate-gradient method outlined in [17]. However, in the interest of time and other constraints a less refined method was employed in the current work. Namely, construct a grid of data points in six dimensional space to determine the gradient direction. The grid was used to determine a search direction that would theoretically lead to a better approximation of the fitting parameters after each iteration, thus a convergent method. One of the difficulties of the method was the relatively long duration of each nonlinear FEA run. Using dual core Dell personal computers a single run was averaging between four and six hours. Considering each fitting parameter as independent magnified the number of runs by six.

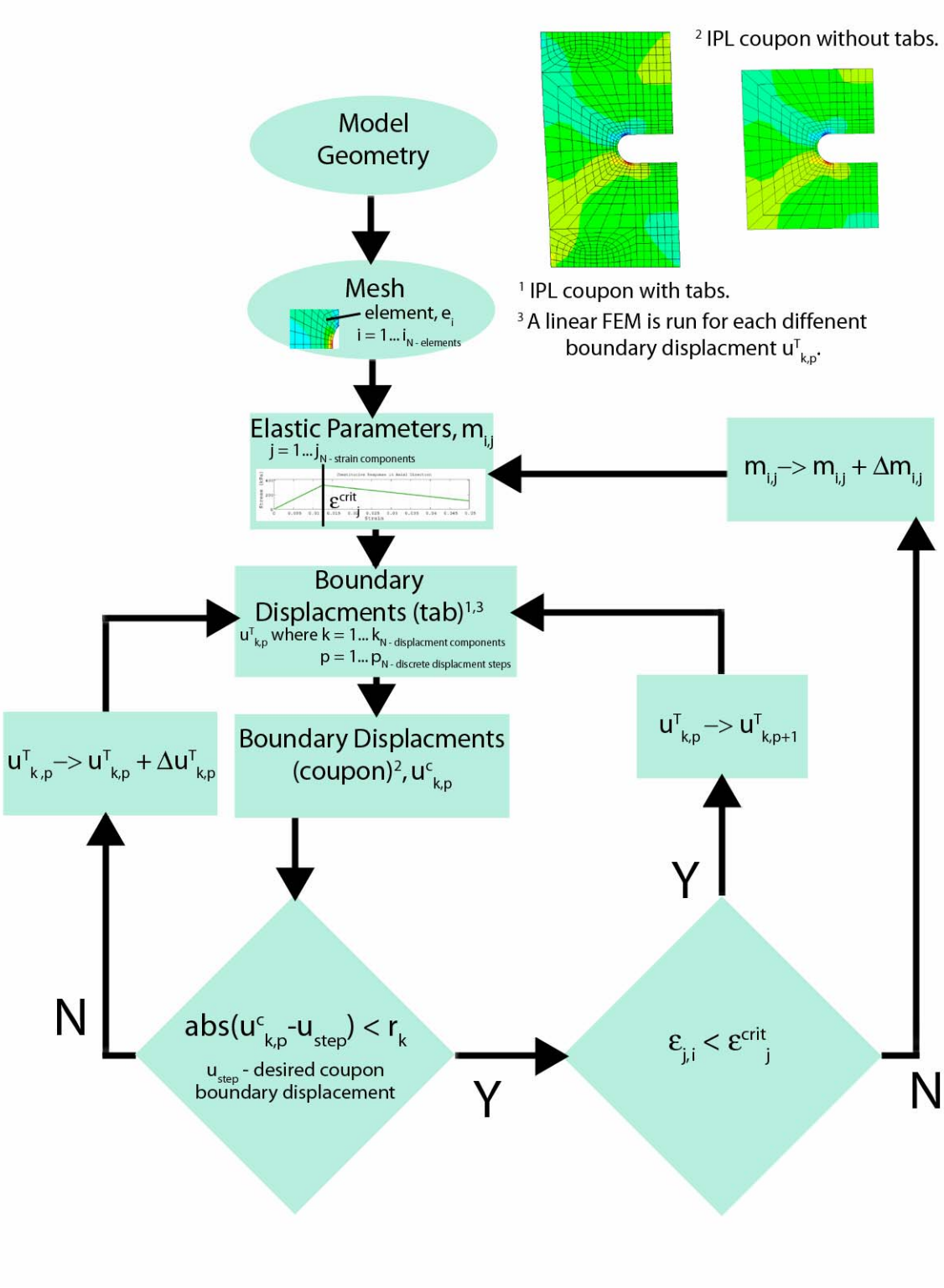


Figure 49. Nonlinear ANSYS model flow diagram.

Even considering a course grid of data points the computational time increases quite dramatically. For example consider a grid containing three data points in each dimension. This corresponds with eighteen runs at an average of five hours per run which equates to a total of ninety hours of computational time.

Figure 50 contains the curves that were determined as the best set of constitutive parameters under the current curve fitting paradigm. These values were obtained by fitting the experimental TDE-displacement curves opposed to the load-displacement curves. The critical strain value in the axial direction was approximately 1.25% which is quite a bit lower than expected value near 3%. Figure 51 and Figure 52 represent the curves that compare the results of the nonlinear FEA model with the experimental data. Extremely good correlation with experimental data for the Y-direction forces is shown in Figure 51. However, the predictions for the X-direction forces become significantly larger than the experimental data. It is interesting to note that the FEA model is capturing the Y-direction behavior quite well even predicting the initial softening corresponding to a displacement of 0.029 cm. Figure 52 represents the TDE curves that were used to determine the best set of parameters. The nonlinear curve fit represents the results of the best set of bi-linear constitutive parameters.

FEA model

To determine the effects of material softening on the strain distribution, a nonlinear FEA model was constructed and the best set of initial linear material constitutive properties was applied. The model followed the programming flow diagram defined in Figure 49.

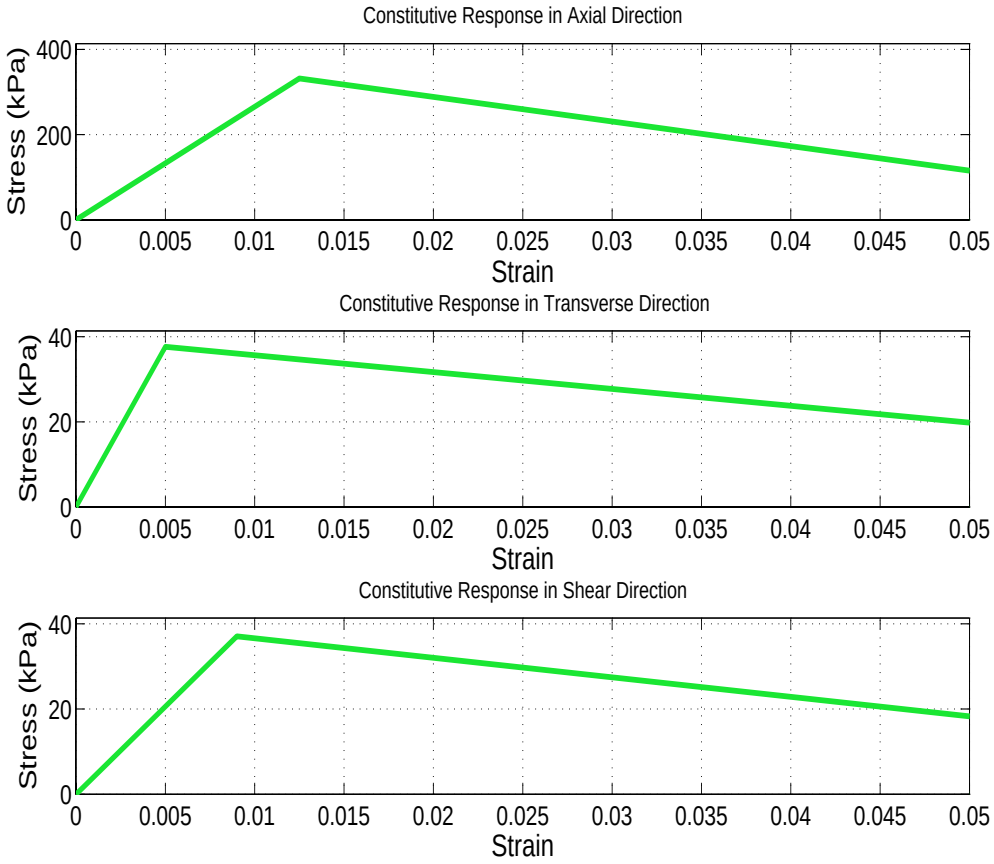


Figure 50. Bi-linear in-situ material constitutive response.

The model was terminated upon reaching coupon boundary displacements of 0.89 cm in both the X and Y direction. Similarly, a linear FEA model was run with the same coupon boundary displacements. Strain field contour plots were retrieved for both of the FEA models and compared side by side in Figure 53, Figure 54 and Figure 55. The ANSYS code written to build the nonlinear model is included as Appendix C.

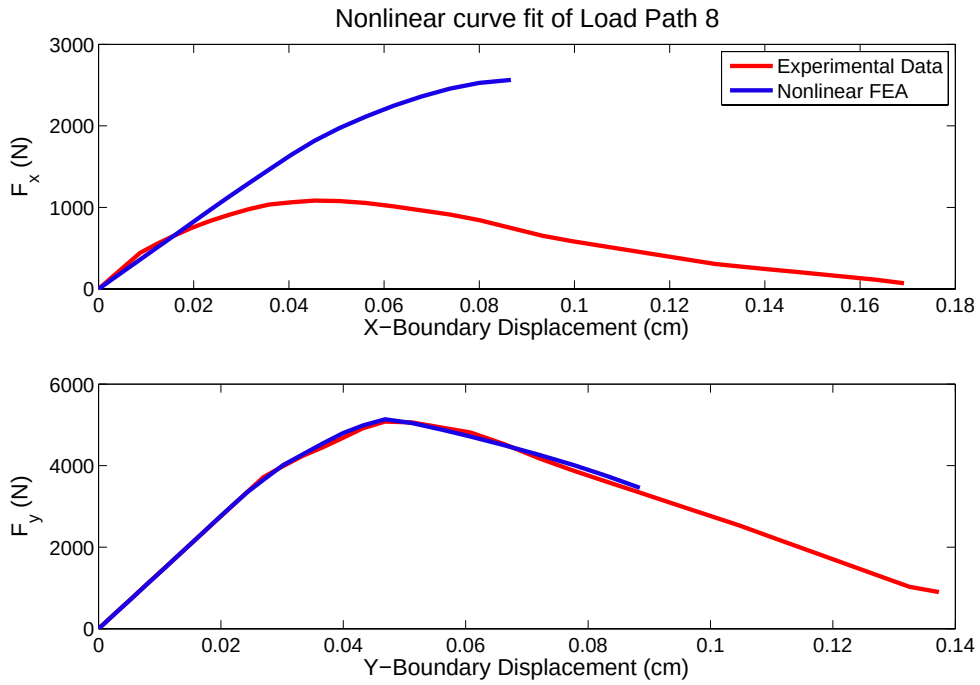


Figure 51. Nonlinear load-displacement curve fit of load path number 8.

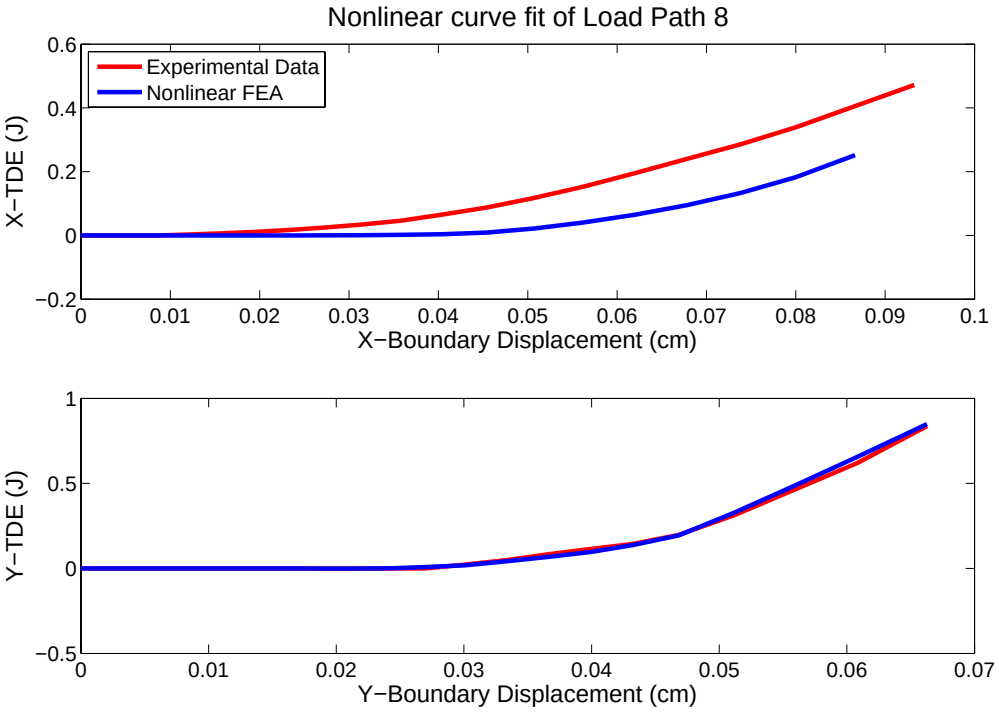


Figure 52. Nonlinear TDE-displacement curve fit of load path number 8.

Based on a visual analysis of the strain field contour plots it is clear that the distribution of strain is different between the linear and nonlinear FEA models. However, each set of strain contours share similarities significant enough not to be categorized as dramatic strain redistribution. A survey of the minimum and maximum strain yields the results in Table 4. Without presenting results explicitly, based on the procedures described in chapter four used to calculate the DED function suggests that changes in strain values of this approximate magnitude would most likely affect computations significantly.

Table 4. Minimum and maximum strain comparison between linear and nonlinear FEA models.

	Linear FEA	Nonlinear FEA	% Difference
X-Strain			
Min. Strain	-0.04	-0.131	227%
Max. Strain	0.061	0.133	118%
Y-Strain			
Min. Strain	-0.019	-0.064	236%
Max. Strain	0.106	0.159	50%
XY-Strain			
Min. Strain	0.132	0.101	23%
Max. Strain	0.04	0.013	67%

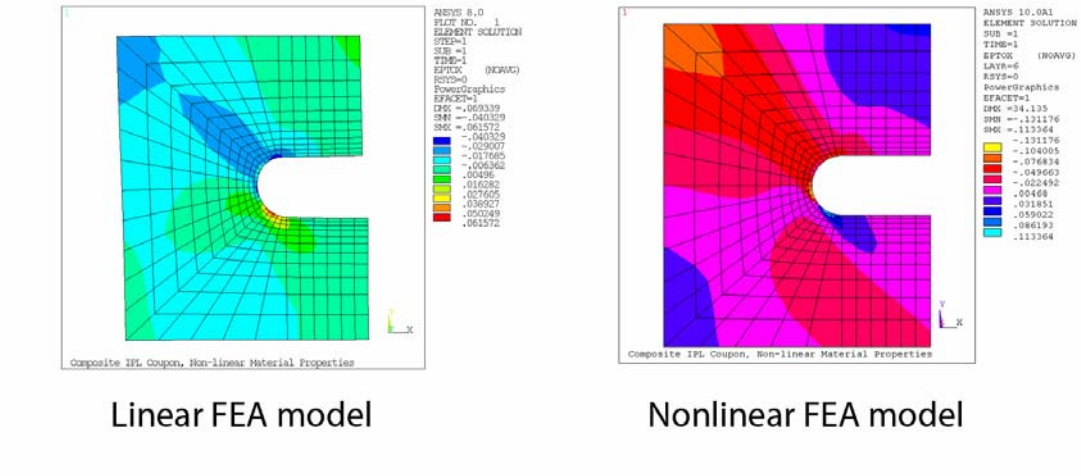


Figure 53. Linear vs. nonlinear X-direction normal strain distribution for load path number 8.

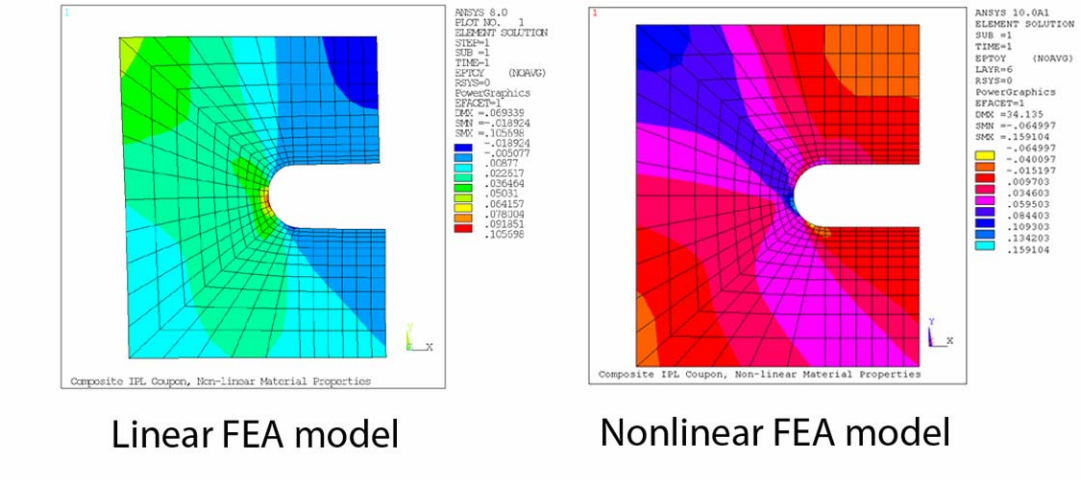


Figure 54. Linear vs. nonlinear Y-direction normal strain distribution for load path number 8.

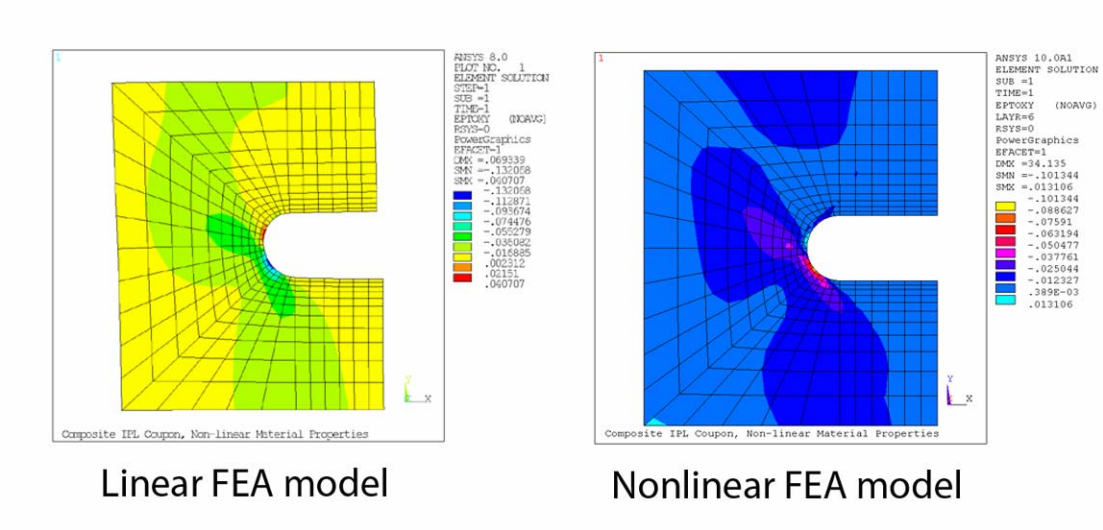


Figure 55. Linear vs. nonlinear shear strain distribution for load path number 8.

Based on the somewhat preliminary information presented in the current work the effects of material strain redistribution as a result of material softening is significant enough to warrant consideration with respect to the calculation of the DED function. Thus, the assumption that the strain field can be linearly scaled from the model of the pristine material to represent the damaged material is somewhat unstable. The validity of linearly scalable strains needs to be further evaluated based on the potential difference in the DED function using both the linear and nonlinear strain fields. In the interest of time, the implementation of the nonlinear FEA strains into the calculation of the DED function was not included as a part of the current work.

CONCLUSIONS

The goal for this thesis was to explore the current predictive capability and application of a composite material database produced using data from the IPL. The experimental database was used to obtain a DED function of the three in-plane strains. The database created for the current work was used to predict the behavior of representative composite material structure, namely open-hole compression and bearing tension coupons. Each of the comparisons with composite structure showed promise for future study. However, both of the predictions had some unresolved issues resulting in an incomplete representation of all of the material behavior. The validity of a critical assumption of the DED method was studied. Information was presented to help steer the direction future study regarding the use of linear finite element models to obtain the strain distribution in a damaged composite coupon.

Table 5. Composite structure prediction compared with experiment.

	Onset of damage	Ultimate capability	R-squared statistic
Open-hole compression	-39%	-8%	0.99
Bearing Tension	87%	133%	0.15

The following results for the open-hole compression and bearing tension tests are summarized in Table 5. It is clear that the DED database used in the current work performed much better predicting the open-hole compression compared to the bearing tension. Based on the assumptions of the DED method presented in the current work it is not surprising to deviate from the experimental data at ultimate capability, however it is

surprising that better results were not achieved predicting the onset of damage. Predicting the onset of damage was inaccurate due to over predicting the total dissipated energy curve using the linear least squares fit of the experimental data. To alleviate the problem a weighted curve fit of the data should be used such that a higher priority would be given to the fit of the total dissipated energy data near the onset of damage. The prediction of the onset of damage is compromised at the expense of fitting the total dissipated energy curve over the entire vector displacement range.

The influence of neglecting the nonlinear effects becomes greater at ultimate capability. The results of chapter six given in Table 4 show that including the nonlinear effects increases the extreme strains. The consequence of neglecting the nonlinear effects translate into an over prediction of the TDE. Thus, providing an explanation for the over prediction of the bearing tension coupon.

The results of the modeling the nonlinear effect assuming a in-situ bi-linear constitutive response were positive. A set of parameters corresponding with the best fit of the TDE curve was obtained. The set of material parameters plotted as stress-strain curves is shown in Figure 50. The set of parameters were applied to a nonlinear model to obtain the strain distribution for an IPL coupon. Ignoring the nonlinear effects may not be possible, but the preliminary nature of the current study precludes any strong conclusion.

Future Work

In-Plane Loader

The IPL was a corner stone for the current work. However, there are some unresolved issues surrounding the experimental data produced on the IPL. All of the suggestions included below are focused around improving the robustness of the entire testing process on the IPL. The current issues will be listed independently with a brief justification, however it should be noted that these issues may not occur independently.

1. Repetitive testing efficiency needs to be a design requirement of future iterations to the IPL. The cycle time for a single test on the IPL in the current configuration is approximately 20-30 minutes. The rather lengthy cycle time is associated with the manual nature of the IPL operations. One of the primary drivers of the undesirable cycle time is the nature of the Labview program. A number of things must be manually checked to ensure the correct test settings prior to initiating an IPL test. The database presented in the current work was derived based on the average results of two tests per load path. As a rule of thumb, to obtain statistically significant data at least five test per load path are necessary. Considering seventeen hours of testing was tied up in building the current database, the duration required to test five coupons per load path was prohibitive. More advanced Labview programming would be advantageous.

2. Frame compliance issues must be addressed in any derivative IPL. Many of the issues that have been identified would dissolve with the proper reiteration of the IPL frame

structure. Currently, the IPL frame compliance drives testing control issues that affect the test results. Hence, the problem is more than a matter of convenience. While some attempt was made herein to address the compliance during data reduction, future improvement should remain as an issue.

3. Data acquisition must be capable to capture data accurately during damage progression. Strain energy in the frame is available for further damage propagation in the coupon. The Labview program used to acquire data is dependent on actively driven boundary displacement convergence, hence the data sampling rate is not constant with respect to time. The strain energy released by the frame is transferred to the coupon via increased boundary displacement as the coupon softens due to damage. The data acquisition hardware is programmed to correspond with actively driven boundary displacement convergence, thus boundary displacements resulting from strain energy transfer are not captured with the IPL in the current configuration.

4. Alternative coupon displacement measuring techniques need to be investigated. Assuming that 100 percent displacement continuity at the coupon grip interface takes place is more theoretical than practical. Consequently, measuring coupon displacements at the grips should be abandoned. An alternative that has recently been used to obtain some promising preliminary results is high speed video with advanced image analysis algorithms.

5. The discrepancy between the prediction of the FEA model and the experimental results shown in Figure 17 must be resolved. The current method relies heavily on the ability to predict the coupon strains using a finite element model. Any deviation between the experimental results and the finite element model could lead to an erroneous DED database.

6. The Labview routine used to control the IPL coupon boundary displacement should be checked. The program that is currently being used has grown to be quite large making it overly complicated. Streamlining the code would make it much more reliable because it could be easily checked by multiple engineers and it would be much easier to work with. The basis for this recommendation comes from the curve presented for load path 8 in Figure 12. Load path eight represents a path with equal X and Y boundary displacement, hence at any arbitrary point the displacement components should be the same. Each data point should have the same X and Y boundary displacement. The discrepancy could be born from an error in the Labview program controlling the coupon boundary displacement convergence.

Future Database Development

The results of the two dissipated energy density (DED) databases produced for the current work showed good promise. Figure 23 and Figure 28 are most representative of the quality of the DED database for the bounded solution derived using all seventeen load paths and the bounded solution derived using select load paths respectively. Specifically for the bounded solution using select load paths the fit of the experimental total dissipated

energy (TDE) was quite encouraging. Future research should not overlook the importance of fitting the TDE data well when the total dissipated energy is relatively small. Through the use a weighted linear least squares routine a higher priority fit could be given to the portion of the curve corresponding to low TDE values, thus capturing the initial material damage response more accurately.

In-situ Ply Properties

The study of the in-situ ply constitutive response presented in the current work was quite preliminary. However, the methods used to obtain the results presented in chapter six have the potential of determining both dissipated energy and nonlinear constitutive response simultaneously. There are two key components required to proceed with future research. First, write the ANSYS code more immune to runtime errors. Nonlinear material softening requires some finesse to have robust runs using ANSYS. Under certain conditions associated with the nonlinear material softening a lengthy ANSYS run would fail to execute completely. The results derived for the current work relied heavily on user input making the process labor intensive. Second, use a more sophisticated nonlinear fitting routine to determine the set of constitutive parameters associated with the material model. In the interest of time, the current work used a manual iterative technique coupled with engineering judgment to converge on the best set of material parameters. Even thought the results turned out quite well, future study should employ a more sophisticated nonlinear optimization routine. The ultimate goal is to find the set of material parameters that reproduces the load-displacement curves experimentally determined with IPL testing. Currently, results were presented for load

path eight only, however future research would most likely want to include the information from multiple load paths in order to obtain the most representative set of material parameters.

Concluding Remarks

Overall, the current work provided some answers to questions surrounding the DED method and how the IPL data should be applied within the method. The current work also uncovered some new questions for future research in composite material characterization at MSU. The results contained within were intended to be used as a stepping stone along a path with a beginning and most importantly end corresponding to a mature method of characterizing composite materials. It is the author's intent for the current work to help minimize the discontinuities associated with the serial connection between researchers adjacent in time.

REFERENCES

[1] Jones, Robert. Mechanics of Composite Materials. New York: Hemisphere, 1975.

[2] Barbero, Ever. Introduction to Composite Materials Design. New York: Taylor & Francis, 1999.

[3] Soden, P.D., Hinton, M.J. and Kaddour, A.S. "A Comparison of the Predictive Capabilities of Current Failure Theories for Composite Laminates." Composites Science and Technology 58 (1998): 1225-1254.

[4] Hinton, M.J., Kaddour, A.S. and Soden, P.D. "Evaluation of Failure Prediction in Composite Laminates: Background to 'Part B' of the Exercise." Composites Science and Technology 62 (2002): 1481-1488.

[5] Toray: Technical Reference Page. Toray Industries, Inc., 29 May 2006<http://www.torayca.com/techref/>

[6] Adams, Donald F., Leif A. Carlsson, and R. Byron Pipes. Experimental Characterization of Advanced Composite Materials. Boca Raton: CRC Press, 2003.

[7] Collett, Aaron. "A Validation Study of the Montana State University In-Plane Loader." Diss. Montana State University, 2006.

[8] Ritter, William. "Application of Energy Methods to Modeling Failures in Composite Materials and Structures." Diss. Montana State University, 2004.

[9] Carbonite Metal Coatings. Carbonite Metal Coatings., 7 July 2006 <http://www.carbinite.com/>

[10] Adams, Donald. Telephone interview. 3 March 2006.

[11] Mast, P.W., G.E. Nash, J.G. Michopoulos, R. Thomas, R. Badaliance, and I. Wolock. "Characterization of strain-induced damage in composite based on the dissipated energy density Part I. Basic scheme and formulation." Theoretical and Applied Fracture Mechanics 22 (1995): 71-96.

[12] "Space Simulation; Aerospace and Aircraft; Composite Materials." Annual Book of ASTM Standards. 2003 ed.

[13] "Plastics (I): D 256-D 2343." Annual Book of ASTM Standards. 2003 ed.

[14] "Metals-Mechanical Testing; Elevated and Low-Temperature Tests; Metallography." Annual Book of ASTM Standards. 2003 ed.

[15] Niu, Michael C.Y. Composite Airframe Structures. Hong Kong: Conmilit Press, 1992.

[16] Shames, Irving H. and Francis A.Cozzarelli. Elastic and Inelastic Stress Analysis. New York: Taylor & Francis, 1997.

[17] Gill, Philip E., Walter Murray, and Margaret H. Wright. Practical Optimization. New York: Academic Press, 1981.

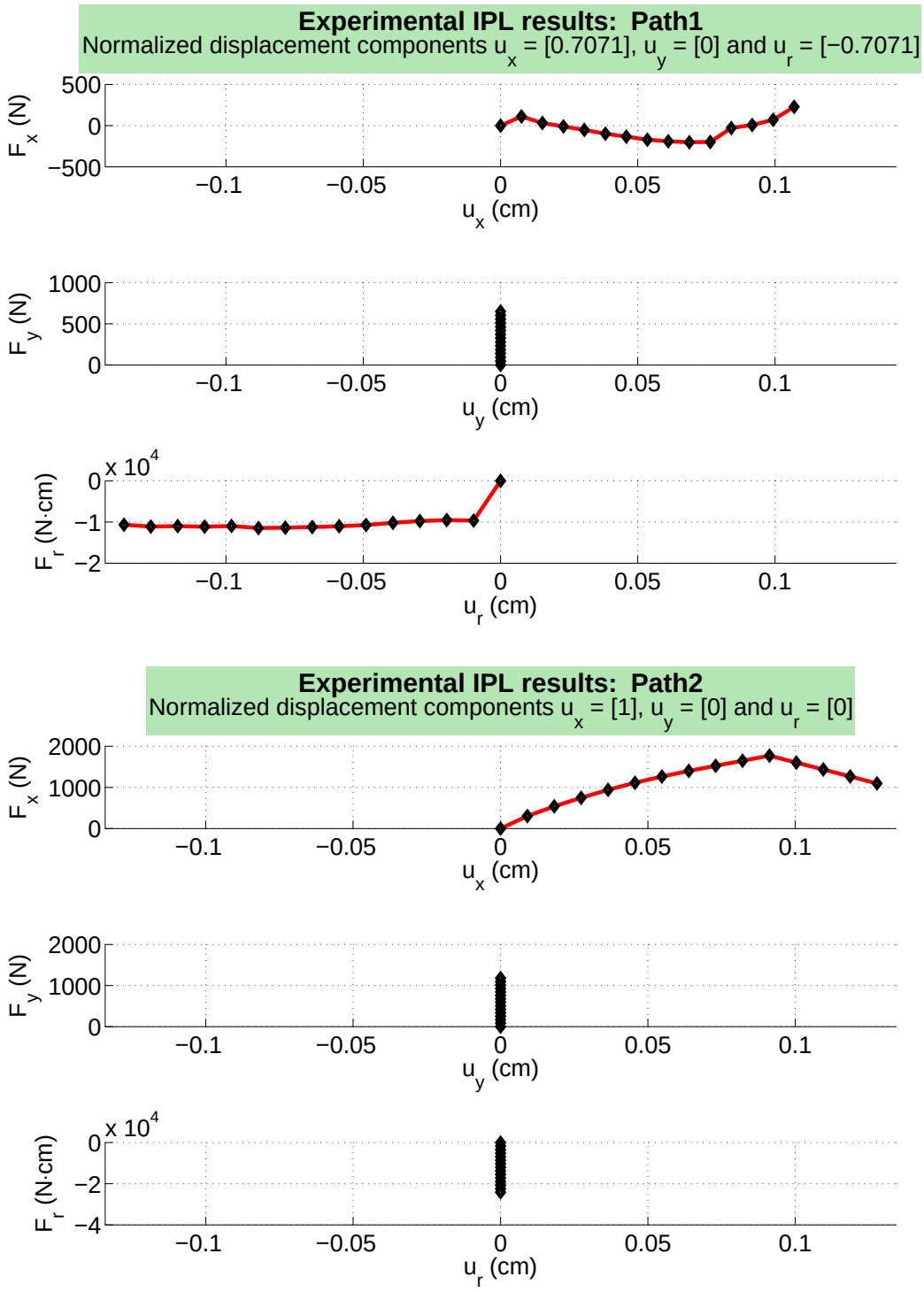
[18] "De Havilland Comet." Wikipedia. 16 December 2006 <http://en.wikipedia.org/wiki/De_Havilland_Comet>

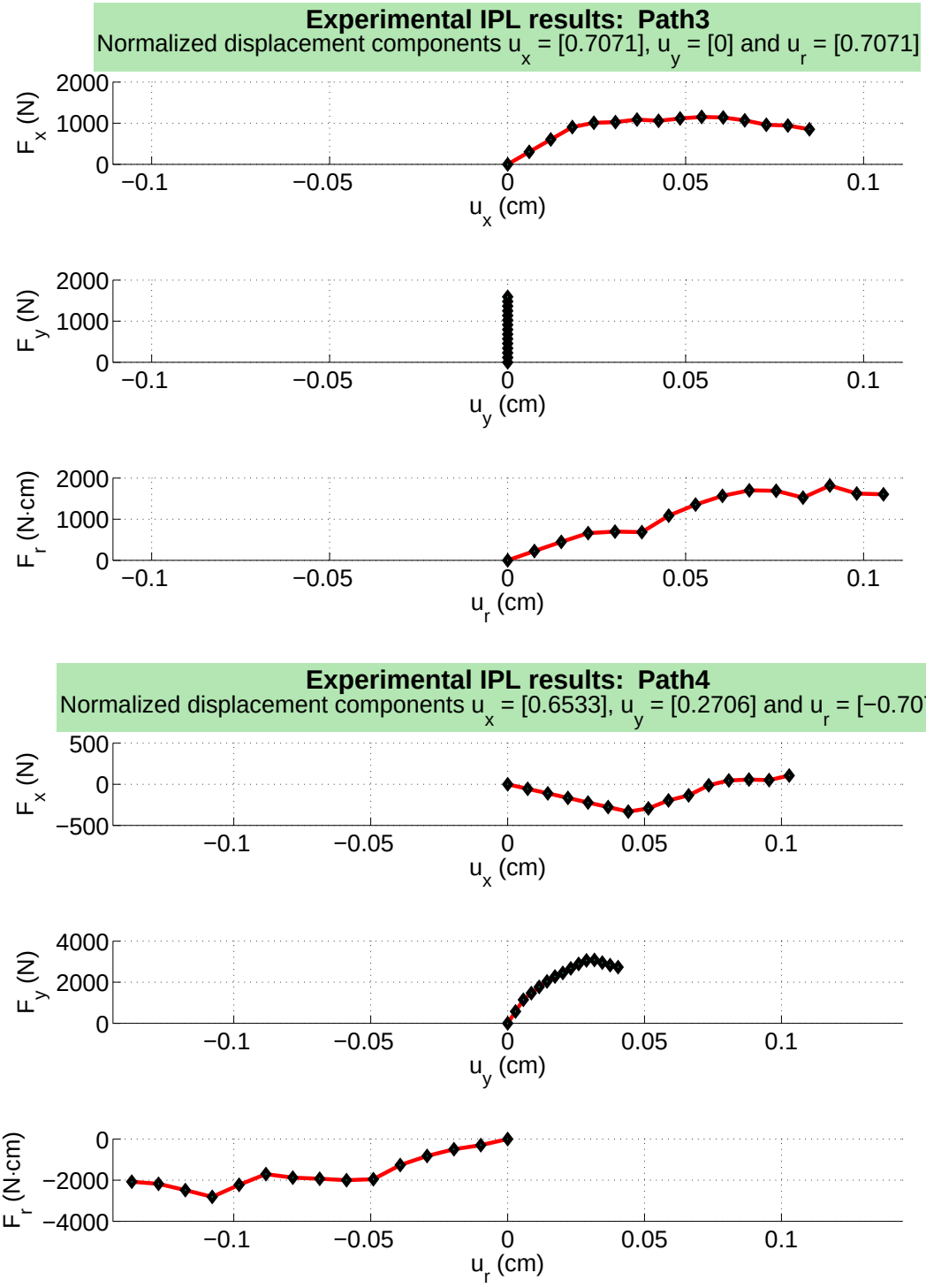
[19] Welsh, Jeffry S., and Donald F. Adams. "An experimental investigation of the biaxial strength of IM6/3501-6 carbon/epoxy cross-ply laminates using cruciform specimens." Composites: Part A 33 (2002).

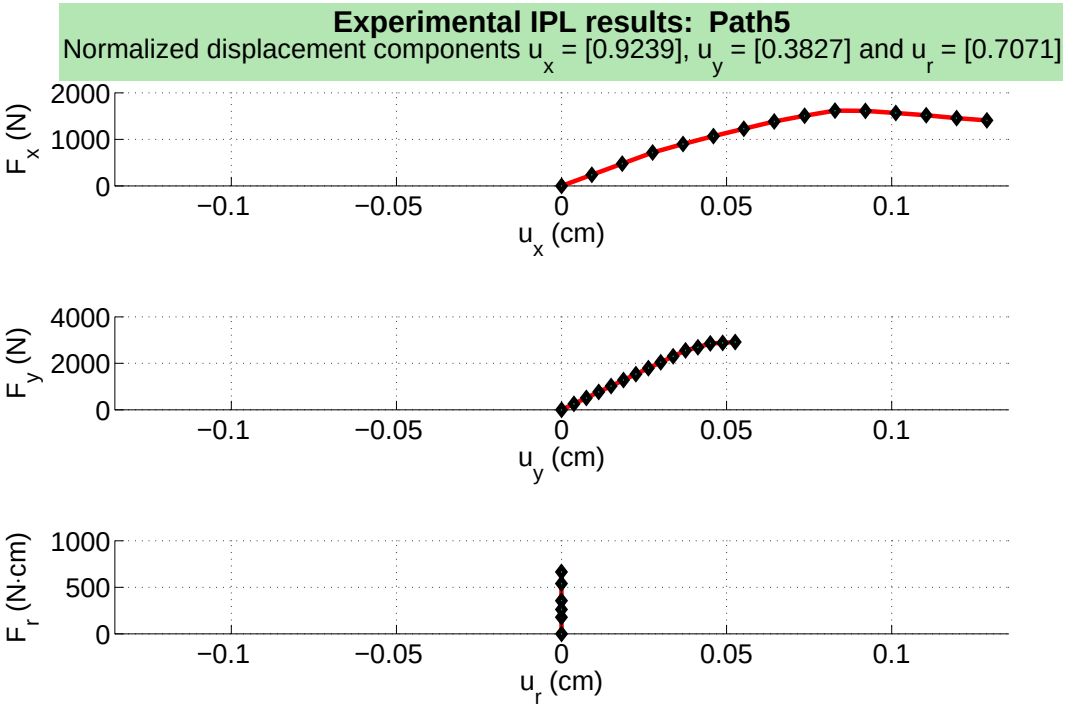
APPENDICES

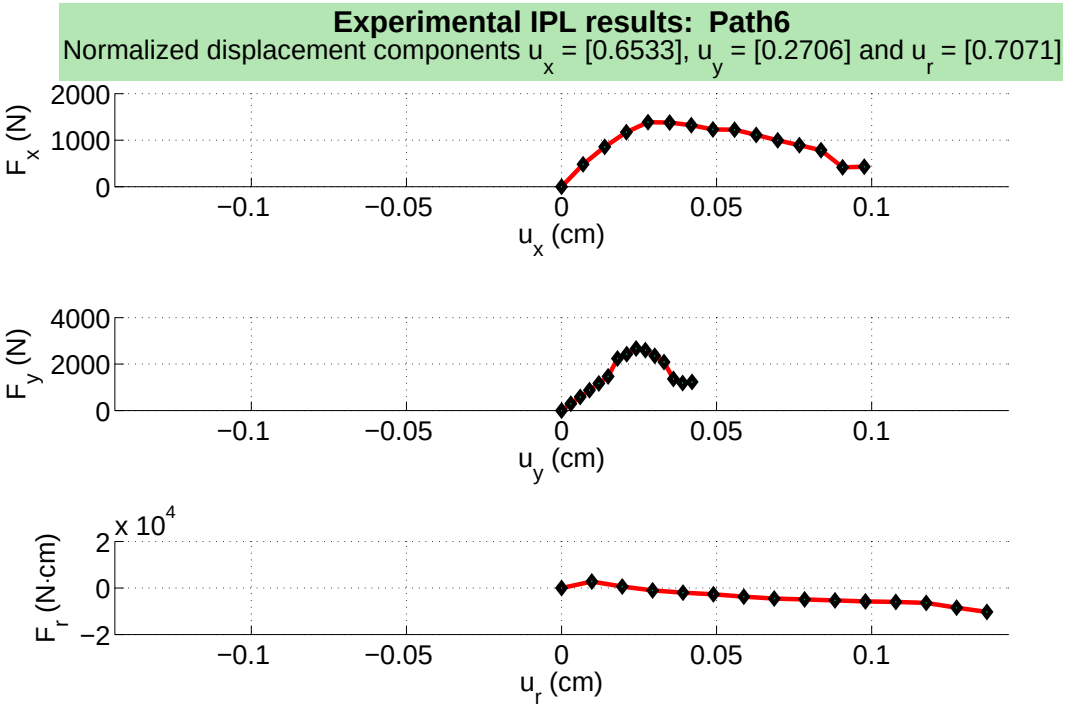
APPENDIX A

IPL EXPERIMENTAL DATA


Experimental IPL results: Path1
Normalized displacement components u_x = [0.7071], u_y = [0] and u_r = [−0.7071]
F_x (N)
500
0
−500
−0.1
−0.05
0
0.05
0.1
u_x (cm)
F_y (N)
1000
500
0
u_y (cm)
x 10^4
F_r (N·cm)
−1
−2
u_r (cm)
Experimental IPL results: Path2
Normalized displacement components u_x = [1], u_y = [0] and u_r = [0]
2000
1000
−4

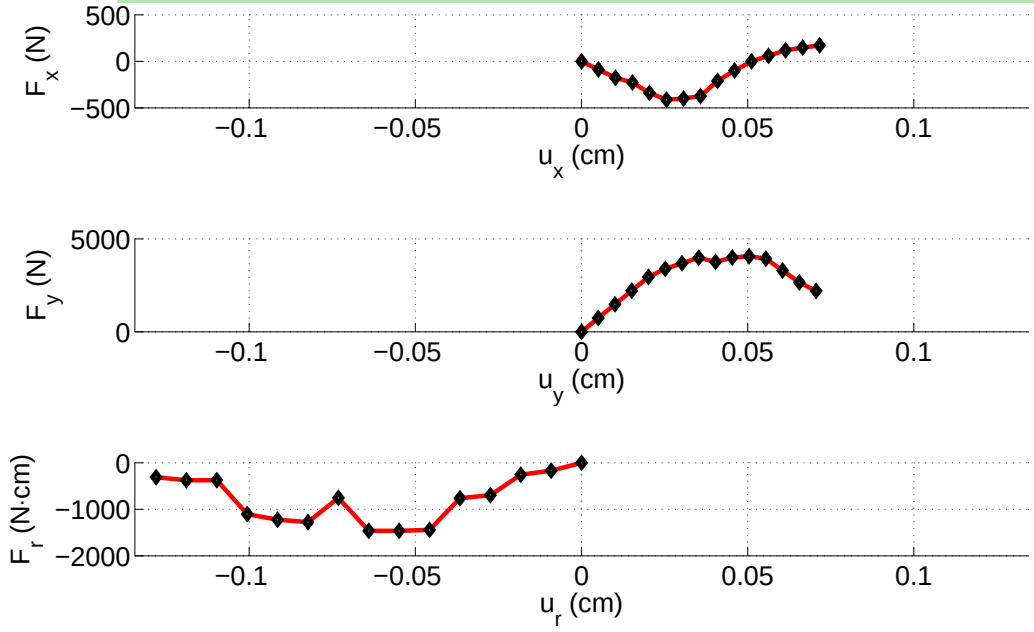

Experimental IPL results: Path3
Normalized displacement components u_x = [0.7071], u_y = [0] and u_r = [0.7071]
F_x (N)
F_y (N)
F_r (N·cm)
u_x (cm)
u_y (cm)
u_r (cm)
Experimental IPL results: Path4
Normalized displacement components u_x = [0.6533], u_y = [0.2706] and u_r = [−0.7071]
F_x (N)
F_y (N)
F_r (N·cm)
u_x (cm)
u_y (cm)
u_r (cm)


Experimental IPL results: Path5
Normalized displacement components u_x = [0.9239], u_y = [0.3827] and u_r = [0.7071]
F_x (N)
2000
1000
0
−0.1
−0.05
0
0.05
0.1
u_x (cm)
F_y (N)
4000
2000
0
u_y (cm)
F_r (N·cm)
1000
500
u_r (cm)


Experimental IPL results: Path6
Normalized displacement components u_x = [0.6533], u_y = [0.2706] and u_r = [0.7071]
F_x (N)
2000
1000
0
−0.1
−0.05
0
0.05
0.1
u_x (cm)
F_y (N)
4000
2000
u_y (cm)
F_r (N·cm)
x 10^4
2
−2
u_r (cm)

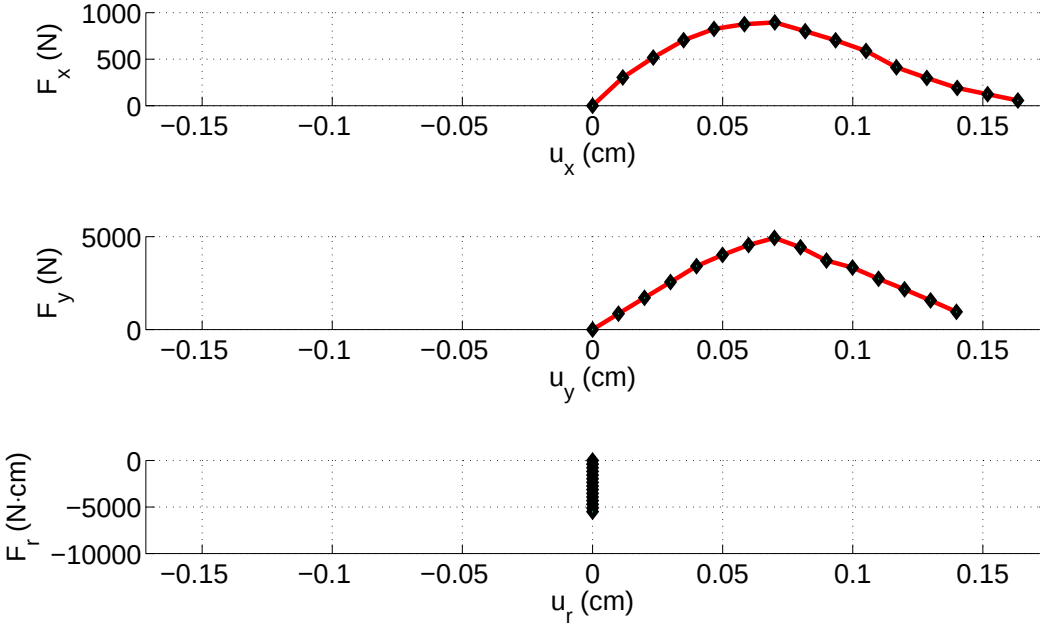
Experimental IPL results: Path7

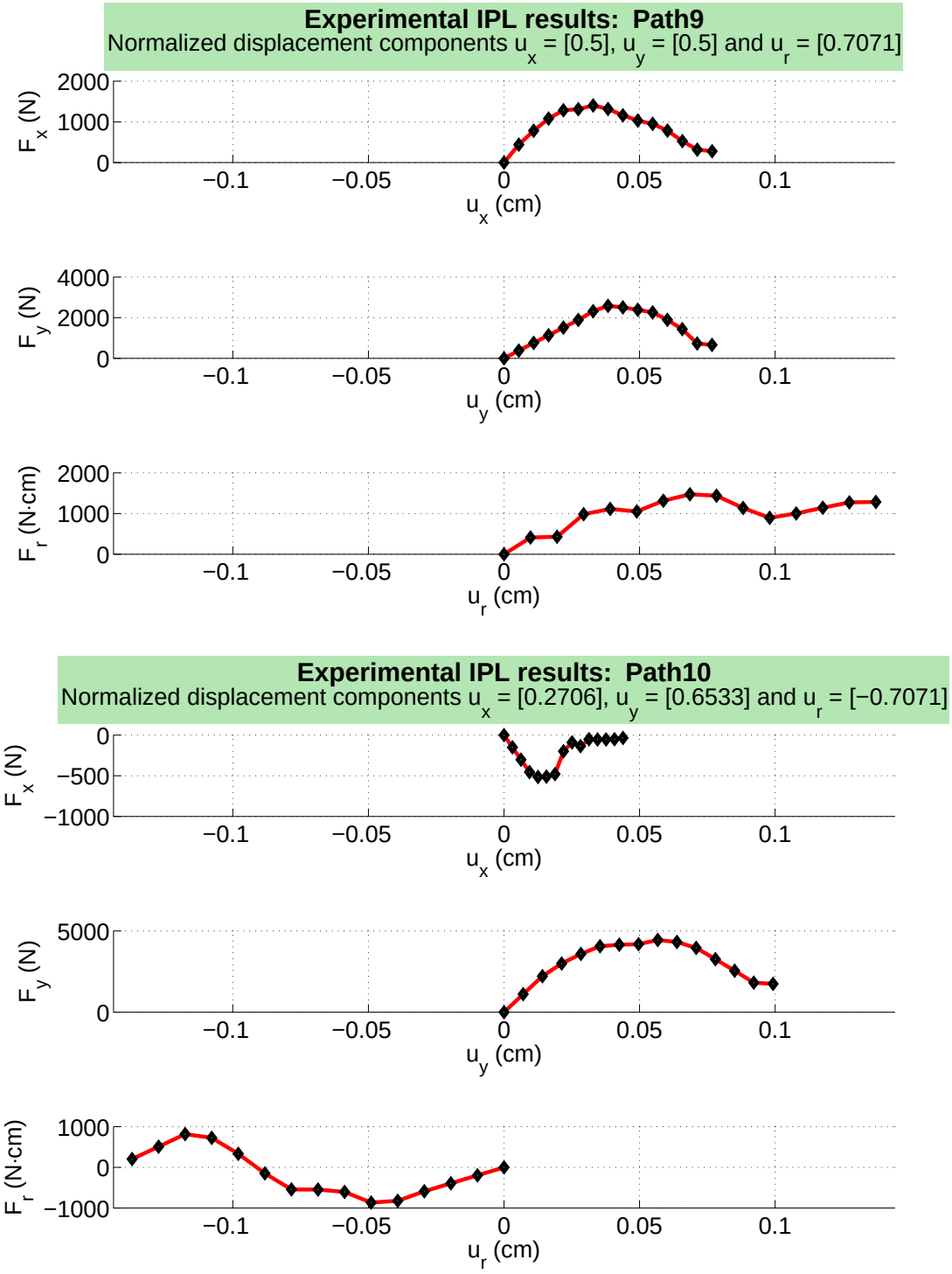
Normalized displacement components u_x = [0.5], u_y = [0.5] and u_r = [−0.7071]

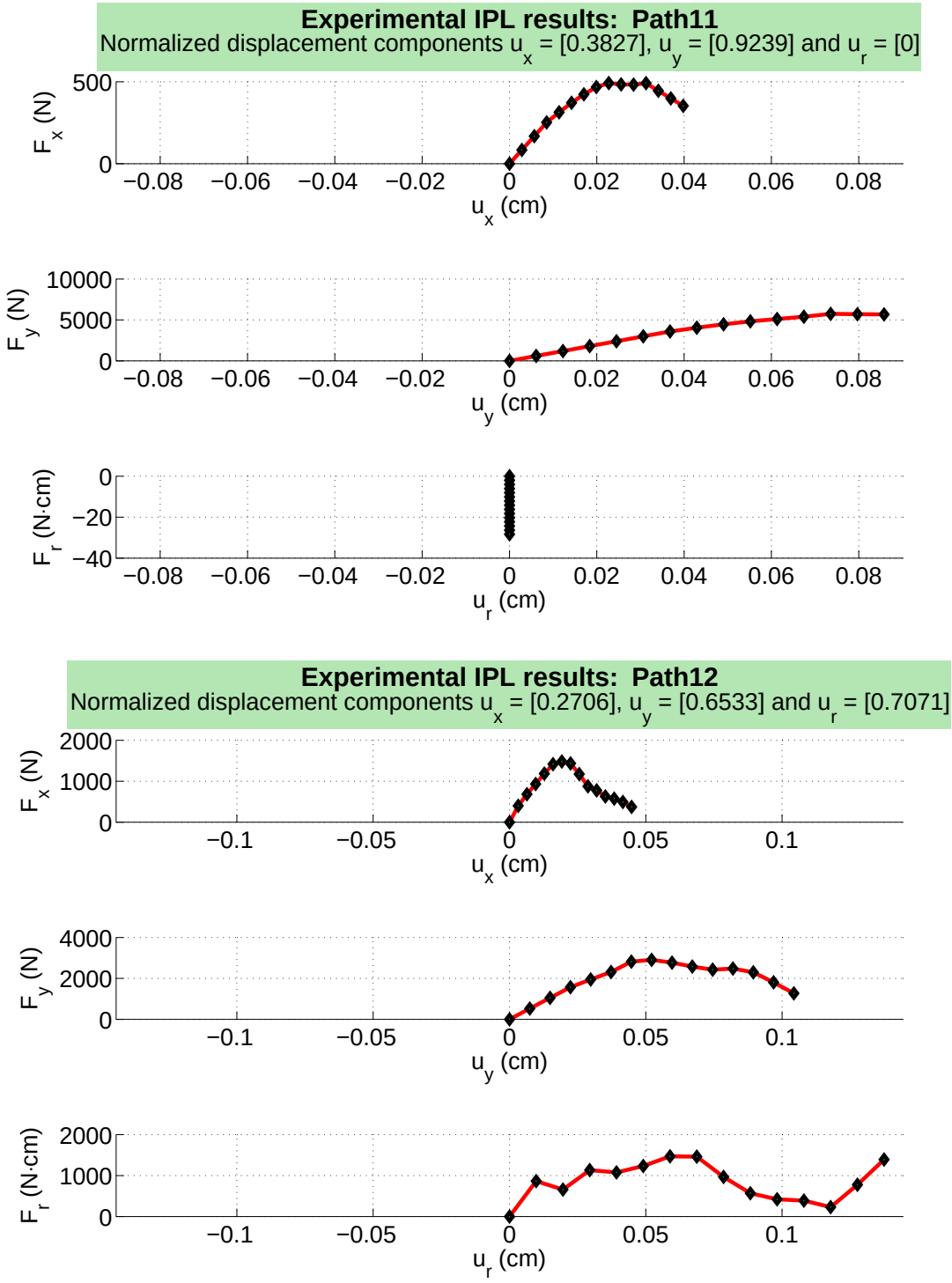


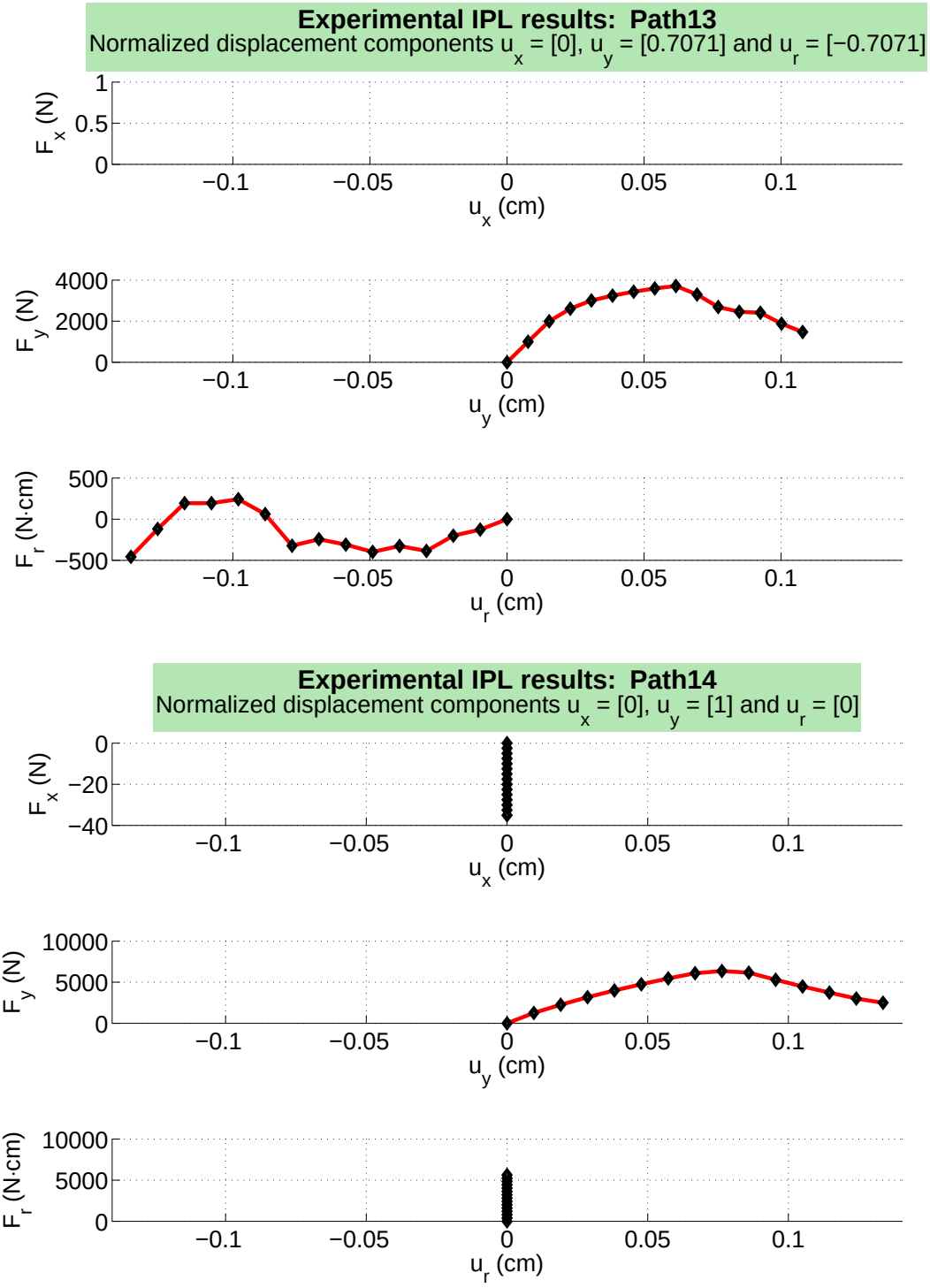
Experimental IPL results: Path8

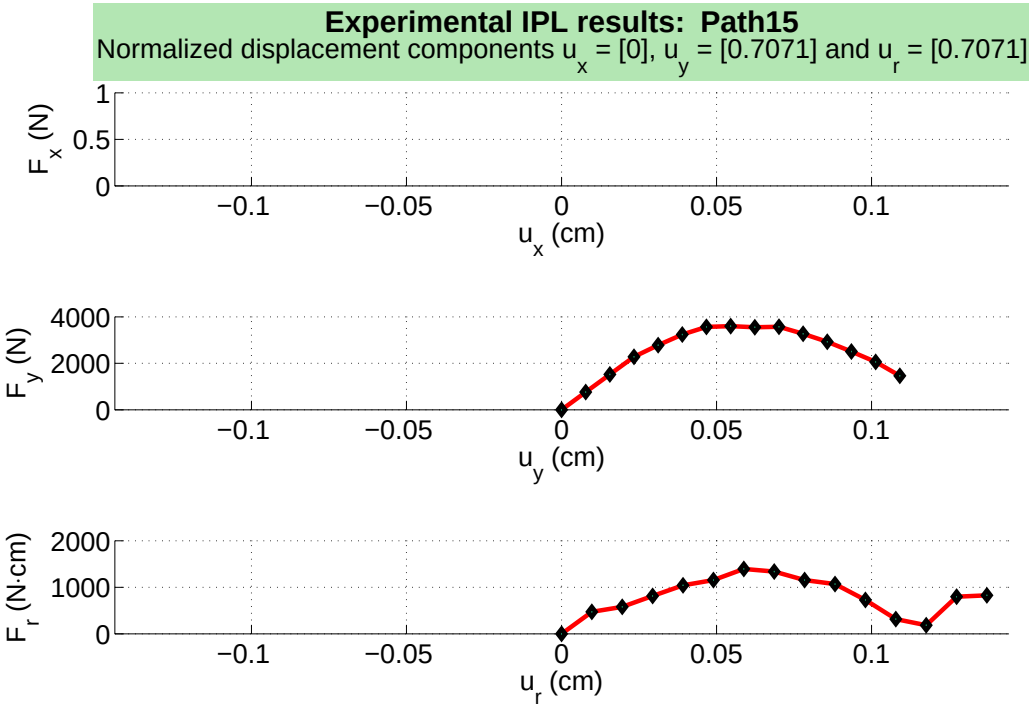
Normalized displacement components u_x = [0.7071], u_y = [0.7071] and u_r = [0]

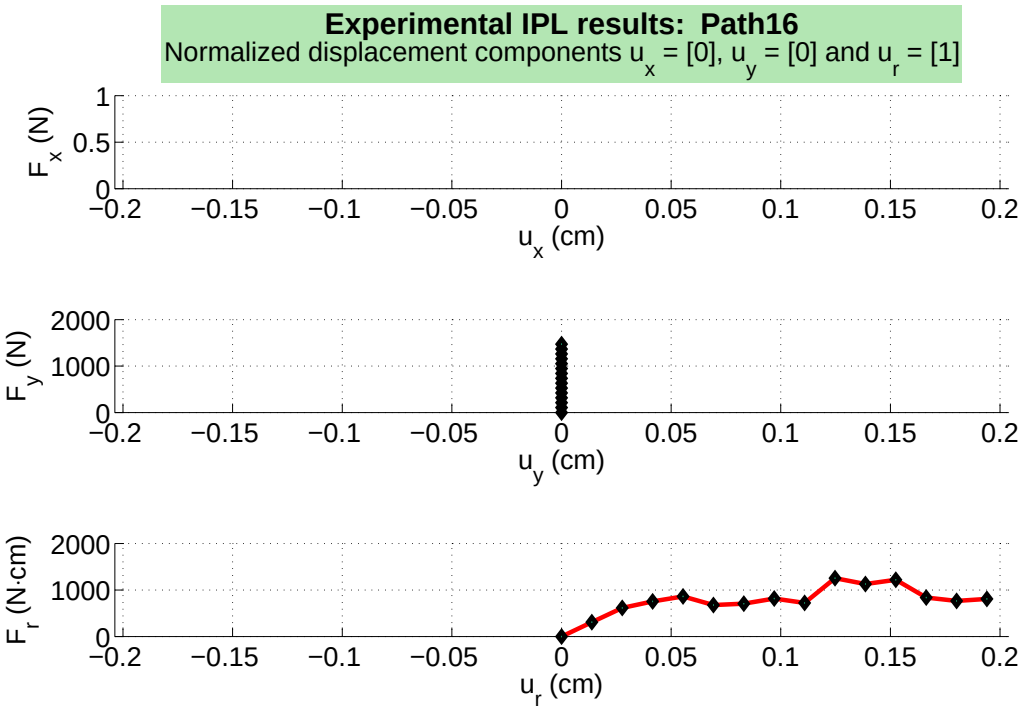


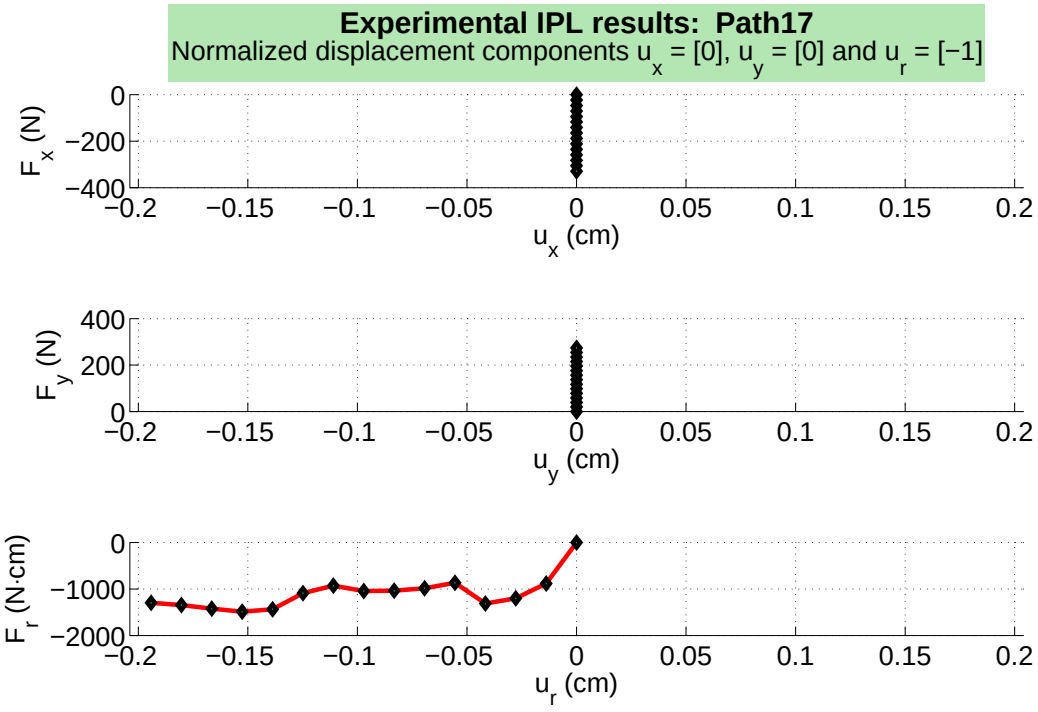

Experimental IPL results: Path9
Normalized displacement components u_x = [0.5], u_y = [0.5] and u_r = [0.7071]
F_x (N)
2000
1000
0
−0.1
−0.05
0
0.05
0.1
u_x (cm)
F_y (N)
4000
2000
0
u_y (cm)
F_r (N·cm)
u_r (cm)
Experimental IPL results: Path10
Normalized displacement components u_x = [0.2706], u_y = [0.6533] and u_r = [−0.7071]
−500
−1000
5000


Experimental IPL results: Path11
Normalized displacement components u_x = [0.3827], u_y = [0.9239] and u_r = [0]
F_x (N)
500
0
−0.08 −0.06 −0.04 −0.02 0 0.02 0.04 0.06 0.08
u_x (cm)
F_y (N)
10000
5000
0
−0.08 −0.06 −0.04 −0.02 0 0.02 0.04 0.06 0.08
u_y (cm)
F_r (N·cm)
0
−20
−40
−0.08 −0.06 −0.04 −0.02 0 0.02 0.04 0.06 0.08
u_r (cm)
Experimental IPL results: Path12
Normalized displacement components u_x = [0.2706], u_y = [0.6533] and u_r = [0.7071]
F_x (N)
2000
1000
0
−0.1 −0.05 0 0.05 0.1
u_x (cm)
F_y (N)
4000
2000
0
−0.1 −0.05 0 0.05 0.1
u_y (cm)
F_r (N·cm)
2000
1000
0
−0.1 −0.05 0 0.05 0.1
u_r (cm)


Experimental IPL results: Path13
Normalized displacement components u_x = [0], u_y = [0.7071] and u_r = [−0.7071]
F_x (N)
u_x (cm)
F_y (N)
u_y (cm)
F_r (N·cm)
u_r (cm)
Experimental IPL results: Path14
Normalized displacement components u_x = [0], u_y = [1] and u_r = [0]
F_x (N)
u_x (cm)
F_y (N)
u_y (cm)
F_r (N·cm)
u_r (cm)


Experimental IPL results: Path15
Normalized displacement components u_x = [0], u_y = [0.7071] and u_r = [0.7071]
F_x (N)
u_x (cm)
F_y (N)
u_y (cm)
F_r (N·cm)
u_r (cm)


Experimental IPL results: Path16
Normalized displacement components u_x = [0], u_y = [0] and u_r = [1]
F_x (N)
u_x (cm)
F_y (N)
u_y (cm)
F_r (N·cm)
u_r (cm)


Experimental IPL results: Path17
Normalized displacement components u_x = [0], u_y = [0] and u_r = [−1]
F_x (N)
0
−200
−400
−0.2
−0.15
−0.1
−0.05
0
0.05
0.1
0.15
0.2
u_x (cm)
F_y (N)
400
200
0
u_y (cm)
F_r (N·cm)
−1000
−2000
u_r (cm)

APPENDIX: B

MATLAB CODE

MATLAB FILES

DED_func_eff.m
ANSYS_test_ixy.m
DE_total_eff.m
Data_fit.m
Secant_fit.m
Diss_E_calc.m
TDE_check_eff.m
IPL_aquire.m
Ply_Principal_Strain.m
Local_index_new.m
Solve_full.m
Finite_diff_bad_implicit.m
Max_E.m
No_data_elem.m
Check_uyy_eff.m
FvsD_IPL.m
IPL_aquire.m
Fc.m
Density.m

```
%DED_func_eff.m

tic

%Define the total number of load paths.
nl = 17;

%Define minimum total points on all load paths
total_points = 1000;  % off by 3 or 4

%Define the number of plys in the test coupon.
plys = [2 2 2];

%{
Define a vector containing the different ply orientations
in IPL
coordinates.
%}
theta_vec_deg = [90 45 -45];

%Get the combined results from the FEA model and IPL data.
global n_elem; global VE; global DET; global u_paths
[epsilon_global,DET,VE,n_elem,u_paths] =
ANSYS_test_ixy(nl,total_points);

%Calculate the ply principal strains.
global epsilon_local
[epsilon_local] =
Ply_Pricipal_Strain(epsilon_global,theta_vec_deg);

%Calculate the fraction of each ply in the laminate.
global frac_plys
frac_plys = plys/sum(plys);

%Get the extreme strain values to define the solution
space.
global Y
index = 0;
for i = 1:size(epsilon_local,1)
    for j = 1:size(epsilon_local,2)
        if ~isempty(epsilon_local{i,j})
            index = index + 1;
            Y(index,1) = DET{i,j}(1,1);
            min_cell(index,:,:) = min(epsilon_local{i,j});
            max_cell(index,:,:) = max(epsilon_local{i,j});
            size_A_dim1 = index;
        end
    end
end
global e_max; global e_min
e_min = min(min(min_cell,[],3));
```

```
e_max = max(max(max_cell,[],3));

%Define the number of nodes that span the solution space.
global n_nodes
n_nodes = [5 5 7];

%Mesh each strain direction with nodes.  Origin is a
required node.
global e_mesh

for i = 1:3
    R = abs(e_min(i))/abs(e_max(i));
    n_right = n_nodes(i)/(R+1);
    n_left = R*n_right;
    if ceil(n_left)<2
        n_left = n_left + 1;
        n_right = n_nodes(i) - n_left;
    end
    e_mesh{i} =
unique([linspace(e_min(i),0,ceil(n_left)).';linspace(0,e_ma
x(i),ceil(n_right)).']);
end

%{
Assemble the element dimensions vector. (Assume each
element has the same
dimensions.
%}
global a
a{1} = (e_mesh{1}(2:end) - e_mesh{1}(1:end-1))/2;
a{2} = (e_mesh{2}(2:end) - e_mesh{2}(1:end-1))/2;
a{3} = (e_mesh{3}(2:end) - e_mesh{3}(1:end-1))/2;

%Assemble the local coordinate location vector.
global d
d{1} = (e_mesh{1}(2:end) + e_mesh{1}(1:end-1))/2;
d{2} = (e_mesh{2}(2:end) + e_mesh{2}(1:end-1))/2;
d{3} = (e_mesh{3}(2:end) + e_mesh{3}(1:end-1))/2;

%Define the global node numbers.
global global_nodes
for i = 1:length(n_nodes)
    for k = 1:(n_nodes(i)-1)
        global_nodes{i}(k,:) = cat(2,k,k+1);
    end
end

%Define the linear interpolation functions.
Nm =@(Xi) (1/8)*(1+Xi(1))*(1+Xi(2))*(1-Xi(3));
Nn =@(Xi) (1/8)*(1-Xi(1))*(1+Xi(2))*(1-Xi(3));
No =@(Xi) (1/8)*(1+Xi(1))*(1-Xi(2))*(1-Xi(3));
```

```
Np =@(Xi) (1/8)*(1-Xi(1))*(1-Xi(2))*(1-Xi(3));

Nq =@(Xi) (1/8)*(1+Xi(1))*(1+Xi(2))*(1+Xi(3));
Nr =@(Xi) (1/8)*(1-Xi(1))*(1+Xi(2))*(1+Xi(3));
Ns =@(Xi) (1/8)*(1+Xi(1))*(1-Xi(2))*(1+Xi(3));
Nt =@(Xi) (1/8)*(1-Xi(1))*(1-Xi(2))*(1+Xi(3));

%Define a strain tolerance for the extreme interpolation
nodes.
global epsilon
epsilon = 1e-5;

%Define a function handle to calculate the local coordinate
index.
loc_coord_index =@(t1,t2,t3) t1+(n_nodes(1)-1)*(t2-
1)+(n_nodes(1)-1)*(n_nodes(2)-1)*(t3-1);

%Define a function handle to calculate the global node
number.
g_node =@(t1,t2,t3) t1+n_nodes(1)*(t2-
1)+n_nodes(1)*n_nodes(2)*(t3-1);

times(1) = toc;

tic
A = zeros(size_A_dim1,prod(n_nodes));
global loc_coord
kdx = 0;
jdx = 0;
%Build the set of equations to solve.
for j = 1:size(epsilon_local,1)       %load paths
    for m = 1:size(epsilon_local,2)       %points on load
path
        if ~isempty(epsilon_local{j,m})
            jdx = jdx + 1;
            temp_A = zeros(1,prod(n_nodes));
            for k = 1:n_elem      %elements
                for i = 1:length(plys)        %plys

                    kdx = kdx+1;

                    %Define an arbitraty x point in the
solution space.
                    x0 = epsilon_local{j,m}(k,:,i);

                    %Compute the local coordinate indices.
                    i0 = loc_index_new(x0);


%!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!
```

```
                    %Calculate the correct local
coordinates.
                    under = find(epsilon > abs(x0 -
e_min));
                    over = find(epsilon > abs(x0 - e_max));
                    inside = setdiff(1:3,[under over]);

                    Xi(under) = -1;
                    Xi(over) = 1;
                    for mdx = inside
                        Xi(mdx) = (x0(mdx) -
d{mdx}(i0(mdx)))/a{mdx}(i0(mdx));
                    end;
                    if ~isempty(find((abs(Xi) - 1) >
epsilon))
                        display('Incorrect local coordinate
system is selected, out of bounds.')
                        return
                    end

%!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!

                    N = zeros(1,prod(n_nodes));

                    %Determine n and m in global node
numbers.
                    q1 = global_nodes{1}(i0(1),1);
                    q2 = global_nodes{1}(i0(1),2);
                    e1 = global_nodes{2}(i0(2),1);
                    e2 = global_nodes{2}(i0(2),2);
                    v1 = global_nodes{3}(i0(3),1);
                    v2 = global_nodes{3}(i0(3),2);

                    g_m = g_node(q2,e2,v1);
                    g_n = g_node(q1,e2,v1);
                    g_o = g_node(q2,e1,v1);
                    g_p = g_node(q1,e1,v1);

                    g_q = g_node(q2,e2,v2);
                    g_r = g_node(q1,e2,v2);
                    g_s = g_node(q2,e1,v2);
                    g_t = g_node(q1,e1,v2);

                    N(g_m) = Nm(Xi);
                    N(g_n) = Nn(Xi);
                    N(g_o) = No(Xi);
                    N(g_p) = Np(Xi);

                    N(g_q) = Nq(Xi);
                    N(g_r) = Nr(Xi);
```

```
                    N(g_s) = Ns(Xi);
                    N(g_t) = Nt(Xi);

                    %Build the A matrix.
                    %temp_A = temp_A +
N*VE{j,m}(k)*frac_plys(i);
                    temp_A = temp_A + N*VE{j,m}(k);

                    %Define the element number for each
strain point.
                    loc_coord(kdx,1) =
loc_coord_index(i0(1),i0(2),i0(3));
                end
            end
        end
        A(jdx,:) = temp_A;
    end
end
times(2) = toc;

%Select solution technique.
solve_full
%solve_part

clock
```

```
function[epsilon_global,DET,VE,n_elem,u_out]=ANSYS_test_ixy
(nl,total_points)
%{
This function takes all the strain values from the three
FEA models and
manipolates it in order to get strain values for all the
load points and
all the load paths.
%}

%{
If 'plots = 1' then the figures of this function will be
created.
If 'plots = 0' then none of the figures will be created
%}
plots = 0;

%{
Define the distance from the origin of the coupon
coordinates to the top
grip face.
%}
arm = 0.625;        %(in)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%
%Define the displacement value that the FEA model was run
at.
model_uxx = 0.00993;        %negative in FEA
coordinates(in)
                            %positive in IPL coordinates

%Load strain file to determine number of elements
global_sizer = dlmread('uxx_strain.csv',',',0,0);
n_elem_uxx = size(global_sizer,1);

%Define the strain values for each displacement step
1:num_load_points.
epsilon_global_uxx = global_sizer(:,1:3);
%epsilon_global_uxx = randn(n_elem_uxx,3);

%Defines the element volumes.
vol_elem_uxx = global_sizer(:,4);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%
%Define the displacement value that the FEA model was run
at.
model_uyy = 0.01;       %(in)

%Load strain file to determine number of elements
```

```
global_sizer = dlmread('uyy_strain.csv',',',0,0);
n_elem_uyy = size(global_sizer,1);

%Define the strain values for each displacement step
1:num_load_points.
epsilon_global_uyy = global_sizer(:,1:3);
%epsilon_global_uyy = randn(n_elem_uyy,3);

%Defines the element volumes.
vol_elem_uyy = global_sizer(:,4);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%
theta_min = -12;
theta_max = 12;
d_theta = 1;

model_uxy = theta_min:d_theta:theta_max;

for theta = model_uxy
    global_sizer = dlmread(['uxy_eff' int2str(theta)
'.csv'],',',0,0);
    epsilon_global_uxy(:,:,theta - theta_min + 1) =
global_sizer(:,1:3);
end
n_elem_uxy = size(global_sizer,1);
vol_elem_uxy = global_sizer(:,4);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%


%Assign number of elements for uniform models
n_elem = n_elem_uxx;

%Build a matrix structure containing all the strain values.
epsilon_global_full =
cat(3,epsilon_global_uxx,epsilon_global_uyy,epsilon_global_
uxy);

%Build a matrix structure containing all the volumes of
each element.
vol_elem_all =
cat(3,vol_elem_uxx,vol_elem_uyy,vol_elem_uxy);

%{
Define a matrix of the FEA model displacement values, and
convert the
rotational displacements to linear displacements using the
variable 'arm'.
%}
```

```
delta_u = [model_uxx  model_uyy  model_uxy*arm*(2*pi/360)];
%{
Create a matrix structure of strain values normalized wrt
FEA model
displacements.
%}
%Column swap bewteen x and y strains.
norm_epsilon_global = epsilon_global_full;
for i=1:2
    norm_epsilon_global(:,:,i) =
epsilon_global_full(:,:,i)/delta_u(i);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%

% Determine range of displacement data for even data
fitting
u_sum = 0;
for j = 1:nl
    Lpath = ['Path' num2str(j)];
    [DE_total_paths,u_fit] = DE_total_eff(Lpath,12);

    if ~isempty(DE_total_paths)
        lp_max(j) = max(max(abs(u_fit)));
    else
        lp_max(j) = 0;
    end
    u_sum = u_sum + lp_max(j);
end

u_step = u_sum/total_points;
clear Lpath

% Calculate number of points for each load path
global n_points_lp
n_points_lp = ceil(lp_max/u_step);

clear u_fit

index = 0;
for j=1:nl

    Lpath = ['Path' num2str(j)];

    %Call the 'DET' function to calculate the dissipated
energy.
    [DE_total_paths,u_fit] =
DE_total_eff(Lpath,n_points_lp(j));

    if ~isempty(DE_total_paths)
        index = index+1;
```

```
        for i=2:length(DE_total_paths)
            epsilon_global{index,i-1} =
(norm_epsilon_global(:,:,1)*u_fit(i,1)...
                +norm_epsilon_global(:,:,2)*u_fit(i,2)...

+permute(interp1(delta_u(3:end),permute(norm_epsilon_global
(:,:,3:end),[3 1 2]),u_fit(i,3)),[2 3 1]));

            %Build a cell of dissipated energy values.
            DET{index,i-1} = DE_total_paths(i,1);
            %{
            Assume that all the finite element models have
the same mesh
            and therefore the same element volume.
            %}
            VE{index,i-1} = vol_elem_all(:,1,1);
            u_out{index,i-1} = u_fit(i,:);
        end
        clear DE_total_paths
        clear u_fit
    end
end
```

```
function[DE_total_paths,u_final] =
DE_total_eff(Lpath,n_data)


%If 'plots = 1' then the figures of this function will be
created.
%If 'plots = 0' then none of the figures will be created
plots = 0;

%If 'trunc = 1' then the data will be truncated at the
trunc_lim value.
%If 'trunc = 0' then none of the data will be truncated.
trunc = 0;
trunc_lim = 5;

%If 'file_write = 1' then a .csv file will be created for
the load disp data.
%If 'file_write = 0' then .csv files will not be created.
file_write = 0;

%Define the test path as you want it to appear on the
graphs.
path_data = struct('Path1',[0.7071  0  -0.7071]...
            ,'Path2',  [1.0000  0  0]...
            ,'Path3',  [0.7071  0  0.7071]...
            ,'Path4',  [0.6533  0.2706  -0.7071]...
            ,'Path5',  [0.9239  0.3827  0.7071]...
            ,'Path6',  [0.6533  0.2706  0.7071]...
            ,'Path7',  [0.5000  0.5000  -0.7071]...
            ,'Path8',  [0.7071  0.7071    0]...
            ,'Path9',  [0.5000  0.5000  0.7071]...
            ,'Path10', [0.2706  0.6533  -0.7071]...
            ,'Path11', [0.3827  0.9239    0]...
            ,'Path12', [0.2706  0.6533  0.7071]...
            ,'Path13', [0  0.7071  -0.7071]...
            ,'Path14', [0  1.0000  0]...
            ,'Path15', [0  0.7071  0.7071]...
            ,'Path16', [0  0  1.0000]...
            ,'Path17', [0  0  -1.0000]);
 %Include all load paths.
 global np
 %      1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17
 %np = [10 12 13 10 16 14 13 15 14 10 10 10 11 12 14 16
16];
 %Include only 4 load paths.
 np = [0 11 0 0 0 0 0 15 0 0 10 0 0 12 0 0 0];
 %Include single load path.
 %np = [0 0 0 0 0 0 0 0 0 0 0 0 0 12 0 0 0];
 %np = [0 11 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];

 for i = 1:length(np)
```

```
     path_info = ['Path' num2str(i)];
     np_data.(path_info) = np(i);
 end

%Define the number of load points for the current load
path.
n_points = np_data.(Lpath);
%n_points = 'all';

%Define the correct load path components.
load_path = path_data.(Lpath);

%{
Define the name of the .txt file that contains the
experimental data for
each load path.
%}

test_sets = 'ABC';
j = 0;
points = NaN;
F_mean = 0;
u_mean = 0;

root = 'C:\Documents and Settings\Jay Smith\My
Documents\Classes\Thesis\James & Jay\Dissipated Energy
Density Calculation\IPL Data\';
for i=1:3
    file = [root 'Data Tests ' test_sets(i) '\test' ' '
Lpath ' ' test_sets(i) '.txt'];
    if exist(file)
        j = j+1;
        [big_step_data] = IPL_aquire(file);

        %{
        If the variable 'n_points' is equal to a character,
the program will keep
        all the experimental data points.  If 'n_points' is
defined numerically
        the program will keep only that number of data
points.
        %}

        if n_points=='all'
            points = min(points,size(big_step_data,1));
        else
            points = n_points;
        end

        %Select only the desired points for subsequent
manipulations.
```

```
        big_step_data_plot = big_step_data(1:points,:);

        %Force data for test A.
        F = [big_step_data_plot(:,10)
big_step_data_plot(:,11) big_step_data_plot(:,12)];

        %Displacement data for test A.
        u = [big_step_data_plot(:,13)
big_step_data_plot(:,14) big_step_data_plot(:,15)];

        if size(F,1)<2
            u_final = [];
            DE_total_paths = [];
            return
        else
            %Calibrate the data so that the first point is
at the origin.
            F_cal = F - ones(size(F(:,1)))*[F(1,1) F(1,2)
F(1,3)];

            %{
            Define the moment arm. i.e. (the distance from
the notch centered
            coordinates to the grip face measured in
inches.)
            %}
            arm = 0.625;

            %Calibrate the data so that the first point is
at the origin.
            u_cal = u - ones(size(u(:,1)))*[u(1,1) u(1,2)
u(1,3)];
            u_cal(:,3) = 2*pi/360*arm*u_cal(:,3);

            if i==1
                F_mean = F_cal;
                u_mean = u_cal;
            else
                F_mean = F_mean(1:points,:) +
F_cal(1:points,:);
                u_mean = u_mean(1:points,:) +
u_cal(1:points,:);
            end
        end
    end
end

F_mean = F_mean/j;
u_mean = u_mean/j;
```

```
%Set the displacement values to zero if the displacement
component is zero.
if find(load_path==0)
    u_mean(:,find(load_path==0)) = 0;
end

%Create a piecewise linear fit for the data.
[u_fit,F_fit] = data_fit(u_mean,F_mean,n_data);

% %Set the displacement values to zero if the displacement
component is zero.
% if find(load_path==0)
%     u_fit(:,find(load_path==0)) = 0;
% end

%Create an initial secant modulus.
[u_scnt,F_scnt,stiff_scnt,u_linear,f_linear] =
secant_fit(u_fit,F_fit);


if file_write == 1
    dlmwrite(['M:\James & Jay\Nonlinear MP\ANSYS\test\'
'disp' Lpath '.csv'], u_scnt, 'delimiter', ',','precision',
'%+5.4E', 'newline', 'pc')
    dlmwrite(['M:\James & Jay\Nonlinear MP\ANSYS\test\'
'load' Lpath '.csv'], F_scnt, 'delimiter', ',','precision',
'%+5.4E', 'newline', 'pc')
end

%{
Calculate the dissipated energy for each of the mean
experimental data
points.
%}
[dissE_mean] = diss_E_calc(u_mean,F_mean);

%{
Calculate the dissipated energy for each of the mean
experimental data
points.
%}
[dissE_scnt] = diss_E_calc(u_scnt,F_scnt);

%{
Calculate the total dissipated energy in the coupon from
the sum of the
dissipated energies for each displacement component.
%}
%Find NaN's probably only for load path 5.
kdx = isnan(dissE_scnt);
dissE_scnt(find(kdx)) = 0;
```

```
DE_total_cut = sum(dissE_scnt,2);

%Check that the dissipated values are positive etc.
[u_final,F_final,DE_total_paths] =
TDE_check_eff(u_scnt,F_scnt,DE_total_cut);

%!! Modified to accomodate Ritter's functions
if ~n_data
    u_final = u_mean;
    DE_total_paths = dissE_mean;
end

if trunc
    max_idx = find(DE_total_paths>trunc_lim,1)-1;
    u_trunc = u_final(1:max_idx,:);
    F_trunc = F_final(1:max_idx,:);
    %Create a piecewise linear fit for the data.
    [u_trunc,F_trunc] = data_fit(u_trunc,F_trunc,n_data);
    %{
    Calculate the dissipated energy for each of the mean
experimental data
    points.
    %}
    [dissE_scnt] = diss_E_calc(u_trunc,F_trunc);

    %{
    Calculate the total dissipated energy in the coupon
from the sum of the
    dissipated energies for each displacement component.
    %}
    %Find NaN's probably only for load path 5.
    kdx = isnan(dissE_scnt);
    dissE_scnt(find(kdx)) = 0;

    DE_total_cut = sum(dissE_scnt,2);

    %Check that the dissipated values are positive etc.
    [u_final,F_final,DE_total_paths] =
TDE_check_eff(u_trunc,F_trunc,DE_total_cut);
end

%Correct for linear compliance in the IPL.
machine_comp

%Convert the load path components to characters.
temp_2 = sprintf('[%.4g]',load_path);

%Create the first line of the figure title.
heading1 = ['                                ' 'uxx' '  '
'uyy' '  ' 'uxy'];
```

```
%Create the second line of the figure title.
heading2 = ['Secant/Raw/Fit ' ' ' Lpath '  ' temp_2];

%{
Create the total path title by vertical catenation of
previously defined
lines.
%}
sub_heading = strvcat(heading1,heading2);

%Switch for plotting 'on' or 'off' defined above.
if plots == 1
    global Hf
    if ischar(Hf)==1
        %Define the first figure if necessary.
        Hf = 1;
    else
        %Get the current figure.
        Hf = gcf;
        %Assign the new current figure.
        Hf = Hf+1;
    end

    %Define the maximum displacement value for plotting.
    u_max = max([max(abs(u_scnt)) max(abs(u_mean))]);

    %Plot the mean and secant load-displacement curves.
    figure(Hf)
    subplot(3,1,1)
    plot(u_mean(:,1),F_mean(:,1),'--bp','LineWidth',1,...
                    'MarkerEdgeColor','k',...
                    'MarkerFaceColor',[0.11 0.56 0.39],...
                    'MarkerSize',6)

    t =
text(u_mean(:,1),F_mean(:,1),num2str((1:size(u_mean,1))'));

set(t,'VerticalAlignment','bottom','FontWeight','bold',
'FontSize',...
            8);
    hold on
    plot(u_final(:,1),F_final(:,1),'-r','LineWidth',1,...
                    'MarkerEdgeColor','k',...
                    'MarkerFaceColor',[0.11 0.56 0.39],...
                    'MarkerSize',6)
    if load_path(1) ~= 0
        plot(u_scnt_alt(:,1),F_final(:,1),'-
g','LineWidth',1,...
                        'MarkerEdgeColor','k',...
                        'MarkerFaceColor',[0.11 0.56
0.39],...
```

```
                        'MarkerSize',6)
    end
    hold off

    title(sub_heading,'BackgroundColor',[0.7 0.9 0.7],...
                    'FontSize',16)
    xlabel('uxx (in)')
    ylabel('Fxx (lbf)')
    box off, grid on
    xlim([-u_max,u_max])

    subplot(3,1,2)
    plot(u_mean(:,2),F_mean(:,2),'--bp','LineWidth',1,...
                    'MarkerEdgeColor','k',...
                    'MarkerFaceColor',[0.11 0.56 0.39],...
                    'MarkerSize',6)
    t =
text(u_mean(:,2),F_mean(:,2),num2str((1:size(u_mean,1))'));

set(t,'VerticalAlignment','bottom','FontWeight','bold',
'FontSize',...
             8);
    hold on
    plot(u_final(:,2),F_final(:,2),'-r','LineWidth',1,...
                    'MarkerEdgeColor','k',...
                    'MarkerFaceColor',[0.11 0.56 0.39],...
                    'MarkerSize',6)
    if load_path(2) ~= 0
        plot(u_scnt_alt(:,2),F_final(:,2),'-
g','LineWidth',1,...
                        'MarkerEdgeColor','k',...
                        'MarkerFaceColor',[0.11 0.56
0.39],...
                        'MarkerSize',6)
    end
    hold off
    xlabel('uyy (in)')
    ylabel('Fyy (lbf)')
    box off, grid on
    xlim([-u_max,u_max])

    subplot(3,1,3)
    plot(u_mean(:,3),F_mean(:,3),'--bp','LineWidth',1,...
                    'MarkerEdgeColor','k',...
                    'MarkerFaceColor',[0.11 0.56 0.39],...
                    'MarkerSize',6)
    t =
text(u_mean(:,3),F_mean(:,3),num2str((1:size(u_mean,1))'));

set(t,'VerticalAlignment','bottom','FontWeight','bold',
'FontSize',...
```

```
            8);
    hold on
    plot(u_final(:,3),F_final(:,3),'-r','LineWidth',1,...
                    'MarkerEdgeColor','k',...
                    'MarkerFaceColor',[0.11 0.56 0.39],...
                    'MarkerSize',6)
    if load_path(3) ~= 0
        plot(u_scnt_alt(:,3),F_final(:,3),'-
g','LineWidth',1,...
                        'MarkerEdgeColor','k',...
                        'MarkerFaceColor',[0.11 0.56
0.39],...
                        'MarkerSize',6)
    end
    hold off
    xlabel('uxy (in)')
    ylabel('Fxy (lbf-in)')
    box off, grid on
    xlim([-u_max,u_max])

    %Create the first line of the figure title.
    heading1 = ['                                        '
'uxx' '  ' 'uyy' '  ' 'uxy'];
    %Create the second line of the figure title.
    heading2 = ['Dissipated Energy' ' ' Lpath '  ' temp_2];
    %Create the third line of the figure title.
    heading3 = ['Secant/Fit'];

    %Create the second line of the figure title.
    sub_heading = strvcat(heading1,heading2,heading3);


    %Plot the dissipated energy curves.
    figure(Hf+1)
    if size(dissE_mean(:,1))~=[0 0]
        subplot(3,1,1)
        plot(u_mean(:,1),dissE_mean(:,1),'--
bp','LineWidth',1,...
                    'MarkerEdgeColor','k',...
                    'MarkerFaceColor',[0.11 0.56 0.39],...
                    'MarkerSize',6)

        t =
text(u_mean(:,1),dissE_mean(:,1),num2str((1:size(u_mean,1))
'));

set(t,'VerticalAlignment','bottom','FontWeight','bold', ...
                'FontSize',8)
    end
    hold on
    if size(dissE_scnt(:,1))~=[0 0]
```

```
        plot(u_final(:,1),dissE_scnt(:,1),'-
r','LineWidth',1,...
                    'MarkerEdgeColor','k',...
                    'MarkerFaceColor',[0.11 0.56 0.39],...
                    'MarkerSize',6)
        hold off
        xlabel('uxx (in)')
        ylabel('Phi_x_x (lbf-in)')
        box off, grid on
        xlim([-u_max,u_max])
    end
    title(sub_heading,'BackgroundColor',[0.7 0.9 0.7],...
                        'FontSize',16)
    if size(dissE_mean(:,2))~=[0 0]
        subplot(3,1,2)
        plot(u_mean(:,2),dissE_mean(:,2),'--
bp','LineWidth',1,...
                        'MarkerEdgeColor','k',...
                        'MarkerFaceColor',[0.11 0.56
0.39],...
                        'MarkerSize',6)
        t =
text(u_mean(:,2),dissE_mean(:,2),num2str((1:size(u_mean,1))
'));

set(t,'VerticalAlignment','bottom','FontWeight','bold',...
                'FontSize',8);
    end
    hold on
    if size(dissE_scnt(:,2))~=[0 0]
        plot(u_final(:,2),dissE_scnt(:,2),'-
r','LineWidth',1,...
                    'MarkerEdgeColor','k',...
                    'MarkerFaceColor',[0.11 0.56 0.39],...
                    'MarkerSize',6)
        hold off
        xlabel('uyy (in)')
        ylabel('Phi_y_y (lbf-in)')
        box off, grid on
        xlim([-u_max,u_max])
    end
    if size(dissE_mean(:,3))~=[0 0]
        subplot(3,1,3)
        plot(u_mean(:,3),dissE_mean(:,3),'--
bp','LineWidth',1,...
                        'MarkerEdgeColor','k',...
                        'MarkerFaceColor',[0.11 0.56
0.39],...
                        'MarkerSize',6)
```

```
        t =
text(u_mean(:,3),dissE_mean(:,3),num2str((1:size(u_mean,1))
'));

set(t,'VerticalAlignment','bottom','FontWeight','bold',...
                'FontSize',8);
    end
    hold on
    if size(dissE_scnt(:,3))~=[0 0]
        plot(u_final(:,3),dissE_scnt(:,3),'-
r','LineWidth',1,...
                    'MarkerEdgeColor','k',...
                    'MarkerFaceColor',[0.11 0.56 0.39],...
                    'MarkerSize',6)
        hold off
        xlabel('uxy (in)')
        ylabel('Phi_x_y (lbf-in)')
        box off, grid on
        xlim([-u_max,u_max])
    end
    figure(Hf+2)

plot([1:length(DE_total_cut)],DE_total_cut,[1:length(DE_tot
al_paths)],DE_total_paths,'r')
end
```

```
function[epsilon_global,DET,VE,n_elem,u_out]=ANSYS_test_ixy
(nl,total_points)
%{
This function takes all the strain values from the three
FEA models and
manipolates it in order to get strain values for all the
load points and
all the load paths.
%}

%{
If 'plots = 1' then the figures of this function will be
created.
If 'plots = 0' then none of the figures will be created
%}
plots = 0;

%{
Define the distance from the origin of the coupon
coordinates to the top
grip face.
%}
arm = 0.625;         %(in)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%
%Define the displacement value that the FEA model was run
at.
model_uxx = 0.00993;        %negative in FEA
coordinates(in)
                            %positive in IPL coordinates

%Load strain file to determine number of elements
global_sizer = dlmread('uxx_strain.csv',',',0,0);
n_elem_uxx = size(global_sizer,1);

%Define the strain values for each displacement step
1:num_load_points.
epsilon_global_uxx = global_sizer(:,1:3);
%epsilon_global_uxx = randn(n_elem_uxx,3);

%Defines the element volumes.
vol_elem_uxx = global_sizer(:,4);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%
%Define the displacement value that the FEA model was run
at.
model_uyy = 0.01;       %(in)

%Load strain file to determine number of elements
```

```
global_sizer = dlmread('uyy_strain.csv',',',0,0);
n_elem_uyy = size(global_sizer,1);

%Define the strain values for each displacement step
1:num_load_points.
epsilon_global_uyy = global_sizer(:,1:3);
%epsilon_global_uyy = randn(n_elem_uyy,3);

%Defines the element volumes.
vol_elem_uyy = global_sizer(:,4);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%
theta_min = -12;
theta_max = 12;
d_theta = 1;

model_uxy = theta_min:d_theta:theta_max;

for theta = model_uxy
    global_sizer = dlmread(['uxy_eff' int2str(theta)
'.csv'],',',0,0);
    epsilon_global_uxy(:,:,theta - theta_min + 1) =
global_sizer(:,1:3);
end
n_elem_uxy = size(global_sizer,1);
vol_elem_uxy = global_sizer(:,4);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%


%Assign number of elements for uniform models
n_elem = n_elem_uxx;

%Build a matrix structure containing all the strain values.
epsilon_global_full =
cat(3,epsilon_global_uxx,epsilon_global_uyy,epsilon_global_
uxy);

%Build a matrix structure containing all the volumes of
each element.
vol_elem_all =
cat(3,vol_elem_uxx,vol_elem_uyy,vol_elem_uxy);

%{
Define a matrix of the FEA model displacement values, and
convert the
rotational displacements to linear displacements using the
variable 'arm'.
%}
```

```
delta_u = [model_uxx  model_uyy  model_uxy*arm*(2*pi/360)];
%{
Create a matrix structure of strain values normalized wrt
FEA model
displacements.
%}
%Column swap bewteen x and y strains.
norm_epsilon_global = epsilon_global_full;
for i=1:2
    norm_epsilon_global(:,:,i) =
epsilon_global_full(:,:,i)/delta_u(i);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%

% Determine range of displacement data for even data
fitting
u_sum = 0;
for j = 1:nl
    Lpath = ['Path' num2str(j)];
    [DE_total_paths,u_fit] = DE_total_eff(Lpath,12);

    if ~isempty(DE_total_paths)
        lp_max(j) = max(max(abs(u_fit)));
    else
        lp_max(j) = 0;
    end
    u_sum = u_sum + lp_max(j);
end

u_step = u_sum/total_points;
clear Lpath

% Calculate number of points for each load path
global n_points_lp
n_points_lp = ceil(lp_max/u_step);

clear u_fit

index = 0;
for j=1:nl

    Lpath = ['Path' num2str(j)];

    %Call the 'DET' function to calculate the dissipated
energy.
    [DE_total_paths,u_fit] =
DE_total_eff(Lpath,n_points_lp(j));

    if ~isempty(DE_total_paths)
        index = index+1;
```

```
        for i=2:length(DE_total_paths)
            epsilon_global{index,i-1} =
(norm_epsilon_global(:,:,1)*u_fit(i,1)...
                +norm_epsilon_global(:,:,2)*u_fit(i,2)...

+permute(interp1(delta_u(3:end),permute(norm_epsilon_global
(:,:,3:end),[3 1 2]),u_fit(i,3)),[2 3 1]));

            %Build a cell of dissipated energy values.
            DET{index,i-1} = DE_total_paths(i,1);
            %{
            Assume that all the finite element models have
the same mesh
            and therefore the same element volume.
            %}
            VE{index,i-1} = vol_elem_all(:,1,1);
            u_out{index,i-1} = u_fit(i,:);
        end
        clear DE_total_paths
        clear u_fit
    end
end
```

```
function[u_scnt,f_scnt,stiff_scnt,u_linear,f_linear] =
secant_fit(u_inpt, f_inpt)

TDE_frac = 0.05;

for j=1:size(u_inpt,2)
    u_mean = u_inpt(:,j);
    f_mean = f_inpt(:,j);

    %Set the proper initial values.
    E_frac = 0;
    E_frac_last = 0;
    index = 1;
    origin = [0 0];
    %This while loop defines the points for which to fit a
secant modulus.
    while E_frac<TDE_frac
        if index == size(u_mean,1)
            break
        else
            index = index + 1;
            dpoint = [u_mean(index,1) f_mean(index,1)];
            dline = [origin; dpoint];
            recov_E = trapz(dline(:,1),dline(:,2));
            total_E =
trapz(u_mean([1:index],1),f_mean([1:index],1));
            diss_E = total_E - recov_E;

            if total_E~=0
                E_frac = diss_E/total_E;
            else
                E_frac = 0;
            end

            if E_frac-E_frac_last>TDE_frac
                index = index-1;
                break
            end

            E_frac_last = E_frac;
        end
    end

    %Create the new points for plotting the secant curve.
    temp_f_secant =
linspace(f_mean(1),f_mean(index),index)';
    temp_u_secant =
linspace(u_mean(1),u_mean(index),index)';

    f_scnt(:,j) = [temp_f_secant;
f_mean(index+1:length(f_mean))];
```

```
    u_scnt(:,j) = [temp_u_secant;
u_mean(index+1:length(u_mean))];

    if u_mean(index)-u_mean(1)
        %Calculate the secant stiffness.
        stiff_scnt{j}= (f_mean(index)-
f_mean(1))/(u_mean(index)-u_mean(1));
    elseif index == length(u_mean)+1
        stiff_scnt{j} = (f_mean(end)-
f_mean(1))/(u_mean(end)-u_mean(1));
    else
        %Assign the secant stiffness a string for divide by
zero.
        stiff_scnt{j} = 'Not Applicable';
    end

    if index == length(u_mean)+1
        u_linear(:,j) = u_mean(end);
        f_linear(:,j) = f_mean(end);
    else
        u_linear(:,j) = u_mean(index);
        f_linear(:,j) = f_mean(index);
    end
end
```

```
function[DE_out]=diss_E_calc(u_inpt,f_inpt)
%Calculate the dissipated energy at each load point.

for j=1:size(u_inpt,2)
    u = u_inpt(:,j);
    f = f_inpt(:,j);
    if size(u,1)<2
        diss_E = [];
    else
        total_E(1) = 0;
        diss_E(1) = 0;
        for i=2:size(u,1)
            total_E(i) = total_E(i-1) + 0.5*(u(i) - u(i-
1))*(f(i) + f(i-1));
            recov_E(i) = 0.5*f(i)*u(i);
            diss_E(i) = total_E(i) - recov_E(i);
        end
    end
    DE_out(:,j) = diss_E';
end
```

```
function[u_good,f_good,TDE_out]=TDE_check_eff(u_in,f_in,TDE
_in)


% Selects only monotonous DEs
mon = TDE_in(1);
g_index = 1;
for i = 2:length(TDE_in)
    if TDE_in(i) >= mon
        mon = TDE_in(i);
        g_index = [g_index i];
    end
end
bad = setdiff(1:g_index(end),g_index);
TDE_out(g_index,1) = TDE_in(g_index,1);
u_good = u_in(1:g_index(end),:);
f_good = f_in(1:g_index(end),:);

for i = 1:length(bad)
    TDE_out(bad(i),1) =
interp1(g_index,TDE_in(g_index),bad(i));
end
```

```
function[big_step_data]=IPL_aquire(file)

if exist(file)
    %Use 'textscan' to bring in the data from the .txt
file.
    fid = fopen(file);
    temp_data_cell = textscan(fid,'%n','headerLines',1');
    fclose(fid);

    %Define the number of column headings.
    n_col = 24;

    %Convert the aquired cell structure to a matrix.
    temp_data = cell2mat(temp_data_cell);

    %Reassembles the data file into the proper dimensions.
    for i=1:size(temp_data,1)/n_col;
        for j=1:n_col
            index = j+n_col*(i-1);
            raw_data(i,j) = temp_data(index);
        end
    end

    %Find the number of total steps in data set A.
    num_steps = raw_data(size(raw_data,1),1);

    %Select only the last small step for each big step.
    for i=0:num_steps
        k = find(raw_data(:,1)==i,1,'last');
        big_step_data(i+1,:) = raw_data(k,:);
        clear k
    end
else
    big_step_data = 0;
end
```

```
function[epsilon_local]=Ply_Pricipal_Strain(epsilon_global,
theta_vec_deg)


%Convert the theta vector from degrees to radians.
tran_theta_vec_rad = theta_vec_deg*2*pi/360;

for j=1:size(epsilon_global,1)
    for i=1:size(epsilon_global,2)
        if ~isempty(epsilon_global{j,i})

            %Permutes the strain structure to be of the
following dimensions:
            tran_epsilon = permute(epsilon_global{j,i},[2
1]);

            %Rotate the strains into ply principal
coordinates.
            for k=1:length(theta_vec_deg)
                temp_theta = tran_theta_vec_rad(k);
                m = cos(temp_theta);
                n = sin(temp_theta);

                temp_T = [m^2   n^2     m*n;
                          n^2   m^2    -m*n;
                        -2*m*n 2*m*n  2*m^2-1];

                temp_epsilon_local(:,:,k) =
temp_T*tran_epsilon;
            end

            epsilon_local{j,i} =
permute(temp_epsilon_local,[2 1 3]);
            clear temp_epsilon_local
        end
    end
end
```

```
function[out]=loc_index_new(x)

global e_max
global e_min
global e_mesh

for i=1:length(x)
    %Compute the local coordinate indices.

    out(i) = find(e_mesh{i} >= x(i),1) - 1;

    if isempty(out(i))
        display('x value is greater than the upper limit of
the solution domain.')
        return
    elseif out(i) == 0
        if x(i) == e_min(i)
            out(i) = 1;
        else
            display('x values is less than the lower limit
of solution domain.')
            return
        end
    end
end
```

```
%solve_full.m

%Zero the nodal value of DED at the origin.
%zero_origin

%Solve the set of equations for the nodal values.
c = A\Y;
c_nonneg = lsqnonneg(A,Y);

%Define the maximum number of iteration in the constrained
linear least
%squares algorithm.
 max_iterations = 100000;
 opt = optimset('maxiter',max_iterations,'TolFun',1E-8);

%c_guess = rand(1,size(A,2));
ub = max_E();
lb = zeros(size(A,2),1);

c_guess = ub/10;

%{
Call on matlabs built in function 'lsqlin' to solve the
constrained linear
least squares problem. x =
lsqlin(C,d,Aeq,beq,A,b,lb,ub,x0,options)
%}
[c_lsqlin,error_norm,residual,exitflag] =
lsqlin(A,Y,[],[],[],[],lb,ub,c_guess,opt);
%error_norm
exitflag
%residual

% %Add the zero the nodal value at the origin in the
solution vector.
% if length(c) == prod(n_nodes)-1
%     c = [c(1:g_nn_origin-1);0;c(g_nn_origin:end)];
%     c_nonneg = [c_nonneg(1:g_nn_origin-
1);0;c_nonneg(g_nn_origin:end)];
%     c_lsqlin = [c_lsqlin(1:g_nn_origin-
1);0;c_lsqlin(g_nn_origin:end)];
% end

%Determine which elements have no data and are therefore
not good.
no_data_elem

%Use finite difference to find nodal values with no
experiment data.
c = finite_diff_bad_implicit(A,c);
```

```
c_nonneg = finite_diff_bad_implicit(A,c_nonneg);
c_lsqlin = finite_diff_bad_implicit(A,c_lsqlin);

%Plot the dissipated energy density.
DED_plot(A,[c c_nonneg c_lsqlin] , 40 , {'C' 'C nonnegative
restriction' 'C linear least squares'} )
```

```
function[c_out] = finite_diff_bad_implicit(A,c_inpt)

global n_nodes
global global_nodes
global a

%Define a function handle to calculate the global node
number.
g_node =@(w1,w2,w3) w1+n_nodes(1)*(w2-
1)+n_nodes(1)*n_nodes(2)*(w3-1);

for i=1:size(A,2)
    n_nonzero{i,1}=length(find(A(:,i)));
end

non_zero_columns = find(cell2mat(n_nonzero));
zero_columns = setdiff([1:size(A,2)],non_zero_columns);


%Find the coordinates of each bad node.
for i=1:length(zero_columns)
    global_nn = zero_columns(i);
    for j = 1:n_nodes(1)
        for k = 1:n_nodes(2)
            for n = 1:n_nodes(3)
                global_nn_guess = g_node(j,k,n);
                if global_nn_guess == global_nn
                    np(i,:) = [j k n];
                end
            end
        end
    end
end

if length(c_inpt) == length(non_zero_columns)
    c_inpt(non_zero_columns) = c_inpt;
end

m = zeros(length(zero_columns));
b = zeros(length(zero_columns),1);
count = 0;
for i = 1:length(zero_columns)

    idx_nearest = nearest_nodes(np(i,:),n_nodes);

    for j = 1:6
        if idx_nearest(j,:)
            global_idx(j) =
g_node(idx_nearest(j,1),idx_nearest(j,2),idx_nearest(j,3));
            dis_btw(j) = a{ceil(j/2)}(np(i,ceil(j/2)) - ((-
1)^j < 0));
```

```
        end
    end
    for j = 1:6
        if idx_nearest(j,:)
            other_dirs = setdiff(1:3,ceil(j/2));
            coeff = (dis_btw(other_dirs(1)*2) +
dis_btw(other_dirs(1)*2-1))*...
                    (dis_btw(other_dirs(2)*2) +
dis_btw(other_dirs(2)*2-1))/...
                    (4*dis_btw(j));
            m(i,i) = m(i,i) + coeff;
            idx = find(global_idx(j) == zero_columns);
            if isempty(idx)
                b(i) = b(i) + coeff*c_inpt(global_idx(j));
            else
                m(i,idx) = m(i,idx) - coeff;
            end
        end
    end
end

cfd = m\b;

c_out = c_inpt;
c_out(zero_columns) = cfd;
```

```
function[max_DE] = max_E()

global e_mesh
global n_nodes

E11 = 3.858E6;
E22 = 1.091E6;
G12 = 5.97E5;
nu12 = 0.322;

delta = 1 - nu12^2*E22/E11;

Stiffness = [E11/delta      nu12*E22/delta 0 ;...
             nu12*E22/delta E22/delta      0 ;...
             0              0              G12];

g_node =@(t1,t2,t3) t1+n_nodes(1)*(t2-
1)+n_nodes(1)*n_nodes(2)*(t3-1);

max_DE = zeros(prod(n_nodes),1);
for i = 1:n_nodes(1)
    for j = 1:n_nodes(2)
        for k = 1:n_nodes(3)
            strain = [e_mesh{1}(i) ; e_mesh{2}(j) ;
e_mesh{3}(k)];
            max_DE(g_node(i,j,k)) =
0.5*strain'*Stiffness*strain;
            if max_DE(g_node(i,j,k)) == 0
                max_DE(g_node(i,j,k)) = eps;
            end
        end
    end
end
```

```
%no_data_elem.m


for i=1:size(A,2)
    n_nonzero{i,1}=length(find(A(:,i)));
end

non_zero_columns = find(cell2mat(n_nonzero));
zero_columns = setdiff([1:size(A,2)],non_zero_columns);

idx = 0;
for i1 = 1:n_nodes(1)-1
    for i2 = 1:n_nodes(2)-1
        for i3 = 1:n_nodes(3)-1

            %Define the local coordinate indices.
            i0 = [i1 i2 i3];

            %Determine n and m in global node numbers.
            q1 = global_nodes{1}(i0(1),1);
            q2 = global_nodes{1}(i0(1),2);
            e1 = global_nodes{2}(i0(2),1);
            e2 = global_nodes{2}(i0(2),2);
            v1 = global_nodes{3}(i0(3),1);
            v2 = global_nodes{3}(i0(3),2);

            g_m = g_node(q2,e2,v1);
            g_n = g_node(q1,e2,v1);
            g_o = g_node(q2,e1,v1);
            g_p = g_node(q1,e1,v1);

            g_q = g_node(q2,e2,v2);
            g_r = g_node(q1,e2,v2);
            g_s = g_node(q2,e1,v2);
            g_t = g_node(q1,e1,v2);

            g_nn = [g_m g_n g_o g_p g_q g_r g_s g_t];

            if length(intersect(g_nn,non_zero_columns)) ~=
length(g_nn)
                idx = idx + 1;
                ws(idx,:) = i0;
                ng_elem(idx,1) =
loc_coord_index(i0(1),i0(2),i0(3));

            end
        end
    end
end
```

```
%check_uyy_eff.m

%Define the solution vector to use.
c_sol = c_lsqlin;

path_idx = 14;

%Determine what 1 direction location load path 14 exists.
global np
[junk lp_used] = find(np);
idx_path = find(lp_used == path_idx);

index = 0;
for i = idx_path
    for j = 1:size(epsilon_local,2)
        if ~isempty(epsilon_local{i,j})
            index = index + 1;
            uyy_data(index) = u_paths{i,j}(2);
            test_epsilon_local{index} = epsilon_local{i,j};
            test_epsilon_global{index} =
epsilon_global{i,j};
            test_VE{index} = VE{i,j};
        end
    end
end

clear uyy_DissEn
mdx = 0;
for m = 1:3:length(uyy_data)
    mdx = mdx + 1;
    Y_single = zeros(1,length(c_sol));

    for i = 1:size(test_epsilon_local{m},1)    %index
element
            for j = 1:size(test_epsilon_local{m},3)
%index ply

                x0 = test_epsilon_local{m}(i,:,j);

                %Compute the local coordinate indices.
                i0 = loc_index_new(x0);


%!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!
                %Calculate the correct local coordinates.
                under = find(epsilon > abs(x0 - e_min));
                over = find(epsilon > abs(x0 - e_max));
                inside = setdiff(1:3,[under over]);
                Xi(under) = -1;
                Xi(over) = 1;
```

```
                for k = inside
                    Xi(k) = (x0(k) -
d{k}(i0(k)))/a{k}(i0(k));
                end;
                if ~isempty(find((abs(Xi) - 1) > epsilon))
                    display('Incorrect local coordinate
system is selected, out of bounds.')
                    return
                end

%!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!

                N = zeros(1,prod(n_nodes));

                %Determine n and m in global node numbers.
                q1 = global_nodes{1}(i0(1),1);
                q2 = global_nodes{1}(i0(1),2);
                e1 = global_nodes{2}(i0(2),1);
                e2 = global_nodes{2}(i0(2),2);
                v1 = global_nodes{3}(i0(3),1);
                v2 = global_nodes{3}(i0(3),2);

                g_m = g_node(q2,e2,v1);
                g_n = g_node(q1,e2,v1);
                g_o = g_node(q2,e1,v1);
                g_p = g_node(q1,e1,v1);

                g_q = g_node(q2,e2,v2);
                g_r = g_node(q1,e2,v2);
                g_s = g_node(q2,e1,v2);
                g_t = g_node(q1,e1,v2);

                N(g_m) = Nm(Xi);
                N(g_n) = Nn(Xi);
                N(g_o) = No(Xi);
                N(g_p) = Np(Xi);

                N(g_q) = Nq(Xi);
                N(g_r) = Nr(Xi);
                N(g_s) = Ns(Xi);
                N(g_t) = Nt(Xi);

                Y_single = Y_single +
N*test_VE{m}(i)*frac_plys(j);

            end % for j
    end % for i

    uyy_DissEn(mdx,:) = [uyy_data(m) Y_single*c_sol*3];
```

```
end %for m

[DE_tp,Fall,uall,u_lim,f_lim] =
FvsD_IPL('Path14',length(uyy_DissEn));

tens_mod = max(f_lim(2)/u_lim(2));
tens_lim = u_lim(2);

f_tens =
fc(uyy_DissEn(:,1),uyy_DissEn(:,2),tens_mod,find(uyy_DissEn
(:,1) >= tens_lim,1)+2);

Y_sol = A*c_sol;
n_points_used(lp_used) = n_points_lp(lp_used)-1;
start_pt = sum(n_points_used(1:(path_idx-1)))+1;
DE_plot = Y_sol(start_pt:(start_pt+length(uyy_data)-1))';

figure(52)
plot(uall(:,2),DE_tp,uyy_DissEn(:,1),uyy_DissEn(:,2),'or',u
yy_data,DE_plot)
title({'Uyy';'Experimental vs. Predicted DE'})
legend('Experimental','Predicted','Predicted... sort of')
xlabel('Displacement (in)')
ylabel('Dissipated Energy (lb-in)')

figure(53)
plot(uall(:,2)*2.54,Fall(:,2)*4.448,uyy_DissEn(:,1)*2.54,f_
tens*4.448,'--r','linewidth',3)
%title({'Uyy';'Experimental vs. Predicted Load'})
legend('Experimental',['Predicted' ' R^2='
num2str(corr2(Fall(:,2),f_tens(1:size(Fall(:,2),1))'))],'Lo
cation','northwest')
xlabel('Displacement (cm)','fontsize',16)
ylabel('Load (N)','fontsize',16)
set(gca,'fontsize',14)

['DE Correlation = '
num2str(corr2(DE_tp,uyy_DissEn(1:size(DE_tp,1),2)))]

['FU correlation = '
num2str(corr2(Fall(:,2),f_tens(1:size(Fall(:,2),1))'))]
```

```
function[DE_total_paths,F_scnt,u_scnt,u_linear,f_linear] =
FvsD_IPL(Lpath,n_data)


%If 'plots = 1' then the figures of this function will be
created.
%If 'plots = 0' then none of the figures will be created
plots = 1;

%If 'file_write = 1' then a .csv file will be created for
the load disp data.
%If 'file_write = 0' then .csv files will not be created.
file_write = 0;

%Define the test path as you want it to appear on the
graphs.
path_data = struct('Path1',[0.7071  0  -0.7071]...
            ,'Path2',  [1.0000  0  0]...
            ,'Path3',  [0.7071  0  0.7071]...
            ,'Path4',  [0.6533  0.2706  -0.7071]...
            ,'Path5',  [0.9239  0.3827  0.7071]...
            ,'Path6',  [0.6533  0.2706  0.7071]...
            ,'Path7',  [0.5000  0.5000  -0.7071]...
            ,'Path8',  [0.7071  0.7071    0]...
            ,'Path9',  [0.5000  0.5000  0.7071]...
            ,'Path10', [0.2706  0.6533  -0.7071]...
            ,'Path11', [0.3827  0.9239    0]...
            ,'Path12', [0.2706  0.6533  0.7071]...
            ,'Path13', [0  0.7071  -0.7071]...
            ,'Path14', [0  1.0000  0]...
            ,'Path15', [0  0.7071  0.7071]...
            ,'Path16', [0  0  1.0000]...
            ,'Path17', [0  0  -1.0000]);

 %Include all load paths.
 global np

 for i = 1:length(np)
     path_info = ['Path' num2str(i)];
     np_data.(path_info) = np(i);
 end

%Define the number of load points for the current load
path.
n_points = np_data.(Lpath);
%n_points = 'all';

%Define the correct load path components.
load_path = path_data.(Lpath);

%{
```

```
Define the name of the .txt file that contains the
experimental data for
each load path.
%}

test_sets = ['A' 'B' 'C'];
j = 0;
points = NaN;
F_mean = 0;
u_mean = 0;

root = 'C:\Documents and Settings\Jay Smith\My
Documents\Classes\Thesis\James & Jay\Dissipated Energy
Density Calculation\IPL Data\';
for i=1:3
    file = [root 'Data Tests ' test_sets(i) '\test' ' '
Lpath ' ' test_sets(i) '.txt'];
    if exist(file)
        j = j+1;
        [big_step_data] = IPL_aquire(file);

        %{
        If the variable 'n_points' is equal to a character,
the program will keep
        all the experimental data points.  If 'n_points' is
defined numerically
        the program will keep only that number of data
points.
        %}

        if n_points=='all'
            points = min(points,size(big_step_data,1));
        else
            points = n_points;
        end

        %Select only the desired points for subsequent
manipulations.
        big_step_data_plot = big_step_data(1:points,:);


        %Force data for test A.
        F = [big_step_data_plot(:,10)
big_step_data_plot(:,11) big_step_data_plot(:,12)];

        %Displacement data for test A.
        u = [big_step_data_plot(:,13)
big_step_data_plot(:,14) big_step_data_plot(:,15)];

        if size(F,1)<2
            u_final = [];
```

```
            DE_total_paths = [];
            return
        else
            %Calibrate the data so that the first point is
at the origin.
            F_cal = F - ones(size(F(:,1)))*[F(1,1) F(1,2)
F(1,3)];

            %{
            Define the moment arm. i.e. (the distance from
the notch centered
            coordinates to the grip face measured in
inches.)
            %}
            arm = 0.625;

            %Calibrate the data so that the first point is
at the origin.
            u_cal = u - ones(size(u(:,1)))*[u(1,1) u(1,2)
u(1,3)];
            u_cal(:,3) = 2*pi/360*arm*u_cal(:,3);

    %         figure(j+12)
    %         plot(u,F)
    %
    %         figure(j+6)
    %         plot(u_cal,F_cal)

            if i==1
                F_mean = F_cal;
                u_mean = u_cal;
            else
                F_mean = F_mean(1:points,:) +
F_cal(1:points,:);
                u_mean = u_mean(1:points,:) +
u_cal(1:points,:);
            end
        end
    end
end

F_mean = F_mean/j;
u_mean = u_mean/j;

%Create a piecewise linear fit for the data.
[u_fit,F_fit] = data_fit(u_mean,F_mean,n_data);

%Set the displacement values to zero if the displacement
component is zero.
if find(load_path==0)
    u_fit(:,find(load_path==0)) = 0;
```

```
end

%Create an initial secant modulus.
[u_scnt,F_scnt,stiff_scnt,u_linear,f_linear] =
secant_fit(u_fit,F_fit);

%***  Scale IPL data to account for machine compliance ***
fea_linear(1,:) = csvread('fea_linear_0.01_0_0.csv');
fea_linear(2,:) = csvread('fea_linear_0_0.01_0.csv');
fea_linear(3,:) = csvread('fea_linear_0_0_0.01.csv');

u_scnt_alt = u_scnt;

eq_fea = load_path*fea_linear;
for j = find(load_path)
    fea_u_diss = f_linear(j)*eq_fea(2*j-1)/eq_fea(2*j);
    m = (u_linear(j) - fea_u_diss)/f_linear(j);
    u_scnt_alt(:,j) = u_scnt(:,j) - m*F_scnt(:,j);
end
%***  ---------------------------------------------  ***

u_scnt = u_scnt_alt;

%{
Calculate the dissipated energy for each of the mean
experimental data
points.
%}
[dissE_mean] = diss_E_calc(u_mean,F_mean);

%{
Calculate the dissipated energy for each of the mean
experimental data
points.
%}
[dissE_scnt] = diss_E_calc(u_scnt,F_scnt);

%{
Calculate the total dissipated energy in the coupon from
the sum of the
dissipated energies for each displacement component.
%}
%Find NaN's probably only for load path 5.
kdx = isnan(dissE_scnt);
dissE_scnt(find(kdx)) = 0;

DE_total_cut = sum(dissE_scnt,2);

%Check that the dissipated values are positive etc.
[u_final,F_final,DE_total_paths] =
TDE_check_eff(u_scnt,F_scnt,DE_total_cut);
```

```
function[F] = fc(u,DE,k,start)


F(1) = 0;
%F(2) = k*(u(2)-u(1));
F(2:start) = k*u(2:start);
A_total(start-1) = 0.5*k*u(start-1)^2;

temp = 0;
A_total(1) = 0;

for i=start:length(u)-1

    temp = 0.5*(F(i)+F(i-1))*(u(i)-u(i-1));
    A_total(i) = A_total(i-1) + temp;
    F(i+1) = (2/u(i))*(A_total(i)-(DE(i+1)-
DE(start))+0.5*F(i)*(u(i+1)-u(i)));

end
```

```
function[ ] = density( epsilon_local, p, e_mesh)

% if lsqlin solution is plotted, plots on top of figure 3
overlay = false;

clock

% Displays p random points from set xk
% Works best if rows(X)*cols(X) >> p

% If p = 0 then all points in xk are displayed

for i = 1:3
    n_nodes(i) = length(e_mesh{i});
end

index = 0;
for i=1:size(epsilon_local,1)
    for j=1:size(epsilon_local,2)
        if ~isempty(epsilon_local{i,j})
            index = index +
size(epsilon_local{i,j},1)*size(epsilon_local{i,j},3);
        end
    end
end
xk = zeros(index,3);

index = 0;
for i=1:size(epsilon_local,1)
    for j=1:size(epsilon_local,2)
        if ~isempty(epsilon_local{i,j})
            count =
size(epsilon_local{i,j},1)*size(epsilon_local{i,j},3);
            xk(index+1:index+count,:) =
reshape(permute(epsilon_local{i,j},[1 3 2]),[count 3]);
            index = index + count;
        end
    end
end


h = figure(4 - overlay);
hold on
%view([0 90])
view(3)
axis equal
xlabel('\epsilon_1_1','fontsize',16);
ylabel('\epsilon_2_2','fontsize',16);
zlabel('\epsilon_1_2','fontsize',16);
box on
grid on
```

```
xlim(e_mesh{1}([1 end]));
ylim(e_mesh{2}([1 end]));
zlim(e_mesh{3}([1 end]));

a = get(h,'CurrentAxes');

set(a,'XTick',e_mesh{1});
set(a,'YTick',e_mesh{2});
set(a,'ZTick',e_mesh{3});
set(gca,'fontsize',14)


if p > 0
    index = ceil(rand(1,p)*index);
else
    index = 1:index;
end

for i = index
    el = loc_index_new(xk(i,:));

    color = el./(n_nodes - 1);

    b =
plot3(xk(i,1),xk(i,2),xk(i,3),'o','MarkerFaceColor',color,'
MarkerEdgeColor','none');
end
```

APPENDIX: C

ANSYS CODE

ANSYS FILE LIST

Regress.txt
Geom.txt
Bound_cond.txt
Mesh.txt
Elem.txt
Pre_info.txt
Dim.txt
Bilin.txt
Converge.txt
Read.txt
Post_info.txt
File_save.txt

```
FILENAME regress.txt
/Title, Composite IPL Coupon, Non-linear Material
Properties

/prep7

/nerr,,,-1
/uis,msgpop,4

pi = 3.1415962

!!!!!!! User defined parameters
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

num_plys = 6

*dim,theta,array,num_plys
theta(1) = 90,45,-45,-45,45,90     !deg.

t_layer = 0.0101            !ply thickness (in)

E1 = 3.858E6        !(psi)
E2 = 1.091E6        !(psi)
E3 = 1.648E6        !(psi)
G12 = 5.97E5        !(psi)
G13 = 5.133E5       !(psi)
G23 = 4.79E5        !(psi)
pr12 = 0.322
pr13 = 0
pr23 = 0

!Define the model displacement fractions.
!xx => 1; yy => 2; xy=>3

!Define the maximum boundary displacements for each
component.(IPL coordinates)
*dim,ipl_disp,array,3
ipl_disp(1) = 0.035 !Total xx displacement in inches.
ipl_disp(2) = 0.035 !Total yy displacement in inches.
ipl_disp(3) = 0          !Total xy displacement in inches.

Lpath = 8

!Define the number of displacement and max convergence
steps.
disp_steps = 15
conv_steps = 8

!Define fitting tolerances
fit_steps = 10
fit_tol = 0.01
```

```
!Set parameters for displacement convergence.
conv_tol = 0.0001

!Define the initial constitutive parameters.
*dim,epsilon_c_initial,array,3
epsilon_c_initial(1) = 0.005,0.01,0.01
*dim,epsilon_u_initial,array,3
epsilon_u_initial(1) = 0.1,0.1,0.1

!Define limiting secant modulus DE fraction
sm_frac = 0.05

e_min = 0.003
e_max = 0.009
e_steps = 10

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!

/com *** Model parameters ***

l = 1.25       !coupon test length (in)
w = 1.03006          !coupon width (in)
D = 0.2295           !Notch diameter (in)
n_length = 0.4579   !Notch length (in)

!Define the distance between the coupon notch center and
the top grip.
arm = l/2

!Correct for ipl coordinates
*dim,max_disp,array,3
max_disp(1) = -ipl_disp(1),ipl_disp(2),ipl_disp(3)

!Apply no correction in directions with no displacement
*if,max_disp(1),eq,0,then
     cor_c_x = 0
     cor_u_x = 0
*endif
*if,max_disp(2),eq,0,then
     cor_c_y = 0
     cor_u_y = 0
*endif
*if,max_disp(3),eq,0,then
     cor_c_r = 0
     cor_u_r = 0
*endif

!Create the model geometry.
/input,geom,txt
```

```
!/input,geom_rect,txt

!Create the model mesh.
/input,mesh,txt
!/input,mesh_rect,txt

!Define info extraction parameters in the preprocessor.
/input,pre_info,txt

!Determine the number of elements in the coupon excluding
the tabs.
cmsel,s,coupon_area,area
nsla,s,1
esla,s
*get,total_coupon_elem,elem,0,count
*get,max_coupon_elem,elem,0,num,max

!Define the tab model scale factor.
model_frac = (l+2*ext_length)/l

!Set the target displacment values to the desired
displacement values.
*dim,disp,array,3

!Scale the target displacement by the model scale factor
for model with tabs.
*dim,total_disp,array,3
*voper,total_disp,max_disp,mult,model_frac

!Load experimental data
/input,read,txt

!Dimension the parameters within the do loop.
/input,dim,txt

*dim,residuals,array,e_steps

!fit_count = 8

*do,fit_count,1,e_steps

     !this is the parameter being varied
     epsilon_c_initial(1) = (fit_count-1)*(e_max -
e_min)/(e_steps - 1) + e_min

     !Initialize element properties.
     /input,elem,txt

     *do,wdx,1,3
          *do,jdx,1,total_coupon_elem
               *do,idx,1,num_plys
```

```
                    ref_num = idx + num_plys*(jdx-1)
                    epsilon_c(ref_num,wdx) =
epsilon_c_initial(wdx)
                    epsilon_u(ref_num,wdx) =
epsilon_u_initial(wdx)
                    E_current(ref_num,wdx) = E_initial(wdx)
               *enddo
          *enddo
     *enddo


     !Determine load-displacement curve for current
constitutives
     *do,kdx,1,disp_steps

          !Define the load fraction.
          disp_frac = kdx/disp_steps

          !Define the desired displacement for the current
step
          *voper,target_disp,max_disp,mult,disp_frac
          *if,kdx,eq,1,then
               *voper,disp,target_disp,mult,model_frac
          *else
               *voper,disp,disp,mult,kdx/(kdx-1)
          *endif


          !Iterate until the desired displacement is equal
to the true displacement.
          /input,converge,txt

          !Check Load step for failure.
          /input,bilin,txt

          data_x(kdx,1) = -disp_x_true
          data_x(kdx,2) = -f_x_true
          data_y(kdx,1) = disp_y_true
          data_y(kdx,2) = f_y_true
          data_r(kdx,1) = disp_arc_true
          data_r(kdx,2) = moment

          data_x_int(kdx,0) = -disp_x_true
          data_x_int(kdx,1) = -f_x_true
          data_y_int(kdx,0) = disp_y_true
          data_y_int(kdx,1) = f_y_true
          data_r_int(kdx,0) = disp_arc_true
          data_r_int(kdx,1) = moment

          data_out(kdx,1) = -disp_x_true
          data_out(kdx,2) = -f_x_true
```

```
          data_out(kdx,3) = disp_y_true
          data_out(kdx,4) = f_y_true
          data_out(kdx,5) = disp_arc_true
          data_out(kdx,6) = moment

     *enddo


     !Calculate residuals between experimental and
calculated load-displacement
     residuals(fit_count) = 0
     *do,ldx,1,exp_data_pts
          *vitrp,int_fx,data_x_int,x_table(ldx,1)
          *vitrp,int_fy,data_y_int,y_table(ldx,1)
          *vitrp,int_fr,data_r_int,r_table(ldx,1)
          res_x(ldx,1) = int_fx(1) - x_table(ldx,2)
          res_y(ldx,1) = int_fy(1) - y_table(ldx,2)
          res_r(ldx,1) = int_fr(1) - r_table(ldx,2)
          residuals(fit_count) = residuals(fit_count) +
res_x(ldx,1)**2 + res_y(ldx,1)**2 + res_r(ldx,1)**2
     *enddo


     !Save the output data to a .csv file.
     par0(1,1) = strcat(par3(1,1),chrval(fit_count))
     /input,file_save,txt

*enddo
```

```
FILENAME geom.txt
/com *** Define the geometrical keypoints ***

!  #    x                    y
k, 1,   n_length-w,          -l/2
k, 2,   n_length-w,           l/2
k, 3,   n_length,            -l/2
k, 4,   n_length,             l/2
k, 5,   n_length,             D/2
k, 6,   n_length,            -D/2
k, 7,   D/2,                   0
k, 8,   n_length,              0
k, 9,   D/2,                  l/2
k, 10,  D/2,                 -l/2
k, 11,  n_length-w,      0
k, 12,  n_length-w/2,         0
k, 13,  n_length-w/2,         l/2

!Keypoints to make up additional extension tabs.

ext_length = 0.25*l

k, 19,  n_length-w, l/2+ext_length
k, 20,  n_length-w, -(l/2+ext_length)
k, 21,  n_length,   l/2+ext_length
k, 22,  n_length,   -(l/2+ext_length)
k, 23,   D/2,           l/2+ext_length
k, 24,   D/2,       -(l/2+ext_length)


/com *** Define coordinate systems ***
cskp,11,1,7,8,9          !Notch centered cylindrical
cskp,12,0,7,8,9          !Notch centered cartesian
cskp,13,0,12,8,13   !Coupon centered cartesian
csys,0              !Notch tip cartesian

circle,7,D/2,,9,180,4


/com *** Define Lines ***
l,10,18
l,1,17
l,11,16
l,2,15
l,9,14
l,4,5
l,3,6
l,3,10
l,10,1
l,1,11
l,11,2
```

```
l,2,9
l,9,4
l,4,5
l,5,14
l,6,18

!Lines to make up additional extension tabs.
l,2,19
l,9,23
l,4,21
l,19,23
l,23,21

l,3,22
l,10,24
l,1,20
l,20,24
l,24,22

/com *** Define Areas ***

al,7,2,8,15
al,8,1,9,16
al,9,18,10,17
al,6,3,7,14
al,5,4,6,13
al,11,19,5,12

!Areas to make up additional extension tabs.
al,16,21,23,20
al,17,22,24,21
al,13,27,28,26
al,12,26,29,25

/com *** Create Boundary Line Components ***

lsel,s,line,,13
lsel,a,line,,12
cm,bottom_boundary,line
alls

lsel,s,line,,16
lsel,a,line,,17
cm,top_boundary,line
alls

lsel,s,line,,28
lsel,a,line,,29
cm,tab_bottom_boundary,line
alls
```

```
lsel,s,line,,23
lsel,a,line,,24
cm,tab_top_boundary,line
alls
```

```
FILENAME bound_cond.txt
/com *** Apply the boundary conditions ***

cmsel,s,tab_bottom_boundary,line
dl,all,,ux,0
dl,all,,uy,0
dk,20,uz,0
alls

!Define the length of the top boundary in coordinate system
13.

csys,13
*get,x_kp_19,kp,19,loc,x
*get,x_kp_21,kp,21,loc,x
top_length = x_kp_21-x_kp_19

!Define the number of points to represent the boundary
condition.

delta_point = top_length/(n_points-1)

disp_theta = disp(3)/(arm+ext_length)

*do,eval,1,n_points,1
     x_eval = ((eval-1)*delta_point)*(x_kp_21-
x_kp_19)+x_kp_19

     !Define the y boundary values.

     y_bc_table(eval,0) = x_eval
     y_bc_table(eval,1) =
(l/2+ext_length)*(cos(disp_theta)-1)-
x_eval*sin(disp_theta)+disp(2)

     !Define the x boundary values.

     x_bc_table(eval,0) = ((eval-1)*delta_point)*(x_kp_21-
x_kp_19)+x_kp_19
     x_bc_table(eval,1) =
(l/2+ext_length)*sin(disp_theta)+x_eval*(cos(disp_theta)-
1)+disp(1)
*enddo
y_bc_table(0,1) = 1
x_bc_table(0,1) = 1


cmsel,s,tab_top_boundary,line
dl,all,,ux,%x_bc_table%
dl,all,,uy,%y_bc_table%
alls
```

```
FILENAME mesh.txt
/com *** Define the mesh spacing ***

et,1,shell99,,0
keyopt,1,11,0
keyopt,1,10,1
keyopt,1,8,1


*get,length_13,line,13,leng

e_arc_length = pi*D/8

aspect = 1

ratio_8_6 = aspect*(e_arc_length/length_13)
ratio_7 = aspect*(e_arc_length/length_13)
ratio_9_5_10_11 = aspect*(e_arc_length/length_13)

ratio_18_19_12_17 = 1
ratio_14_15 = 1
ratio_13_16 = 1

ediv_radial = 11
ediv_wing = 6
ediv_out = 5

!Radial Lines
lesize,8,,,ediv_radial,ratio_8_6
lesize,6,,,ediv_radial,ratio_8_6
lesize,7,,,ediv_radial,ratio_7
lesize,5,,,ediv_radial,ratio_9_5_10_11
lesize,9,,,ediv_radial,ratio_9_5_10_11
lesize,10,,,ediv_radial,ratio_9_5_10_11
lesize,11,,,ediv_radial,ratio_9_5_10_11

!Vertical Lines
lesize,14,,,ediv_out,ratio_14_15
lesize,15,,,ediv_out,ratio_14_15

!Horizontal Lines
lesize,13,,,ediv_out,ratio_13_16
lesize,16,,,ediv_out,ratio_13_16
lesize,12,,,ediv_wing,ratio_18_19_12_17
lesize,17,,,ediv_wing,ratio_18_19_12_17
lesize,18,,,ediv_wing,ratio_18_19_12_17
lesize,19,,,ediv_wing,ratio_18_19_12_17

!Circumferential Lines
lesize,1,,,ediv_out
lesize,2,,,ediv_out
```

```
lesize,3,,,ediv_out
lesize,4,,,ediv_out

!Tab lines
lesize,20,,,5
lesize,21,,,5
lesize,22,,,5

lesize,25,,,5
lesize,26,,,5
lesize,27,,,5

!lesize,28,,,ediv_out,ratio_13_16
!lesize,29,,,ediv_wing,ratio_18_19_12_17
!lesize,23,,,ediv_out,ratio_13_16
!lesize,24,,,ediv_wing,ratio_18_19_12_17

lesize,28,,,15,ratio_13_16
lesize,29,,,ediv_wing,ratio_18_19_12_17
lesize,23,,,15,ratio_13_16
lesize,24,,,ediv_wing,ratio_18_19_12_17


/com *** Mesh Selected Areas ***

!Mesh the coupon areas.

asel,s,area,,1,6
cm,coupon_area,area

type,ref_num
mat,ref_num
real,ref_num
amesh,all
alls

!Mesh the tab areas.

asel,s,area,,7,10
mshkey,0
amesh,all
alls

!Orient all the element coordinate systems with coordinate
system 12.

csys,12
emodif,all,esys,12
csys,13
```

```
FILENAME elem.txt
/com *** Define the element type ***

*do,jdx,1,total_elem

     esel,s,elem,,jdx

     r,jdx,num_plys,0

     *do,idx,1,num_plys

          ref_num = idx + num_plys*(jdx-1)

          mp,ex,ref_num,E1
          mp,ey,ref_num,E2
          mp,ez,ref_num,E3
          mp,gxy,ref_num,G12
          mp,gyz,ref_num,G23
          mp,gxz,ref_num,G13
          mp,prxy,ref_num,pr12
          mp,pryz,ref_num,pr23
          mp,prxz,ref_num,pr13

          rmodif,jdx,13+(idx-
1)*3,ref_num,theta(idx),t_layer

     *enddo

*enddo

alls
```

```
FILENAME pre_info.txt
/com *** Get model information for data extraction ***


cmsel,s,bottom_boundary,lines
nsll,s,1

!Get the number of nodes along the bottom coupon boundary.
*get,num_nodes,node,,count
*get,b_min,node,,num,min

*dim,b_nnum,array,num_nodes

!Create an array consisting of the node numbers along the
bottom coupon boundary.
b_nnum(1) = b_min
*do,eval,2,num_nodes,1
     b_nnum(eval) = ndnext(b_min)
     b_min = ndnext(b_min)
*enddo
alls



cmsel,s,top_boundary,lines
nsll,s,1

!Get the number of nodes along the top coupon boundary.
*get,t_min,node,,num,min

*dim,t_nnum,array,num_nodes

!Create an array consisting of the node numbers along the
top coupon boundary.
t_nnum(1) = t_min
*do,eval,2,num_nodes,1
     t_nnum(eval) = ndnext(t_min)
     t_min = ndnext(t_min)
*enddo
alls
```

```
FILENAME dim.txt
/com *** Define the dimensions of parameters within loop
***


!OOO! For main !OOO!
*dim,epsilon_loc,array,total_coupon_elem*num_plys,3

!Set the initial change in displacement to zeros.
*dim,delta_disp,array,3
delta_disp(1) = 0,0,0

*dim,E_initial,array,3
E_initial(1) = E1,E2,G12

*dim,epsilon_c,array,total_coupon_elem*num_plys,3
*dim,epsilon_u,array,total_coupon_elem*num_plys,3
*dim,E_current,array,total_coupon_elem*num_plys,3

*dim,target_disp,array,3

*dim,data_x,array,disp_steps,2
*dim,data_y,array,disp_steps,2
*dim,data_r,array,disp_steps,2
*dim,data_out,array,disp_steps,6

*dim,data_x_int,table,disp_steps,1
*dim,data_y_int,table,disp_steps,1
*dim,data_r_int,table,disp_steps,1

*dim,par0,string,128
*dim,par1,string,128
*dim,par2,string,128
*dim,par3,string,128

par1(1,1) = 'fea_data_Path'
par2(1,1) = chrval(Lpath)
par3(1,1) =
strcat('feaFvsD\',strcat(par1(1,1),strcat(par2(1,1),'_')))

*dim,int_fx,array,1,1
*dim,int_fy,array,1,1
*dim,int_fr,array,1,1

*dim,res_x,array,exp_data_pts,1
*dim,res_y,array,exp_data_pts,1
*dim,res_r,array,exp_data_pts,1

*dim,fea_u_secant,array,3
```

```
!OOO! For secant function !OOO!
*dim,u_data,array,max(disp_steps,exp_data_pts),3
*dim,f_data,array,max(disp_steps,exp_data_pts),3

*dim,u_secant,array,3

*dim,d_enter,array,3
*dim,d_TE,array,3
*dim,d_DE,array,3
*dim,DE_prev,array,3


!OOO! For bound_cond !OOO!
n_points = 10
*dim,y_bc_table,table,n_points,1,1,x,,,13
*dim,x_bc_table,table,n_points,1,1,x,,,13


!OOO! For post_info !OOO!
*dim,b_disp_x,array,num_nodes
*dim,b_disp_y,array,num_nodes
*dim,t_disp_x,array,num_nodes
*dim,t_disp_y,array,num_nodes
*dim,t_vec,array,2
*dim,b_vec,array,2
*dim,ms_b,array,2
*dim,center,array,2
*dim,ms_t_rot,array,2
*dim,ms_t_true,array,2
*dim,disp_true,array,2
*dim,rot_true,array,2,2
```

```
FILENAME bilin.txt
! Define minimum modulus as a fraction of original
min_mod = 0.01

*do,idx,1,num_plys
     /post1

     layer,idx
     rsys,lsys

     etable,11_strain,epel,x
     etable,22_strain,epel,y
     etable,12_strain,epel,xy

     /solution
     *do,jdx,1,total_coupon_elem

          ref_num = idx + num_plys*(jdx-1)

          *get,temp_11,etab,1,elem,jdx
          *get,temp_22,etab,2,elem,jdx
          *get,temp_12,etab,3,elem,jdx


          *if,temp_11,abgt,epsilon_c(ref_num,1),then
               *if,temp_11,ablt,epsilon_u(ref_num,1),then
                    E_current(ref_num,1) =
(E_current(ref_num,1)*epsilon_c(ref_num,1)/abs(temp_11))*((
(epsilon_c(ref_num,1)-abs(temp_11))/(epsilon_u(ref_num,1)-
epsilon_c(ref_num,1)))+1)
                    epsilon_c(ref_num,1) = abs(temp_11)
               *else
                    E_current(ref_num,1) = min_mod*E1
               *endif
               *if,E_current(ref_num,1),lt,min_mod*E1,then
                    E_current(ref_num,1) = min_mod*E1
               *endif
               mp,ex,ref_num,E_current(ref_num,1)
          *endif

          *if,temp_22,abgt,epsilon_c(ref_num,2),then
               *if,temp_22,ablt,epsilon_u(ref_num,2),then
                    E_current(ref_num,2) =
(E_current(ref_num,2)*epsilon_c(ref_num,2)/abs(temp_22))*((
(epsilon_c(ref_num,2)-abs(temp_22))/(epsilon_u(ref_num,2)-
epsilon_c(ref_num,2)))+1)
                    epsilon_c(ref_num,2) = abs(temp_22)
               *else
                    E_current(ref_num,2) = min_mod*E2
               *endif
               *if,E_current(ref_num,2),lt,min_mod*E2,then
```

```
                    E_current(ref_num,2) = min_mod*E2
               *endif
               mp,ey,ref_num,E_current(ref_num,2)
          *endif

          *if,temp_12,abgt,epsilon_c(ref_num,3),then
               *if,temp_12,ablt,epsilon_u(ref_num,3),then
                    E_current(ref_num,3) =
(E_current(ref_num,3)*epsilon_c(ref_num,3)/abs(temp_12))*((
(epsilon_c(ref_num,3)-abs(temp_12))/(epsilon_u(ref_num,3)-
epsilon_c(ref_num,3)))+1)
                    epsilon_c(ref_num,3) = abs(temp_12)
               *else
                    E_current(ref_num,3) = min_mod*G12
               *endif
               *if,E_current(ref_num,3),lt,min_mod*G12,then
                    E_current(ref_num,3) = min_mod*G12
               *endif
               mp,gxy,ref_num,E_current(ref_num,3)
          *endif

     *enddo

*enddo
```

```
FILENAME converge.txt
un_conv = 1
count = 0
*dowhile,un_conv
     /prep7

     !Define the change in displacement for a given
iteration.
     *voper,delta_disp,delta_disp,mult,model_frac

     !Defines the current model displacements.
     *voper,disp,disp,sub,delta_disp


     !Define the boundary conditions. Requires disp(1..3).
     /input,bound_cond,txt

     /solution
     solve
     finish
     /post1


     !Find the true boundary displacements.
     /input,post_info,txt

     delta_disp(1) = disp_x_true-target_disp(1)
     delta_disp(2) = disp_y_true-target_disp(2)
     delta_disp(3) = disp_arc_true-target_disp(3)

     count = count+1

     *if,delta_disp(1),ablt,conv_tol,then
          *if,delta_disp(2),ablt,conv_tol,then
               *if,delta_disp(3),ablt,conv_tol,then
                    un_conv = 0
               *endif
          *endif
     *endif

     *if,count,gt,conv_steps,then
          un_conv = 0
     *endif
*enddo
```

```
FILENAME read.txt
/com *** Read Experimental Data ***

!Define the number of data points.
exp_data_pts = 16

*dim,par4,string,128
*dim,par5,string,128
*dim,par6,string,128
*dim,par7,string,128
*dim,par8,string,128
*dim,par9,string,128
*dim,par10,string,128
*dim,par11,string,128
*dim,par12,string,128
*dim,par13,string,128


par4(1,1) = '..\expFvsD\'
par5(1,1) = 'loadPath'
par6(1,1) = 'dispPath'
par7(1,1) = chrval(Lpath)
par8(1,1) = strcat(strcat(par4(1,1),par5(1,1)),par7(1,1))
par9(1,1) = strcat(strcat(par4(1,1),par6(1,1)),par7(1,1))

par10(1,1) = 'load_data'
par11(1,1) = strcat(par10(1,1),par7(1,1))

par12(1,1) = 'disp_data'
par13(1,1) = strcat(par12(1,1),par7(1,1))


!Load the experimental displacement data.
*dim,%par11(1,1)%,array,exp_data_pts,3
*vread,%par11(1,1)%,%par8(1,1)%,csv,,jik,3,exp_data_pts
(E12.6,',',E12.6,',',E12.6)

!Load the experimental load data.
*dim,%par13(1,1)%,array,exp_data_pts,3
*vread,%par13(1,1)%,%par9(1,1)%,csv,,jik,3,exp_data_pts
(E12.6,',',E12.6,',',E12.6)


*dim,x_table,array,exp_data_pts,2
*dim,y_table,array,exp_data_pts,2
*dim,r_table,array,exp_data_pts,2
*do,eval,1,exp_data_pts
     x_table(eval,1) = %par13(1,1)%(eval,1)
     x_table(eval,2) = %par11(1,1)%(eval,1)

     y_table(eval,1) = %par13(1,1)%(eval,2)
```

```
      y_table(eval,2) = %par11(1,1)%(eval,2)

      r_table(eval,1) = %par13(1,1)%(eval,3)
      r_table(eval,2) = %par11(1,1)%(eval,3)
*enddo
```

```
FILENAME post_info.txt
/com *** Find desired quantities ***

!Select nodes and elements of the coupon.
cmsel,s,coupon_area,area
esla
nsla,s,1
rsys,0
plesol,epto,y
alls

cmsel,s,bottom_boundary,lines
nsll,s,1

!Define an array of bottom boundary displacements.
*do,eval,1,num_nodes,1
     b_disp_x(eval) = ux(b_nnum(eval))
     b_disp_y(eval) = uy(b_nnum(eval))
*enddo
alls

cmsel,s,top_boundary,lines
nsll,s,1

!Define an array of top boundary displacements.

*do,eval,1,num_nodes,1
     t_disp_x(eval) = ux(t_nnum(eval))
     t_disp_y(eval) = uy(t_nnum(eval))
*enddo
alls

!Determine the true displacements at the coupon boundary.
t_vec(1) = ux(tr_node)-ux(tl_node)+w
t_vec(2) = uy(tr_node)-uy(tl_node)


b_vec(1) = ux(br_node)-ux(bl_node)+w
b_vec(2) = uy(br_node)-uy(bl_node)

theta_t = -atan(t_vec(2)/t_vec(1))
theta_b = -atan(b_vec(2)/b_vec(1))

!Determine the angular displacement.
disp_theta_true = theta_t-theta_b


!Determine the x and y displacements.
ms_b(1) = (ux(br_node)+ux(bl_node))/2
ms_b(2) = -l/2+(uy(br_node)+uy(bl_node))/2
```

```
center(1) = ms_b(1)+l/2*sin(theta_b)
center(2) = ms_b(2)+l/2*cos(theta_b)

ms_t_rot(1) = center(1)+l/2*sin(theta_t)
ms_t_rot(2) = center(2)+l/2*cos(theta_t)

ms_t_true(1) = (ux(tr_node)+ux(tl_node))/2
ms_t_true(2) = l/2+(uy(tr_node)+uy(tl_node))/2

disp_true(1) = ms_t_true(1)-ms_t_rot(1)
disp_true(2) = ms_t_true(2)-ms_t_rot(2)

! Negative accounts for -theta_b rotation
rot_true(1,1) = cos(theta_b),-sin(theta_b)
rot_true(1,2) = sin(theta_b),cos(theta_b)

!Determine the x and y displacements parallel and perp.
with bottom grip.
disp_x_true =
disp_true(1)*rot_true(1,1)+disp_true(2)*rot_true(1,2)
disp_y_true =
disp_true(1)*rot_true(2,1)+disp_true(2)*rot_true(2,2)
disp_arc_true = disp_theta_true*arm


!Get the reaction forces.
cmsel,s,tab_bottom_boundary,lines
nsll,s,1

moment = 0
x_force = 0
y_force = 0


!Define an array of bottom boundary displacements.
*do,eval,1,num_nodes_tab,1
     *get,temp_fx,node,tt_nnum(eval),rf,fx
     *get,temp_fy,node,tt_nnum(eval),rf,fy

     x_force = temp_fx+x_force
     y_force = temp_fy+y_force

     temp_m = -
(tt_xloc(eval)+ux(tt_nnum(eval)))*temp_fy+(tt_yloc(eval)+uy
(tt_nnum(eval)))*temp_fx

     moment = temp_m+moment

*enddo
alls
```

```
!Determine the x and y forces parallel and perp. with
bottom grip.
f_x_true = x_force*rot_true(1,1)+y_force*rot_true(1,2)
f_y_true = x_force*rot_true(2,1)+y_force*rot_true(2,2)
```

```
FILENAME file_save.txt
/com *** File Save ***


!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!
!This section saves the data.
*Mwrite,data_out,%par0(1,1)%,csv
(G14.5,',',G14.5,',',G14.5,',',G14.5,',',G14.5,',',G14.5)
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!
```