2020

AN INVERSE METHOD FOR MATERIAL PROPERTY DERIVATION USING DIGITAL IMAGE CORRELATION AND THE FINITE ELEMENT METHOD

Sean D. Scro
University of Rhode Island, seanscro3@gmail.com

AN INVERSE METHOD FOR MATERIAL PROPERTY DERIVATION USING DIGITAL IMAGE CORRELATION AND THE FINITE ELEMENT METHOD

BY

SEAN D. SCRO

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE

REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

IN

MECHANICAL ENGINEERING

UNIVERSITY OF RHODE ISLAND

2020

MASTER OF SCIENCE THESIS

OF

SEAN D. SCRO

APPROVED:

Thesis Committee:

Major Professor David G. Taggart

George Tsiatas

Robert Coyne

Brenton LeBoef

DEAN OF THE GRADUATE SCHOOL

UNIVERSITY OF RHODE ISLAND

2020

ABSTRACT

The objective of this research is to develop a standard operating procedure for correlating uniaxial and biaxial digital image correlation (DIC) data and results from numerical finite element analysis (FEA) simulations to determine material elastic properties. An inverse method is developed in which iterative data matching is achieved using the software package, Isight™. The method is applied to characterize the elastic properties of Lexan polycarbonate sheet and a nominal carbon epoxy composite. Displacement and strain fields obtained from DIC data from simulated uniaxial tension and biaxial DIC experiments serve as the target parameter to match in iterative testing where the Young's modulus and Poisson's ratio are updated each cycle. The results of these experiments are used to verify that an accurate approximation of an unknown material's elastic properties can be predicted by this procedure.

ACKNOWLEDGMENTS

I would like to thank Dr. Taggart for his guidance, technical support, and mentorship during my time as his student. I'm very grateful for his patience and assistance throughout the research process. His guidance was invaluable to the completion of my work.

Contents

Page

Page

List of Figures

Figure **Page**

List of Tables

CHAPTER 1

Introduction

Characterization of the constitutive behavior of woven and composite materials is a topic of increasing importance for a variety of commercial, aerospace and military applications. Understanding the material properties and behavior of cutting edge materials as they develop is paramount to their implementation in the field. This research seeks to develop a standard operating procedure for correlating uniaxial and biaxial digital image correlation (DIC) data and results from numerical finite element analysis (FEA) simulations to determine material elastic properties. Iterative data matching will be achieved using the software package, Isight™. This technique will be developed and validated for isotropic and orthotropic materials with known elastic properties.

1.1 Modeling Overview

The models in this research simulate tensile and biaxial tests to iteratively determine the material properties of the test sample. For modeling a tensile test, in which an axial load, P, is applied to a sample and the deformation as characterized by the observed change in gage section length, L, with respect to time. The stress, σ_{22}, of the sample is defined as the longitudinal axial load, P, divided by the initial cross sectional area of the sample, A_o. The strain of the sample is defined as the ratio of the deformed length of the sample, L, minus the original length, L_o, divided by its original length, [4].

$$\sigma_{22} = \frac{P}{A_o} \tag{1}$$

$$\epsilon_{22} = \frac{L - L_o}{L} \tag{2}$$

Hooke's law for an isotropic material subjected to a three dimensional stress

state (Eq. 3) relates the strain components to the stress components in terms of Young's Modulus, E and Poisson's ratio ν. For the case of uniaxial tension in the y-direction, Young's modulus is the ratio of applied stress, σ_{22}, to axial strain, ϵ_{22}. Poisson's ratio is defined as the negative of the ratio of the transverse strain to the axial strains $\nu = -\frac{\epsilon_{11}}{\epsilon_{22}}$ [5].

$$\begin{aligned}
\epsilon_{11} &= \frac{\sigma_{11}}{E} - \frac{\nu}{E}(\sigma_{22} + \sigma_{33}) \\
\epsilon_{22} &= \frac{\sigma_{22}}{E} - \frac{\nu}{E}(\sigma_{11} + \sigma_{33}) \\
\epsilon_{33} &= \frac{\sigma_{33}}{E} - \frac{\nu}{E}(\sigma_{11} + \sigma_{22}) \\
\gamma_{23} &= \frac{1+\nu}{E}\tau_{23} \\
\gamma_{13} &= \frac{1+\nu}{E}\tau_{13} \\
\gamma_{12} &= \frac{1+\nu}{E}\tau_{12}
\end{aligned} \tag{3}$$

Eq. 3 applies to the behavior of linear-elastic isotropic materials. An isotropic material is defined as a material whose properties are independent on the loading direction. Orthotropic materials have different properties in three mutually orthogonal directions. For orthotropic materials, a generalized version of Hooke's law (Eq. 4) is used to describe the relation between stresses and strains.

$$\begin{bmatrix} \epsilon_{11} \\ \epsilon_{22} \\ \epsilon_{33} \\ \gamma_{23} \\ \gamma_{13} \\ \gamma_{12} \end{bmatrix} = \begin{bmatrix} \frac{1}{E_1} & -\frac{\nu_{21}}{E_2} & -\frac{\nu_{31}}{E_3} & 0 & 0 & 0 \\ -\frac{\nu_{12}}{E_1} & \frac{1}{E_2} & -\frac{\nu_{32}}{E_3} & 0 & 0 & 0 \\ -\frac{\nu_{13}}{E_1} & -\frac{\nu_{23}}{E_2} & \frac{1}{E_3} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{G_{23}} & & \\ 0 & 0 & 0 & 0 & \frac{1}{G_{13}} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{G_{12}} \end{bmatrix} \begin{bmatrix} \sigma_{11} \\ \sigma_{22} \\ \sigma_{33} \\ \tau_{23} \\ \tau_{13} \\ \tau_{12} \end{bmatrix} \tag{4}$$

For the case of plane stress, where the out-of-plane stresses σ_{33}, τ_{23} and τ_{13} are zero, Eq. 4 reduces to [4].

$$\begin{bmatrix} \epsilon_{11} \\ \epsilon_{22} \\ \gamma_{12} \end{bmatrix} = \begin{bmatrix} \frac{1}{E_1} & -\frac{\nu_{21}}{E_2} & 0 \\ -\frac{\nu_{12}}{E_1} & \frac{1}{E_2} & 0 \\ 0 & 0 & \frac{1}{G_{12}} \end{bmatrix} \begin{bmatrix} \sigma_{11} \\ \sigma_{22} \\ \tau_{12} \end{bmatrix} \tag{5}$$

Note that the 3x3 matrix on the right hand side of Eq. 5 is referred to as the material compliance matrix and the inverse of this matrix is referred to as the material stiffness matrix. In chapter 3, the material stiffmess matrix will be referenced as the matrix $\boldsymbol{D}$. It should be noted that Eq. 5 reduces to the case of plane stress for isotropic materials, where $E = E_1 = E_2$, $\nu = \nu_{12} = \nu_{21}$ and $G = G_{12} = \frac{E}{1+\nu}$:

$$\begin{bmatrix} \epsilon_{11} \\ \epsilon_{22} \\ \gamma_{12} \end{bmatrix} = \begin{bmatrix} \frac{1}{E} & -\frac{\nu}{E} & 0 \\ -\frac{\nu}{E} & \frac{1}{E} & 0 \\ 0 & 0 & \frac{1}{G} \end{bmatrix} \begin{bmatrix} \sigma_{11} \\ \sigma_{22} \\ \tau_{12} \end{bmatrix} \tag{6}$$

1.2 Motivation

Inflatable fabric structures are of growing importance for application in aerospace, commercial and military fields. Fabrics are used as reinforcement in both stiff laminated composite structures where a resin matrix fills the voids between the woven yarns and flexible skins for inflatable structures where a polymer coating is applied to the surface of the fabric but does not fill the interstices between the yarns. Inflatable fabric structures offer several advantages, including relatively low structural weight, low deflated volume, ease of speedy deployment, and simple transportation, making them an attractive and cost effective solution for a variety of applications. Successful application of these materials requires methods for characterizing the constitutive behavior under a variety of load conditions.

1.3 Objective

The objective of this research is to develop a coupled experimental and numerical technique for obtaining the elastic material properties of a material under

either uniaxial or biaxial tension utilizing a version of the finite element model update method. In this procedure, the material properties are taken to be unknown and are iteratively adjusted until the experimentally observed surface displacement and strain fields match the finite element model predictions. Validation of the method is achieved if the resulting properties match the expected values for the material under consideration.

Another objective of this research is to apply the algorithm to an unknown orthotropic material, such as a laminated composite or a woven fabric material, to identify their elastic material properties. It is anticipated that future research will explore application of this technique to the characterization of the non-linear response of this class of materials subjected to a variety of load conditions.

1.4 Methodology

An initial set of experiments will be devised to characterize the uniaxial and biaxial elastic properties of a known isotropic material, For this study, the material being considered is taken to be Lexan polycarbonate, a widely used stiff polymer with well known material properties. The original research plan was modified to adapt to the COVID-19 pandemic in the Spring of 2020. Rather than using experimentally obtained DIC image data, simulated speckle patterns were utilized. To simulate specimen deformation to be measured with DIC, 4000x4000 pixel images of speckle patterns in the 30mm long by 25mm wide section of the center of a tensile bar were created with a stochastic pattern using a MATLAB script [Appendix B]. The displacement of each speckle was determined using an Abaqus™ finite element model and the deformed speckle images are created using the same MATLAB script. These simulated deformed images are analyzed using the GOM Correlate™ and ARAMIS™ DIC software to compute the full field displacements and strains. These results are then used during the iterative analysis as a target to

match. A second set of experiments seek to characterize the biaxial response of the same material. Prior to these experiments, specifics of the cruciform geometry will be analyzed using finite element analysis. Geometric parameters to be explored include specimen width, thickness, fillet radius and the addition of slits outside of the biaxial stress region to assist in achieving a more uniform biaxial stress state. The objective of this analysis is to identify an appropriate cruciform specimen design. The cruciform specimen images were generated using the same procedure as the dogbone samples, focusing on their center sections behavior. As in the uniaxial tests, simulted DIC images are generated and commercial DIC software is used to measure the deformation and strain fields in the cruciform test section. It is observed that the state of strain in the cruciform test section is non-uniform. To extract material properties from the test data will require correlation of the DIC data with finite element analysis results. This is achieved using an iterative optimization algorithm that is available in the Isight software package. The resulting elastic properties are compared to those obtained using uniaxial test data.

Upon successful correlation of elastic properties measured with uniaxial and biaxial testing, another material which exhibits orthotropic material behavior is examined. Such materials require characterization of four in-plane elastic constants: E_1, the Young's modulus in the longitudinal direction; E_2, the Young's modulus in the transverse direction; ν_{12} , the Poisson's ratio characterizing the transverse contraction due to a loading in the longidutinal direction; and G_{12}, the in plane shear modulus. Using a procedure similar to that implemented by Lecompte et al. [6] to determine the orthotropic properties of a glass fiber reinforced epoxy composite laminate, an iterative data matching analysis of the DIC data and finite element analysis results is used to determine these constants for the orthotropic material of interest. In this study, the Abaqus finite element analysis program, coupled with

Isight's optimization capabilities will be used for the iterative analysis.

Successful demonstration of the validity of this experimental technique will allow for a coated woven fabric material, typical of that used in inflatable structures to be evaluated under biaxial loading. The objective of this experiment is to simulate the effect of inflation pressure on the stiffness of the coated fabric material when subjected to subsequent structural loads. The ability to determine the effect of inflation pressure on inflated panel stiffness will provide a valuable tool in predicting the behavior of inflatable structures.

1.5 Thesis Outline

The purpose of this thesis is to develop a standard operating procedure for correlating uniaxial and biaxial digital image correlation (DIC) data and results from numerical finite element analysis (FEA) simulations to determine the elastic properties of orthotropic materials. Chapter 2 will review published articles and studies relevant to DIC, FEA, and the development of the iterative testing procedure. Chapter 3 will discuss the isoparametric finite element formulation of 4 node quadrilateral elements used to analyze the tensile bar and cruciform models. DIC background and theory, and the optimization methods used in this study's iterative testing procedure will also be discussed. Chapter 4 will describe each step of the experimental procedures, including: sample creation, model creation, DIC camera set up and operation, tensile testing, iterative testing procedure. Chapter 5 will discuss the results obtained throughout this research for both the uniaxial and biaxial test cases. It will also discuss some of the optimization nuances encountered when constructing this technique. Chapter 6 will provide a summary of relevant information and conclusions from the previous chapters and make recommendations for future research possibilities and adaptations.

CHAPTER 2

Literature review

2.1 DIC and Finite Element Modeling

Digital image correlation (DIC) is a process developed by Sutton et al. [7] and Bruck et al. [8] for material analysis. DIC is a full-field optical measurement technique for examining the deformation of materials and structures. It has been used to study the mechanics of textile reinforced concrete by Mobasher [9], woven materials by Baghernezhad et al. [10], as well as metals and orthotropic composites by Tariq et al[11]. The study conducted by Tariq et al. [11] demonstrates that using their MATLAB-based 2D DIC algoritm, the Young's modulus, E, and Poisson's ratio, ν, are accurately obtainable for a variety of materials, including 2024-T6 aluminum, AISI 4140 steel, an aramid/epoxy composite, an E-glass/polyester laminate, and a carbon/epoxy composite. This work verifies that obtaining an accurate uniaxial result is possible for composite materials as well as further verifying the validity of DIC as a method of property identification.

A more recent study done by Dinh et al. [12] utilizes Abaqus to replicate the strain results obtained from DIC using a PVC coated fabric that has undergone biaxial loading. This study investigated the elastic and plastic deformations of the PVC coated fabric. The goal of this research was to validate an orthotropic model for woven coated structures. The material properties E_{warp}, E_{fill}, ν_{12}, ν_{21}, and G_{12} were determined during uniaxial and biaxial testing by analyzing stress strain curves in the warp and fill directions. Because this analysis dealt with the elastic and plastic deformations, the properties were determined for different ranges of loading. The applied properties were used in an Abaqus model to generate a strain field to compare to a strain field obtained from DIC. Dinh et al. developed a biaxial model of their material subjected to an equibiaxial warp-fill direction load

ratio that correlated well with the observed response of their DIC experiment.

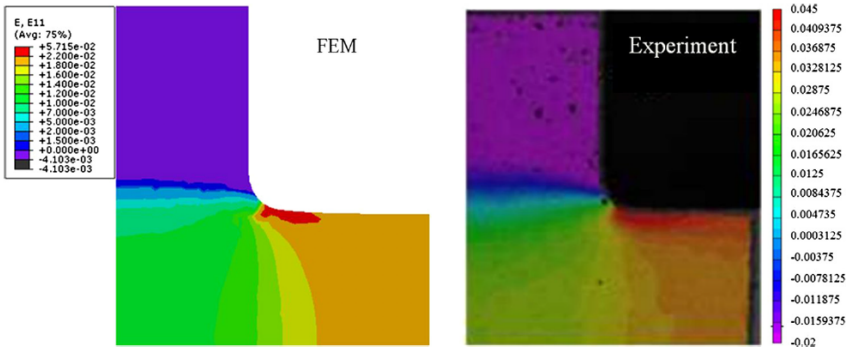


Figure 1: Warp Strain: FEA model (left) and DIC strain (right) with 1:1 loading [12]

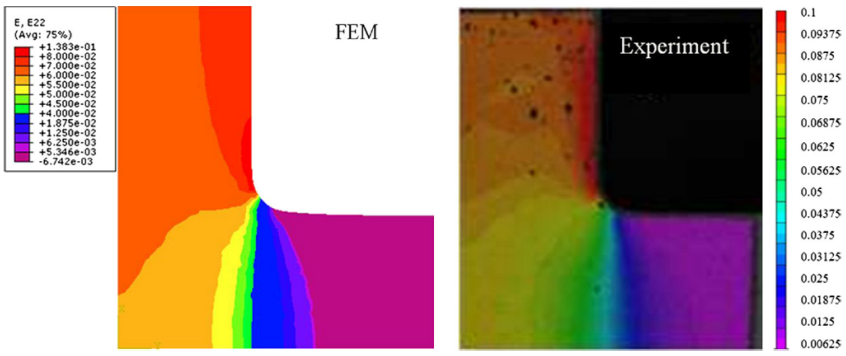


Figure 2: Fill Strain: FEA model (left) and DIC strain (right) with 1:1 loading [12]

2.2 Inverse Method and Iterative Testing

The primary goal of an inverse method is to identify a set of unknown parameters in a mathematical model. In a study done by Cooreman et al. [13] the parameters of the Hill yield surface and swift type hardening law of DC06 steel were determined using a method similar to the present study. To derive the aforementioned constants, the experimenters perform 3D DIC on a biaxial loaded cruciform sample of DC06 steel and correlated the observed response with a finite element model of the same sample. Their algorithm compares the axial and

shear strains measured at a number of points from the DIC experiment and the corresponding strain output from the finite element model. The strains at these points are correlated by computing a least-squares cost function. At each iteration the sensitivity matrix of the system is recalculated and the parameters are updated in accordance with the Gauss-Newton update algorithm. Convergence to a minimized cost function was shown to correspond to the desired material properties. The present research will employ a similar strain matching procedure. In a similar study by Lecompte et al. [6], the investigators attempt to derive four elastic parameters from two different samples of glass fibre reinforced epoxy with an inverse methodology. Both are cruciform in shape, but one is perforated with a hole in the center and the other is not. Similar to the first study mentioned, this method tracks the surface strains of the material with DIC and compares them to a finite element model, calculates the system's sensitivity matrix, and updates the parameters at each iteration. This scheme utilizes the same least cost function and optimization algorithm as the previous study. Both Young's moduli were measured within ± 5 GPa and the Poisson's ratio within ± 0.07 of their value from the classical lamination theory.

CHAPTER 3

Material Modeling

3.1 Finite Element Method

The finite element method is a widely used technique for problem solving in engineering and solid mechanics. It is a numerical technique for solving systems of partial differential equations. The method provides approximate solution to the field problem by solving a system of linear algebraic equations. The domain is discretized into finite sized subregions called *elements* and the solution is determined at key points called *nodes*. The finite element formulation of the partial differential equations consists of the development of element level equations which are then assembled into a larger system of equations that represent the entire domain of interest. For stress analysis problems, the element equations relate each element's nodal displacements to the corresponding nodal forces. The element formulation assumes interpolation functions to describe the displacement field between the nodes. The assembled global equations relate the displacements of all of the nodes in the model to the external nodal forces. These global equations are modified to impose the external loads and boundary conditions and are then solved to determine the unknown nodal displacements. These displacements are then used to compute the strain and stress fields by applying the assumed interpolation functions and the material's stress-strain behavior.

In this section, the formulation of the 4 node quadrilateral elements that were used to create the models used in this research are discussed. The finite element models were constructed to model the behavior of isotropic and orthotropic materials. The material properties of Lexan polycarbonate and a carbon/epoxy composite laminate were used in these models.

3.1.1 Element Formulation and Stiffness Calculation

All of the finite element models employed in this research utilized 4-node quadrilateral elements. Every element of the mesh did not have a rectangular shape, so an isoparametric formulation for their coordinate system was used to simplify the computational process [1]. This takes advantage of an element's natural coordinate system where the origin is placed at the center of the element (see Fig. 3).

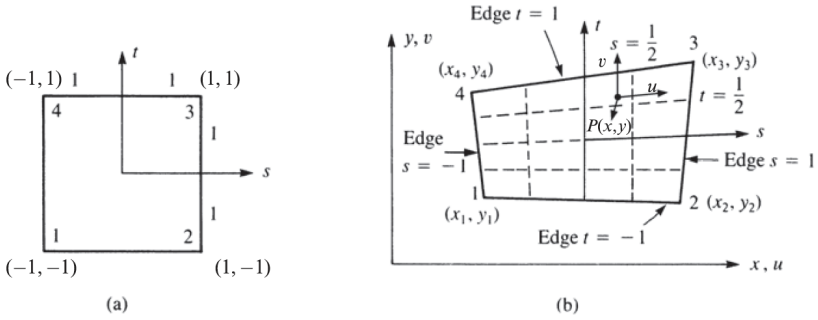


Figure 3: A natural coordinate system mapped [1]

The natural coordinate system is given in terms of s and t, which provide a one-to-one mapping of a global (x, y) coordinate point to a corresponding (s, t) coordinate point. The global x and y coordinates can be written in terms of nodal coordinates (x_1, x_2, x_3, x_4) and (y_1, y_2, y_3, y_4) and the (s, t) position of a given point

$$
\begin{aligned}
x =& \frac{1}{4}\Big[(1-s)(1-t)x_1 + (1+s)(1-t)x_2 + (1+s)(1+t)x_3 + (1-s)(1+t)x_4\Big] \\
y =& \frac{1}{4}\Big[(1-s)(1-t)y_1 + (1+s)(1-t)y_2 + (1+s)(1+t)y_3 + (1-s)(1+t)y_4\Big]
\end{aligned}
\tag{7}
$$

or, in matrix form.

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} N_1 & 0 & N_2 & 0 & N_3 & 0 & N_4 & 0 \\ 0 & N_1 & 0 & N_2 & 0 & N_3 & 0 & N_4 \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ x_3 \\ y_3 \\ x_4 \\ y_4 \end{bmatrix} \tag{8}$$

In Eq. 8, the N_i entries are the shape functions given by

$$\begin{aligned} N_1 &= \frac{(1-s)(1-t)}{4} \\ N_2 &= \frac{(1+s)(1-t)}{4} \\ N_3 &= \frac{(1+s)(1+t)}{4} \\ N_4 &= \frac{(1-s)(1+t)}{4} \end{aligned} \tag{9}$$

Similarly, the displacements of any point with natural coordinates (s, t) can be determined from the nodal coordinates (u_1, u_2, u_3, u_4) and (v_1, v_2, v_3, v_4)

$$\begin{bmatrix} u(s,t) \\ v(s,t) \end{bmatrix} = \begin{bmatrix} N_1 & 0 & N_2 & 0 & N_3 & 0 & N_4 & 0 \\ 0 & N_1 & 0 & N_2 & 0 & N_3 & 0 & N_4 \end{bmatrix} \begin{bmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \\ u_3 \\ v_3 \\ u_4 \\ v_4 \end{bmatrix} \tag{10}$$

Using this isoparametric coordinate transformation, it is possible to derive the so-called $\boldsymbol{B}$ matrix which relates the nodal displacements to the element strains and the terms in the $\boldsymbol{B}$ matrix are functions of $(s,\ t)$. Considering the function $f(x(s,t), y(s,t))$ as a general form were f can represents displacement fields u or v, the strains are defined in terms of derivatives $\frac{\partial f}{\partial x}$ and $\frac{\partial f}{\partial y}$. Relating these derivatives to the partial derivatives $\frac{\partial f}{\partial s}$ and $\frac{\partial f}{\partial t}$ requires application of the chain

rule. For example, for the derivative of $f(x,y)$ we obtain

$$\begin{aligned}\frac{\partial f}{\partial s} &= \frac{\partial f}{\partial x}\frac{\partial x}{\partial s} + \frac{\partial f}{\partial y}\frac{\partial y}{\partial s} \\ \frac{\partial f}{\partial t} &= \frac{\partial f}{\partial x}\frac{\partial x}{\partial t} + \frac{\partial f}{\partial y}\frac{\partial y}{\partial t}\end{aligned} \tag{11}$$

Now that $\frac{\partial f}{\partial s}$, $\frac{\partial f}{\partial t}$, $\frac{\partial x}{\partial s}$, $\frac{\partial x}{\partial t}$, $\frac{\partial y}{\partial s}$, and $\frac{\partial y}{\partial t}$ are defined, Cramer's Rule can be applied to solve for the operators $\frac{\partial}{\partial x}$ and $\frac{\partial}{\partial y}$. For example

$$\frac{\partial f}{\partial x} = \frac{\begin{vmatrix} \frac{\partial f}{\partial s} & \frac{\partial y}{\partial s} \\ \frac{\partial f}{\partial t} & \frac{\partial y}{\partial t} \end{vmatrix}}{\begin{vmatrix} \frac{\partial x}{\partial s} & \frac{\partial f}{\partial s} \\ \frac{\partial x}{\partial t} & \frac{\partial f}{\partial t} \end{vmatrix}} \qquad\qquad \frac{\partial f}{\partial y} = \frac{\begin{vmatrix} \frac{\partial x}{\partial s} & \frac{\partial f}{\partial s} \\ \frac{\partial x}{\partial t} & \frac{\partial f}{\partial t} \end{vmatrix}}{\begin{vmatrix} \frac{\partial x}{\partial s} & \frac{\partial y}{\partial s} \\ \frac{\partial x}{\partial t} & \frac{\partial y}{\partial t} \end{vmatrix}} \tag{12}$$

The operators $\frac{\partial}{\partial x}$ and $\frac{\partial}{\partial y}$ are now defined below where $\boldsymbol{J}$ is the Jacobian matrix

$$\frac{\partial()}{\partial x} = \frac{1}{|\boldsymbol{J}|}\frac{\partial y}{\partial t}\frac{\partial()}{\partial s} - \frac{\partial y}{\partial s}\frac{\partial()}{\partial t} \qquad\qquad \frac{\partial()}{\partial y} = \frac{1}{|\boldsymbol{J}|}\frac{\partial x}{\partial s}\frac{\partial()}{\partial t} - \frac{\partial x}{\partial t}\frac{\partial()}{\partial s} \tag{13}$$

where

$$|\boldsymbol{J}| = \begin{vmatrix} \frac{\partial x}{\partial s} & \frac{\partial y}{\partial s} \\ \frac{\partial x}{\partial t} & \frac{\partial y}{\partial t} \end{vmatrix} \tag{14}$$

Tne element strain fields can now be written in terms of the isoparametric formulation.

$$\begin{bmatrix} \epsilon_{xx} \\ \epsilon_{yy} \\ \gamma_{xy} \end{bmatrix} = \frac{1}{J}\begin{bmatrix} \frac{\partial y}{\partial t}\frac{\partial()}{\partial s} - \frac{\partial y}{\partial s}\frac{\partial()}{\partial t} & 0 \\ 0 & \frac{\partial x}{\partial s}\frac{\partial()}{\partial t} - \frac{\partial x}{\partial t}\frac{\partial()}{\partial s} \\ \frac{\partial x}{\partial s}\frac{\partial()}{\partial t} - \frac{\partial x}{\partial t}\frac{\partial()}{\partial s} & \frac{\partial y}{\partial t}\frac{\partial()}{\partial s} - \frac{\partial y}{\partial s}\frac{\partial()}{\partial t} \end{bmatrix}\begin{bmatrix} u(s,t) \\ v(s,t) \end{bmatrix} \tag{15}$$

Substituting Eq. 10 on the right hand side of Eq. 15, we can define the $[\boldsymbol{B}]$ matrix as

$$\begin{bmatrix} \epsilon_{xx} \\ \epsilon_{yy} \\ \gamma_{xy} \end{bmatrix} = [\boldsymbol{B}]\begin{bmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \\ u_3 \\ v_3 \\ u_4 \\ v_4 \end{bmatrix} \tag{16}$$

where $[\boldsymbol{B}]$, is obtained by applying the differential operators on the right hand side of Eq. 15 to the shape functions in Eq. 9.

The element stiffness matrix, $[\mathbf{k}]$, can now be computed using isoparametric coordinate system. The element stiffness matrix relates the nodal displacements to the nodal forces. In global $(x,\ y)$ coordinates, the stiffness matrix is given by [1]

$$[\mathbf{k}] = \int\int_A [\boldsymbol{B}(x,y)]^T [\boldsymbol{D}][\boldsymbol{B}(x,y]|\boldsymbol{J}|hdxdy \tag{17}$$

where A is the area of the element, $[\mathbf{D}]$ contains the material constants, and h is the element thickness. For the case of an isotropic material subjected to plane stress

$$[\boldsymbol{D}] = \frac{E}{1-\nu^2}\begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & \frac{1-\nu}{2} \end{bmatrix} \tag{18}$$

Converting Eq. 17 to natural coordinates yields

$$[\mathbf{k}] = \int_{-1}^{1}\int_{-1}^{1} [\boldsymbol{B}(s,t)]^T [\boldsymbol{D}][\boldsymbol{B}(s,t)]|\boldsymbol{J}|hdsdt \tag{19}$$

This integral is evaluated using Gaussian quadrature, which is discussed below.

3.1.2 Gaussian Quadrature

Evaluation of the integral in Eq. 19 is approximated by implementing Gaussian quadrature. Gaussian quadrature evaluates the integral by sampling a number of points in the domain and evaluating the integrand at those points. Appropriate weight factors are applied to these terms and the integral is approximated by summing weighted integrand evaluations. For the method of Gaussian quadrature, the sampling point locations and associated weight factors are selected to give best accuracy for polynomial functions. The number of sampling points and associated weight factors can be specified, where increasing the number of points increases the

accuracy of the approximation. For the 4 node quadrilateral elements considered in this study, either a single sampling point, referred to as reduced integration, or a 2x2 array of points (4 sampling points), referred to as full integration, are typically used. Sampling point locations for 2x2 Gaussian quadrature are shown in Fig. 4. Fig. 5 gives the sampling point locations and associated weight factors for 1 point, 2 point, 3 point and 4 point Gaussian quadrature. These sampling point locations are the roots of Legendre polynomials of increasing order and have been shown to provide highly accurate approximations when integrating polynomial functions. For the case of the 4 node quadrilateral element with full integration, samplings points at (x_1,x_2) = $\pm$0.57735026918962 with weight factors $W_1 = W_2 = 1.00$ are utilized.

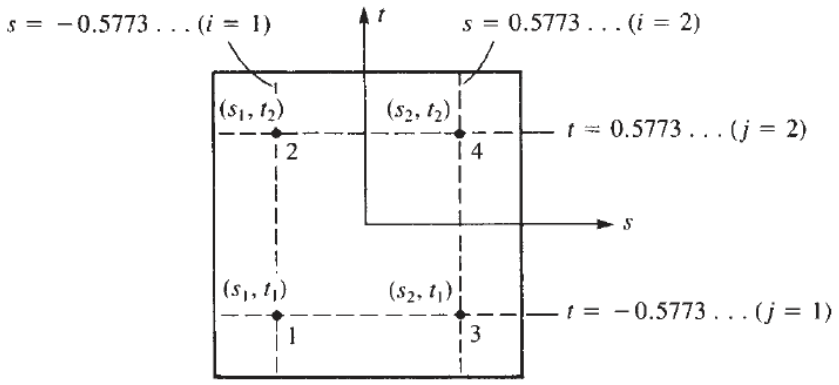


Figure 4: The evaluation points of a 4 node quad element. [1]

Number of Points	Locations, x_i	Associated Weights, W_i
1	$x_1 = 0.000\ldots$	2.000
2	$x_1, x_2 = \pm 0.57735026918962$	1.000
3	$x_1, x_3 = \pm 0.77459666924148$	$\frac{5}{9} = 0.555\ldots$
	$x_2 = 0.000\ldots$	$\frac{8}{9} = 0.888\ldots$
4	$x_1, x_4 = \pm 0.8611363116$	0.3478548451
	$x_2, x_3 = \pm 0.3399810436$	0.6521451549

Figure 5: 2D Gaussian quadrature sampling points and weight factors. [1]

Approximation of the element stiffness matrix is achieved using 4 point, or 2x2, Gaussian quadrature, resulting in a summation over the 4 sampling points as shown in Eq. 20.

$$[\mathbf{k}] = \sum_{i=1}^{4} [\boldsymbol{B}(s_i, t_i)]^T [\boldsymbol{D}][\boldsymbol{B}(s_i, t_i)] |\boldsymbol{J}| h W_i W_i \tag{20}$$

3.2 Digital Image Correlation

Digital Image Correlation (DIC) is a non-contacting full-field measurement technique. DIC tracks the movement of an object's surface to extract the displacement and strain fields across that objects surface [9]. To perform DIC on an object, an area of interest (AOI) must be identified and coated with a stochastic pattern. This pattern should be isotropic, non-repetitive, with high contrast between light and dark regions.

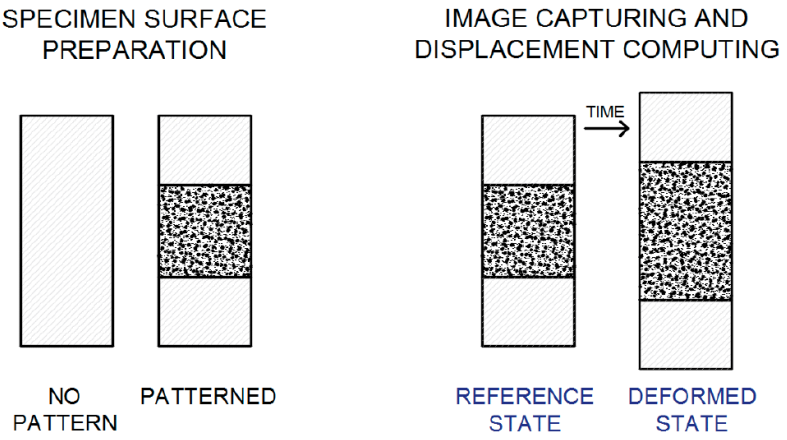


Figure 6: Reference and deformed area of interest[2]

To track image displacements of the surface within the software, a grid of facets, also called subsets, are positioned across the image. Each facet contains an $(n)x(n)$ grid of pixels. Each pixel within the facet is assigned a grey scale value from zero to one hundred as an indicator of how dark or light the pattern is within the pixel.

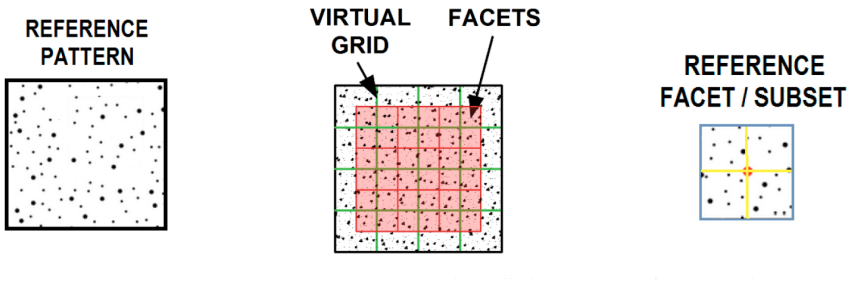


Figure 7: How DIC software interprets and divides a stochastic pattern. [2]

An initial measurement with no deformation is captured by an imaging system and used as a reference from which to calculate displacements. The sample is then deformed and the movement of the information contained within the facets is recorded again when the surface is displaced. The reference image is compared to the deformed image(s) using a sum of squared differences correlation function, $C(x, y, u, v)$ [14].

$$C(x, y, u, v) = \sum_{i=-\frac{n}{2}}^{\frac{n}{2}} \sum_{j=-\frac{n}{2}}^{\frac{n}{2}} (I(x+i, y+j) - I'(x+u+i, y+v+j))^2 \tag{21}$$

$C(x, y, u, v)$ is a function of the facet's center coordinates and it's displacements within the grid on the AOI's surface. The functions $I(x, y)$ and $I'(x+u, y+v)$ represent the reference and deformed images assigned greyscale value for a given pixel, respectively. This comparison is performed at all points within the facet, testing different displacement values in an attempt to minimize the correlation function. When that function reaches a minimum, the best case displacements of the image have been found for the facet[14].

Another process happening simultaneously is the software recognizing the values of pixels adjacent to a given pixel in order to track the deformation of the pattern. The values returned from these displacement analyses are what are used as the values to be matched by the models in this research. For this research, the

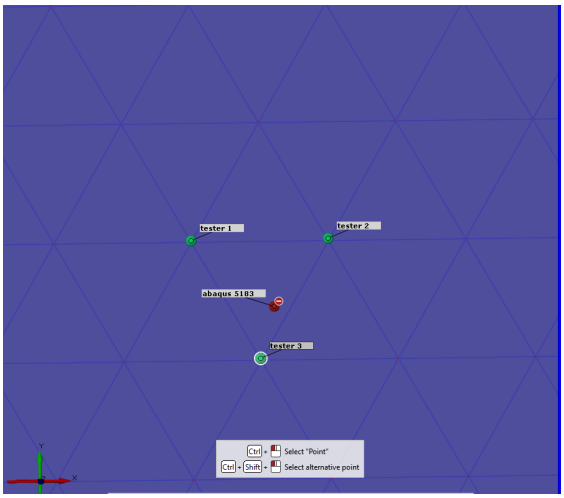


Figure 8: ABAQUS node placed on GOM mesh

GOM ARAMIS commercial DIC software package was utilized.

3.3 Interpolation of Displacements and Strains

A finite element model was developed for each case for use in comparison to the DIC data. These comparisons require that the finite element results be compared to DIC results at specific points. Ths was achieved by selecting key finite element nodes and interpolating the DIC results to give the corresponding DIC result at that node. For this study, the commercial finite element software, Abaqus, was utilized.

To begin the interpolation process a point of interest, (x, y) is specified and the coordinates of the grid triangle that it resides are recorded as $(x_1,\ x_2,\ x_3)$ and $(y_1,\ y_2,\ y_3)$. The displacements of a triangular regions displacement field are defined by

$$\begin{Bmatrix} u(x,y) \\ v(x,y) \end{Bmatrix} = \begin{bmatrix} 1 & x & y & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & x & y \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \end{bmatrix} \tag{22}$$

To derive the constants associated with the x or y components of displacement, equation (17) transforms.

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} = \begin{bmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} \tag{23}$$

The column vector of constants is determined by inverting the 3x3 matrix on the right hand side this equation. The inverse of this matrix is found by the method of cofactors to be

$$\frac{1}{2A} \begin{bmatrix} \alpha_1 & \alpha_2 & \alpha_3 \\ \beta_1 & \beta_2 & \beta_3 \\ \gamma_1 & \gamma_2 & \gamma_3 \end{bmatrix} \tag{24}$$

where

$$2A = \begin{vmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{vmatrix} \tag{25}$$

The entries of the cofactor matrix are

$$\begin{gathered} \alpha_1 = x_2 y_3 - x_3 y_2, \quad \alpha_2 = x_1 y_3 - x_3 y_1, \quad \alpha_3 = x_1 y_2 - x_2 y_1 \\ \beta_1 = y_2 - y_3, \qquad \beta_2 = y_3 - y_1, \qquad \beta_3 = y_1 - y_2 \\ \gamma_1 = x_2 - x_3, \qquad \gamma_2 = x_3 - x_1, \qquad \gamma_3 = x_1 - x_2 \end{gathered} \tag{26}$$

The cofactor matrix is the same for both sets of displacements. A simple formulation for the displacements of the point of interest is assembled by constructing weight factors from the entries of the cofactor matrix and the coordinates of the point of interest.

$$\begin{aligned} N_1 &= \frac{1}{2A}(\alpha_1 + \beta_1 x + \gamma_1 y) \\ N_2 &= \frac{1}{2A}(\alpha_2 + \beta_2 x + \gamma_2 y) \\ N_3 &= \frac{1}{2A}(\alpha_3 + \beta_3 x + \gamma_3 y) \end{aligned} \tag{27}$$

These weight factors are used in conjunction with the displacement of the node their subscript corresponds to to arrive at the value of the interpolated displacements.

$$\begin{aligned} u &= N_1 u_1 + N_2 u_2 + N_3 u_3 \\ v &= N_1 v_1 + N_2 v_2 + N_3 v_3 \end{aligned} \tag{28}$$

The MATLAB code used to interpolate the DIC generated displacements to the desired finite element node location is given in Appendix A. The same formulation was used to interpolate for the strains at each point of interest.

3.4 Optimization Schemes

To correlate DIC and finite element results in this study, two different algorithms were used to optimize the system: the Hooke-Jeeves pattern search [3], and the NLPQLP method that employs sequential quadratic programming [15]. Both schemes operate within bounds or constraints specified at the beginning of the optimization. Both are well suited for optimization of non-linear problems and are available in the ISight software package.

3.4.1 Pattern Search

The pattern search begins with a function $S(\psi)$ where ψ is a set of variables such that $\psi = (\psi_1, \psi_2, \psi_3, ..., \psi_n)$. The $S(\psi)$ function is sought to be minimized. The intuition of the process is that it starts from an initial guess point of all parameters. Each variable is updated individually to determine whether increasing or decreasing a given parameter proceeds in the correct direction of minimizing $S(\psi)$. If a value of ψ_i is successful it is saved as the value to be used in the next calculation of $S(\psi)$. If a value of ψ_i fails to further minimize $S(\psi)$, the value is deemed a failure and its previous value is used in the subsequent step. Each of these steps is an exploratory move that the algorithm takes in exploring the point space where the specified constraints define.

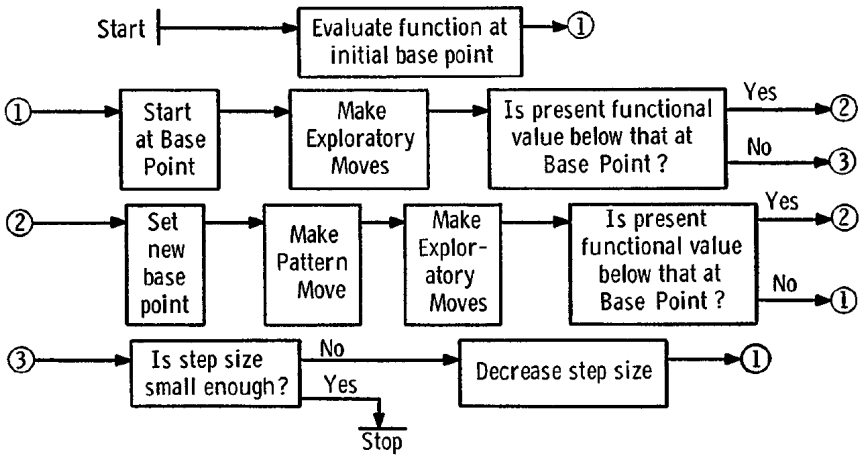


Figure 9: The progression of logic in a pattern search [3]

The optimization converges when the step size of the guess value of ψ_i reaches a cut off point determined at the beginning of the optimization. One advantage of this algorithm is that it will not stop until it reaches its convergence criterion or its iteration limit has been reached. It is a heuristic method of optimization that has proved to be effective for many problems. However, it is not time efficient.

3.4.2 NLPQLP

The NLPQLP algorithm begins similarly to the pattern search; it requires constraints, target parameters, and a step size to be identified [16]. It also requires that the function $f(x)$ that this algorithm attempts to optimize is twice differentiable and continuous over its domain as this is a gradient based method [17]. The parameters of the function are updated every iteration in an attempt to arrive at the target value, a maximum or minimum of the gradient. An advantage of this method is its efficiency. However, it can converge to local minima or maxima and return an inaccurate approximation.

CHAPTER 4

Experimental methods

The original experimental procedure is discussed in Appendix C.

4.1 ABAQUS Modeling

To begin, an ABAQUS model for the uniaxial and biaxial cases needed to be created. A tensile bar and a cruciform shape were used for each type of test respectively. Each of these models were used in the ISight iterative testing that returned the estimates of the elastic properties of the material being examined. The isotropic models were created using the material properties of Lexan (Table 1). Lexan is a common polycarbonate plastic that is available in easily cuttable sheets. The orthotropic models were created using the nominal material properties of a carbon/epoxy composite laminate (Table 2).

Lexan Properties	
Parameter	Value
Young's Modulus	2,100 MPa
Poisson's Ratio	0.37
Material Thickness	2 mm

Table 1: Isotropic Material Properties

Orthotropic Properties	
Parameter	Value
E_1	32,500 MPa
E_2	77,000 MPa
ν_{12}	0.297
G_{12}	22,400 MPa
Material Thickness	2 mm

Table 2: Orthotropic Material Properties

These material properties were the values sought to be returned from the iterative testing.

4.1.1 Tensile Bar

A tensile bar model was created in Abaqus with the following dimensions.

Tensile Bar Dimensions	
Dimension	Size (mm)
End Tab Width	40
End Tab Length	25
Taper Length	25
Bar Length	150
Bar Width	25
Bar Thickness	2

Table 3: Tensile bar model dimensions

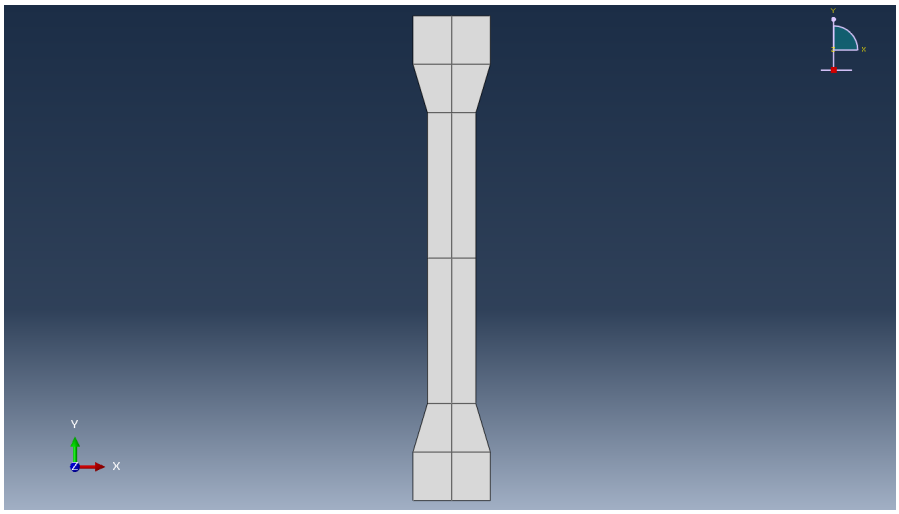


Figure 10: Unmeshed tensile bar model

Then, the part was partitioned to constrain the deformation of the end tabs to simulate the presence of a grip from a load frame.

Pressure loads of 40 MPa were applied to the tabs on the ends of the isotropic tensile bar. The loads were selected because they induce a three percent strain on the object. For the orthotropic case, this load would likely cause fiber failure and/or matrix cracking if applied to a physical sample. However, for the purpose of verifying the algorithm for an orthotropic material, it is convenient to idealize the part to exhibit linear elastic deformation. Three boundary conditions of the model

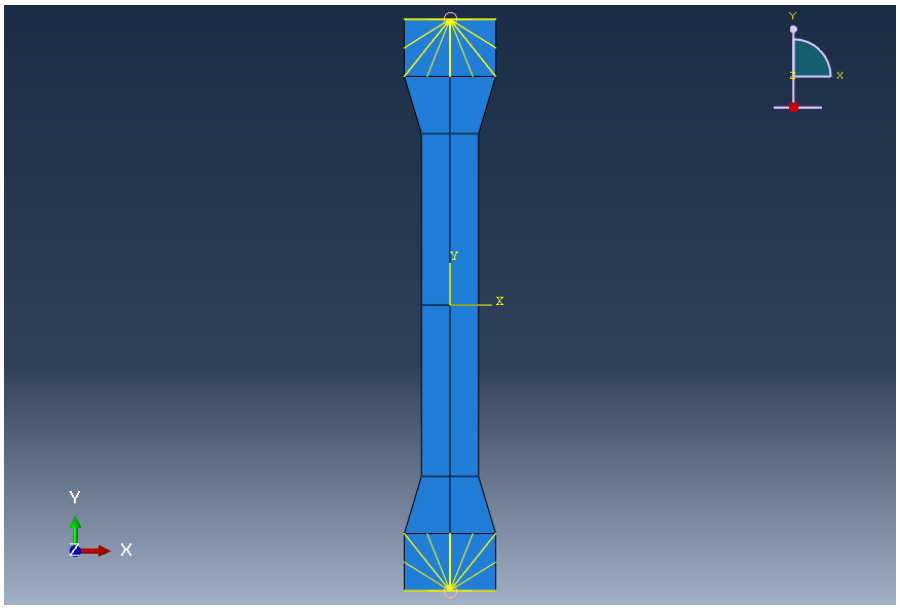


Figure 11: Constrained tensile bar

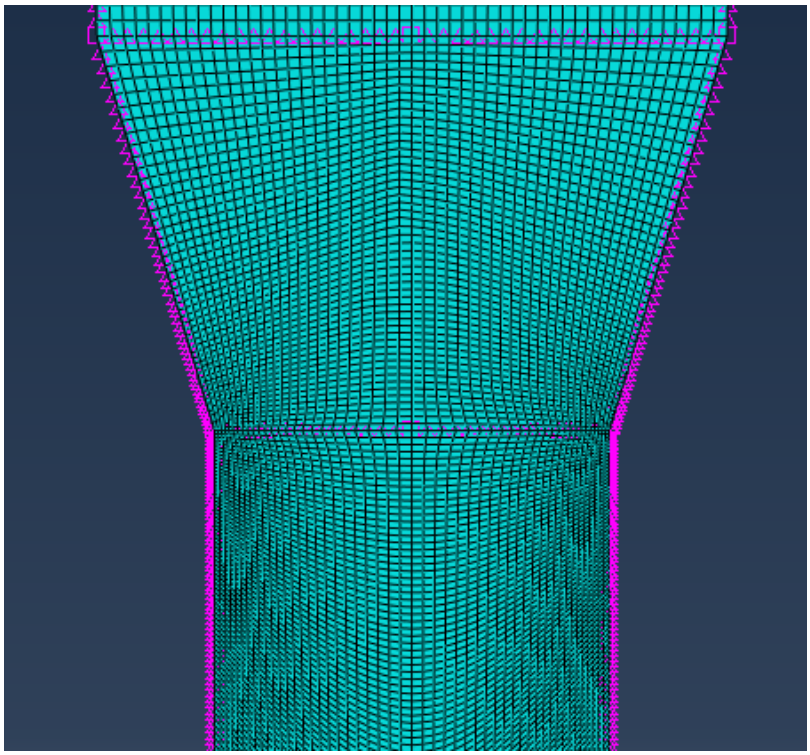
were specified to give a rigid uniform displacement of the tab region, zero horizontal displacement along the vertical center line of the tensile bar, and the center point of the model was fixed in both directions. The tensile bar was meshed with 4-node quadrilateral elements with biases toward the location where the tapered region meets the bar.

4.1.2 Cruciform

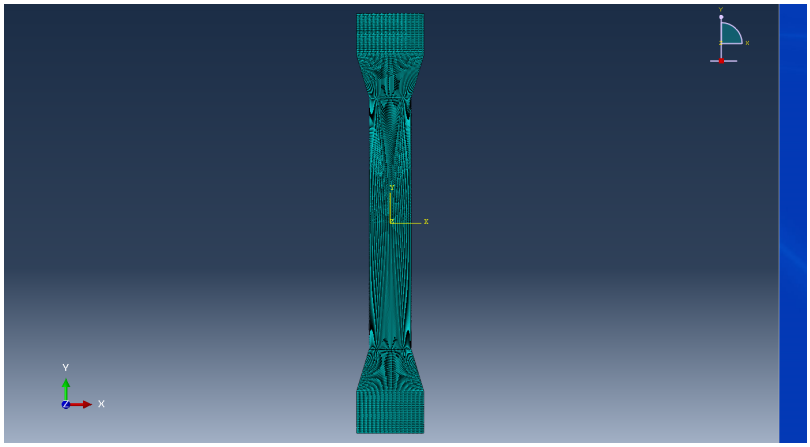
A cruciform shaped model was created in Abaqus with the following dimensions.

Cruciform Dimensions	
Dimension	Size (mm)
Tab Width	50
End Tab Length	25
Tab end to center	100
Model thickness	2

Table 4: Cruciform model dimensions



(a) Biased seeds



(b) Meshed tensile bar

Figure 12: The seeded and meshed tensile bar model

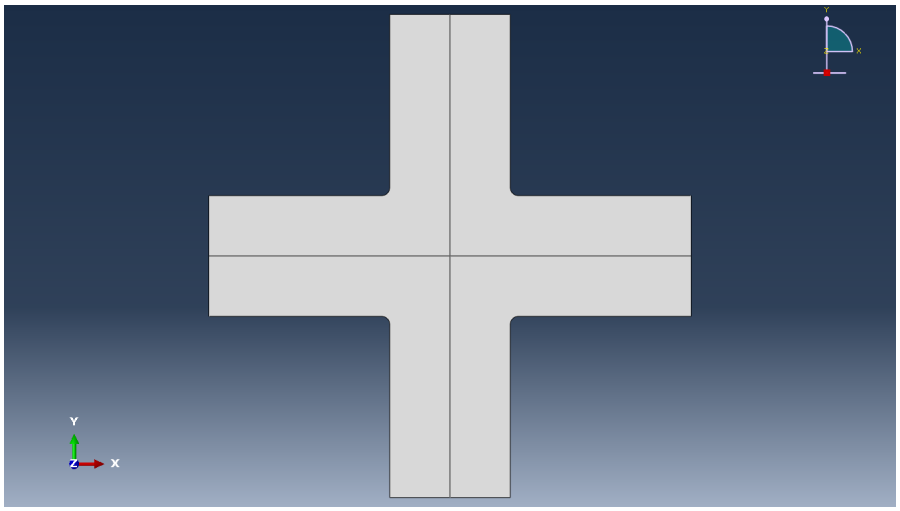


Figure 13: The unmeshed cruciform model

The part was partitioned along its x and x axes and -25 MPa loads were applied to the ends of all of the tabs for the isotropic model. For the orthotropic case loads of 975 MPa were applied to all tabs. Both of these load selections were made to generate strains that were around three percent. Similar to the orthotropic tensile bar case, the loads of the orthotropic cruciform sample would likely cause fiber failure and/or matrix cracking if applied to a physical sample. Two boundary conditions were applied to the model to constrain lateral displacement of the nodes along the x and y axes. The nodes on the x axis had their vertical displacement constrained, and the nodes on the x axis had their horizontal displacement constrained. Similar to the tensile bar models, this model was meshed with 4-node quadrilateral elements with element size biased toward the fillets.

4.1.3 Point Selection

To sample the strains and displacements of the Abaqus model, a collection of 10 points within the area of interest for the tensile bar and cruciform samples were selected. The node sets for both samples were selected from the upper right hand quadrant of the respective models. The nodes selected represent the strain

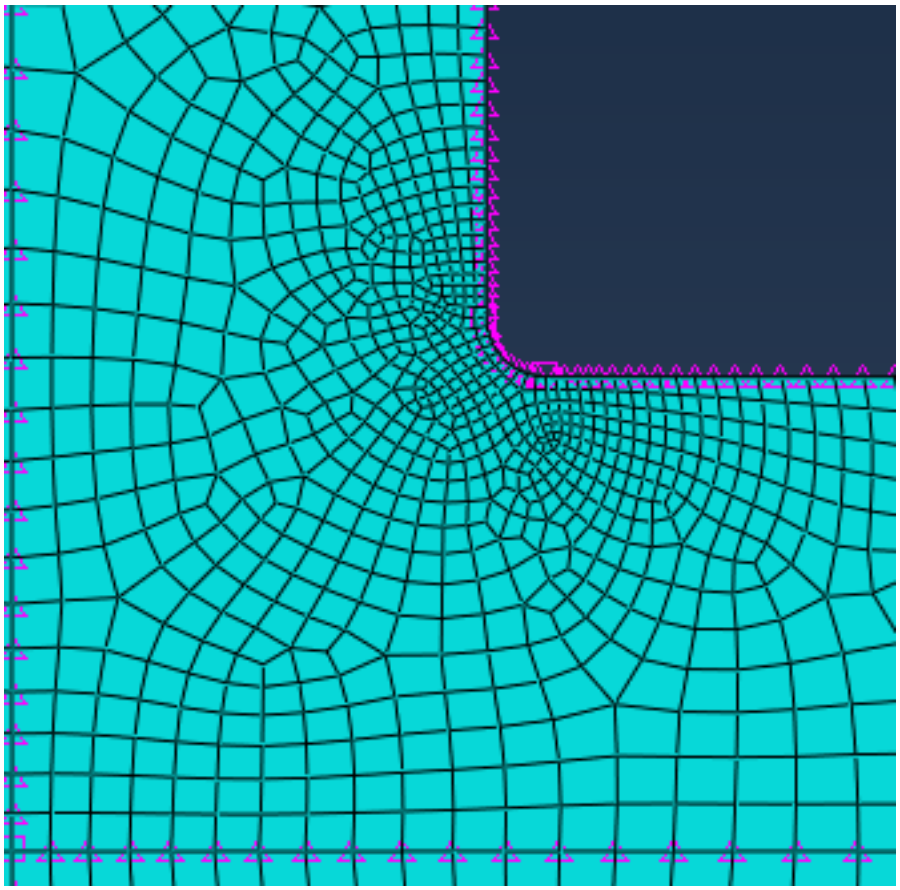


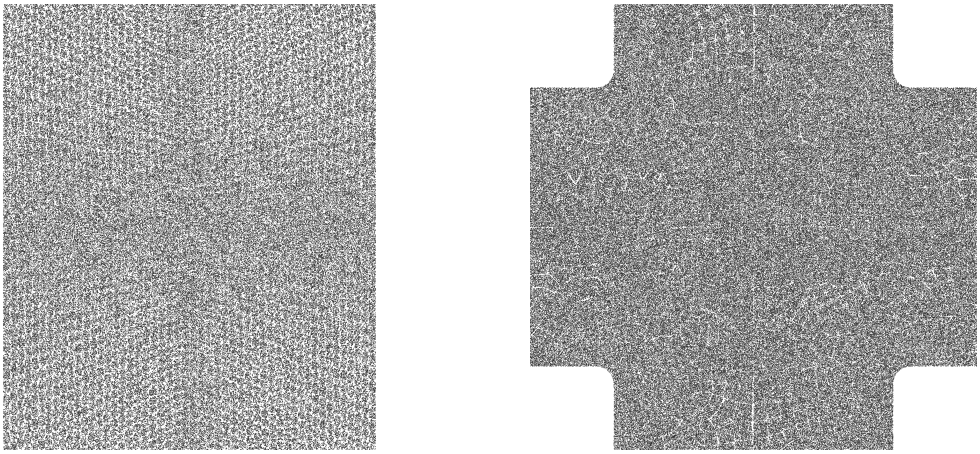
Figure 14: Seeded and meshed cruciform model

and displacement fields across the whole area of interest i.e. the points selected are scattered about the region, not focused in a cluster. For the tensile bar, this region is the center 30 mm length of the bar. For the cruciform shape, this area is the center square. Each node was assigned a specific set and a field output was requested for each individual set to extract the strains and displacements. The node number and initial positions of all selected points were recorded from the Abaqus input file and stored in a .csv file for later use.

4.2 Sample Preparation

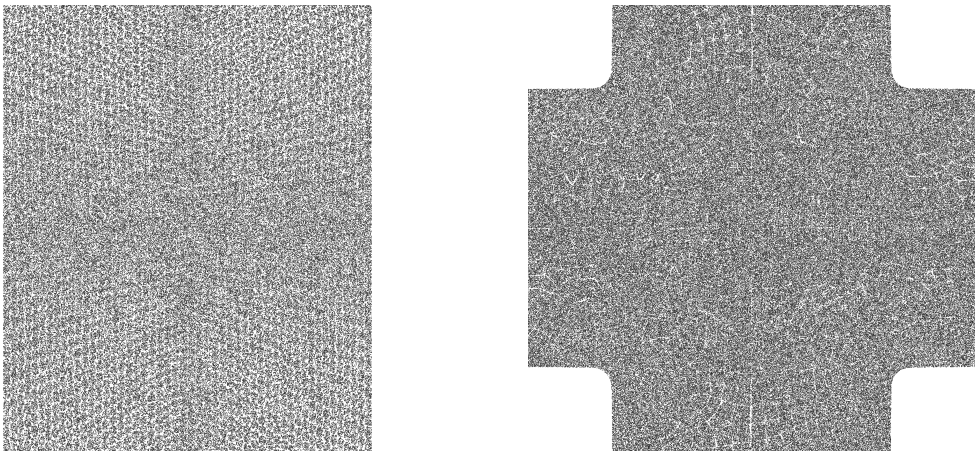
Due to limited access to facilities and equipment during the COVID-19 pandemic adjustments had to be made to the samples that were tested. Instead of per-

forming DIC on physical samples, a series of numerically generated, stochastically-patterned images of the tensile bar and cruciform models were created with a MATLAB script (see Appendix B). The code generates an initial speckle image by reading in the number of elements, number of nodes, nodes and elements themselves to create the surface to be virtually speckle coated. The image resolution, image scale, and number of deformation steps were specified at the start of the image creation. The amount of space between speckles, the maximum and minimum radii of speckles and their ability to overlap with one another when they are placed were also parameters to be specified for each case.



(a) Undeformed tensile bar speckle pattern (b) Meshed tensile bar

Figure 15: Undeformed cruciform speckle pattern



(a) Deformed tensile bar speckle pattern (b) Deformed cruciform speckle pattern

Figure 16: Deformed cruciform speckle pattern

An initial speckle image is created with the same footprint as the Abaqus model it imported data from. Then, a series of speckle images are created that deform in accordance with nodal displacements imported from Abaqus. The images were generated using the loads from the Abaqus model to induce a three percent strain.

4.3 DIC

The DIC analysis in this study was performed using the GOM ARAMIS software. The same process was used for both the tensile bar and cruciform case.

4.3.1 Software Routines

To begin, the numerically generated deformation image series was imported into the ARAMIS software for analysis. The first undeformed image is defined to be the reference state. The initial image analysis step is to scale the images within ARAMIS. Two pixels on either edge of the speckled region that were at the same height on the image were selected. For the tensile bar the image series focused on the center region of the bar and the distance was set to 25 mm. For the cruciform case the width of the tabs were set to 50 mm. With the reference

stage in view in the interface, a surface component with a facet size of 34 pixels and a point distance of 30 pixels was created for both sets of images. The facet size specifies the nxn pixel regions that are tracked by the software. The point distance is the distance from the center of one facet to the center of the adjacent facets [18]. The aforementioned parameter amounts were chosen for the surface

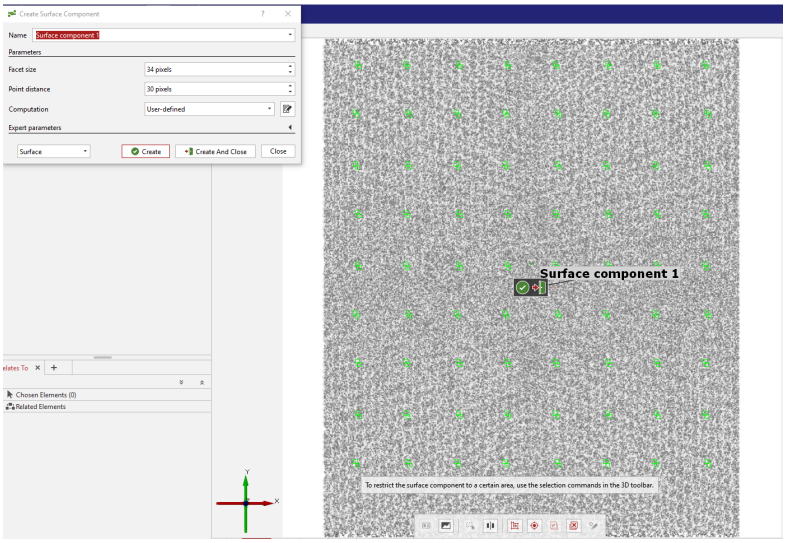


Figure 17: Surface component creation

component through a process of trial and error. If the facets were too small and the point distances too short, the mesh would give highly non-uniform strain and displacement fields for images with small deformations. With larger deformations, the mesh would develop holes where the software was unable to track the stochastic pattern. The rationale behind the procedure is to choose the facet size to be as small as possible, but still large enough to be calculated [18]. A facet's ability to be computed relative to its size is determined by the density of the stochastic pattern on the surface that is being analyzed.

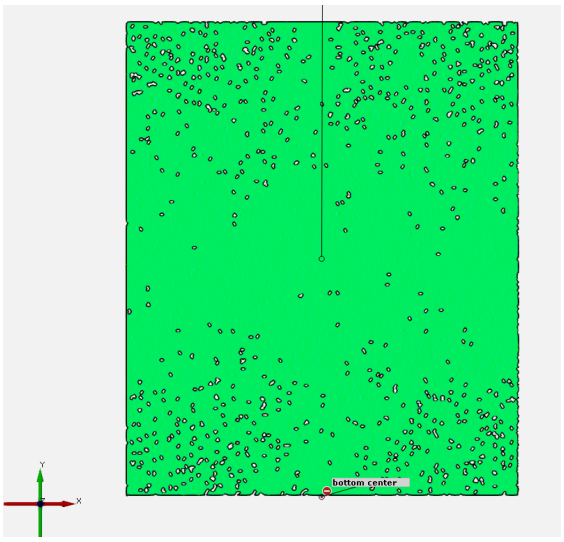


Figure 18: An example of a mesh with data loss

To view the mesh that is created by the surface component, select the surface component. Turn on the option to view the points and the mesh in the display tab find the properties menu. This may assist the process of determination of the facet size and point distance.

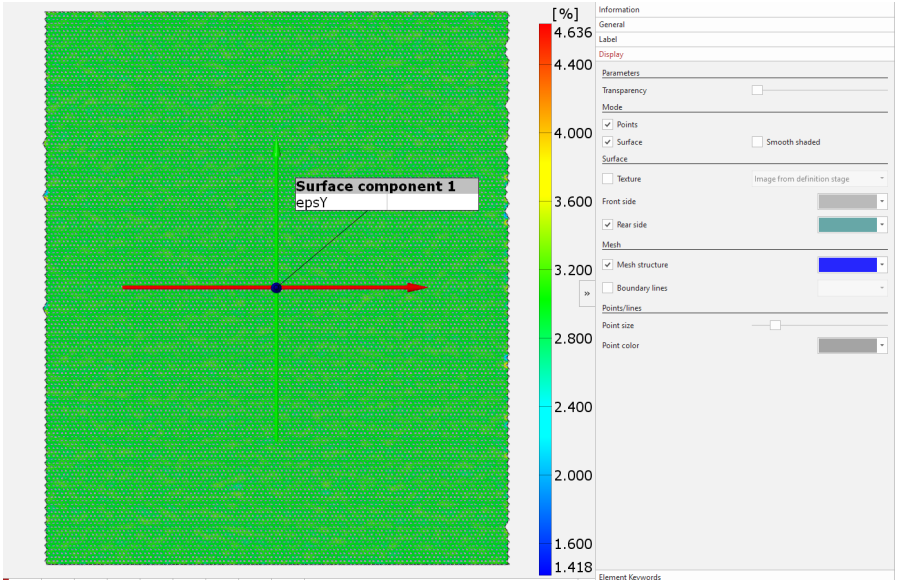


Figure 19: How to view the mesh and points

Normally after a surface component is created, an alignment must be per-

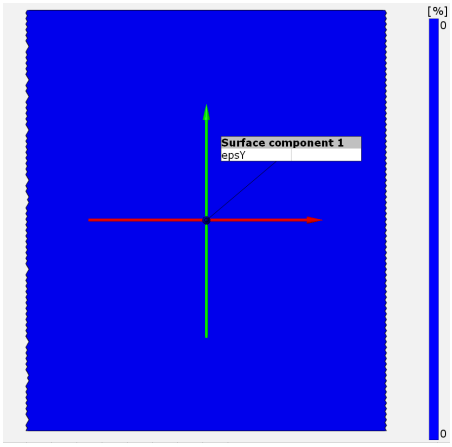
formed to define a coordinate system for the surface component. Because these images were created computationally, rather than captured with a camera, an accurate alignment was assigned to them automatically as they were imported. The alignment native to the imported images had the same orientation as the coordinate system in the Abaqus models.

4.3.2 Analysis

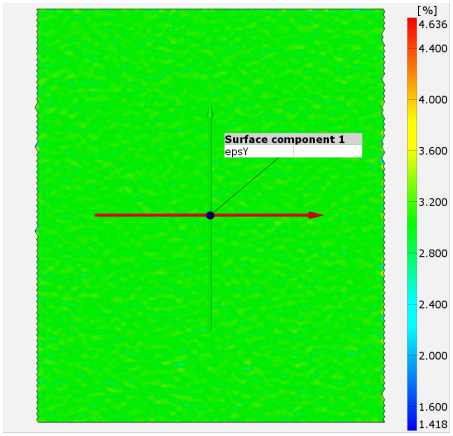
To begin extracting the data from the deformed images, the surface component was selected and an "inspection" of the part was performed. An inspection is a process in the software that calculates a desired field from the deformed image.

The isotropic cases were analyzed first. Initially, it was unclear which field, displacement or strain, would yield a more accurate result for optimization. The x and y strains and displacements were all extracted to determine which field was a better option to use as target values for iterative testing. To extract a desired field, all nodes of the surface component were selected after checking if the field of interest was displayed in the interface window and that that field was the only field currently being examined. The node number, (x, y) coordinates, and field value at the node are then exported as a .csv file. If multiple fields are being sampled at once the exported file will not record any field data, only the nodes and their locations.

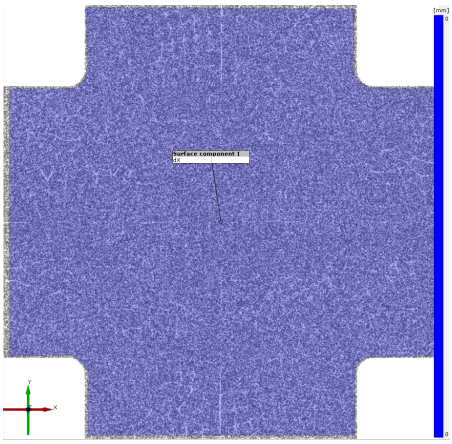
Once exported from ARAMIS, the displacement data and the strain data were combined in .csv files. These values, in concert with the recorded initial positions of the Abaqus nodes of interest were used to compute the strains and displacements at the locations selected in Abaqus on the surface component defined in ARAMIS. The Abaqus finite element mesh and the mesh of the surface component generated by ARAMIS were not identical to each other. Based on the nodal displacements and strains recorded by ARAMIS, the corresponding DIC



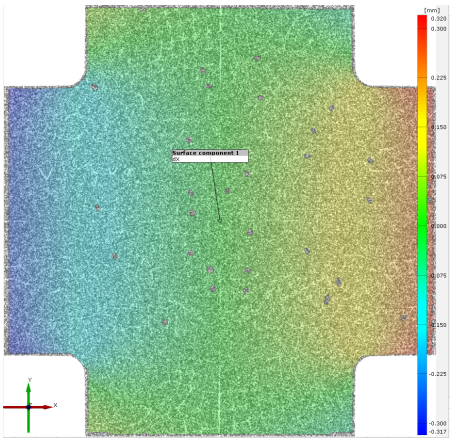
(a) Undeformed tensile bar surface component



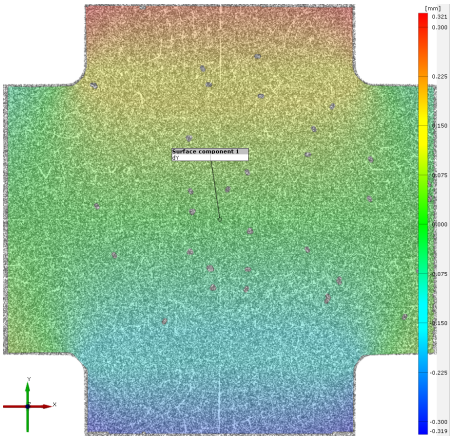
(b) Deformed tensile bar surface component



(c) Undeformed cruciform surface component



(d) Deformation in the x direction



(e) Deformation in the x direction

Figure 20: Deformation fields of the tensile bar and cruciform samples.

displacements and strains at the desired Abaqus node points were calculated by interpolation of the data recorded for the three nearest ARAMIS nodes.

4.4 Iteration

The driving concept behind an inverse method of property determination is that from the outset you have a data set obtained by experimentation that is sought to be matched to the finite element model by iterative model updating. When the model can no longer be improved, i.e. the optimization converges, the input values of the most accurate iteration are the properties of the sample being investigated. For the isotropic cases, those parameters are the Young's Modulus, E, and Poisson's ratio, ν, and for the an orthotropic material under biaxial load, the longitudinal and lateral Young's moduli, E_1 and E_2, Poisson ratio, ν_{12}, and sheer modulus, G_{12}. The optimization schemes used to extract the material properties were NQLPQL that uses sequential quadratic programming [15], and a Hooke-Jeeves pattern search [3]. ISight, by Simulia, was the software used to execute the iterative analysis. ISight is an interface that combines multiple applications in a process flow, analyzes a system, and automates their execution to extract the optimal parameters of a given system. For the systems investigated in this research only Abaqus and Excel components were needed.

4.4.1 Parameter Selection and Mapping

After establishing the component flow of the loop the corresponding models and spreadsheets were imported into ISight and the type of simulation flow was defined as an optimization.

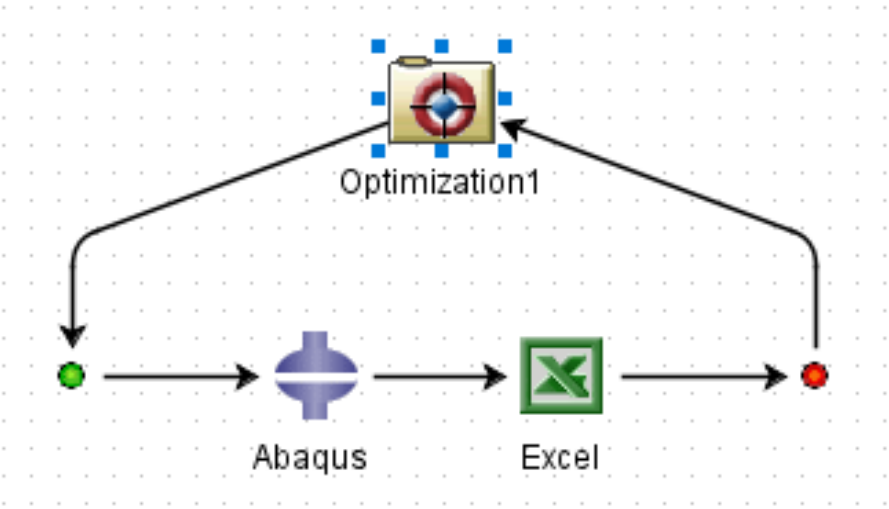


Figure 21: How to view the mesh and points

For the Abaqus component, ISight required two files: the .inp input, and the .odb output database files associated with the model. The orthotropic models require the .cae file associated with the model instead of the .inp file. The table of desired material properties that this research is focused on determining is retrievable from the .cae file, but not from the .inp file. From the "input" tab in the component design window, the various moduli and Poisson's ratios were identified as parameters to be sampled. In the "Extraction" tab of the component design window, it was specified that all field outputs from the .odb file would be extracted. From the "output" tab in the component design window the maximum x and y nodal displacements of all the previously identified nodes were chosen as outputs to sample.

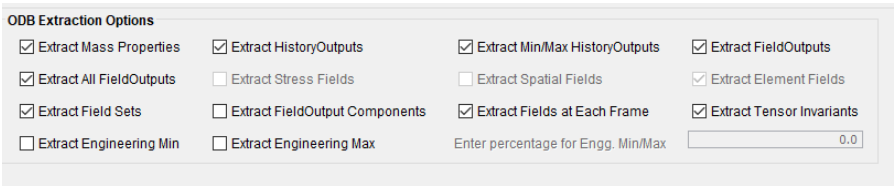


Figure 22: Execution settings for parameter extraction.

Before an Excel sheet could be imported into ISight, it had to be formatted.

The spreadsheet for each case contained the Abaqus node number, x and y nodal coordinates, x and y displacements, x and y strains, and the interpolated x and y strains from ARAMIS. Each cell containing data that was to be mapped in the data flow had to be named so that ISight recognized it as either an input or output. ISight recognizes cells with just information as inputs and cells containing formulas as outputs.

Once all the component parameters were identified, their flow from one another was mapped in the design gateway. The moduli and Poisson's ratios were identified as inputs to the Abaqus file and remained unmapped because they will be specified variables that are updated at the start of each iteration in the next section. The x and y components of the nodal displacements from the Abaqus model were mapped to import into their respective cells in the Excel spreadsheet. The Excel component outputs the strains or displacements from the current iteration based on what was examined.

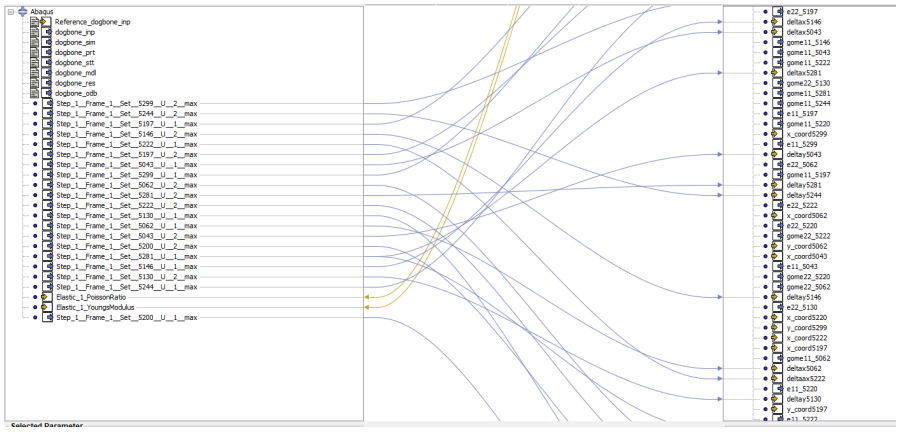


Figure 23: Flow of Abaqus outputs to Excel component

4.4.2 Optimization

With the parameter flow mapped, the optimization structure can then be defined. As mentioned above, the isotropic models were evaluated with two different optimization algorithms to determine which process could yield a more accurate

conclusion. In the optimization component editor, for each scheme, the default optimization technique options were used with the exception of updating the maximum number of iterations from 40 to 100.

In the variables tab, all of the parameters previously defined as inputs are available to be defined as variables. Only the moduli and Poisson's ratios of interest were selected to be updated. This is where the initial prediction for the variables and the upper and lower bound of allowed values for the variables are defined. ISight will not use values outside of these bounds as inputs. In Table 5 the bounds of the parameters that are updated in each iteration are displayed. These only apply to the isotropic experiments. The bounds of the orthotropic experiments were determined experimentally and will be discussed with the results.

Isotropic Variable Bounds		
Property	Upper	Lower
E	3000	1000
ν	0.1	0.5

Table 5: Isotropic variable bounds

In the tab furthest to the right of the optimization component editor exists the objective tab. The target values for the desired output parameters were defined in the "Target" column and their direction was set to "Target." The other specification instructs the software to find the combination of parameters that best satisfy the specified target values. The previously calculated interpolated strains or displacements were used as the target values to be optimized. Lastly, the constraints for the target values were specified in the "Constraints" tab. The optimization of the model is only as accurate or inaccurate as the constraints allow. Several different options for constraints were explored: the least and greatest displacements used as the lower and upper bounds respectively, the difference of the interpolated displacement/strain value and the calculated Abaqus value of each point, and the

average difference in displacement/strain of each component, as well as an arbitrarily determined value based on the precision of the Abaqus measurements versus the interpolated results. With all targets and constraints identified, the models were allowed to iterate until their termination accuracy had been satisfied or the maximum number of iterations had been reached.

Isotropic Constraint Formulations		
Formulation	Tensile Bar	Cruciform
$\lvert u_{i,exp} - u_{i,num} \rvert$	point dependent	point dependent
$\frac{1}{n}\sum_{i=1}^{n} \lvert u_{i,exp} - u_{i,num} \rvert$	$\pm$0.002872	$\pm$0.008
Maximum and Minimum u,v	-0.11, 0, 0.02, 0.21	0.008, 0.15
Arbitrary value	$\pm$0.02	$\pm$0.03

Table 6: Isotropic constraint values for ISight target values.

Orthotropic Constraint Formulations	
Formulation	Cruciform
$\lvert u_{i,exp} - u_{i,num} \rvert$	point dependent
$\frac{1}{n}\sum_{i=1}^{n} \lvert u_{i,exp} - u_{i,num} \rvert$	$\pm$0.008
Maximum and Minimum u,v	0.33, 0.01
Arbitrary value	$\pm$0.03

Table 7: Isotropic constraint values for ISight target values.

All isotropic models were started with the initial conditions of 0.2 and 1500 for Poisson's ratio and Young's modulus, respectively. The results of the simulations combining the different algorithms and constraint criteria will be discussed in the next section. The initial conditions of the orthotropic models will be discussed in the discussion in Chapter 6. Like the orthotropic bounds, their determination became part of the interpretation of results.

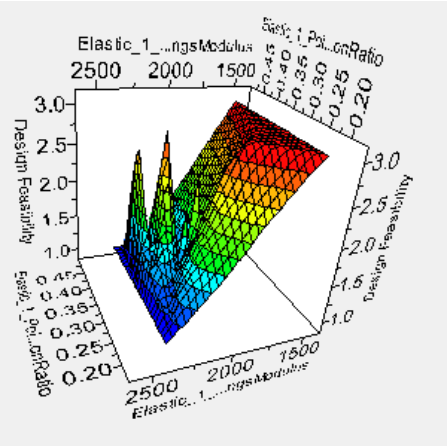
CHAPTER 5

Results

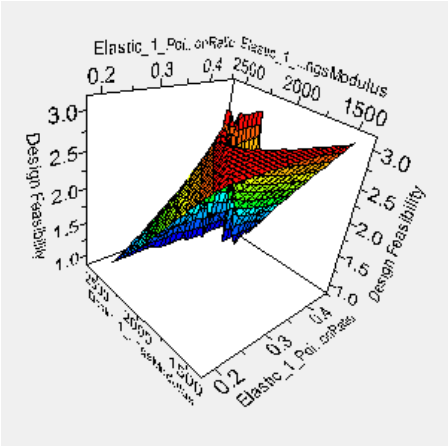
From the outset, this study sought to analyze the strain and displacement fields of the objects of interest. After analyzing the strains and displacements of the isotropic cases using the NLPQLP, it was determined that when using the displacements as the target, the Isight optimization gave more accurate results. For this reason, in subsequent experiments, only the displacement fields were used as optimization targets.

5.1 Isotropic Cases

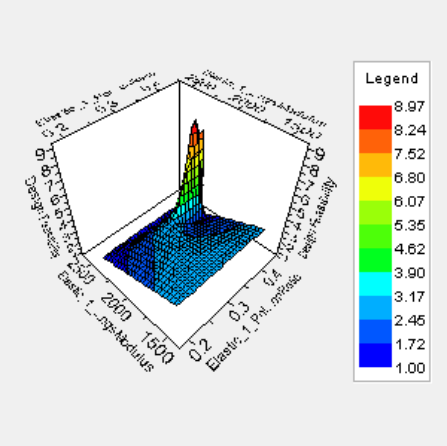
As previously stated, each model was optimized using two different schemes with four different constraint formulations. Only the results of the optimization using the displacements as targets will be discussed. When interpreting the results, it's important to recall that the speckle images for all of the tests were created from the nodal displacements of the Abaqus finite element model. Because ARAMIS tracks the displacement of the speckle pattern based on how fine or coarse the mesh used to create the images was, there is an inherent amount of error to what data is able to be collected. In the simulated stochastic pattern of the deformed images, the speckles do not change shape as they would on a physical sample.



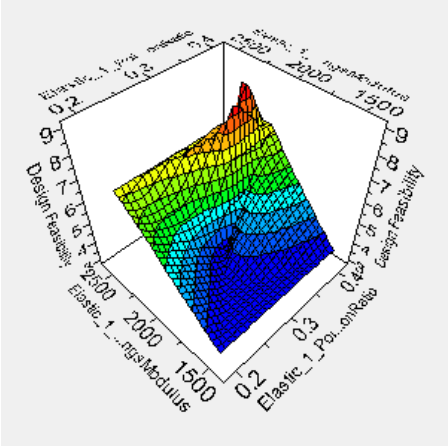
(a) $|exp - num|$



(b) Average difference



(c) Maximum and minimum u and v



(d) Arbitrary Value

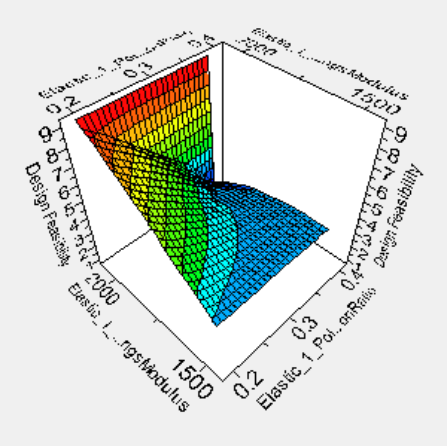
Figure 24: Surface plots of the design feasibility of parameters combinations for the isotropic tensile bar design feasibility using the pattern search scheme. ν, E, and the design feasibility are plotted on the x, y, and z axes, respectively.

Isotropic Tensile Bar Pattern Search Results				
Constrain Case	E	ν	E% error	ν% error
$\lvert u_{i,exp} - u_{i,num} \rvert$	2243 MPa	0.43	6.8%	16.2%
$\frac{1}{n}\sum_{i=1}^{n} \lvert u_{i,exp} - u_{i,num} \rvert$	2221 MPa	0.42	5.7%	13.5%
Max Min u,v	2203 MPa	0.41	4.9%	10.8%
Arbitrary value	2203 MPa	0.41	4.9%	10.8%

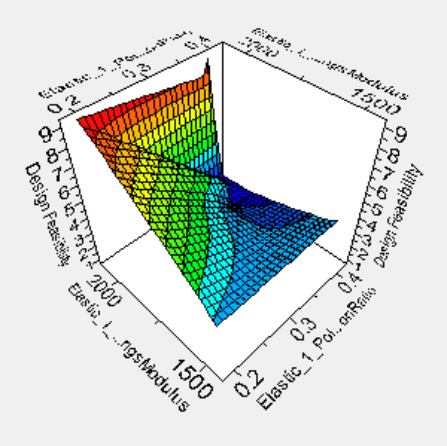
Table 8: Isotropic tensile bar optimization results using the pattern search scheme.

5.1.1 Tensile Bar

All of the pattern search optimizations were allowed to run for 100 iterations. Recall the initial conditions for the model were 0.2 and 1500 MPa The initial conditions were selected to be far enough away from the known material parameters to allow the algorithm to arrive at those values independent of the initial guess. Recall that the Poisson ratio and Young's modulus of Lexan are 0.37 and 2100 MPa. The maximum and minimum displacements and arbitrarily chosen constraint value produced the lowest error in use with the pattern search algorithm. In 24 surface plots display the feasibility of the combination of values as a function of E and ν. The higher the value of the design feasibility, the better the combination of E and ν satisfy the targets and constraints.



(a) Maximum and minimum u and v



(b) Arbitrary Value

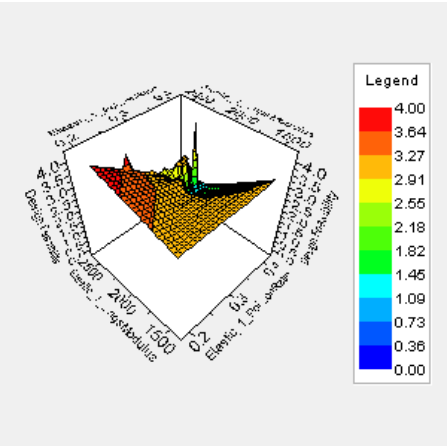
Figure 25: Surface plots of the design feasibility of parameters combinations for the isotropic tensile bar design feasibility using the NLPQLP scheme. ν, E, and the design feasibility are plotted on the x, y, and z axes, respectively.

Isotropic Tensile Bar NLPQLP Results				
Constrain Case	E	ν	E% error	ν% error
$\lvert u_{i,exp} - u_{i,num} \rvert$	N/A	N/A	N/A	N/A
$\frac{1}{n}\sum_{i=1}^{n} \lvert u_{i,exp} - u_{i,num} \rvert$	N/A	N/A	N/A	N/A
Max Min u,v	2106 MPa	0.39	0.2%	5.4%
Arbitrary value	2120 MPa	0.39	0.9%	5.4%

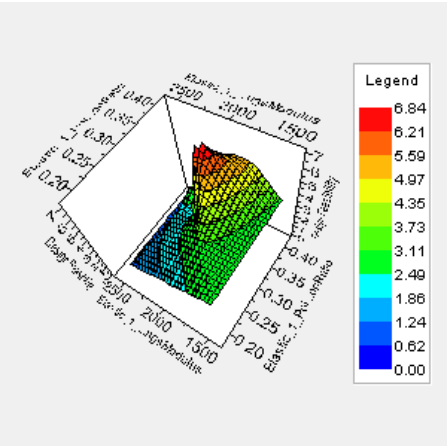
Table 9: Isotropic tensile bar optimization results using the NLPQLP scheme.

The NLPQLP optimizations used the same initial conditions and constraint formulations as the pattern search. The first two constraint formulations caused errors for the optimization process. No matter where the initial conditions were placed, the algorithm would determine the conditions that optimize the design parameters. The first two constraint formulations are orders of magnitude stricter than the latter two, indicating that this process does not work well if its area of interest is too confined. The maximum and minimum displacement and arbitrary value constraint cases paired with this algorithm returned the material property values with less than 1% error for Young's modulus and less than 6% error for Poisson's ratio.

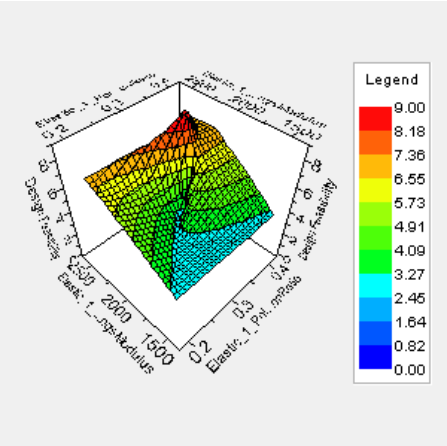
5.1.2 Cruciform



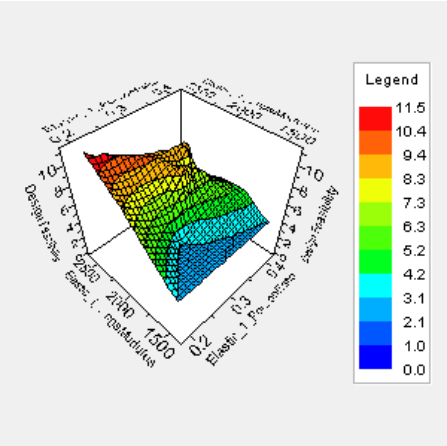
(a) $|exp - num|$



(b) Average difference

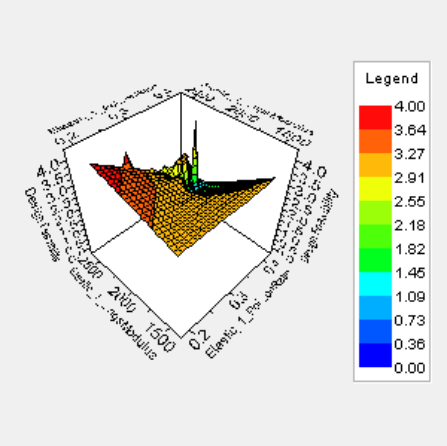


(c) Maximum and minimum u and v

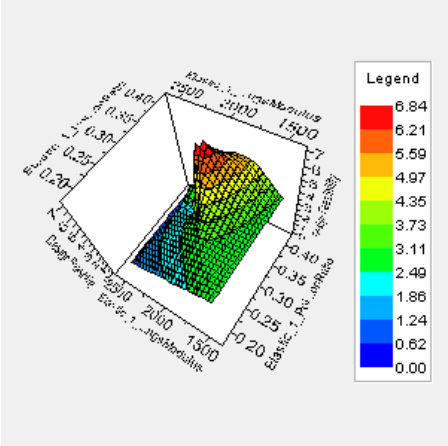


(d) Arbitrary Value

Figure 26: Surface plots of the design feasibility of parameters combinations for the isotropic cruciform design feasibility using the pattern search scheme. ν, E, and the design feasibility are plotted on the x, y, and z axes, respectively.



(a) Maximum and minimum u and v



(b) Average difference

Figure 27: Surface plots of the design feasibility of parameters combinations for the isotropic cruciform design feasibility using the NLPQLP scheme. ν, E, and the design feasibility are plotted on the x, y, and z axes, respectively.

Isotropic Cruciform Pattern Search Results				
Constraint Case	E	ν	E %error	ν %error
$\lvert u_{i,exp} - u_{i,num} \rvert$	2291 MPa	0.3228	9.12%	12.7%
$\frac{1}{n}\sum_{i=1}^{n} \lvert u_{i,exp} - u_{i,num} \rvert$	2103 MPa	0.3712	0.14%	0.32%
Max Min u,v	2123 MPa	0.3664	1.13%	0.98%
Arbitrary value	2104	0.3718	0.21%	0.47%

Table 10: Isotropic cruciform optimization results using pattern search scheme.

The biaxial isotropic pattern search optimizations were performed using the same initial conditions, bounds, and constraint formulations as the uniaxial experiments. The goal was a combination of optimization process and constraint formulation is to return the same material properties as in the uniaxial tests. Using the average difference between the analytical model and the experimental results as the target constraints for the simulation returned the most accurate parameter estimations with 0.14% error for Young's modulus and 0.32% error for Poisson's ratio. The arbitrary value and max-min constraints returned similarly accurate values to the average difference case. The NLPQLP scheme encountered errors with the $|exp - num|$ and arbitrary value constraint cases and couldn't complete

Isotropic Cruciform NLPQLP Results				
Constraint Case	E	ν	E %error	ν %error
$\frac{1}{n}\sum_{i=1}^{n}\lvert u_{i,exp} - u_{i,num}\rvert$	2019 MPa	0.37952	3.86%	6.81%
Max Min u,v	1952 MPa	0.425	7.1%	14.9%

Table 11: Isotropic cruciform optimization results using NLPQLP scheme.

the optimization. The percent error of the values returned from the NLPQLP scheme were much higher than the pattern search and the choice was made to not use it to evaluate the orthotropic cases.

5.2 Orthotropic Cases

The DIC images for the orthotropic cruciform sample were created using a different set of parameters to better simulate the imperfect nature of applying a stochastic pattern with spray paint. Most notable, blurred edges were applied to the outside of the speckles to better resemble a sprayed pattern.

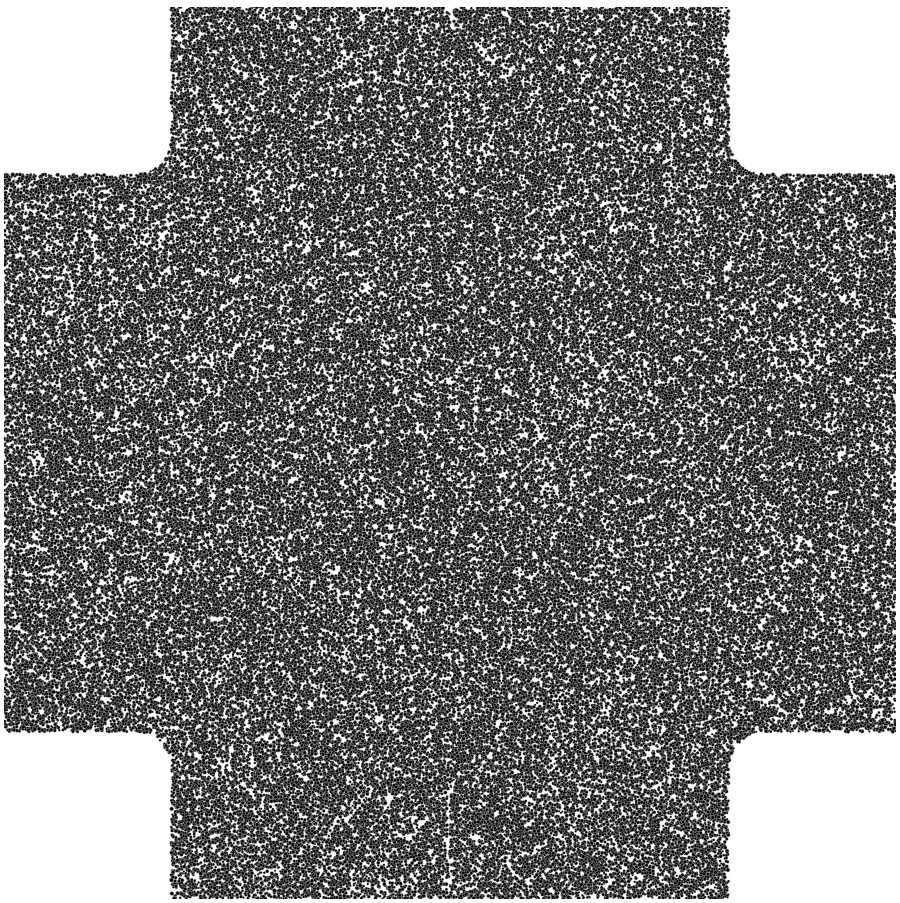
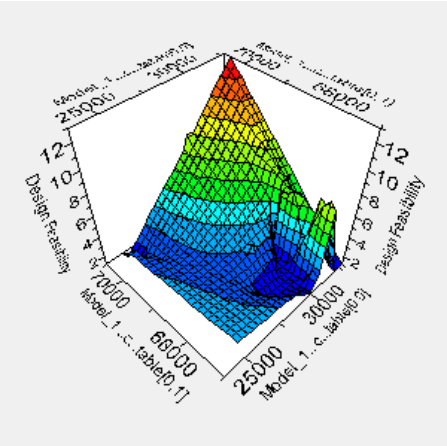


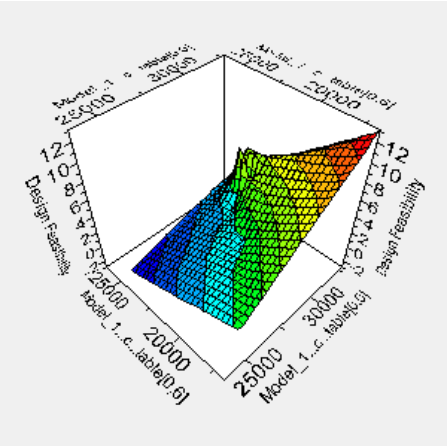
Figure 28: New speckle pattern

To begin evaluation of the orthotropic cruciform sample, the pattern search algorithm paired with the maximum and minimum displacements as target constraints were investigated. Three trials of this case were run, updating the variable bounds each trial, to refine the result based on the design feasibility plots of variable pairings. The first trial had variable bounds of $\pm 10,000$, the second $\pm 5,000$, and the third with bounds chosen based on regions of high design feasibility seen in the surface plots of the first and second trials. In the second and third trials, the bounds were chosen so the returned estimate of the variable was still contained within the updated bounds. The initial guess of the third trial was set to the center of the range of values for each variable. If the best estimate of the parameters were truly the values determined by the previous trial, the optimization would

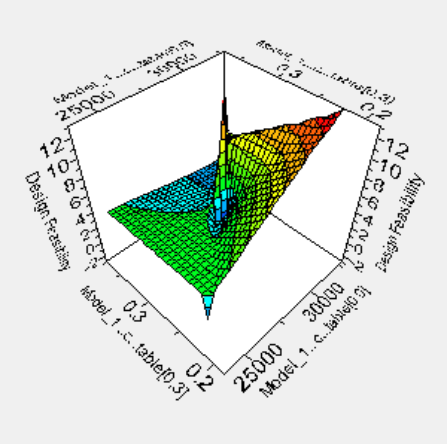
converge to those values as they also exist inside the bounds. A single trial using the NLPQLP scheme was conducted using the same bounds and initial conditions as the third trial and returned values of similar accuracy.



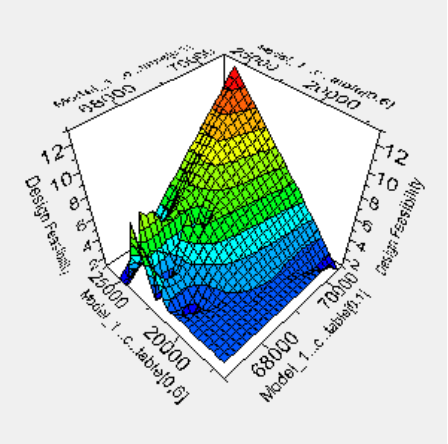
(a) design feasibility E_1 vs E_2



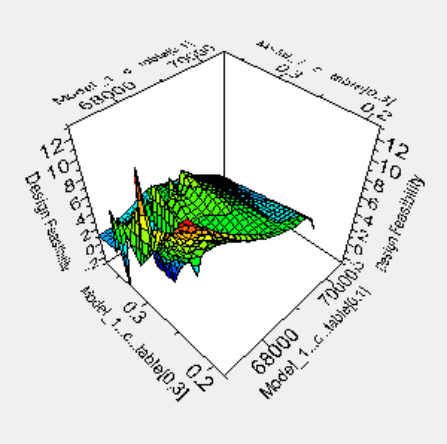
(b) design feasibility E_1 vs G_{12}



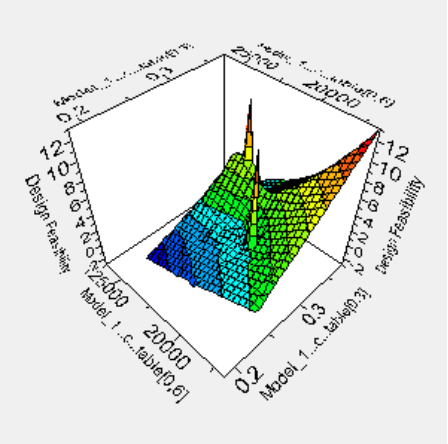
(c) design feasibility E_1 vs ν_{12}



(d) design feasibility E_2 vs G_{12}

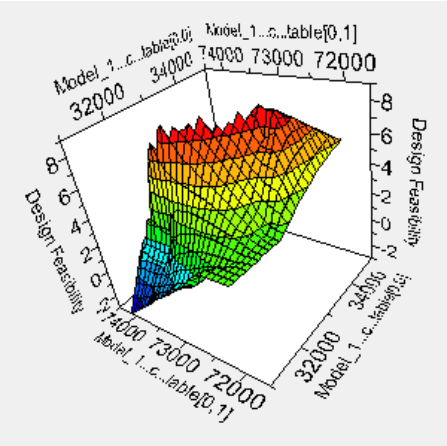


(e) design feasibility E_2 vs ν_{12}

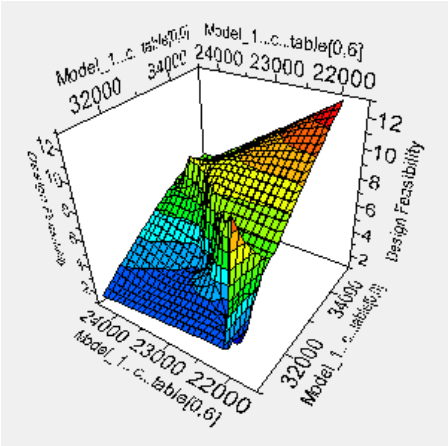


(f) design feasibility ν_{12} vs G_{12}

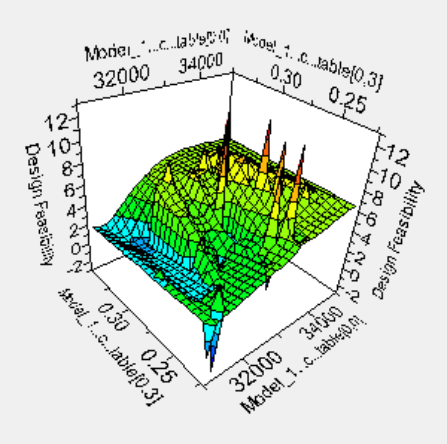
Figure 29: Surface plots of the design feasibility of parameter combinations for the first orthotropic cruciform trial using the pattern search scheme and max min constraints.



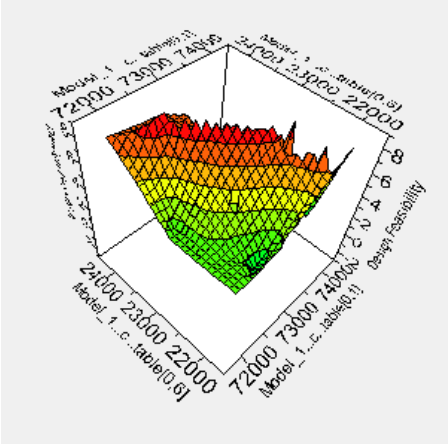
(a) design feasibility E_1 vs E_2



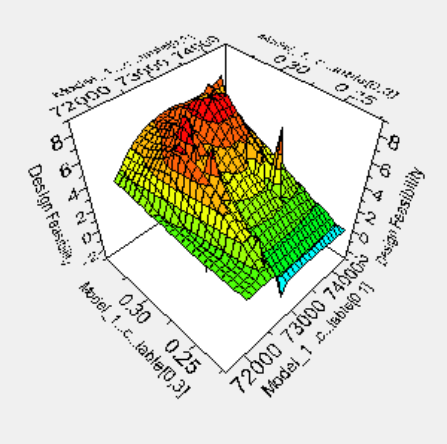
(b) design feasibility E_1 vs G_{12}



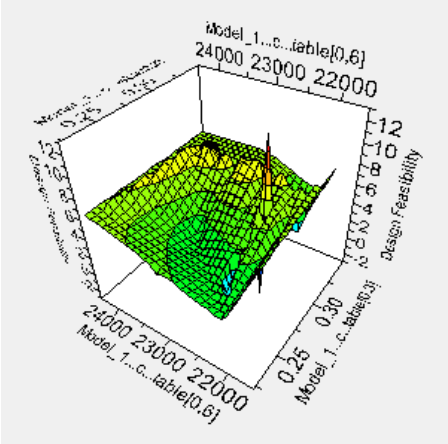
(c) design feasibility E_1 vs ν_{12}



(d) design feasibility E_2 vs G_{12}

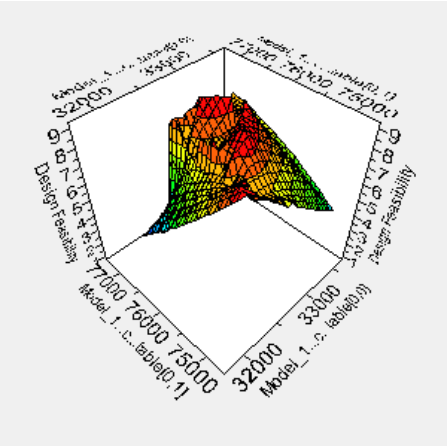


(e) design feasibility E_2 vs ν_{12}

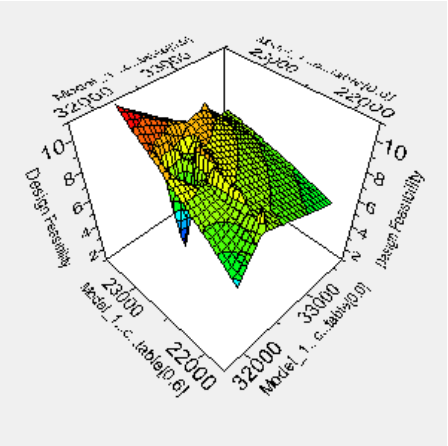


(f) design feasibility ν_{12} vs G_{12}

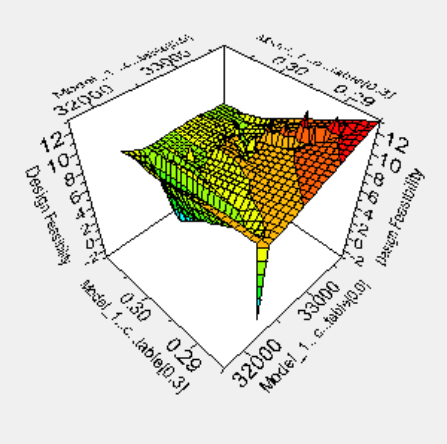
Figure 30: Surface plots of the design feasibility of parameter combinations for the first orthotropic cruciform trial using the pattern search scheme and max min constraints.



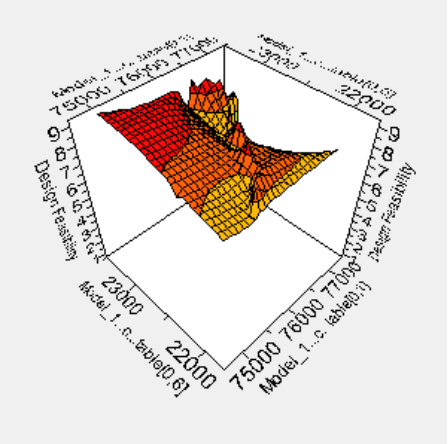
(a) design feasibility E_1 vs E_2



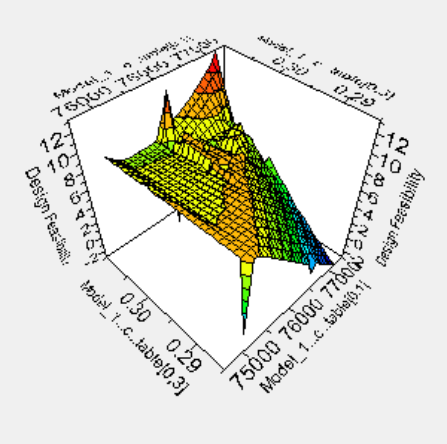
(b) design feasibility E_1 vs G_{12}



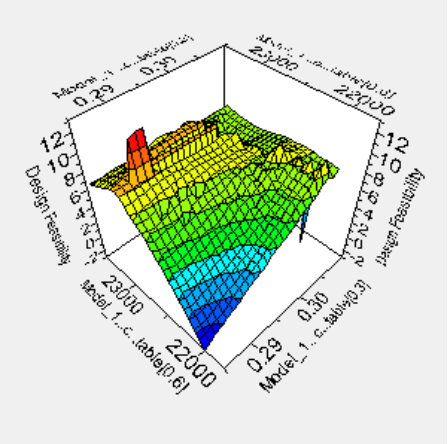
(c) design feasibility E_1 vs ν_{12}



(d) design feasibility E_2 vs G_{12}



(e) design feasibility E_2 vs ν_{12}



(f) design feasibility ν_{12} vs G_{12}

Figure 31: Surface plots of the design feasibility of parameter combinations for the first orthotropic cruciform trial using the pattern search scheme and max min constraints.

Orthotropic Cruciform Pattern Search First Trial			
	Literature Value	Experimental Value	% error
E_1	32500 MPa	31012 MPa	4.57%
E_2	77000 MPa	68325 MPa	11.26%
G_{12}	22400 MPa	22053 MPa	1.55%
ν_{12}	0.297	0.33472	13.5%

Table 12: Orthotropic cruciform pattern search first trial results.

Orthotropic Cruciform Pattern Search Second Trial			
	Literature Value	Experimental Value	% error
E_1	32500 MPa	32176 MPa	0.99%
E_2	77000 MPa	73763 MPa	4.20%
G_{12}	22400 MPa	22022 MPa	1.68%
ν_{12}	0.297	0.31475	5.97%

Table 13: Orthotropic cruciform pattern search second trial results.

Orthotropic Cruciform Pattern Search Third Trial			
	Literature Value	Experimental Value	% error
E_1	32500 MPa	32526 MPa	0.08%
E_2	77000 MPa	75481 MPa	1.97%
G_{12}	22400 MPa	22001 MPa	1.87%
ν_{12}	0.297	0.3083	3.8%

Table 14: Orthotropic cruciform pattern search third trial results.

Orthotropic Cruciform NLPQLP			
	Literature Value	Experimental Value	% error
E_1	32500 MPa	32500 MPa	0%
E_2	77000 MPa	75575 MPa	1.85%
G_{12}	22400 MPa	2300 MPa	2.5%
ν_{12}	0.297	0.30703	3.37%

Table 15: Orthotropic cruciform pattern search third trial results.

CHAPTER 6

Discussion and Conclusions

6.1 Isotropic Cases

6.1.1 Tensile Bar

The isotropic tensile bar experiments were able to return a minimum error of 0.2% and 5.4% for Young's modulus and Poisson's ratio, respectively, by using the NLPQLP algorithm and constraining the target values with the maximum and minimum displacements of each node. Since the first two constraint formulations (see Table 9) did not yield any results, it is reasonable to assume that those constraint formulations confine the model too tightly for the NLPQLP algorithm. The pattern search returned values for Young's modulus and Poisson's ratio that were precise relative to one another, but were not accurate in returning the properties of Lexan. To support the claim that the first two constraint cases confine the model too tightly, Figure 24 shows that the design feasibility for the same two constraint cases never reaches a value higher than 3 where the other two cases return values between 8 and 9. It is possible that if it were allowed more time to optimize, the pattern search paired with the tighter constraint cases may have arrived at the correct parameters, but, to maintain uniformity, this possibility was not explored. The isotropic tensile bar was the simplest case examined in this study. The optimization resulted in values of less than 6% error utilizing the NLPQLP optimization algorithm was taken to be acceptable and will be used in examining the cruciform case.

6.1.2 Cruciform

The NLPQLP algorithm failed to return accurate results for any of the constraint cases for the isotropic cruciform experiments. The cruciform experiments

were able to return a minimum error of 0.14% and 0.32% for Young's modulus and Poisson's ratio, respectively, by combining the pattern search and constraining the target values with the average difference constraint case. The max-min displacements and arbitrary value constraints also returned values with similarly low error percentages. Like to the tensile bar, the design feasibility of the $|exp - num|$ constraint case never achieves a value greater than 3, where all other constraint formulations reach values of 9 (see Figure 26), Having approximated Lexan's material properties of less than 2% error in a more complex loading scenario. The model was accepted and the orthotropic cruciform sample was examined.

6.2 Othotropic Cases

The max-min target constraints were used in the orthotropic experiments because they were consistently the most reliable in terms of allowing the algorithm to run without error. In Figure 29 and Figure 30 there are very clearly ranges of values that are within the bounds of the optimization that are not explored by the algorithm. These regions are characterized by steep gradients at the edges of the plots and regions that look like they are forming a peak. In Figure 31, we see a realization of these peaks, or regions of consistently high design feasibility. The development of the peak when plotting E_1 vs. E_2 displays this phenomena the best. Updating the bounds between which the variable values could exist in accordance with the design feasibility (see Figures 29, 30, and 31) returned values for the elastic constants of the composite with less than 2% error when compared to the classically determined values. This process of bound determination relied on the interpretation of the design feasibility plots and not the a priori knowledge of the material's elastic constants. This fact verifies that it is a suitable method of testing an unknown orthotropic material.

6.3 Future Research

The most intuitive next step for this research is verifying the orthotropic results with loading that is within the elastic limits of the composite material. This relies on the ability of the DIC software to accurately report displacements that are orders of magnitude less than the current model. The next topic that should be investigated is verifying this technique with displacements obtained from DIC on a physical sample of the isotropic and orthotropic materials in this research. If the values for these materials can be returned from an analysis of a physical sample with a similar degree of accuracy, it stands to reason that this method could be applied to more complex woven structures and new alloys that emerge as candidates for commercial applications. While the isotropic results came to an acceptable conclusion, the behavior of both algorithms in the cruciform case require further study. These cases should be reexamined taking care to properly set the length scale for the simulated DIC images. In the same revised experiment, the parameters used to generate the DIC images in the orthotropic experiments should be used for the isotropic cases. The DIC software was able to track their motion more accurately and with less data loss.

LIST OF REFERENCES

[1] D. L. Logan, *A First Course in the Finite Element Method*. 200 First Stamford Place, Suite 400 Stamford, CT 06902: Cengage Learning, 2012.

[2] J. Górszczyk, K. Malicki, and T. Zych, "Application of digital image correlation (dic) method for road material testing," *Materials*, vol. 12, no. 15, p. 2349, 2019.

[3] R. Hooke and T. A. Jeeves, "" direct search" solution of numerical and statistical problems," *J. ACM*, vol. 8, no. 2, p. 212–229, Apr. 1961. [Online]. Available: https://doi.org/10.1145/321062.321069

[4] D. Roylance, "Mechanical properties of materials," Massachusetts Institute of Technology, 2008. [Online]. Available: http://web.mit.edu/course/3/3.225/book.pdf

[5] "Chapter 4 - material behavior-linear elastic solids," in *Elasticity (Third Edition)*, third edition ed., M. H. Sadd, Ed. Boston: Academic Press, 2014, pp. 505 – 529. [Online]. Available: http://www.sciencedirect.com/science/article/pii/B9780124081369000167

[6] S. Cooreman, D. Lecompte, H. Sol, J. Vantomme, and D. Debruyne, "Identification of mechanical material behavior through inverse modeling and dic," *Experimental Mechanics*, vol. 48, no. 4, pp. 421 – 433, 2008. [Online]. Available: https://doi.org/10.1007/s11340-007-9094-0

[7] M. Sutton, W. Wolters, W. Peters, W. Ranson, and S. McNeill, "Determination of displacements using an improved digital correlation method," *Image and Vision Computing*, vol. 1, no. 3, pp. 133 – 139, 1983. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0262885683900641

[8] M. Sutton, W. Wolters, W. Peters, W. Ranson, and S. McNeill, "Digital image correlation using newton-raphson method of partial differential correction," *Experimental Mechanics*, vol. 29, no. 3, pp. 261 – 267, 1989. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0262885683900641

[9] B. Mobasher, "5 - textile fiber composites: Testing and mechanical behavior," in *Textile Fibre Composites in Civil Engineering*, T. Triantafillou, Ed. Woodhead Publishing, 2016, pp. 101 – 150. [Online]. Available: http://www.sciencedirect.com/science/article/pii/B9781782424468000069

[10] S. Baghernezhad, M. Ghane, and M. Moezzi, "Strain monitoring in woven fabrics with locally induced mass irregularities using an image based method," *Fibres and Textiles in Eastern Europe*, vol. 24, pp. 73–80, 01 2016.

[11] F. Tariq, M. Z. Siddqui, N. Naz, and M. F. Ahmed, "Application of digital image correlation technique in determination of elastic constants of materials," 2012.

[12] T. Dinh, A. Rezaei, L. D. Laet], M. Mollaert, D. V. Hemelrijck], and W. V. Paepegem], "A new elasto-plastic material model for coated fabric," *Engineering Structures*, vol. 71, pp. 222 – 233, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0141029614002405

[13] D. Lecompte, A. Smits, H. Sol, J. Vantomme, and D. Van Hemelrijck, "Mixed numerical–experimental technique for orthotropic parameter identification using biaxial tensile tests on cruciform specimens," *International Journal of Solids and Structures*, vol. 44, no. 5, pp. 1643 – 1656, 2007. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0020768306002848

[14] M. A. S. a. Hubert Schreier, Jean-José Orteu, *Image Correlation for Shape, Motion and Deformation Measurements: Basic Concepts,Theory and Applications*. Springer US, 2009. [Online]. Available: http://gen.lib.rus.ec/book/index.php?md5=85c8b67b9f567265060c2cccf9b75c61

[15] K. Schittkowski, "Nlpqlp: A fortran implementation of a sequential quadratic programming algorithm with distributed and non-monotone line search–user's guide," *Report, Department of Computer Science, University of Bayreuth*, 2006.

[16] J. Nocedal and S. Wright, *Numerical Optimization: Springer Series in Operations Research and Financial Engineering*. Springer, 2006.

[17] K. Schittkowski, "Nlpqlp: A new fortran implementation of a sequential quadratic programming algorithm for parallel computing," *Report, Department of Mathematics, University of Bayreuth*, 2001.

[18] *GOM Correlate Professional: V8 SR1 Manual Basic*.

APPENDIX A

Displacement Interpolation Function

```
function nodal_displacements_from_DIC_data
%
%  1. reads finite element nodal coordinates from file
%        'nodal_coordinates.csv'
%  2. reads DIC displacment fields at triangular grid points from
%         file 'grid_ displacment.csv'
%  3. interpolates the grid point displacements to node locations
%  4. writes the interpolated nodal displacements to file
%        'nodal_displacements.csv'
%
clc; clear all; close all; format compact
%
% Read nodal coordinates
%
disp('Finite element nodal coordinates')
nodal_coordinates=csvread('nodal_coordinates.csv');
disp(nodal_coordinates)
nodal_coordinate_size=size(nodal_coordinates);
num_nodes=nodal_coordinate_size(1);
%
% Read DIC grid data
%
data=csvread('grid_displacements.csv');
data_size=size(data);
numpoints=data_size(1);
grid_point=data(:,1);
xgrid=data(:,2);
ygrid=data(:,3);
u=data(:,5);
v=data(:,6);
%
plot(xgrid,ygrid,'r.')
xlabel('x')
ylabel('y')
figure
plot(xgrid,ygrid,'ro')
disp('Grid points in range 0<=(x,y)<=1')
for i=1:numpoints
    if 0<=xgrid(i)&& xgrid(i)<=1 && 0<=ygrid(i) && ygrid(i)<=1
        disp([xgrid(i) ygrid(i) u(i) v(i)])
        text(xgrid(i),ygrid(i),num2str(i))
    end
end
%
% zoom in to region near (0,0)
%
axis([0 1 0 1])
%
% Interpolate to desired nodal locations
%
nodal_displacements=[];
for node_point=1:num_nodes
    node_number=nodal_coordinates(node_point,1);
    x=nodal_coordinates(node_point,2);
    y=nodal_coordinates(node_point,3);
    disp('--------------------------')
    disp(['(x_node,y_node)= ' num2str([x,y])])
```

```
    hold on
    plot(x,y,'kx')
    xlabel('x')
    ylabel('y')
    %
    % find 3 nearest grid points
    %
    distance=sqrt((x-xgrid).^2+(y-ygrid).^2);
    [~,igrid]=sort(distance);
    pt1=igrid(1); pt2=igrid(2); pt3=igrid(3);
    disp(['Nearest grid points are:     ' ...
        num2str([pt1 pt2 pt3])])
    disp([' u values are:           ' num2str([u(pt1) ...
        u(pt2) u(pt3)])])
    disp([' v values are:           ' num2str([v(pt1) ...
        v(pt2) v(pt3)])])
    %
    x1=xgrid(pt1);
    y1=ygrid(pt1);
    x2=xgrid(pt2);
    y2=ygrid(pt2);
    x3=xgrid(pt3);
    y3=ygrid(pt3);
    %
    alpha_1=x2*y3-x3*y2;
    alpha_2=x3*y1-x1*y3;
    alpha_3=x1*y2-x2*y1;
    beta_1=y2-y3;
    beta_2=y3-y1;
    beta_3=y1-y2;
    gamma_1=x3-x2;
    gamma_2=x1-x3;
    gamma_3=x2-x1;
    A=det([1 x1 y1;1 x2 y2;1 x3 y3])/2;
    %
    N1=(alpha_1+beta_1*x+gamma_1*y)/(2*A);
    N2=(alpha_2+beta_2*x+gamma_2*y)/(2*A);
    N3=(alpha_3+beta_3*x+gamma_3*y)/(2*A);
    %
    node_u=N1*u(pt1)+N2*u(pt2)+N3*u(pt3);
    node_v=N1*v(pt1)+N2*v(pt2)+N3*v(pt3);
    %
    disp([' Interpolated displacementss are:  ' ...
        num2str(node_u) ' ' num2str(node_v)])
    nodal_displacements=[nodal_displacements; ...
        node_number, x, y, node_u, node_v ];

end
nodal_displacements
%
% Display and save interpolated nodal displacements
%
csvwrite('nodal_displacements.csv',nodal_displacements)
```

APPENDIX B

Stochastic Pattern Generator

```
function simulated_speckle_generator
%
clear all; close all; clc; format compact
%
global num_elems num_nodes elems nodes u_nodes v_nodes
%
% create speckle pattern
%
tic
%
% USER INPUT - turn on/off blurred edges
%
blurred_edge=1;
%
% USER INPUT - set desired resolution here:
%
% N=500;         % image size = (N+1) x (N+1)
% Ns=10000;        % number of speckles in frame
% N=1000;      % image size = (N+1) x (N+1)
% Ns=25000; % 17000;         % number of speckles in frame
N=4000;          % image size = (N+1) x (N+1)
Ns=150000;       % 250000     % number of speckles in frame
%
N_center=N/2+1;  % pixel number at center of image
%
% USER INPUT - set number steps & range of pixel sizes
%
Nstep=10;   % number of load steps (Number of images = Nstep+1)
rmin=5; rmax=10;  % speckle size range in pixels4000
%
% USER INPUT - set if overlap is allowed when finding ...
%                   locations of speckles
%
allow_overlap=0; % turn on/off overlapped speckles
if allow_overlap==1
    xp=N*rand(1,Ns);
    yp=N*rand(1,Ns);
else
    min_space=3; % 5; % minimum spacing between speckle centers
    [xp,yp]=getpoints(N,Ns,min_space);
end
disp([num2str(Ns) ' speckles generated'])
toc
%
% generate image with undeformed speckles
%
rp=zeros(1,Ns);
for i=1:Ns
    %     if i<Ns/2
    %         rp(i)=rmin;
    %     else
    %         rp(i)=rmax;
    %     end
    rp(i)=rmax-((i-1)/(Ns-1))*(rmax-rmin); % random speckle
    %                                       size between rmin and rmax
end
image_size_x=N+1;
```

```
image_size_y=N+1;
a=uint8(255*ones(image_size_y,image_size_x));
% 6/21
gray_max=100;
gray_level=gray_max*rand(1,Ns);
% gray_level=zeros(1,Ns);
%
on_sample=true(1,Ns);
[a,cmap]=create_image(Ns,xp,yp,rp,image_size_x,image_size_y, ...
    gray_max,gray_level,on_sample,blurred_edge);
gray_pixels=sum(sum(a<255));
num_pixels=(N+1)*(N+1);
af=gray_pixels/num_pixels; % fraction of speckles
%                            (non-white pixels)
disp(['area fraction of speckles = ' num2str(af)])
toc
title('undeformed speckles')
plotname=['undeformed_speckles_af=_' num2str(af) '.png'];
imwrite(a,cmap,plotname,'png')
disp(['initial_speckle pattern created'])
toc
% pause
%
% USER INPUT - set scale (match fea coordinates)
%    scale for dogbone 30mm = N pixels
%
scale=30/N;  % (= 30 mm/N pixels)
for i=1:Ns
    x_mm(i)=(xp(i)-N_center)*scale;
    y_mm(i)=(yp(i)-N_center)*scale;
end
disp('undeformed scaling complete')
toc
%
get_fea(Nstep+1)
disp('fea data read from data file')
toc
%
% find element number for each speckle
%
% gray_level=255*ones(1,Ns);
% first - find centroid of each element
for ielem=1:num_elems
    elem_node=elems(ielem,2:5);
    xv=[nodes(elem_node(1),2) nodes(elem_node(2),2) ...
        nodes(elem_node(3),2) nodes(elem_node(4),2)] ...
        /scale+N_center;
    yv=[nodes(elem_node(1),3) nodes(elem_node(2),3) ...
        nodes(elem_node(3),3) nodes(elem_node(4),3)] ...
        /scale+N_center;
    x_centroid(ielem)=mean(xv);
    y_centroid(ielem)=mean(yv);
end
% second - for each speckle, find nearest element
for ispeckle=1:Ns
    clear dist
    dist=sqrt((xp(ispeckle)-x_centroid).^2+ ...
        (yp(ispeckle)-y_centroid).^2);
    [~,elem_number(ispeckle)]=min(dist);
end
% third - check to see if speckle is on the sample and if not,
%          set gray_level to 255 (white)
on_sample=true(1,Ns);
for ispeckle=1:Ns
    elem_node=elems(elem_number(ispeckle),2:5);
    xv=[nodes(elem_node(1),2) nodes(elem_node(2),2) ...
        nodes(elem_node(3),2) nodes(elem_node(4),2)] ...
        /scale+N_center;
```

```
    yv=[nodes(elem_node(1),3) nodes(elem_node(2),3) ...
        nodes(elem_node(3),3) nodes(elem_node(4),3)] ...
        /scale+N_center;
    in = inpolygon(xp(ispeckle),yp(ispeckle),xv,yv);
    if in~=1
%
        gray_level(ispeckle)=255;
        on_sample(ispeckle)=0;
    end
end
%
size(on_sample);
on_sample;
%
%
disp('elements found for each speckle')
toc
%
% create image for test sample
%
figure
image_size_x=N+1;
image_size_y=N+1;
a=uint8(255*ones(image_size_y,image_size_x));
[a,cmap]=create_image(Ns,xp,yp,rp,image_size_x,image_size_y, ...
    gray_max,gray_level,on_sample,blurred_edge);
title(['Deformed speckles - step 0'])
plotname='deformed_speckles-step_0.png';
imwrite(a,cmap,plotname,'png')
disp('Deformed speckles, step 0 created')
toc
% initial video
v = VideoWriter('dogbone.avi');
v.FrameRate=1;
open(v);
frame = getframe(gcf);
writeVideo(v,frame);
%
% create images for each load step (starting with step 2)
%
%   find (s,t)coordinate for each speckle

for ispeckle=1:Ns
    pointxy=[x_mm(ispeckle) y_mm(ispeckle)];
%     pointxy=[xp(ispeckle) yp(ispeckle)];
    if on_sample(ispeckle)==1
        elem_node=elems(elem_number(ispeckle),2:5);
        x_nodes=[nodes(elem_node(1),2) nodes(elem_node(2),2) ...
            nodes(elem_node(3),2) nodes(elem_node(4),2)];
        y_nodes=[nodes(elem_node(1),3) nodes(elem_node(2),3) ...
            nodes(elem_node(3),3) nodes(elem_node(4),3)];
        st=find_s_t(x_nodes,y_nodes,pointxy);
        st_speckle(ispeckle,:)=[st(1) st(2)];
    end
end
disp('(s,t) coordinates found for each speckle')
toc
%
%   determine displacement of each speckle and update position
%
for istep=2:Nstep+1
    for ispeckle=1:Ns
%         pointxy=[x_mm(ispeckle) y_mm(ispeckle)];
        if on_sample(ispeckle)==1
            elem_node=elems(elem_number(ispeckle),2:5);
            x_nodes=[nodes(elem_node(1),2) nodes(elem_node(2),2) ...
                nodes(elem_node(3),2) nodes(elem_node(4),2)];
            y_nodes=[nodes(elem_node(1),3) nodes(elem_node(2),3) ...
                nodes(elem_node(3),3) nodes(elem_node(4),3)];
```

```
            %
            s=st_speckle(ispeckle,1);
            t=st_speckle(ispeckle,2);
            un=[u_nodes(elem_node(1),istep) ...
                u_nodes(elem_node(2),istep) ...
                u_nodes(elem_node(3),istep) ...
                u_nodes(elem_node(4),istep)];
            vn=[v_nodes(elem_node(1),istep) ...
                v_nodes(elem_node(2),istep) ...
                v_nodes(elem_node(3),istep) ...
                v_nodes(elem_node(4),istep)];
            u_mm=((1-s)*(1-t)*un(1)+(1+s)*(1-t)*un(2)+ ...
                (1+s)*(1+t)*un(3)+(1-s)*(1+t)*un(4))/4;
            v_mm=((1-s)*(1-t)*vn(1)+(1+s)*(1-t)*vn(2)+ ...
                (1+s)*(1+t)*vn(3)+(1-s)*(1+t)*vn(4))/4;
            x_mm_new=x_mm(ispeckle)+u_mm;
            y_mm_new=y_mm(ispeckle)+v_mm;
            xp(ispeckle)=x_mm_new/scale+N_center;
            yp(ispeckle)=y_mm_new/scale+N_center;
        end
        %
    end
    %
    % create image for each load step
    %
    figure
    a=uint8(255*ones(image_size_y,image_size_x));
    [a,cmap]=create_image(Ns,xp,yp,rp,...
        image_size_x,image_size_y, ...
        gray_max,gray_level,on_sample,blurred_edge);
    title(['Deformed speckles - step  ' num2str(istep-1)])
    plotname=[['deformed_speckles-step_' ...
        num2str(istep-1) '.png']];
    imwrite(a,cmap,plotname,'png')
    disp(['Deformed speckles, step ' ...
        num2str(istep-1) ' created'])
    toc
    % save movie frame
    frame = getframe(gcf);
    writeVideo(v,frame);
    %
end
%
drawnow
% end
[s,Fs] = audioread('blip.wav');
sound(s,Fs);
close(v)
toc
% ------------------------------------------------------------
%
function [xp,yp]=getpoints(N,Ns,min_space)
%
%  - N gives image size = (N+1) x (N+1) in pixels
%  - Ns is desired number of speckles
%  - min_space is the minimum spacing between pixel centers
%  - xp and yp are the (x,y) coordinates of generated speckles
%
% ref: https://www.mathworks.com/matlabcentral/answers/158357-...
%      create-random-points-in-a-rectangular-domain-but-...
%      with-minimum-separation-distance
x = N*rand(1, 10000000);
y = N*rand(1, 10000000);
% x = N*rand(1, 100000);
% y = N*rand(1, 100000);
minAllowableDistance = min_space;
numberOfPoints = 10000000;
```

```
% numberOfPoints = 100000;
% Initialize first point.
keeperX = x(1);
keeperY = y(1);
% Try dropping down more points.
counter = 2;
for k = 2 : numberOfPoints
    % Get a trial point.
    thisX = x(k);
    thisY = y(k);
    % See how far is is away from existing keeper points.
    distances = sqrt((thisX-keeperX).^2 + (thisY - keeperY).^2);
    minDistance = min(distances);
    if minDistance >= minAllowableDistance
        keeperX(counter) = thisX;
        keeperY(counter) = thisY;
        counter = counter + 1;
    end
    if counter==Ns+1
        break
    end
end
% disp([num2str(counter) ' points found'])
xp=keeperX;
yp=keeperY;
%
% ------------------------------------------------------------
%
function [a,cmap]=create_image(Ns,xp,yp,rp,image_size_x,...
    image_size_y,gray_max,gray_level,on_sample,blurred_edge)
%
a=uint8(255*ones(image_size_y,image_size_x));
for k=1:Ns
    if on_sample(k)==1
        i_mid=round(yp(k));
        j_mid=round(xp(k));
        r_search=2*round(rp(k));
        for i=max(1,i_mid-r_search):...
                min(i_mid+r_search,image_size_y)
            for j=max(1,j_mid-r_search): ...
                    min(j_mid+r_search,image_size_x)
                dist=sqrt((j-xp(k))^2+(i-yp(k))^2);
                if blurred_edge==1
                    if dist<=.7*rp(k)
                        a(i,j)=gray_level(k);
                    elseif dist<=.85*rp(k)
                        a(i,j)=gray_max-...
                            0.8*(gray_max-gray_level(k));
                    elseif dist<=rp(k)
                        a(i,j)=gray_max-...
                            0.4*(gray_max-gray_level(k));
                    end
                else
                    if dist<=rp(k)
                        a(i,j)=gray_level(k);
                    end
                end
            end
        end
    end
end
%
image(a)
cmap=colormap(gray(256));
axis equal
drawnow
%
% ------------------------------------------------------------
```

```
%
function get_fea(nsteps)
%
% read finite element results
%
global num_elems num_nodes elems nodes u_nodes v_nodes
%
% read finite element model parameters
elems=load('elems.dat');
elems_size=size(elems);
num_elems=elems_size(1);
nodes=load('nodes.dat');
nodes_size=size(nodes);
num_nodes=nodes_size(1);
% displacements=load('increment_10.dat');
displacements=load('displacements.dat');
size(displacements);
%
u=zeros(num_nodes,nsteps);
v=zeros(num_nodes,nsteps);
% u_size=size(u)
mag_factor=1;
for i=2:nsteps
    u_nodes(:,i)=mag_factor*(i-1)*displacements(:,2)/(nsteps-1);
    v_nodes(:,i)=mag_factor*(i-1)*displacements(:,3)/(nsteps-1);
end
%
% ------------------------------------------------------------
%
function st=find_s_t(x_nodes,y_nodes,pointxy)
global xn yn ptxy
ptxy=pointxy;
xn=x_nodes;
yn=y_nodes;
options = optimset('MaxFunEvals',1e8);
st=fminsearch(@point_s_t,[0,0],options);
%
% objective function to find (s,t) for a point
%
function f = point_s_t(x)
%
global xn yn ptxy
%
f = 0;
%
s=x(1);
t=x(2);
f=sqrt((ptxy(1)-((1-s)*(1-t)*xn(1)+(1+s)*(1-t)*xn(2)+ ...
    (1+s)*(1+t)*xn(3)+ 1-s)*(1+t)*xn(4))/4)^2+ ...
    (ptxy(2)-((1-s)*(1-t)*yn(1)+(1+s)*(1-t)*yn(2)+ ...
    (1+s)*(1+t)*yn(3)+(1-s)*(1+t)*yn(4))/4)^2);
```

APPENDIX C

Original DIC Procedure

C.0.1 Dog Bone

A sample of an isotropic material needed to be prepared to for uniaxial tensile testing. To create an exact and duplicatable shape, a MATLAB script was used to generate a set of points in an X-Y coordinate system that were stored in an .svg file. The .svg file was uploaded into the Inventables Carvey online GUI. The cutting path of the tool of the Carvey was set to cut outside of the perimeter of the shape. This was done to ensure that the dimensions of the final piece most exactly matches the dimensions of the ABAQUS model. The dimensions of the sample were chosen to be compatible withe the biaxial load frame.

C.0.2 Cruciform

A cruciform of an isotropic material needed to be prepared to for biaxial tensile testing. To create another exact and duplicatable shape, a MATLAB script was used to generate a set of points in an X-Y coordinate system that were stored in an .svg file. The .svg file was uploaded into the Inventables Carvey online GUI. The dimensions of the sample were chosen to be compatible with the biaxial load frame. The corners of the outside of the center region of the cruciform shape use a small arc instead of a sharp corner to take into account the shape of the bit of the CNC machine.

C.1 Sample Preparation

To begin preparing the Lexan tensile bars and cruciform samples, their areas of interest (AOI) should be isolated using painters tape. A base coat of white spray paint was applied to the area of interest and allowed to dry for 24 hours. After drying, a medium-fine speckle coat of black spray paint was applied to the AOI

and allowed to dry for at least 6 hours. After the second drying, the samples are ready for tensile testing.

C.2 DIC

C.2.1 Camera Set Up

The camera array should be assembled and placed in a way such that the area of interest to be examined is normal to the camera array. Based on the size of the object being observed, the measuring distance, angle, and suggested aperture opening of the cameras is obtained from the GOM Manual[18]. Using the lights in the DIC kit, place the lights to illuminate the subject taking care to not place the lights behind, to the side of or above the line of sight of the cameras. The heat radiating from the lights can distort the camera view of the object. Once the object is in view of the cameras and the lights have been placed, the cameras can be focused and calibrated.

C.2.2 Focusing

To begin, close the apertures of the camera as much as they can be by adjusting the collar and finger screw near their lens. Next, in the ARAMIS software, adjust the exposure time of the cameras by clicking and dragging on the feed from the camera. The image is taken to be good when it is illuminated with minimal red over exposed patches. After that, the cameras can be focused. Each camera much be focused individually. To begin focusing, zoom in on the area of interest almost as far as the software permits. Loosen the screw of the silver collar near the lens of the camera and rotate the barrel containing the lens until the image obtained from each camera has the desired level of clarity. Once focused, a final adjustment of the aperture is done using the false color option. The false color option is another way to gauge the exposure of the object. Ideally, the apertures of each camera should be adjusted so that the false color seen on screen is yellow. White indicates

over exposure and blue indicates underexposure.

C.2.3 Calibration

To calibrate the camera system, indicate the desire to calibrate in the ARAMIS interface. once prompted, the software with guide the user step by step through the process. A calibration panel with precisely placed dots is observed by the camera array and photographed at different angles and orientations in order to create a 3-D calibrated space. Once the system is calibrated the cameras are ready to capture and track the deformation of any patterned object.

BIBLIOGRAPHY

GOM Correlate Professional: V8 SR1 Manual Basic.

"Chapter 4 - material behavior-linear elastic solids," in *Elasticity (Third Edition)*, third edition ed., Sadd, M. H., Ed. Boston: Academic Press, 2014, pp. 505 – 529. [Online]. Available: http://www.sciencedirect.com/science/article/pii/B9780124081369000167

Baghernezhad, S., Ghane, M., and Moezzi, M., "Strain monitoring in woven fabrics with locally induced mass irregularities using an image based method," *Fibres and Textiles in Eastern Europe*, vol. 24, pp. 73–80, 01 2016.

Cooreman, S., Lecompte, D., Sol, H., Vantomme, J., and Debruyne, D., "Identification of mechanical material behavior through inverse modeling and dic," *Experimental Mechanics*, vol. 48, no. 4, pp. 421 – 433, 2008. [Online]. Available: https://doi.org/10.1007/s11340-007-9094-0

Dinh, T., Rezaei, A., Laet], L. D., Mollaert, M., Hemelrijck], D. V., and Paepegem], W. V., "A new elasto-plastic material model for coated fabric," *Engineering Structures*, vol. 71, pp. 222 – 233, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0141029614002405

Górszczyk, J., Malicki, K., and Zych, T., "Application of digital image correlation (dic) method for road material testing," *Materials*, vol. 12, no. 15, p. 2349, 2019.

Hooke, R. and Jeeves, T. A., "" direct search" solution of numerical and statistical problems," *J. ACM*, vol. 8, no. 2, p. 212–229, Apr. 1961. [Online]. Available: https://doi.org/10.1145/321062.321069

Hubert Schreier, Jean-José Orteu, M. A. S. a., *Image Correlation for Shape, Motion and Deformation Measurements: Basic Concepts,Theory and Applications*. Springer US, 2009. [Online]. Available: http://gen.lib.rus.ec/book/index.php?md5=85c8b67b9f567265060c2cccf9b75c61

Lecompte, D., Smits, A., Sol, H., Vantomme, J., and Van Hemelrijck, D., "Mixed numerical–experimental technique for orthotropic parameter identification using biaxial tensile tests on cruciform specimens," *International Journal of Solids and Structures*, vol. 44, no. 5, pp. 1643 – 1656, 2007. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0020768306002848

Logan, D. L., *A First Course in the Finite Element Method*. 200 First Stamford Place, Suite 400 Stamford, CT 06902: Cengage Learning, 2012.

Mobasher, B., "5 - textile fiber composites: Testing and mechanical behavior," in *Textile Fibre Composites in Civil Engineering*, Triantafillou, T., Ed. Woodhead Publishing, 2016, pp. 101 – 150. [Online]. Available: http://www.sciencedirect.com/science/article/pii/B9781782424468000069

Nocedal, J. and Wright, S., *Numerical Optimization: Springer Series in Operations Research and Financial Engineering*. Springer, 2006.

Roylance, D., "Mechanical properties of materials," Massachusetts Institute of Technology, 2008. [Online]. Available: http://web.mit.edu/course/3/3.225/book.pdf

Schittkowski, K., "Nlpqlp: A new fortran implementation of a sequential quadratic programming algorithm for parallel computing," *Report, Department of Mathematics, University of Bayreuth*, 2001.

Schittkowski, K., "Nlpqlp: A fortran implementation of a sequential quadratic programming algorithm with distributed and non-monotone line search–user's guide," *Report, Department of Computer Science, University of Bayreuth*, 2006.

Sutton, M., Wolters, W., Peters, W., Ranson, W., and McNeill, S., "Determination of displacements using an improved digital correlation method," *Image and Vision Computing*, vol. 1, no. 3, pp. 133 – 139, 1983. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0262885683900641

Sutton, M., Wolters, W., Peters, W., Ranson, W., and McNeill, S., "Digital image correlation using newton-raphson method of partial differential correction," *Experimental Mechanics*, vol. 29, no. 3, pp. 261 – 267, 1989. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0262885683900641

Tariq, F., Siddqui, M. Z., Naz, N., and Ahmed, M. F., "Application of digital image correlation technique in determination of elastic constants of materials," 2012.