8-7-2010

# The effect of porosity distribution on the predicted mechanical response of die cast AM60B magnesium

David Barrett Hardin

# THE EFFECT OF POROSITY DISTRIBUTION ON THE PREDICTED MECHANICAL RESPONSE OF DIE CAST AM60B MAGNESIUM

By

David Barrett Hardin

A Thesis
Submitted to the Faculty of
Mississippi State University
in Partial Fulfillment of the Requirements
for the Degree of Masters of Science
in Civil Engineering
in the Department of Civil Engineering

Mississippi State, Mississippi

August 2010

THE EFFECT OF POROSITY DISTRIBUTION ON THE PREDICTED MECHANICAL RESPONSE OF DIE CAST AM60B MAGNESIUM

By

David Barrett Hardin

Approved:

| | |
|---|---|
| Philip M. Gullett<br>Assistant Professor of Civil Engineering<br>Major Professor | Mark F. Horstemeyer<br>Professor of Mechanical Engineering<br>Committee Member |
| Douglas J. Bammann<br>Professor of Mechanical Engineering<br>Committee Member | Sarah A. Rajala<br>Dean, Bagley College of Engineering |
| James Martin<br>Professor of Civil Engineering<br>Graduate Coordinator | |

Name: David Barrett Hardin

Date of Degree: August 7, 2010

Institution: Mississippi State University

Major Field: Civil Engineering

Major Professor: Dr. Philip M. Gullett

Title of Study: THE EFFECT OF POROSITY DISTRIBUTION ON THE PREDICTED MECHANICAL RESPONSE OF DIE CAST AM60B MAGNESIUM

Pages in Study: 119

Candidate for Degree of Masters of Science

In this paper, it is clearly shown that the distribution of the initial porosity is a critical factor in the prediction of damage evolution and initiation of failure in a cast AM60B magnesium notch Bridgeman tensile specimen. Using X-ray computed tomography, the actual initial porosity distribution was obtained, and this distribution was input into a finite element code as an initial condition. The predicted damage evolution from this simulation was compared to the damage evolution of the experimental specimen as well as other simulated porosity distributions. This study shows that the simulation of the actual porosity distribution predicted well the damage evolution observed in the experiment. It is also shown that the initial distribution of porosity plays a vital role in the predicted elongation to failure of a notched specimen. The actual distribution was shown to fail at a significantly lower strain than random or uniformly distributed damage.

DEDICATION

I dedicate this work to my wife Kate, without whom I would probably be working at some job I really wouldn't enjoy.

ACKNOWLEDGEMENTS

. I would like to thank the many people that have aided me in finishing this Master's Thesis. First, I would like to thank Dr. Gullett, my advisor, because not only has he taught me through my research, he also single-handedly convince me that graduate school was a good idea for me. Next, I want to thank my other committee members who have spent time illuminating ideas and concepts that I was struggling with. The next group of people are the grad students who worked near me and tolerated incessant questions and rants over the past 2 years. Through conversations with Matthew, Kyle, and Jason, many of the ideas taught in classes became knowledge that I can use. Drs. Youssef and Koffe also helped greatly in understanding the operation of the umat and vumat codes. Finally, I would like to thank Kate, my wife, for keeping me grounded in reality. Without these people, this work would not have been possible. Thank you.

TABLE OF CONTENTS

Page

DEDICATION ..... ii

ACKNOWLEDGEMENTS ..... iii

LIST OF TABLES ..... vi

LIST OF FIGURES ..... vii

CHAPTER

I. INTRODUCTION ..... 1

II. EXPERIMENTAL RESULTS AND FINITE ELEMENT SETUP ..... 5

2.1 Introduction ..... 5
2.2 Experimental Testing ..... 5
2.2.1 Material ..... 6
2.2.2 Geometry ..... 7
2.2.3 Microstructure ..... 7
2.3 Model Setup ..... 9

III. DMG MATERAL MODEL ..... 12

3.1 Introduction ..... 12
3.2 Kinematics ..... 13
3.3 Macroscale Elastic-Plastic Model ..... 17
3.4 Damage ..... 20
3.5 Numerical Implementation ..... 22

IV. IMPLICIT SIMULATION RESULTS ..... 26

4.1 Introduction ..... 26
4.2 Calibration ..... 27
4.2.1 DMG Material Constants ..... 27
4.2.2 Damage Evolution Response ..... 30
4.2.3 Time Step Size Evaluation ..... 31
4.3 Notch Bridgeman Simulation Results ..... 32

4.3.1 Comparison of Simulation to Experimental Results....33
4.3.2 Effect of Initial Porosity Distribution on Evolution of Damage....39
4.4 Discussion....46

V. EXPLICIT SIMULATION RESULTS....48

5.1 Introduction....48
5.2 Verification....48
5.2.1 Reduce Time Step Size (Without Damage)....52
5.2.2 Increase Element Size (Without Damage)....52
5.2.3 Increase Element Size (With Damage)....56
5.3 Fixed Mass Scaling....59
5.3.1 Mass Scaling Without Damage....60
5.3.2 Mass Scaling with Damage....64
5.3.3 Notched Bridgeman Simulations....67
5.4 Without Rate Sensitivity on the Initiation of Yield....71
5.4.1 One Element Simulations....72
5.4.2 Notched Bridgeman Simulations....75
5.5 Discussion....78

VI. CONCLUSIONS AND FUTURE WORK....81

REFERENCES....83

APPENDIX

A CONDITIONS OF SIMULATION EXIT IN THE PRESENCE OF A FLOATING POINT ERROR....86

A.1 Mass Scaling Simulations....87
A.1.1 Sample Input File....87
A.2 Sample Input File for Simulations with C1 = 0....89
A.3 Condition at Simulation Failure....92

B OPTIMIZATION OF EXPLICIT NUMERICAL IMPLEMENTATION....93

C UMAT REFERENCED IN THIS STUDY....97

D VUMAT REFERENCED IN THIS STUDY....109

LIST OF TABLES

TABLE Page

2.1 Average ultimate tensile strength of all H series experiments and percent loads calculated using known notch radius (Waters 2001). Strain values are the strain levels at which each load occurred in the experiment. ..... 6

5.1 Kinetic and total energy in each of the mass scaling simulations. ..... 62

LIST OF FIGURES

FIGURE Page

2.1 Three AM60B magnesium notched Bridgeman tensile specimens with three different notch geometries. The geometry used in this study is series H, the top specimen, with a notch radius of 0.635 cm. .....7

2.2 Experimental notch specimen for AM60 Mg showing the three trial specimens (a) Trial 1, (b) Trial 2, and (c) Trial 3 with the same geometry for which the 3D porosity was measured by computed tomography for the longitudinal studies performed at the Stanfor Synchrotron Radiation Laboratory (Waters et al. 2000). .....8

2.3 The axisymmetrically averaged void volume fractions found from CT scans of the experimental specimen at various points in the testing (Waters 2001). .....9

2.4 X-Ray CT results (Waters 2001) of the untested notch Bridgeman specimen (above) and the axisymmetrically averaged finite element values of void volume fraction used in the simulations (below). Labeled are the large pore “a” and the flow line “b”. .....10

2.5 Notch Bridgeman tensile specimen final mesh configuration and coordinate system with labeled boundary conditions.....11

3.1 Multiplicative decomposition of the deformation gradient into elastic, deviatoric, and dilational parts. .....14

4.1 Comparison of the implicit finite element simulation to uniaxial tension experimental data. .....27

4.2 Comparison of simulations run with different values of C5. When C5 = 1E-5, the simulation predicts elevated stresses for quasi-static test (top) and when C5 = 1E-4, the simulation predicts stresses corresponding well with the experimental data (bottom). .....29

4.3 Comparison of initial porosity versus elongation to failure for the implicit simulations and the experimental data. .....31

4.4 This figure shows a series of simulations designed to determine the accuracy of the implicit solution with various time steps. .....32

4.5 The black curve shows the mechanical response of the experimental tensile test. The blue curve is the mechanical response of the finite element simulation. The green curve is equal to the blue curve with the force values multiplied by 0.7. The dashed lines represent, from left to right, 60%, 87%, and 93% of the total load; the solid line represents the first element failure in the finite element simulation. .....33

4.6 Damage levels at 60% of the failure load for the experiment (left) and the simulation (right). .....34

4.7 Damage levels at 87% of the failure load for the experiment (left) and the simulation (right). .....35

4.8 Damage levels at 93% of the failure load for the experiment (left) and the simulation (right). .....36

4.9 Simulation contour plot of the damage distribution at the time of the first element failure for the implicit finite element simulation of the notch Bridgeman tensile specimen. .....37

4.10 Contour plots of von Mises equivalent stress (top left), effective plastic strain (top right), pressure (bottom left), and stress triaxiality (bottom right) for the simulation of the notch Bridgeman tensile specimen with the actual initial porosity distrbution. .....38

4.11 The graph shows the force versus strain response for each of the three porosity distributions, and the figures are damage contours taken as the first element failed in each simulation. .....40

4.12 Comparison of the damaged states of the real (top left), random (top right), and uniform (bottom) porosity distribution simulations at 3.3% true strain, which corresponds to first element failure of the real porosity distribution simulation. .....41

4.13 Comparison of the damaged states of the random (left) and uniform (right) porosity distribution simulations at 8.8% true strain, which corresponds to first element failure of the random porosity distribution simulation. .....42

4.14 Contour plots of von Mises equivalent stress (top left), effective plastic strain (top right), pressure (bottom left), and stress triaxiality (bottom right) for the simulation of the notch Bridgeman tensile specimen with the random initial porosity distribution. .....43

4.15 Contour plots of von Mises equivalent stress (top left), effective plastic strain (top right), pressure (bottom left), and stress triaxiality (bottom right) for the simulation of the notch Bridgeman tensile specimen with the uniform initial porosity distribution. .....45

5.1 Comparison of the stress versus strain response of a 1mm cube as predicted by the implicit and explicit implementations of the DMG model. 49

5.2 This illustrates clearly the difference in the evolution of the isotropic hardening variable kappa between the implicit and explicit schemes. .....51

5.3 Stress versus strain responses of a series of simulations of increasing element size with a constant strain rate, boundary conditions, and material parameters. .....54

5.4 Plot showing the error (MPa and %) at the last step in each simulation. .....55

5.5 This figure shows the stress versus strain response of a series of explicit finite element simulations with an initial porosity of 3% and increasing in element size from 50 mm to 300 mm. .....57

5.6 Plot showing the percent error between the stress at the last step in each simulation. .....58

5.7 Truncated plot showing the percent error between the stress at the last step in each simulation. Each simulation contains 3% initial porosity, element sizes vary from 210 mm to 300 mm. .....59

5.8 Stress versus strain responses for a series of simulations with increasing mass scaling factors. All simulations are run at the same strain rate and all elements are the same size. .....61

5.9 Plot showing the percent error of the true stress at 4.8% true strain for simulations with mass scaling factors ranging from 1 E 1 to 1 E 12. The error is measured between the stress in each simulation and the stress of the converged solution to this problem, 199.4 MPa. .....63

5.10 Stress versus strain responses for a series of simulations with increasing mass scaling factors and a non-zero initial porosity (3%). All simulations are run at the same strain rate and all elements are the same size. 64

5.11 Plot showing the percent error of the true stress at 4.8% true strain for simulation with an initial porosity of 3% and mass scaling factors ranging from 1 E 1 to 1 E 13. The error is measured between the stress in each simulation and the stress of the converged solution to this problem, 195.1 MPa. ....65

5.12 Plot comparing the elongation to failure of simulations run with various initial porosity levels and experimental data. The simulations were run on 1 mm elements with mass scaling factors of 1 E 9 and 1 E 10. ....66

5.13 Damage levels at 60% of the failure load for the experiment ( top left) and the simulations with mass scaling factors of 1 E 6 (top right), 1 E 8 (bottom left), and 1 E 10 (bottom right). The damage contours were taken at 0.6% true strain. ....67

5.14 Damage levels at 87% of the failure load for the experiment ( top left) and the simulations with mass scaling factors of 1 E 6 (top right), 1 E 8 (bottom left), and 1 E 10 (bottom right). The damage contours were taken at 2.1% true strain. ....69

5.15 Damage levels at 93% of the failure load for the experiment ( top left) and the simulations with mass scaling factors of 1 E 6 (top right), 1 E 8 (bottom left), and 1 E 10 (bottom right). The damage contours were taken at 2.8% true strain. ....70

5.16 Damage contour at the final increment of the simulation of the notched Bridgeman tensile specimen applying a mass scaling factor of 1 E 10. ....71

5.17 This plot shows the stress versus strain response for an implicit and explicit simulation with no damage and C1 = 0. The results are plotted over experimental data for a quasi static tensile test. ....72

5.18 Plot showing the stress versus strain response for simulations run in ABAQUS Explicit on 1 mm elements and having a range of mass scaling factors from 100 to $1 \; x \; 108$. ....73

5.19 Stress versus strain results of simulations run with 3% initial porosity and C1 = 0. ....74

5.20 Damage contours at 0.6% true strain for the experiment (left) and simulation (right). ....75

5.21 Damage contours at 2.1% true strain for the experiment (left) and simulation (right). ....76

5.22 Damage contours at 2.8% true strain for the experiment (left) and simulation (right). ....77

5.23 Damage contour at the final increment of the simulation of the notched Bridgeman tensile specimen with C1 = 0 and applying a mass scaling factor of 1 E 6. ..... 77

# CHAPTER I

# INTRODUCTION

The mechanical response of a solid material to external loading is fully dependent on its internal structure as well as the way in which it is loaded. In order to predict the behavior of a material when used in a part, numerous models with varying degrees of sophistication have been created and used. The earliest and simplest predictive material model is an isotropic linear elastic model, which while computationally efficient, is only capable of accurately predicting material behavior in the elastic range. With the significant advances in computational power, models have been developed as need has required that can model not only the elastic range but now the inelastic range and eventual failure of the material. Then, it was found that initial porosity is an important factor in determining the mechanical response of many cast materials (Herrera and Kondic 1977, Eady and Smith 1986, Surappa et al. 1986, Caceres 1995, Caceres and Selling 1996, Weiler et al. 2005, Lee 2007, Lee and Shin 2007). Therefore for simulation based design, the simulation of damage processes is critical to failure prediction in ductile materials.

The effect of porosity on the tensile properties of cast metals has been the focus of many studies. Herrera and Kondic (1977) and Surappa et al. (1986) studied the effect of average porosity levels of $< 0.4\%$ on the tensile response of aluminum alloys. When attempting to correlate the tensile strength to average volume fraction of porosity, Herrera and Kondic found very large scatter in the data; Surappa et al. actually showed

that it is possible for the elongation to failure to increase with an increase in average porosity. Surappa et al. also speculated that this might be caused by non-uniform distribution of pores within the test bars. Later, Horstemeyer and Gokhale (1999) introduce a phenomenological void-crack nucleation model with the purpose of predicting damage evolution for design applications. This model is a function of stress state, microstructural features, fracture toughness, and strain level. This paper by Horstemeyer and Gokhale represents a major step towards creating a tool capable of accurately predicting damage evolution as well as the mode and location of fracture. More recently, Weiler et al. (2005) used X-Ray tomography to non-destructively measure the size and location of internal pores in a tensile specimen of AM60B magnesium. A critical strain model was then implemented to predict the mechanical properties of the tensile specimens during the fracture process. The predicted values of fracture stress, fracture strain, and plane of fracture were then compared to experimental tensile tests.

This study utilizes an internal state variable (ISV) model to incorporate physical details into a finite element simulation of response and failure of a magnesium alloy. Possible details include particle size, grain size, and internal porosity. In this study the initial porosity distribution is mapped directly from X-Ray CT scans into the finite element mesh as an initial condition. Simulation results are compared to detailed experimental efforts that include, X-Ray tomography of the actual internal porosity distribution at several steps during the tensile test. This represents a significant step in simulation technology because in previous studies, if porosity was considered at all it has typically been homogeneously or randomly distributed throughout the specimen. While Weiler et al. did consider the actual initial internal porosity distribution, this study also uses an ISV model to track the evolving damage. Inputting an actual porosity distribution

into an ISV model designed to predict the evolution of porosity and damage allows the possibility for the possibility of increased accuracy in predictive modeling of parts tested to near failure.

This study focuses on examining the ability of a specific ISV model to predict the evolution of damage over the course of a notched Bridgeman tensile test. The actual internal porosity distribution for the specimen was found initially using X-Ray CT and at four different times during the tensile test using the same method. The study also examines the effect of the initial distribution on the simulated damage evolution and on the overall behavior of the material. This was accomplished by running simulations using initially uniform and initially random damage distributions and comparing the results with the actual distribution results.

Chapter II presents background information concerning the experimental testing and the setup of the finite element simulation. The chemical composition, forming process, and geometry of the tensile specimen will be discussed. This chapter will also present the results from the X-Ray CT and how these results were incorporated into the finite element simulation.

Chapter III presents an introduction to the DMG model in use at Mississippi State University. A description of the model as well as the mathematical basis for it will be covered briefly. Sections detailing the kinematics, macroscale elastic- plastic material response, damage framework, and numerical implementation are included in this chapter.

Chapter IV presents the results from ABAQUS Standard for the quasi-static case of the DMG model described in Chapter III. Comparison of simulations with extensive mechanical testing are included. The effect of initial porosity distribution on the damage

evolution and mechanical response is studied using simulations with various porosity distributions run until the first element failure.

Chapter V presents the results of simulations run in ABAQUS Explicit. Included in this chapter is a verification that the implicit and explicit versions of the ABAQUS user defined material model were equivalent and several limitations that were discovered when running quasi-static simulations.

Finally, Chapter VI presents conclusions drawn from the results presented in the earlier chapters. Along with these conclusions, some recommendations for future work in this area will also be presented.

# CHAPTER II

# EXPERIMENTAL RESULTS AND FINITE ELEMENT SETUP

## 2.1 Introduction

This section details the experimental testing done for this study and the setup of the finite element simulation to be used. The testing was performed in Waters (2001). Along with mechanical results, X-Ray CT scans were taken of the specimen before and at 3 specific points during the loading. It is these scans that will be used to evaluate the ability of the DMG model to predict the evolution of damage.

## 2.2 Experimental Testing

The test specimen being modeled in this study was originally tested by Waters (2001). The specimen was analyzed using X-Ray Computed Tomography in order to obtain a quantification of the porosity distribution existing within the specimen. First, a 3D reconstruction was created containing all of the porosity data for the entire specimen. Then, a routine was created to determine the total void volume fraction for a series of increasing diameter rings for one value in the y-direction, as labeled in Figure 2.5. The void volume fraction was found for other y-values until the entire notch region had been analyzed. This analysis routine provided a map of the void volume fraction as a function of radius and the y-value; this mapped void volume fraction data is averaged over the entire ring and therefore assumes axial symmetry. As the specimen was tested, CT scans were performed at various strain levels in order to track the evolution of damage within the part over the range of the test.

### 2.2.1 Material

The specimens in this study are made of die-cast AM60B magnesium alloy. This alloy is often used in the production of automotive parts having complex geometries and requiring good elongation, good toughness, and low weight. The chemical composition of the tensile specimens in weight percent, is 5.67 % Al, 0.377% MN, 0.15% SI, 0.0011% Fe, <0.0035% Cu, <0.0001% Ni, <0.0033% Ca, and the remainder Mg. The samples were cast in a 600 ton cold chamber die-casting machine with a 600 ton locking force. The injection temperature was between 675 and 690 degrees Celsius, the metal temperature was 750 degrees Celsius, and the die temperature was 300 degrees Celsius. The shot weight used was 1.63 kg, the shot sleeve diameter was 3.81 cm, and had a shot stroke of 29.21 cm. The average gate velocity was 41.15 m/s, and the process had an average cycle time of 45 seconds (Waters 2001).

Table 2.1 Average ultimate tensile strength of all H series experiments and percent loads calculated using known notch radius (Waters 2001). Strain values are the strain levels at which each load occurred in the experiment.

| Measure | 60% load | 87% load | 93% load | $\sigma_{uts}$ (Mpa) |
|---|---|---|---|---|
| Force (N) | 9176 | 13304 | 14225 | 207 |
| Strain | 0.60% | 2.06% | 2.81% | |

Table 2.1 displays the points during the experimental loading at which CT scans were taken. The strain values found in Table 2.1 are the strains at which the corresponding loads are first experienced. For comparison between simulations and experimental results, the damage distribution at the given strain levels will be evaluated.

### 2.2.2 Geometry

The notched Bridgeman tensile specimens were obtained in three different notch geometries. For each geometry, three specimens were obtained and tested. Each bar was 11.4 cm long and had an outer diameter of 1.27 cm. The difference in the three geometries was the notch root radii; the three different radii were 0.635 cm, 0.794 cm, and 1.27 cm. A picture of the geometries is shown in Figure 2.1 (Waters 2001). The geometry used for the remainder of this study is on the top, has the smallest notch root radius, and is referred to as series H. An H series specimen was chosen for this study because detailed CT data was obtained for the specific test H-24.

Figure 2.1 Three AM60B magnesium notched Bridgeman tensile specimens with three different notch geometries. The geometry used in this study is series H, the top specimen, with a notch radius of 0.635 cm.

### 2.2.3 Microstructure

The microstructure of these particular samples was studied in great detail by Waters et al. (2000). CT scans of three test specimens with the same geometry were taken before testing and at 60, 87, and 93% of the total load. The CT scans taken of the

specimens prior to loading are located in Figure 2.2 Figure 2.3 shows the evolution of damage in the experimental specimen at 60, 87, and 93% of the total load. All of the data in Figure 2.3 underwent the process described in Section 2.2 to convert the 3D CT data into 2D axisymmetrically averaged void volume fractions.

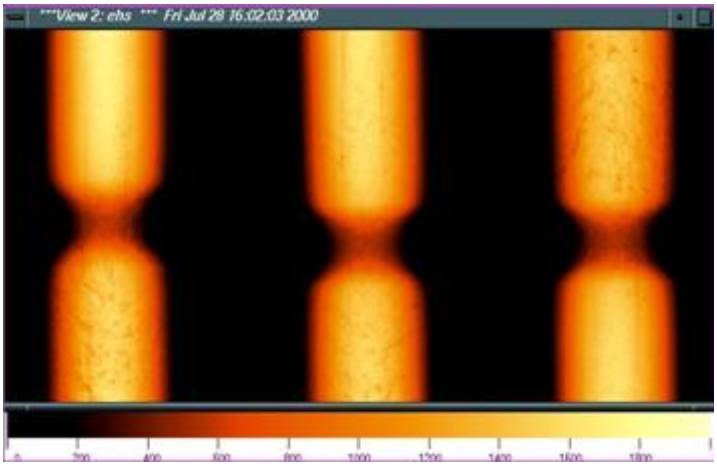



Figure 2.2 Experimental notch specimen for AM60 Mg showing the three trial specimens (a) Trial 1, (b) Trial 2, and (c) Trial 3 with the same geometry for which the 3D porosity was measured by computed tomography for the longitudinal studies performed at the Stanford Synchrotron Radiation Laboratory (Waters et al. 2000).

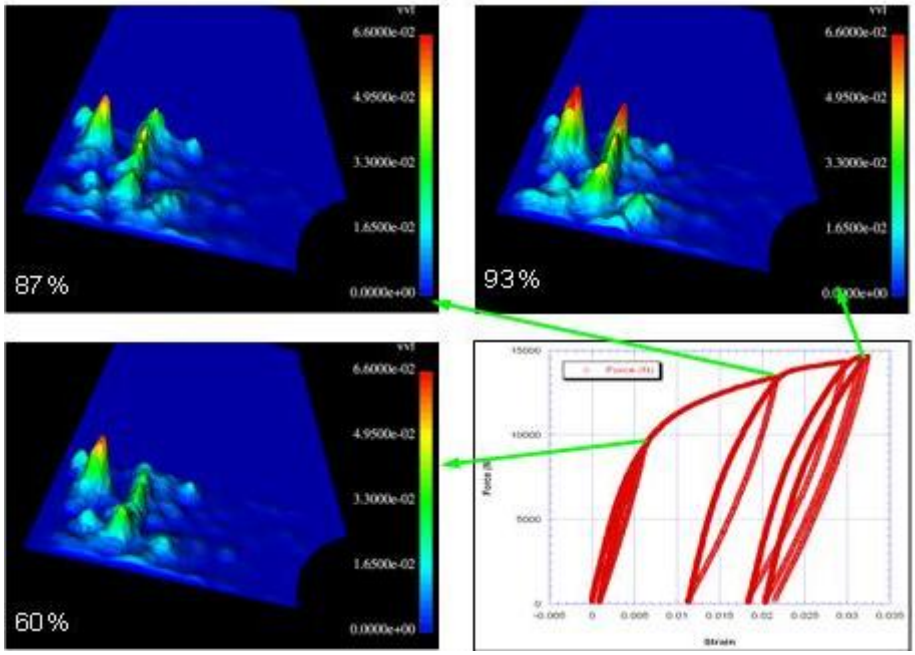



Figure 2.3 The axisymmetrically averaged void volume fractions found from CT scans of the experimental specimen at various points in the testing (Waters 2001).

## 2.3 Model Setup

After running the experiments and obtaining the results, the next step is to design a finite element simulation that physically represents the state of the material and the shape of the specimen before any testing is done. For the purposes of this study the tensile specimens are modeled as axisymmetric about the y-axis and assumed to by symmetric about the x-axis, axis labels are as shown in Figure 2.5. In order to use the results of the CT scans for a now two dimensional simulation, the void data is averaged axisymmetrically. The resulting averaged void data are discretized and mapped onto a computational mesh using a moving least squares interpolation (Lancaster and Salkuaskas, 1981). Figure 2.4 shows both the axisymmetrically averaged CT data and the initial discretized distribution of the damage onto the finite element mesh. In the CT scan, the large internal pore and the flow line are labeled as “a” and “b”, respectively.

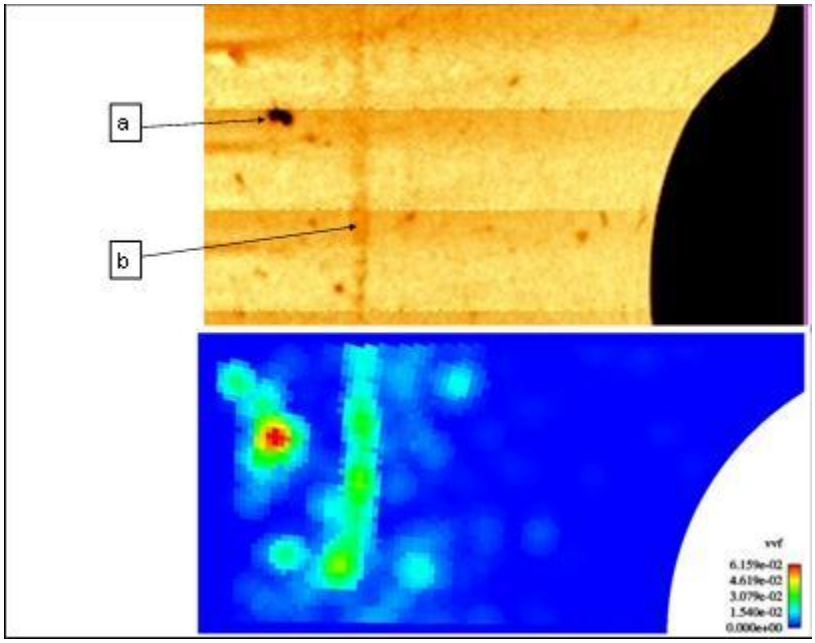



Figure 2.4 X-Ray CT results (Waters 2001) of the untested notch Bridgeman specimen (above) and the axisymmetrically averaged finite element values of void volume fraction used in the simulations (below). Labeled are the large pore “a” and the flow line “b”.

The mesh was created using 4 noded axisymmetric elements with reduced integration. For use in ABAQUS version 6.9, the CAX4R element fits the requirements. The boundary conditions applied are axisymmetry about the left-hand side, symmetry along the bottom face, and a constant velocity of 0.005 mm/s on the top edge of the simulation. The simulation mesh contained 12800 four noded elements. Figure 2.5 shows the final mesh and the boundary conditions.

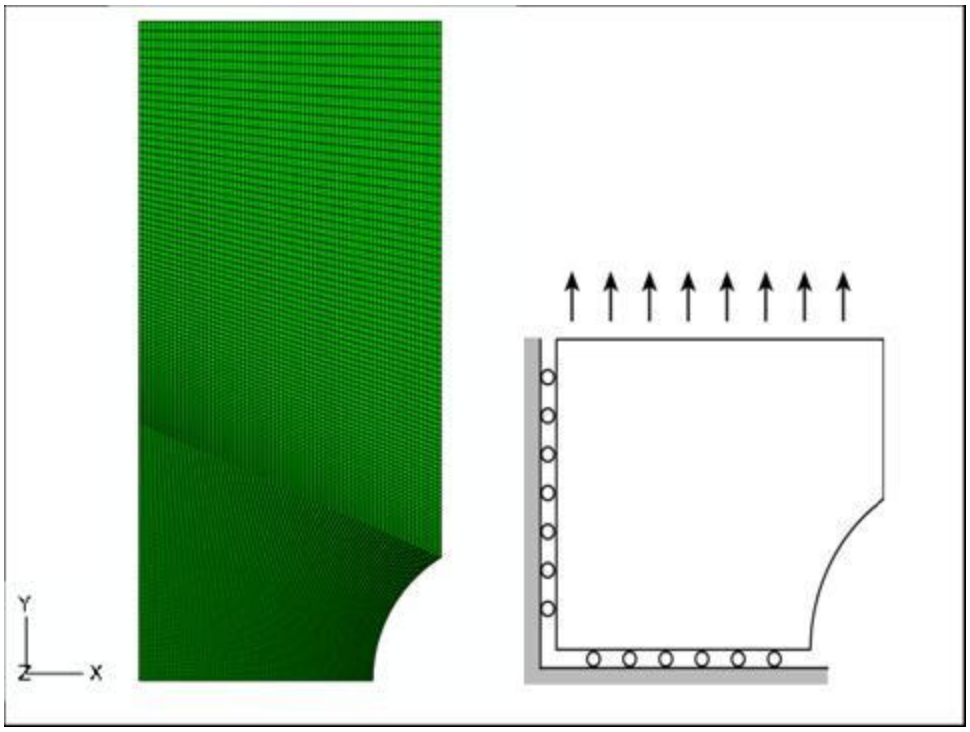



Figure 2.5 Notch Bridgeman tensile specimen final mesh configuration and coordinate system with labeled boundary conditions.

# CHAPTER III

# DMG MATERAL MODEL

## 3.1 Introduction

The microstructural details of a material and three dimensional stress state can play a vital role in determining the overall performance and eventual failure of structural components. The material being used, forming processes, and loading histories all affect the way a component will respond to a subsequent load. Voids, particles, and cracks seen in the microstructure of the material can create local regions of increased stress and plastic strain. It is common for simple failure criteria such as Tresca or Von Mises equivalent stresses to be used in the analysis and design of components but do not always produce results that are both sound and economical. Without accounting for the effect of microstructure or loading history, these failure criteria could lead to insufficient or inefficient designs.

This chapter will describe the internal state variable model used in this study. This model includes thermodynamically consistent kinematics, damage progression, and an elastic-plastic framework. A major motivation for the development of this model is to create a physically based mathematical model that is capable of tracking the material's history and using the state of the material to consistently and accurately predict the mechanical response of a material under complex loading conditions. The model used in this study is based on the constitutive model developed by Bammann et al., 1984, 1989, 1990, 1993, 1996. The model uses internal state variables to characterize the motion and

density dislocations. The evolution of these dislocations and their effect on material behavior is a controlling factor on overall material response. The damage evolution equations were introduced to the model by Horstemeyer and Gokhale, 1999, and Horstemeyer and Lathrop et al., 2000. These damage equations included void nucleation and growth evolutions and a phenomenological formulation for the coalescence term. The McClintock void growth rule (1968) is utilized to describe the growth of damage associated with particles in the material, and the Cocks and Ashby void growth rule (1982) is used to describe the growth of pores. A subsequent coalescence evolution equation was added by Allison, 2009 in order to take into account the findings of Horstemeyer and Matalanis et al., 2000. It was determined that the influence of coalescence could be described as a function of pore diameter divided by nearest neighbor distance.

### 3.2 Kinematics

The kinematic formulation used in this model is very similar to that of Davison et al. (1977), Bammann and Aifantis (1989), Bammann et al. (1993), Horstemeyer et al. (2000), and Horstemeyer et al. (2007). The kinematics of motion use a multiplicative decomposition of elastic straining, plastic flow, and damage formation and growth. The deformation gradient, $\underline{F}$, is decomposed into elastic($\underline{F^e}$), deviatoric inelastic($\underline{F_d^p}$), and volumetric inelastic parts ($\underline{F_v^p}$) illustrated in Figure 3.1 and given by

$$\underline{\boldsymbol{F}} = \underline{\boldsymbol{F}}^{\boldsymbol{e}}\underline{\boldsymbol{F}}_{\boldsymbol{v}}^{\boldsymbol{p}}\underline{\boldsymbol{F}}_{\boldsymbol{d}}^{\boldsymbol{p}}. \qquad \textit{Equation 3.1}$$

Equation 3.1 assumes that the motion is described by a continuous function. The elastic deformation gradient, $\underline{F^e}$, represents recoverable lattice displacements from their equilibrium position. The deviatoric deformation gradient, $\underline{F_d^p}$, represents dislocations in

a continuous field whose motion is both permanent and volume preserving. The volumetric deformation gradient, $\underline{F}_v^p$, represents a continuous field of voids and is responsible for the volume change in material during plastic deformation.

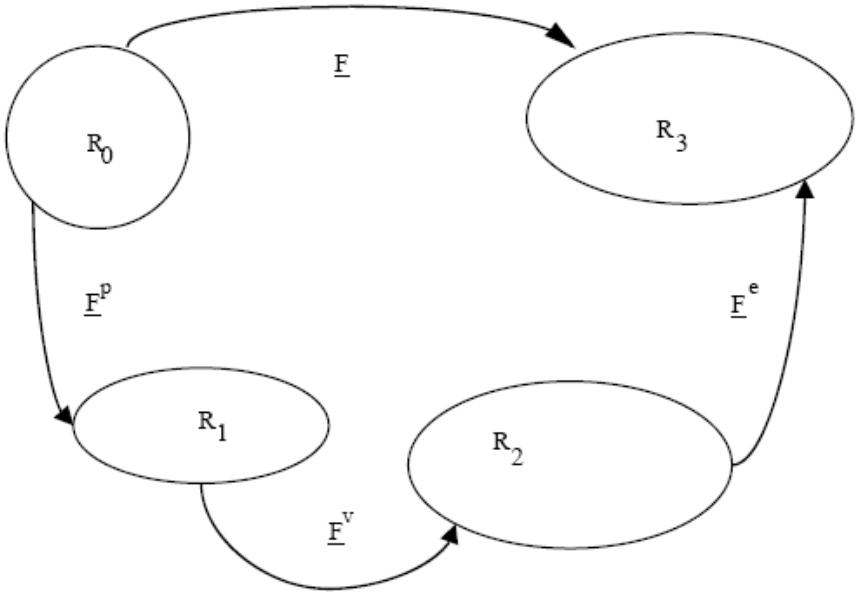



Figure 3.1 Multiplicative decomposition of the deformation gradient into elastic, deviatoric, and dilational parts.

The Jacobian of Equation 3.1 is related to the volume change or the change in density for a constant mass system and is given by

$$\boldsymbol{J = det\underline{F}_v^p = \frac{V_2}{V_0} = \frac{\rho_0}{\rho_2}} \qquad \textit{Equation 3.2}$$

and is required to be positive. The change in volume from the reference configuration (State 0) to the intermediate configuration (State2) in Figure 3.1 is $V_2=V_0 + V_v$. This relationship is true since there is no volume change between States 0 and 1. In moving the configuration from State 0 to State 2, a volume of voids, $V_v$, is added to the initial volume, $V_0$, to give the total volume in State 2, $V_2$. Damage, φ, is then introduced as the

ratio of the volume of voids divided by the total volume of voids in State 2 or the elastically unloaded or unstressed state and is described by Equation 3.3.

$$\boldsymbol{\phi} = \frac{\boldsymbol{V_v}}{\boldsymbol{V_2}} \qquad \textit{Equation 3.3}$$

From this relationship,

$$\boldsymbol{V_0} = (\mathbf{1} - \boldsymbol{\phi})\boldsymbol{V_2}\,, \qquad \textit{Equation 3.4}$$

$$\boldsymbol{J} = \frac{\mathbf{1}}{\mathbf{1}-\boldsymbol{\phi}}\,. \qquad \textit{Equation 3.5}$$

Using Equation 3.4, Equation 3.5 finds the Jacobian of the deformation gradient in terms of the newly introduced damage parameter, φ. With the assumption that damage produces isotropic volume change, the volumetric portion of the deformation gradient becomes

$$\underline{\boldsymbol{F}}_{\boldsymbol{v}}^{\boldsymbol{p}} = \frac{\mathbf{1}}{(\mathbf{1}-\boldsymbol{\phi})^{\frac{\mathbf{1}}{\mathbf{3}}}}\underline{\boldsymbol{I}} \qquad \textit{Equation 3.6}$$

The velocity gradient, Equation 3.7, associated with the deformation gradient is additively decomposed into elastic, volumetric, and deviatoric parts as seen in Equation 3.8. The velocity gradient can be further decomposed into symmetric and antisymmetric parts, Equations 3.9 and 3.10, respectively.

$$\underline{\boldsymbol{L}} = \underline{\dot{\boldsymbol{F}}\boldsymbol{F}^{-\mathbf{1}}} \qquad \textit{Equation 3.7}$$

$$\underline{\boldsymbol{L}} = \underline{\boldsymbol{L}}^{\boldsymbol{e}} + \underline{\boldsymbol{L}}_{\boldsymbol{v}}^{\boldsymbol{p}} + \underline{\boldsymbol{L}}_{\boldsymbol{d}}^{\boldsymbol{p}} \qquad \textit{Equation 3.8}$$

$$\underline{\boldsymbol{D}} = \frac{\mathbf{1}}{\mathbf{2}}\left(\underline{\boldsymbol{L}} + \underline{\boldsymbol{L}}^{\boldsymbol{T}}\right) \qquad \textit{Equation 3.9}$$

$$\underline{\boldsymbol{W}} = \frac{\mathbf{1}}{\mathbf{2}}\left(\underline{\boldsymbol{L}} - \underline{\boldsymbol{L}}^{\boldsymbol{T}}\right) \qquad \textit{Equation 3.10}$$

Equations of the form of Equations 3.9 and 3.10 hold true for the elastic, volumetric, and deviatoric parts of the velocity gradient. Therefore, the volumetric part of the velocity gradient can then be given as

$$\underline{\dot{\boldsymbol{F}}}_{\boldsymbol{v}}^{\boldsymbol{p}}\underline{\boldsymbol{F}}_{\boldsymbol{v}}^{\boldsymbol{p}-\mathbf{1}} = \frac{\dot{\boldsymbol{\phi}}}{\mathbf{3}(\mathbf{1}-\boldsymbol{\phi})}\underline{\boldsymbol{I}} \qquad \textit{Equation 3.11}$$

and the volumetric rate of deformation is

$$\underline{\boldsymbol{D}}_{\boldsymbol{v}}^{\boldsymbol{p}} = \frac{\dot{\boldsymbol{\phi}}}{\mathbf{3}(\mathbf{1}-\boldsymbol{\phi})}\underline{\boldsymbol{I}}\,. \qquad \textit{Equation 3.12}$$

If we take the trace of the volumetric rate of deformation, we get Equation 3.13 that shows that the damage, φ, is directly related to the volumetric rate of deformation. It is also helpful to notice here that $\underline{W}_v^p = 0$.

$$\boldsymbol{tr}\left(\underline{\boldsymbol{D}}_{\boldsymbol{v}}^{\boldsymbol{p}}\right) = \frac{\dot{\boldsymbol{\phi}}}{\mathbf{1}-\boldsymbol{\phi}} \qquad \textit{Equation 3.13}$$

The elastic rate of deformation is related to the volumetric rate of deformation according to the additive decomposition of the deformation rates,

$$\underline{\boldsymbol{D}}^{\boldsymbol{e}} = \underline{\boldsymbol{D}} - \underline{\boldsymbol{D}}_{\boldsymbol{v}}^{\boldsymbol{p}} - \underline{\boldsymbol{D}}_{\boldsymbol{d}}^{\boldsymbol{p}} \qquad \textit{Equation 3.14}$$

Now that the damage has been related to the rate of deformation, the damaged state can be given in terms of the void nucleation and growth in the unstressed configuration. Next, let the total number of voids be given by N, the representative volume in the reference configuration equal $V_0$, $\eta^*$ is equal to the number of voids per unit volume in the reference configuration, and $v_v$ is the average void volume of a nucleated void. Therefore, the volume of voids is given by

$$\boldsymbol{V}_{\boldsymbol{v}} = \boldsymbol{\eta}^{*}\boldsymbol{V}_{\mathbf{0}}\boldsymbol{v}_{\boldsymbol{v}}\,. \qquad \textit{Equation 3.15}$$

Combine Equation 3.15 with the definition, $V_2 = V_0 + V_v$ , and insert them into Equation 3.3 to get

$$\boldsymbol{\phi} = \frac{\boldsymbol{\eta^* V_0 v_v}}{\boldsymbol{V_0 + \eta^* V_0 v_v}} = \frac{\boldsymbol{\eta^* v_v}}{\boldsymbol{1 + \eta^* v_v}} . \qquad \textit{Equation 3.16}$$

This form of the damage term was used by Davison et al. (1977). If the number of voids per unit volume is known in terms of the intermediate configuration instead of the reference configuration, then damage can be written as Equation 3.17.

$$\boldsymbol{\phi} = \frac{\boldsymbol{V_v}}{\boldsymbol{V_2}} = \frac{\boldsymbol{V_v}}{\boldsymbol{N}} \frac{\boldsymbol{N}}{\boldsymbol{V_2}} = \boldsymbol{v_v \eta} \qquad \textit{Equation 3.17}$$

Where,

$$\boldsymbol{\eta} = \frac{\boldsymbol{N}}{\boldsymbol{V_2}} = \frac{\boldsymbol{N}}{\boldsymbol{V_0}} \frac{\boldsymbol{V_0}}{\boldsymbol{V_2}} = \boldsymbol{\eta^*} \frac{\boldsymbol{V_0}}{\boldsymbol{V_2}} \qquad \textit{Equation 3.18}$$

relates the number of voids per unit volume in the intermediate configuration to the number in the reference configuration. Once again using the relationship additively combining the volume in the reference configuration with the volume of voids to get the volume in the intermediate configuration gives the expression

$$\boldsymbol{\eta^* = \eta / (1 - \phi)} , \qquad \textit{Equation 3.19}$$

which enables us to find $\eta^*$ from an experimentally determined value for $\eta$ obtained from a specimen that has been loaded and then elastically unloaded.

### 3.3 Macroscale Elastic-Plastic Model

The internal state variable model used in this study was developed by Bammann et al. (1993) and modified to account for the stress state dependence of the damage evolution by Horstemeyer and Gokhale (1999) and Horstemeyer, Lathrop, et al. (2000).

The defining equations of this model are given by the rate of change of observable and internal state variables. The equation used to implement the model into the finite element method are given by,

$$\dot{\underline{\sigma}} = \lambda(1-\phi)tr(\underline{D}^e)\underline{I} + 2\mu(1-\phi)\underline{D}^e - \left(\frac{\dot{\phi}}{1-\phi}\right)\underline{\sigma} \qquad \textit{Equation 3.20}$$

where $\underline{\sigma}$ and $\dot{\underline{\sigma}}$ are the Cauchy stress and the co-rotational or objective rate of the Cauchy stress, respectively. In this model the objective Jaumann rate is being used. λ and μ are the elastic Lame constants, and $\underline{I}$ is the second-order identity tensor. The plastic flow rule is given by Equation 3.21.

$$\underline{D}^p = f(T)sinh\left\{\frac{\|\underline{\sigma}'-\underline{\alpha}\|-[R+Y(T)](1-\phi)}{V(T)(1-\phi)}\right\}\frac{\underline{\sigma}'-\underline{\alpha}}{\|\underline{\sigma}'-\underline{\alpha}\|} \qquad \textit{Equation 3.21}$$

where $\underline{\sigma}'$ is the deviatoric portion of the stress tensor, $\underline{\alpha}$ is kinematic hardening, R is isotropic hardening, and T is temperature. The function Y(T) is the rate independent yield stress, V(T) is related to the magnitude of rate dependence on initial yield, and f(T) determines at what strain rate the rate dependence affects yielding. Y(T), V(T), and f(T) are temperature sensitive and given by

$$Y(T) = C_3 exp(C_4/T) \qquad \textit{Equation 3.22}$$

$$V(T) = C_1 exp(-C_2/T) \qquad \textit{Equation 3.23}$$

$$f(T) = C_5 exp(-C_6/T). \qquad \textit{Equation 3.24}$$

$C_1$ through $C_6$ are material parameters obtained from uniaxial mechanical tests performed at various strain rates and temperature and are all related to yield stress.

The hardening rate of the material is separated into Equations 3.25 and 3.26. Equation 3.25 is the co-rotational rate of kinematic hardening, $\underline{\dot{\alpha}}$, and is representative of geometrically necessary dislocations in the material. The material time derivative of the isotropic hardening, $\dot{R}$, is described in Equation 3.26 and represents the uniform distribution of statistically stored dislocations in the material.

$$\underline{\dot{\boldsymbol{\alpha}}} = \left\{ \boldsymbol{h(T)}\underline{\boldsymbol{D}}^{\boldsymbol{p}} - \left[ \sqrt{\frac{2}{3}} \boldsymbol{r_d(T)} \left\| \underline{\boldsymbol{D}}^{\boldsymbol{p}} \right\| + \boldsymbol{r_s(T)} \right] \left\| \underline{\boldsymbol{\alpha}} \right\| \underline{\boldsymbol{\alpha}} \right\} \left[ \frac{\boldsymbol{DCS_0}}{\boldsymbol{DCS}} \right]^{\boldsymbol{z}} \qquad \textit{Equation 3.25}$$

$$\dot{\boldsymbol{R}} = \left\{ \boldsymbol{H(T)}\underline{\boldsymbol{D}}^{\boldsymbol{p}} - \left[ \sqrt{\frac{2}{3}} \boldsymbol{R_d(T)} \left\| \underline{\boldsymbol{D}}^{\boldsymbol{p}} \right\| + \boldsymbol{R_s(T)} \right] \boldsymbol{R^2} \right\} \left[ \frac{\boldsymbol{DCS_0}}{\boldsymbol{DCS}} \right]^{\boldsymbol{z}} \qquad \textit{Equation 3.26}$$

where $DCS_0$, DCS, and z are related to dendrite cell size and grain size. These constants allow for microstructural features to be accounted for in the plastic region of the model. The functions h(T) and H(T) are the anisotropic and isotropic hardening modulus, respectively, and are described as a function of temperature in Equations 3.28 and 3.31. $R_d(T)$ and $r_d(T)$ are scalar functions of temperature that describe the dynamic recovery, and $R_s(T)$ and $r_s(T)$ are scalar functions that describe static (thermal) recovery. These functions are

$$\boldsymbol{r_d(T)} = \boldsymbol{C_7} \left( \mathbf{1} - \boldsymbol{C_a} \left[ \frac{\mathbf{4}}{\mathbf{27}} - \frac{\boldsymbol{J_3^2}}{\boldsymbol{J_2^3}} \right] - \boldsymbol{C_b} \frac{\boldsymbol{J_3}}{\boldsymbol{J_2^{1.5}}} \right) \boldsymbol{exp} \left( {}^{-\boldsymbol{C_8}}/_{\boldsymbol{T}} \right) \qquad \textit{Equation 3.27}$$

$$\boldsymbol{h(T)} = \left\{ (\boldsymbol{C_9} - \boldsymbol{C_{10}T}) \left( \mathbf{1} + \boldsymbol{C_a} \left[ \frac{\mathbf{4}}{\mathbf{27}} - \frac{\boldsymbol{J_3^2}}{\boldsymbol{J_2^3}} \right] + \boldsymbol{C_b} \frac{\boldsymbol{J_3}}{\boldsymbol{J_2^{1.5}}} \right) \right\} \qquad \textit{Equation 3.28}$$

$$\boldsymbol{r_s(T)} = \boldsymbol{C_{11}} \boldsymbol{exp} \left( {}^{-\boldsymbol{C_{12}}}/_{\boldsymbol{T}} \right) \qquad \textit{Equation 3.29}$$

$$\boldsymbol{R_d(T)} = \boldsymbol{C_{13}} \left( \mathbf{1} - \boldsymbol{C_a} \left[ \frac{\mathbf{4}}{\mathbf{27}} - \frac{\boldsymbol{J_3^2}}{\boldsymbol{J_2^3}} \right] - \boldsymbol{C_b} \frac{\boldsymbol{J_3}}{\boldsymbol{J_2^{1.5}}} \right) \boldsymbol{exp} \left( {}^{-\boldsymbol{C_{14}}}/_{\boldsymbol{T}} \right) \qquad \textit{Equation 3.30}$$

$$\boldsymbol{H(T)} = \left\{ (\boldsymbol{C_{15}} - \boldsymbol{C_{16}T}) \left( \mathbf{1} + \boldsymbol{C_a} \left[ \frac{\mathbf{4}}{\mathbf{27}} - \frac{\boldsymbol{J_3^2}}{\boldsymbol{J_2^3}} \right] + \boldsymbol{C_b} \frac{\boldsymbol{J_3}}{\boldsymbol{J_2^{1.5}}} \right) \right\} \qquad \textit{Equation 3.31}$$

$$\boldsymbol{R_s(T)} = \boldsymbol{C_{17}} \boldsymbol{exp} \left( {}^{-\boldsymbol{C_{18}}}/_{\boldsymbol{T}} \right) \qquad \textit{Equation 3.32}$$

where $C_7$ through $C_{12}$ are material parameters related to the kinematic hardening and recovery terms, $C_{13}$ through $C_{18}$ are parameters related to isotropic hardening and recovery, and $C_a$ and $C_b$ are parameters related to dynamic hardening and anisotropic hardening, respectively (Tucker 2009). Also, $J_2 = \frac{1}{2}\left(\underline{\sigma}' - \underline{\alpha}\right)^2$ and $J_3 = \frac{1}{3}\left(\underline{\sigma}' - \alpha 3\right)$where the deviatoric stress σ' is expressed in indicial notation as

$$\boldsymbol{\sigma'_{ij} = \sigma_{ij} - \frac{1}{3}\sigma_{ii}} \, . \qquad \textit{Equation 3.33}$$

Constants C1 through C18 are determined by macroscale experiments conducted at different temperatures and strain rates.

### 3.4 Damage

The damage evolution implemented in this model consists of three major components: void nucleation, void growth, and void coalescence. The damage evolution given by Horstemeyer (2001) is,

$$\boldsymbol{\phi = \left(\eta v + \phi_{pores}\right) C} \qquad \textit{Equation 3.34}$$

where η is the void nucleation, v is the void growth related to particles, $\varphi_{pores}$ is the void growth related to pre-existing pores, and C is the coalescence term.

The nucleation term, η, is the number of voids per unit volume. The evolution equation for nucleation used was developed and discussed in depth in Horstemeyer (2001) as,

$$\boldsymbol{\dot{\eta} = \left\|\underline{D}_d^p\right\| \left(\frac{d^{0.5} C_{coeff}}{K_{IC} f^{1/3}} \eta \left\{ a \left[\frac{4}{27} - \frac{J_3^2}{J_2^3}\right] + b \frac{J_3}{J_2^{3/2}} + c \left\|\frac{I_1}{\sqrt{J_2}}\right\| \right\}\right) exp \frac{C_{T\eta}}{T}} \qquad \textit{Equation 3.35}$$

where, d is the average particle size, $K_{IC}$ is the fracture toughness, f is the initial void volume fraction, a, b, c, $C_{coeff}$, $C_{T\eta}$ are material constants, $I_1$ is the first invariant of

Cauchy stress, and $J_2$ and $J_3$ are the second and third invariants of the deviatoric stress tensor.

The void growth term is then separated into two separate parts, the growth of voids nucleated from particles and the growth of existing pores. This model uses the McClintock (1968) void growth equation to describe the void growth related to nucleated voids from particles. The equation used in the model is

$$\dot{v} = \frac{\sqrt{3}R_0}{2(1-m)}\left[sinh\left(\sqrt{3}(1-m)\frac{\sqrt{2}I_1}{3\sqrt{J_2}}\right)\right]\|\underline{D}_d^p\| \qquad \textit{Equation 3.36}$$

where $R_0$ is the initial void radius and m is the McClintock void growth parameter. In order to describe the growth of existing pores, the Cocks and Ashby (1982) void growth term is implemented as

$$\dot{\phi}_{pores} = \left[\frac{1}{(1-\phi_{pores})^m} - (1-\phi_{pores})\right] sinh\left\{\frac{2(2m-1)}{2m+1}\frac{\sigma_h}{\sigma_e}\right\}\|\underline{D}_d^p\| \qquad \textit{Equation 3.37}$$

where m is the Cocks-Ashby void growth parameter, $\sigma_h$ is the hydrostatic stress, and $\sigma_e$ is the von Mises equivalent stress. The term $\sigma_h/\sigma_e$ is a measure of stress triaxiality.

The third component in the damage evolution is the coalescence term. In early forms of this model, the coalescence is described by a phenomenological relationship developed by Horstemeyer and Gokhale(1999) as

$$C = [C_{D1} + C_{D2}\eta v]\left(\frac{DCS_0}{DCS}\right)^{zz}(TC_{TC}) \qquad \textit{Equation 3.38}$$

where $C_{D1}$ is related to simple coalescence of two voids into one, $C_{D2}$ is related to the nucleation of small voids between larger voids that occurs during microvoid linking, $C_{TC}$ is a material parameter related to the temperature dependence of coalescence. This term

has since been modified to reflect experimental findings of Horstemeyer, Matalanis et al. (2000) by Allison (2009) as

$$\dot{C} = TC_{TC}\left(\frac{4d_0}{NND}\right)^{\zeta}\dot{\varepsilon} \qquad \textit{Equation 3.39}$$

where $d_0$ is the initial void diameter, NND is the average nearest pore neighbor distance, and $\zeta$ is a material parameter. This relationship implies that if two pores are more than 4 diameters apart from one another, there will be relatively low influence on one another in terms of coalescence. However, if the pores are closer than four diameters, coalescence can become a critical factor in the total damage evolution. It is this result that is shown in Horstemeyer, Matalanis et al. (2000) that inspired a coalescence of this form.

## 3.5 Numerical Implementation

The MSU ISV model has been implemented into various finite element codes, including ABAQUS (Bammann et al. 1993 and Horstemeyer, Lathrop et al. 2000). The implementation presented here is very closely related to the implementation presented in Horstemeyer, Lathrop et al. (2000). In order to implement this model into a finite element code, one major simplification is required; the deviatoric plastic rate of deformation, $\underline{\boldsymbol{D}}^{\boldsymbol{p}}$, in the recovery terms of the hardening rate equations 3.25 and 3.26, is replaced with the total rate of deformation, $\underline{\boldsymbol{D}}$. This replacement assumes that $\underline{\boldsymbol{D}} = \underline{\boldsymbol{D}}^{\boldsymbol{p}}$, which means that all deformation is deviatoric plastic deformation. While this assumption is acceptable for large strains, it can induce significant errors for small strain problems. This makes equations 3.25 and 3.26 become equations 3.40 and 3.41, respectively.

$$\underline{\boldsymbol{\alpha}} = \left\{h(T)\underline{\boldsymbol{D}}^{\boldsymbol{p}} - \left[\sqrt{\frac{2}{3}}r_d(T)\left\|\underline{\boldsymbol{D}}\right\| + r_s(T)\right]\|\underline{\boldsymbol{\alpha}}\|\underline{\boldsymbol{\alpha}}\right\}\left[\frac{DCS_0}{DCS}\right]^{z} \qquad \textit{Equation 3.40}$$

$$R = \left\{H(T)\underline{\boldsymbol{D}}^{\boldsymbol{p}} - \left[\sqrt{\frac{2}{3}}R_d(T)\left\|\underline{\boldsymbol{D}}\right\| + R_s(T)\right]|R|R\right\}\left[\frac{DCS_0}{DCS}\right]^{z} \qquad \textit{Equation 3.41}$$

The beginning of each step is to determine the values for the second and third deviatoric invariants, $J_2$ and $J_3$, from the previous step to update the hardening and recovery terms in equations 3.27 – 3.32. Then, trial values for deviatoric stress, kinematic and isotropic hardening, and the yield criterion are calculated by assuming a purely elastic step. These equations are implemented as equations 3.42 – 3.45.

$$\boldsymbol{\sigma^*_{n+1} = \sigma'_n \left(1 - \frac{\dot{\phi}\Delta t}{1+\phi}\right) + 2\mu(1-\phi)\underline{D}'\Delta t} \qquad \textit{Equation 3.42}$$

$$\boldsymbol{\alpha^*_{n+1} = \left\{1 - \left(r_d\sqrt{2/3}\left\|\underline{D}\right\| + r_s\right)\sqrt{2/3}\|\alpha_n\|\Delta t\right\}\alpha_n} \qquad \textit{Equation 3.43}$$

$$\boldsymbol{R^*_{n+1} = \left\{1 - \left(R_d\sqrt{2/3}\left\|\underline{D}\right\| + R_s\right)\sqrt{2/3}|R_n|\Delta t\right\}R_n} \qquad \textit{Equation 3.44}$$

$$\boldsymbol{\Phi^*_{n+1} = \sqrt{\frac{3}{2}}\left\|\underline{\xi}\right\| - (1-\phi)\left[R + Y + Vsinh^{-1}\left(\frac{\sqrt{2/3}\|D\|}{f(T)}\right)\right]} \qquad \textit{Equation 3.45}$$

In Equation 3.45, $(1-\phi)\left[R + Y + Vsinh^{-1}\left(\frac{\sqrt{2/3}\|D\|}{f(T)}\right)\right]$ is the yield radius and $\underline{\xi} = \underline{\sigma}^* - (2/3)\underline{\alpha}$. The equation for PHI is found by inverting the flow rule found in Equation 3.21 and taking the norm of both sides. If $\Phi^*_{n+1}$ is less than or equal to zero, then the assumption of a purely elastic step is valid, and the trial values of $\sigma^*_{n+1}$, $\alpha^*_{n+1}$, $R^*_{n+1}$ are updated as the actual values. If $\Phi^*_{n+1}$ is positive, the deviatoric plastic component of the strain must be found that satisfies Equation 3.46.

$$\boldsymbol{\int_{t_n}^{t_{n+1}} \underline{D}^p dt = \frac{\gamma}{\left\|\underline{\xi}\right\|}\underline{\xi},\ and \quad \int_{t_n}^{t_{n+1}} \left\|\underline{D}^p\right\| dt = \gamma} \qquad \textit{Equation 3.46}$$

It can be shown that γ is defined by Equation 3.47. Using this γ, it is possible to correct the trial values of $\sigma^*_{n+1}$ , $\alpha_{n+1}$, and $R_{n+1}$ by Equations 3.48 – 3.50 to give the actual values of deviatoric stress ($\sigma'_{n+1}$), $\alpha_{n+1}$, and $R_{n+1}$. With these corrected values, substitution into the inverted flow rule shows that the condition PHI = 0 is enforced.

$$\gamma = \frac{\|\xi\| - \sqrt{2/3}(1-\phi)\left(Y + R + V sinh^{-1}\left(\|\underline{D}\|/f\right)\right)}{2\mu(1-\phi) + \frac{2}{3}\left(h + H(1-\phi)\right)\left(\frac{DCS_0}{DCS}\right)^{zz}} \quad \textit{Equation 3.47}$$

$$\sigma'_{n+1} = \sigma^*_{n+1} - \frac{2\mu(1-\phi)\gamma}{\|\xi\|}\xi \quad \textit{Equation 3.48}$$

$$\alpha_{n+1} = \alpha^*_{n+1} + \frac{h\gamma}{\|\xi\|}\xi\left(\frac{DCS_0}{DCS}\right)^{zz} \quad \textit{Equation 3.49}$$

$$R_{n+1} = R^*_{n+1} + \sqrt{\frac{2}{3}} H\gamma\left(\frac{DCS_0}{DCS}\right)^{zz} \quad \textit{Equation 3.50}$$

Using the corrected value of R, the total effective plastic strain can then be computed according to Equation 3.51.

$$\bar{\varepsilon}^p_{n+1} = \bar{\varepsilon}^p_n + \sqrt{2/3}\,\gamma \quad \textit{Equation 3.51}$$

Next, using the updated pressure term located in Equation 3.52, the total stresses (Equation 3.53) and invariants can be found. Using the total stresses, an equivalent stress can be calculated as in Equation 3.54 so that the damage terms can be updated.

$$p_{n+1} = \frac{1}{3}tr(\sigma_n)\left(1 - \frac{\Delta t\dot{\phi}}{(1-\phi)}\right) + (1-\phi)\Delta t K tr\left(\underline{D}\right) \quad \textit{Equation 3.52}$$

$$\sigma_{n+1} = \sigma'_{n+1} + p_{n+1} \quad \textit{Equation 3.53}$$

$$\sigma_e = \sqrt{3J_2} \quad \textit{Equation 3.54}$$

From Equations 3.35 – 3.37, the updated equations for nucleation, void growth from nucleation, and void growth from existing pores become:

$$\eta_{n+1} = C_{coeff} exp\left(\frac{\bar{\varepsilon}^p_{n+1} d^{0.5}}{K_{IC} f^{1/3}\sqrt{2/3}}\left\{a\left[\frac{4}{27} - \frac{J_3^2}{J_2^3}\right] + b\frac{J_3}{J_2^{3/2}} + c\left\|\frac{I_1}{\sqrt{J_2}}\right\|\right\}\right) * exp\frac{C_{T\eta}}{T} \quad \textit{Equation 3.55}$$

$$v_{n+1} = \pi\left[R_0 exp\left\{\frac{3\bar{\varepsilon}^p_{n+1}}{2\sqrt{2}(1-n)} sinh\left(\frac{\sqrt{3}}{2}(1-n)\left(\frac{2p}{\sigma_e} + \frac{1}{3}\right)\right)\right\}\right]^2 \quad \textit{Equation 3.56}$$

$$\boldsymbol{\phi_{pores}}_{n+1} = \mathbf{1} - \left[\mathbf{1} + \left\{\left(\frac{\mathbf{1}}{(\mathbf{1}-\boldsymbol{\phi_{pores}})^{m}} - (\mathbf{1} - \boldsymbol{\phi_{pores}})\right)\right.\right.$$

$$\left.\left. * \boldsymbol{exp}\left(\boldsymbol{\Delta t \dot{\varepsilon}^{p}}(\mathbf{1}+\boldsymbol{m})\boldsymbol{sinh}\left\{\frac{\mathbf{2(2m-1)}}{\mathbf{2m+1}}\frac{p}{\sigma_e}\right\}\right)\right]^{\frac{1}{1+m}} \qquad \textit{Equation 3.57}$$

The above implementation is how the MSU ISV model is used in an ABAQUS explicit, user-defined material model, or vumat. The growth from nucleated voids term, described in Equation 3.57, varies slightly from the form of Equation 3.36, the McClintock void growth equation. This form is found in McVeigh and Liu (2008) and is structured like finding the area of a two dimensional void; this differs from the three dimensional form found in Jordan (2006, 2007) and Horstemeyer, Lathrop et al. (2000). Equation 3.59 shows the three dimensional form of the McClintock void growth equation, and it can be seen that it is structured like the volume of a sphere rather than like the area of a circle like Equation 3.57.

$$\boldsymbol{v_{n+1}} = \frac{\mathbf{4\pi}}{\mathbf{3}}\left[\boldsymbol{R_0 exp}\left\{\frac{\mathbf{3}\bar{\boldsymbol{\varepsilon}}^{p}_{n+1}}{\mathbf{2(1-n)}}\boldsymbol{sinh}\left(\sqrt{\mathbf{3}}(\mathbf{1}-\boldsymbol{n})\left(\frac{\sqrt{\mathbf{2}I_1}}{\mathbf{3}\sqrt{J_2}}\right)\right)\right\}\right]^{\mathbf{3}} \qquad \textit{Equation 3.58}$$

In addition to the calculations in the explicit implementation of the DMG material model, the implicit implementation also requires the calculation of a tangent modulus at the end of each increment. It was shown by Nagtegaal (1982) and Simo and Taylor (1985, 1986) that the Newton-Raphson method for satisfying global equilibrium loses quadratic convergence with the use of a continuous tangent operator. For use with the DMG material model, Hammi and Horstemeyer (2007) derived and implemented a tangent operator that is consistent with the numerical incrementation used in the local constitutive equations in order to improve the quality of global and local convergence.

# CHAPTER IV

# IMPLICIT SIMULATION RESULTS

## 4.1 Introduction

In this chapter, the implicit finite element program ABAQUS Standard will be used to first verify that the DMG material constants are correctly fit to the experimental data for cast AM60B, then to simulate the notch tensile tests. ABAQUS Standard uses an implicit solver to satisfy the momentum balance equations. In general, an implicit integration scheme uses data from the previous step as well as values from the current step to approximately solve the system of equations (Reddy 2004), and as such, this solution approach is iterative. In implicit finite element codes, an initial guess is found using the tangent stiffness from the previous step and then ABAQUS iterates until equilibrium is met within a given tolerance. According to Reddy (2004), implicit schemes are unconditionally stable but the accuracy breaks down when the time step increases with respect to the period of response of the system.

A study similar to that presented here was performed in Weiler et al. (2005). As in this study, Weiler et al. used X-ray CT scans to determine the porosity distribution of a tensile specimen. They then used a critical local strain model to predict the plane of fracture, fracture strain, and fracture stress. Key differences between the approach in this study and that of Weiler include the fact that this study is performed on notched tensile specimens and utilizes a material model capable of tracking the state of the specimen. The importance of the notched specimen is that the notch introduces stress and triaxiality

gradients in the specimen. Unlike Weiler, damage evolution in the DMG model takes these gradients into account.

## 4.2 Calibration

In order to have confidence that the simulation will be correctly predicting the material response on the scale of the entire notched Bridgman tensile specimens, simulations were run to ensure that the model would capture the stress versus strain response of the real material as well as adequately predict the evolution of damage in the material. Figure 4.1 compares a quasi-static simulation to published experimental data (Horstemeyer et al. 2007) for the constants used in this study.

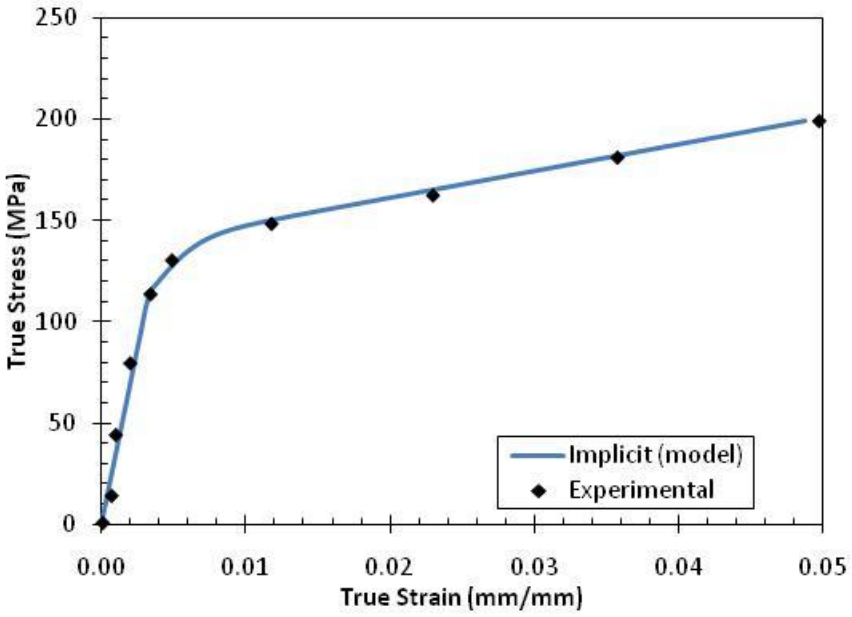



Figure 4.1 Comparison of the implicit finite element simulation to uniaxial tension experimental data.

### 4.2.1 DMG Material Constants

This figure shows a very good correlation between the mechanical response of the simulation and that of the real material. The material constants used in this study were

calibrated in Horstemeyer et al. (2007) and are given in Appendix A. These constants are used here with only minor recalibration to constant, C5. Figure 4.2 illustrates both the motivation and the result of this change in C5. The geometry in the simulations shown in Figure 4.2 is a cube that is 1 mm on each side. Each curve represents one uniaxial tension simulation at a constant engineering strain rate. The only difference between different curves on each graph is strain rate, and the only change between the two graphs is the value used for C5. The green curves in Figure 4.2 correspond to the strain rate used in the experimental testing. It can be seen in Figure 4.2a that the constants provided will slightly over-predict the stress for the QS case, but the response in Figure 4.2b closely matches the experimental data.

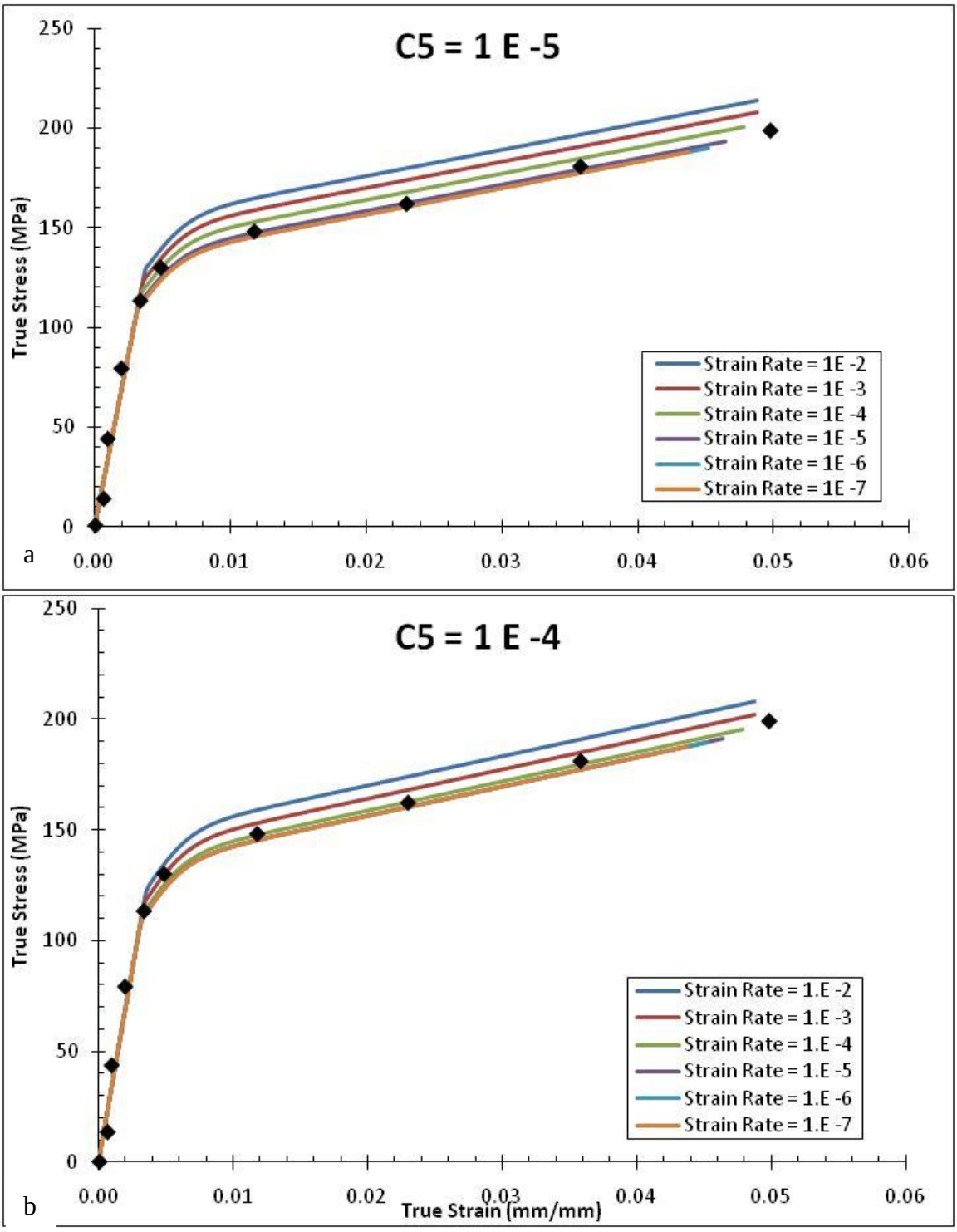



Figure 4.2 Comparison of simulations run with different values of C5. When C5 = 1E-5, the simulation predicts elevated stresses for quasi-static test (top) and when C5 = 1E-4, the simulation predicts stresses corresponding well with the experimental data (bottom).

Figure 4.2a shows that in order for the model to reach its quasi-static, rate independent limit with C5 = 1 x $10^{-5}$, the strain rate must be no greater than 1 x $10^{-5}$ $s^{-1}$. Since the experimental data shown was performed quasi-statically at a strain rate of 1 x $10^{-4}$ $s^{-1}$, the constant C5 was recalibrated such that the model will have approached its quasi-static, rate independent limit at the lower bound of available experimental data (which is 1 x $10^{-4}$ $s^{-1}$). After recalibration, C5 = 1 x $10^{-4}$. Figure 4.2b illustrates the change that this C5 has on model response. It can be seen that the new material constant allows for the model to reach rate independent limit at the same strain rate as the quasi-static experiment used to calibrate the model.

### 4.2.2 Damage Evolution Response

Since the purpose of this study is to evaluate the MSU-ISV models ability to predict the evolution of damage in a part, simulations were run to show percent elongation to failure as a function initial area fraction of porosity. Figure 4.3 shows that the model is capable of capturing the general trend of damage progression in the experimental data for a single element. Since cast AM60 contains large amounts of porosity initially, it is assumed that the initial porosity is the controlling feature in the overall damage evolution and therefore nucleation of new voids and the subsequent growth of the new voids is neglected. Therefore in Figure 4.3, all of the damage in each data point comes solely from the growth of initial pores and the coalescence of those pores.

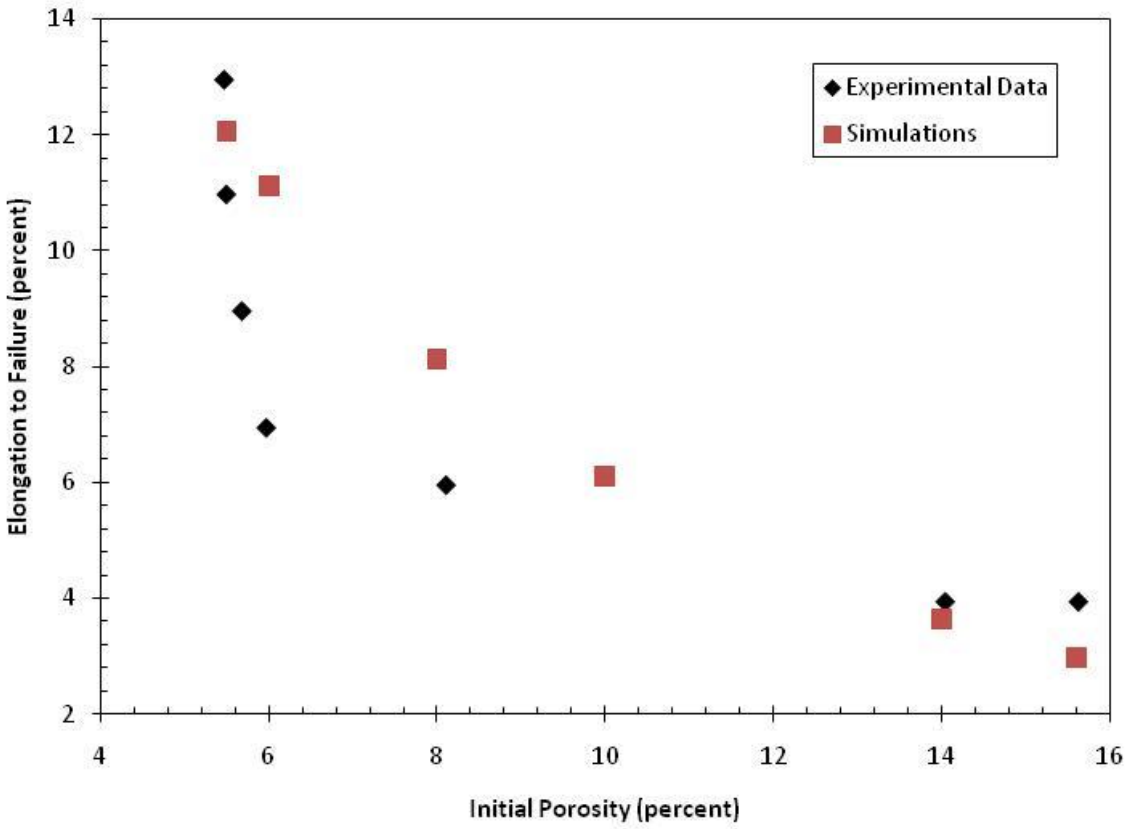



Figure 4.3 Comparison of initial porosity versus elongation to failure for the implicit simulations and the experimental data.

### 4.2.3 Time Step Size Evaluation

This series of simulations was performed in order to ensure that the time step used in the ABAQUS implementation of the problem was sufficiently small to ensure accuracy of the predicted results. Figure 4.4 shows that as long as the time step for this particular simulation is lower than 6 seconds, the implicit solution will not vary with changing time step. However, for time steps larger than 6 seconds, the solution begins to oscillate around the converged solution. If the time step is only slightly higher than the stable threshold, the solution will oscillate near the transition from elastic to elasto-plastic response in the data and then return to the stable solution, but if the time step is about 10 seconds or higher, at 6% true strain the solution continues to oscillate without damping.

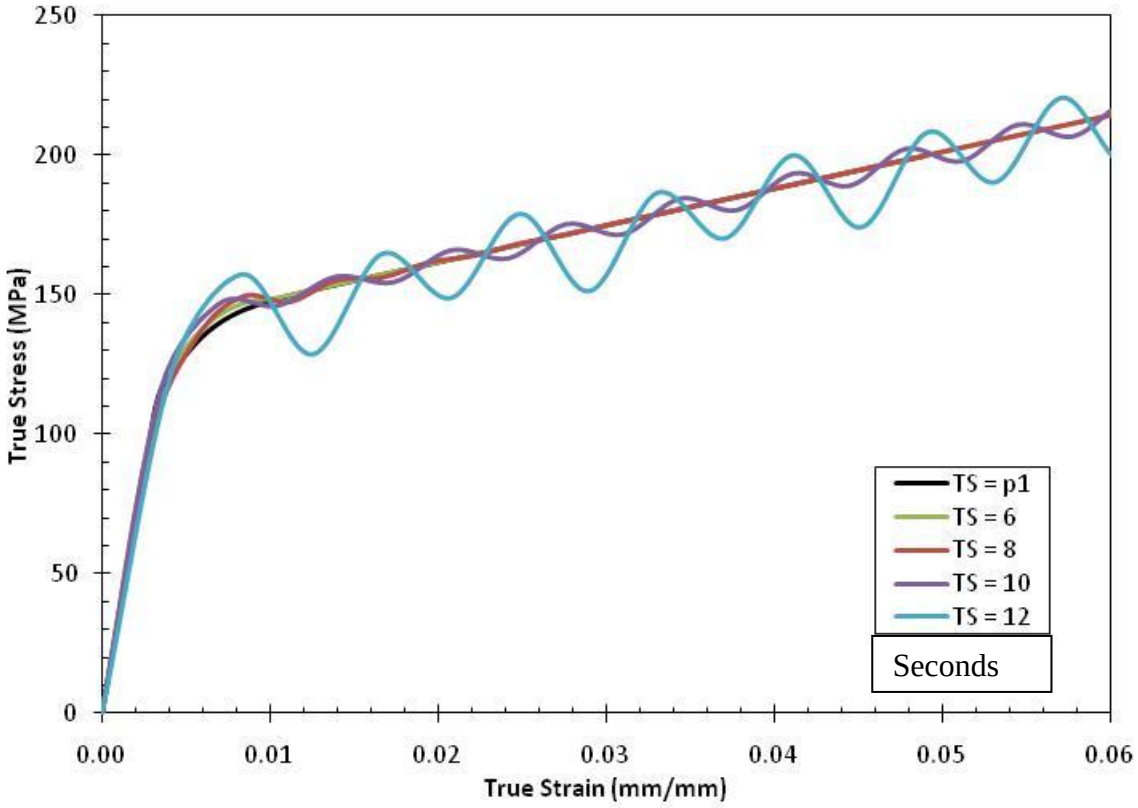



Figure 4.4 This figure shows a series of simulations designed to determine the accuracy of the implicit solution with various time steps.

### 4.3 Notch Bridgeman Simulation Results

Using the model and boundary conditions presented in Section 2.3, simulations corresponding to experiments run by Amy Waters (2001) are presented. In Section 4.3.1, the results of the experimental data are compared to the predicted material behavior from an ABAQUS Standard finite element simulation. Section 4.3.2 explores the effect of various initial porosity distributions on the predicted evolution of damage within a specimen.

### 4.3.1 Comparison of Simulation to Experimental Results

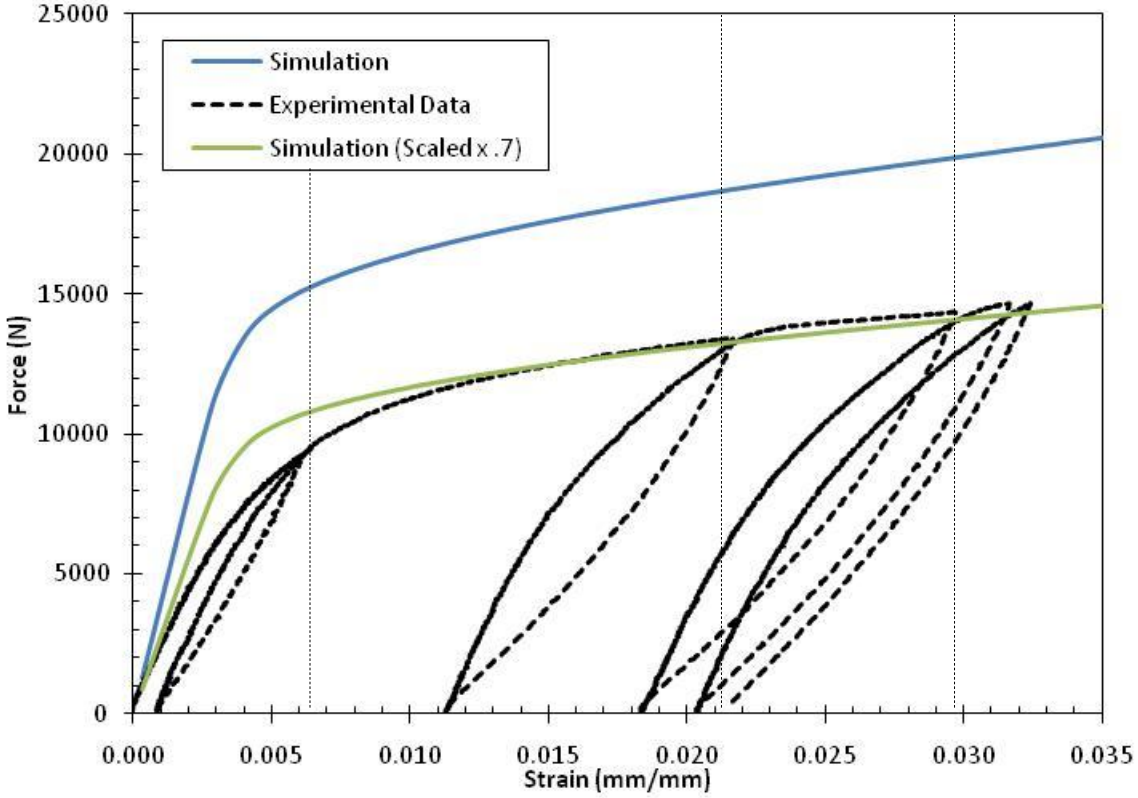



Figure 4.5 The black curve shows the mechanical response of the experimental tensile test. The blue curve is the mechanical response of the finite element simulation. The green curve is equal to the blue curve with the force values multiplied by 0.7. The dashed lines represent, from left to right, 60%, 87%, and 93% of the total load; the solid line represents the first element failure in the finite element simulation.

Figure 4.5 shows the loading history of both the simulation and the experimental data. Marked by vertical dashed lines are the strains represented by 60%, 87%, and 93% of the total "failure load". For this case, the failure load is considered to be an averaged ultimate tensile strength, 207 MPa, as stated in Table 2.1. The strains represented by 60%, 87%, and 93% of the failure load are 0.6%, 2.06%, and 2.81%, respectively. Figures 4.6 -4.9 illustrate the evolution of porosity in the experiment alongside the

simulated specimen. Given a cross sectional area at the notch of 74 $mm^2$, the experimental data predicts a yield stress of approximately 125 MPa. This value is the averaged stress across the minimum cross section of the specimen. Under the same assumptions, the simulation predicts a yield stress of about 170 MPa. As presented in Table 2.1, the average yield point of AM60B magnesium is 130 MPa. Therefore, as far as mechanical response is concerned, the simulation predicts stress levels that are consistently higher than the experimental data. If the value of force for the simulation were multiplied by a constant value of 0.7, the green curve is the result and the experimental data is adequately represented. So, while the values for force are very high, the shape of the curve for the simulation is basically the same as the experiment. In order to compare the damage evolution from the simulation to the experimental data, the strain levels indicated by the dashed lines in Figure 4.5 will be used.

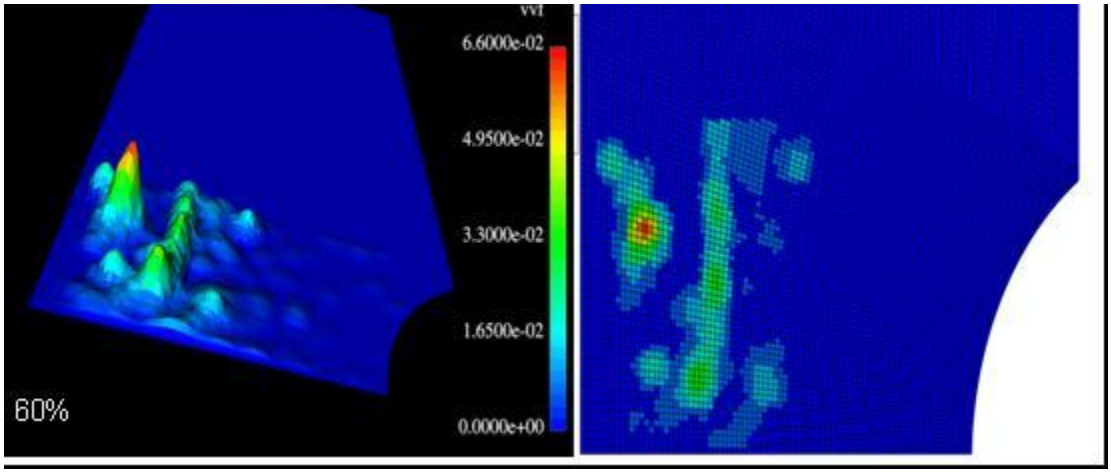



Figure 4.6 Damage levels at 60% of the failure load for the experiment (left) and the simulation (right).

From Figure 4.6, it can be seen that, to this level of strain, the damage distribution in the simulation closely resembles that of the experiment. The peak damage in both

cases is located at the isolated pore located to the left of the flow line in the figure. It should also be noted that at 60% load, there has only been a 1.7% change in the peak damage in the simulation from the initial damage state.

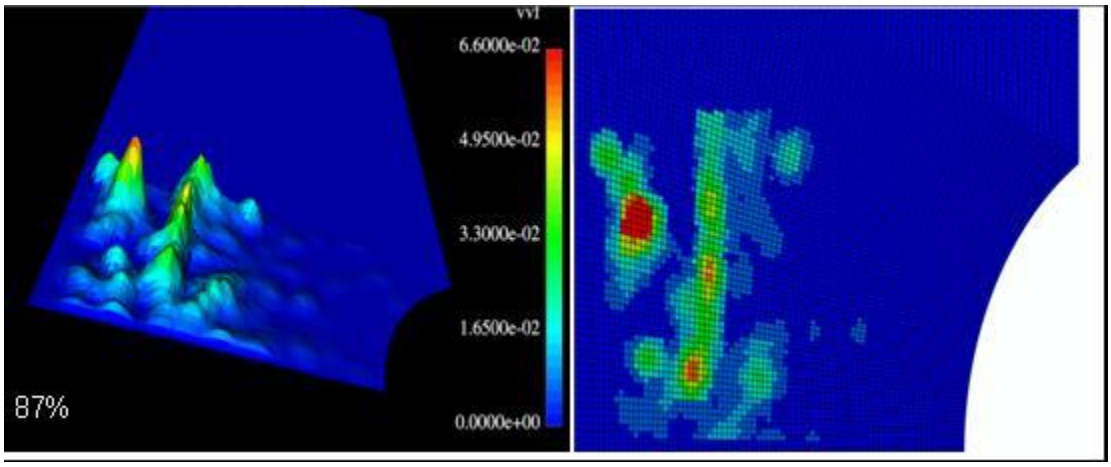



Figure 4.7 Damage levels at 87% of the failure load for the experiment (left) and the simulation (right).

Figure 4.7 shows that, by 87% load, the simulated damage has evolved considerably more than at 60% load. At this point, the simulation is over-predicting the damage evolution at the isolated pore relative to the flow line, according to the experimental data. The simulation, however, does still correctly predict the void growth trends, such as elevated growth rates around the pore and the flow lines.

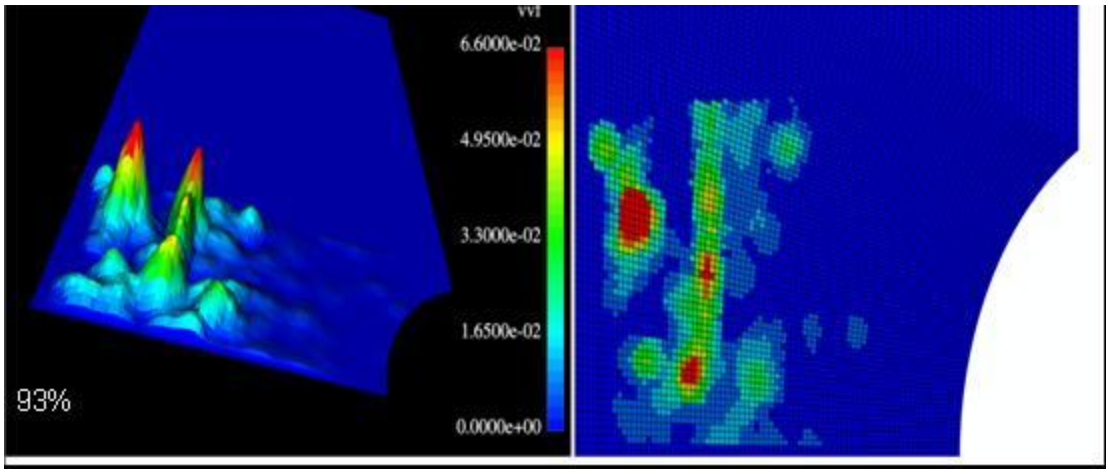



Figure 4.8 Damage levels at 93% of the failure load for the experiment (left) and the simulation (right).

Figure 4.8 show that the simulation continues to grow the porosity in the regions characterized by the pore and the flow lines much more than other regions of the specimen. This trend continues to the end of the simulation just after the first element failure. For this model, an element is considered failed when the damage goes to unity. The first element failure for this simulation occurs at the isolated pore and is shown in the damage contour in Figure 4.9. The first element failure occurred at 3.3% true strain. It can also be seen that the strain to first element failure corresponds to between 95 and 100% of the average "total load" as Figure 4.5 evaluates percent of total load.

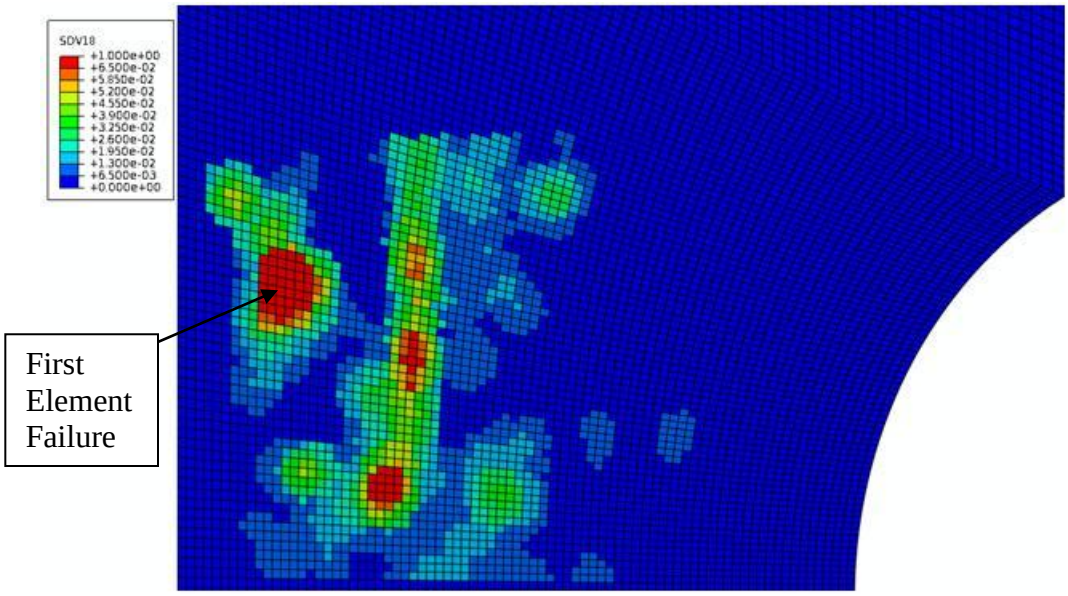



Figure 4.9 Simulation contour plot of the damage distribution at the time of the first element failure for the implicit finite element simulation of the notch Bridgeman tensile specimen.

Other than the porosity directly around the pore, the only damage to evolve to a significant value is that associated with the flow line. These two regions are significant because in the experimental data they are the same regions that experience the most damage. In the experimental results, the damage associated with the flow line experiences more growth than the damage associated with the large pore, but these results show that the model is capable of capturing the trend of elevated pore growth relative to the rest of the specimen in these two regions under the application of a uniaxial tensile load.

Figure 4.10 shows a contour plot of the pressure, von Mises equivalent stress, stress triaxiality, and plastic strain at just prior to the first element failure. The maximum values for each of these different measures of stress and strain occur at various locations around the specimen, however, none of the measures display a maximum at or near the location of first failure.

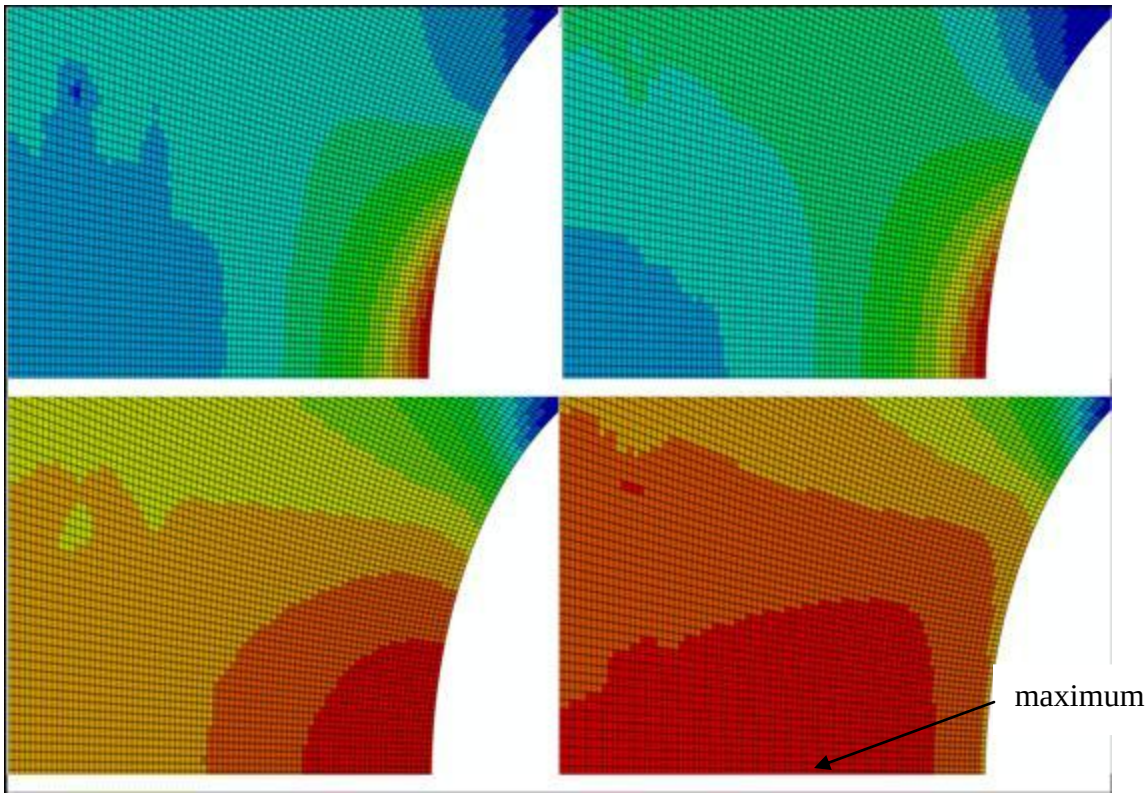



Figure 4.10 Contour plots of von Mises equivalent stress (top left), effective plastic strain (top right), pressure (bottom left), and stress triaxiality (bottom right) for the simulation of the notch Bridgeman tensile specimen with the actual initial porosity distribution.

It can be easily seen from Figure 4.10 that the maximum values for the plastic strain, von Mises stress, and pressure all exist near the notch root. The maximum stress triaxiality exists not at the notch root, but at the point labeled in the figure. In Agarwal et al. (2003), it is shown that pore growth is dependent upon both stress triaxiality and plastic strain; for the real distribution, it seems that the influence of the high porosity being concentrated around the pore and the flow line dominates the influence of plastic strain and stress triaxiality.

### 4.3.2 Effect of Initial Porosity Distribution on Evolution of Damage

After observing that, given the initial porosity distribution of a specimen, the DMG model is capable of predicting the general trends in the evolution of that porosity as the specimen undergoes deformation, a question arises regarding what effect, if any, the porosity distribution actually had on the predicted results in the previous section. That is, if the simulation were run with the same total area of porosity, but distributed in a different manner, what effect would there be on the predicted response of the specimen? In order to answer this question, two additional porosity distributions were developed, one with homogeneously distributed porosity in each element in the notched region and the other with randomly distributed damage. For both of these new porosity distributions, the void volume fraction for the entire part is identical to that of the porosity distribution in the previous section. Figure 4.11 compares the three different simulations by means of mechanical response.

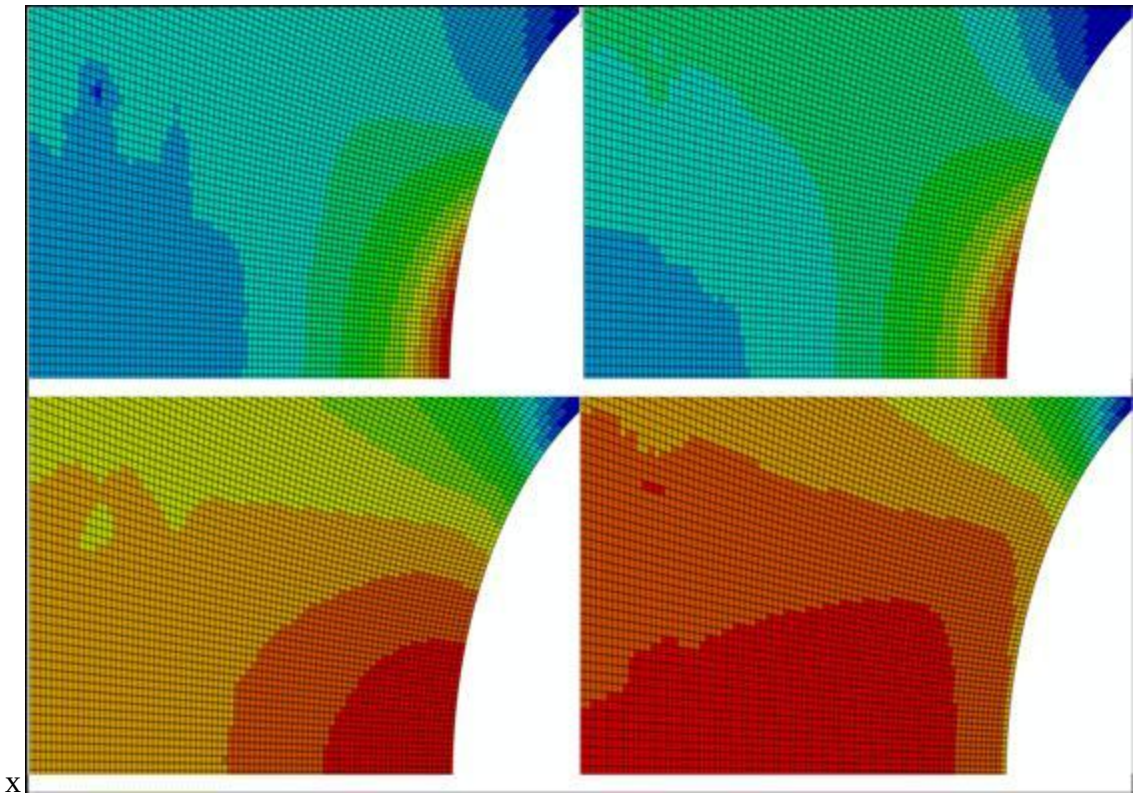



Figure 4.11 The graph shows the force versus strain response for each of the three porosity distributions, and the figures are damage contours taken as the first element failed in each simulation.

It is readily seen in Figure 4.11 that the force versus nominal true strain response for each of the simulations follows the same path regardless of the initial porosity distribution. The difference in the three cases is that the randomly and uniformly distributed cases exhibited first element failure at a much larger strain. Also included in Figure 4.11 are damage contours taken at first element failure for each simulation. It is obvious that the damage evolution in each case is very different. For the real distribution, the damage evolution is highest around the pore and the flow line. In the random distribution, the damage evolution seems to generally increase as the distance from the notch root decreases, but non-uniform initial damage causes non-uniform damage

evolution. Finally, in the uniform distribution, the evolution of damage is focused on the notch root and radiates outward.

In the previous section, the fact that the real porosity distribution displayed increased void growth at the preexisting pore and flow line was discussed in depth; now the current damage state of two new distributions will be compared to the real distribution at the first element failure of the real distribution.

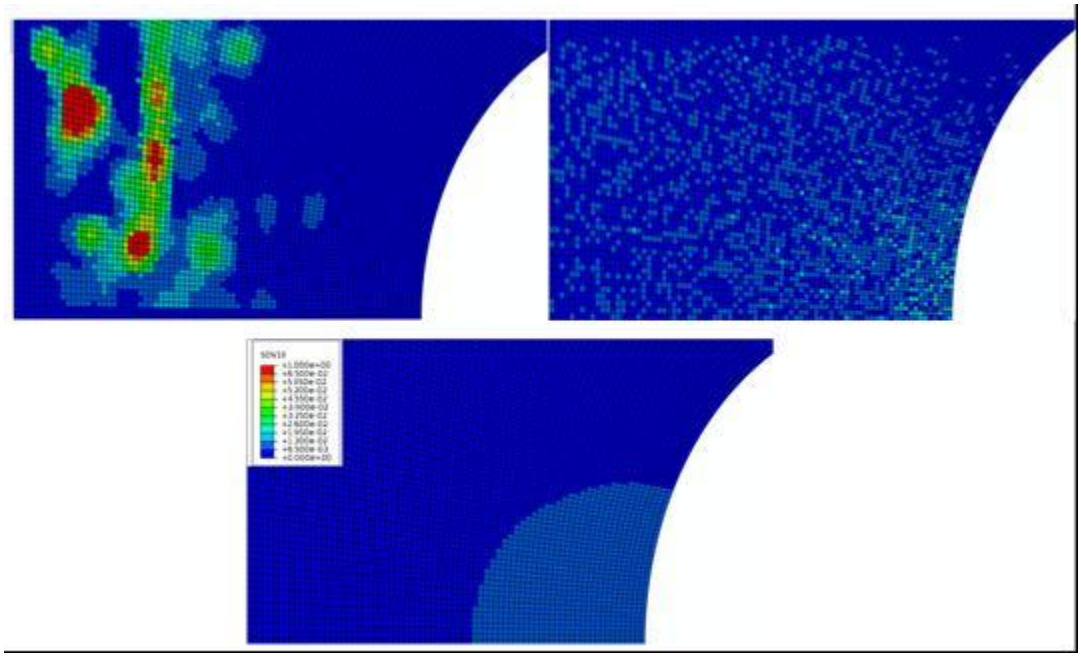


Figure 4.12 Comparison of the damaged states of the real (top left), random (top right), and uniform (bottom) porosity distribution simulations at 3.3% true strain, which corresponds to first element failure of the real porosity distribution simulation.

Figure 4.12 clearly shows that at the first element failure of the real initial porosity distribution, the random and uniform distributions are still far from failure initiation. In fact, at this point the maximum damage in the real distribution is 1 (first element failure); however, the maximum damage in the random and uniform distributions are 2.6% and 1.0%, respectively. Within the random and uniform cases, trends about the

growth of the voids can also be noted. For the random case, the concentration of elements with elevated damage levels gets much higher as the distance to the notch root decreases. However, there are some elements with damage levels near the maximum at this strain level that are located far from the notch. The damage level does not correspond directly to the position relative to the notch root. The same cannot be said about the uniform distribution. It can be seen that the damage growth seems to be radiating out from the notch root.

The next strain level chosen to compare the response of the two new porosity distributions is 8.8%. This strain corresponds to the first element failure for the randomly distributed porosity case. Figure 4.13, shows the damage distribution in the randomly and uniformly distributed porosity cases at the point of first element failure in the randomly distributed case.

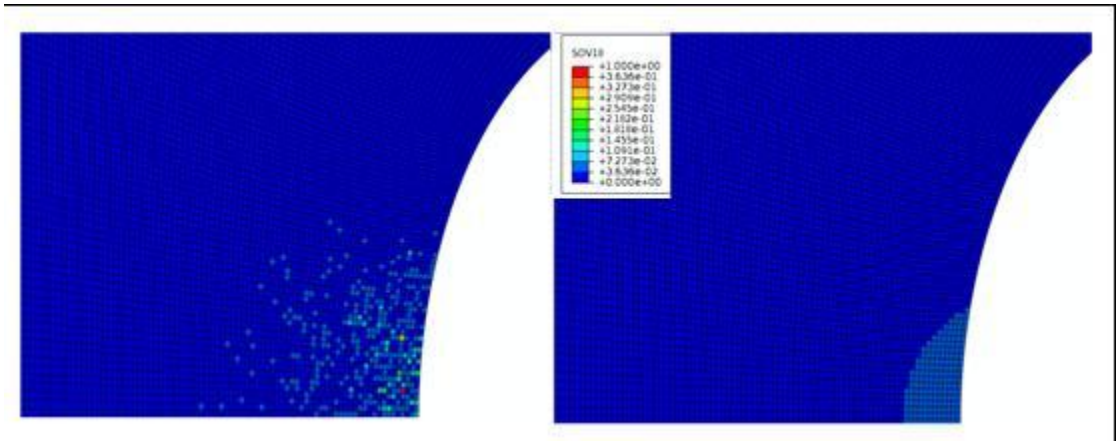


Figure 4.13 Comparison of the damaged states of the random (left) and uniform (right) porosity distribution simulations at 8.8% true strain, which corresponds to first element failure of the random porosity distribution simulation.

Figure 4.13 shows that the failure of the randomly distributed case occurs neither directly at the notch root nor at the element with the highest initial porosity. At this level

of strain, the uniformly distributed case still continues the trend of highest damage occurring at the notch root and decreasing as the distance from the notch increases. The maximum damage present in the uniformly distributed case is 4.5%.

Now, seeing that the element that fails first in the random distribution simulation, Figure 4.14 examines the state of the material in that simulation and to determine the cause for failure of the first element. As Figure 4.10 illustrates the state of the material just prior to the first element failure for the real damage distribution, Figure4.14 shows the same for the randomly distributed case.

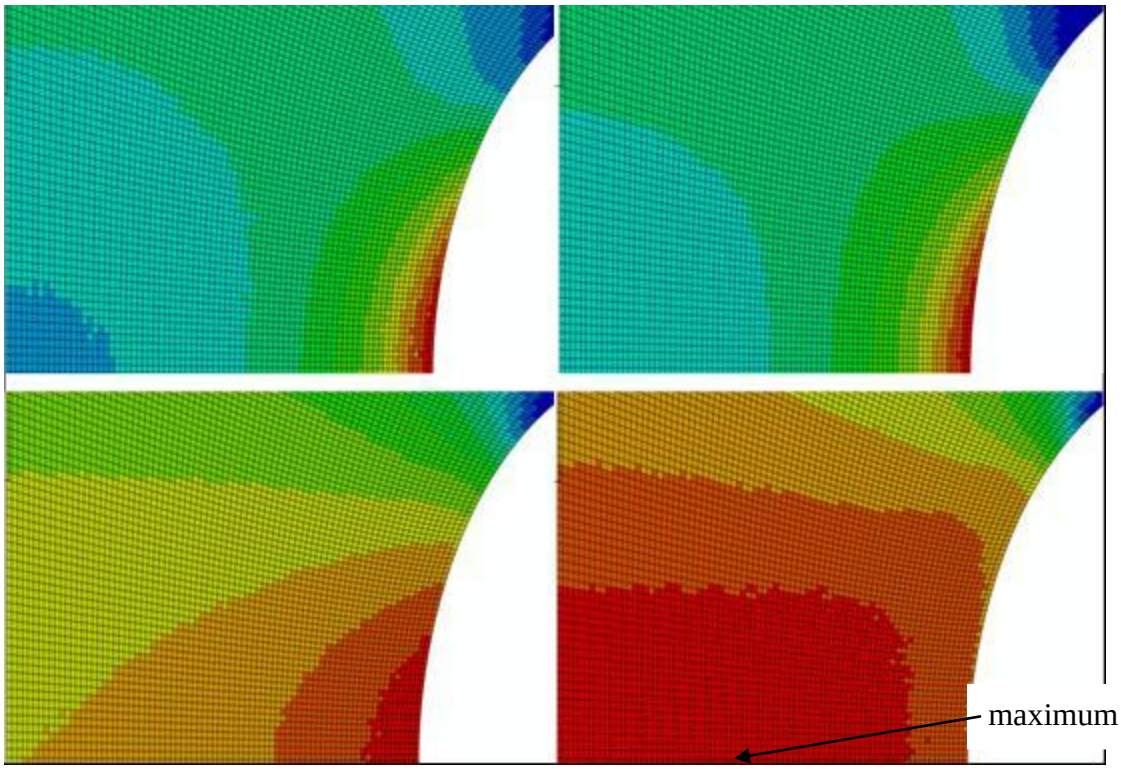



Figure 4.14 Contour plots of von Mises equivalent stress (top left), effective plastic strain (top right), pressure (bottom left), and stress triaxiality (bottom right) for the simulation of the notch Bridgeman tensile specimen with the random initial porosity distribution.

Figure 4.14 shows that the von Mises equivalent stress, the equivalent plastic strain, and the pressure are all maximum at the notch root, but the stress triaxiality is not. The maximum stress triaxiality occurs near the center of the specimen but not near the element that fails first. Therefore, for the case of randomly distributed initial porosity, the first element to fail is not located at the site of the maximum initial porosity, von Mises stress, equivalent plastic strain, pressure, or triaxiality. The element to fail first had slightly higher initial porosity than the other elements in its proximity, was in an area with high plastic strain levels, and was experiencing high stress triaxialities. So, while the first element failure did not correspond exactly to the maximum value for any of these measures, failure did occur at an element where porosity, plastic strain, and stress triaxiality were all elevated and working in tandem.

All that remains is to simulate the uniform distribution to first element failure and compare the results with those from the other two distributions. Figure 4.15 shows the von Mises equivalent stress, plastic strain, pressure, and stress triaxiality of the uniformly damaged case immediately prior to the first element failure.

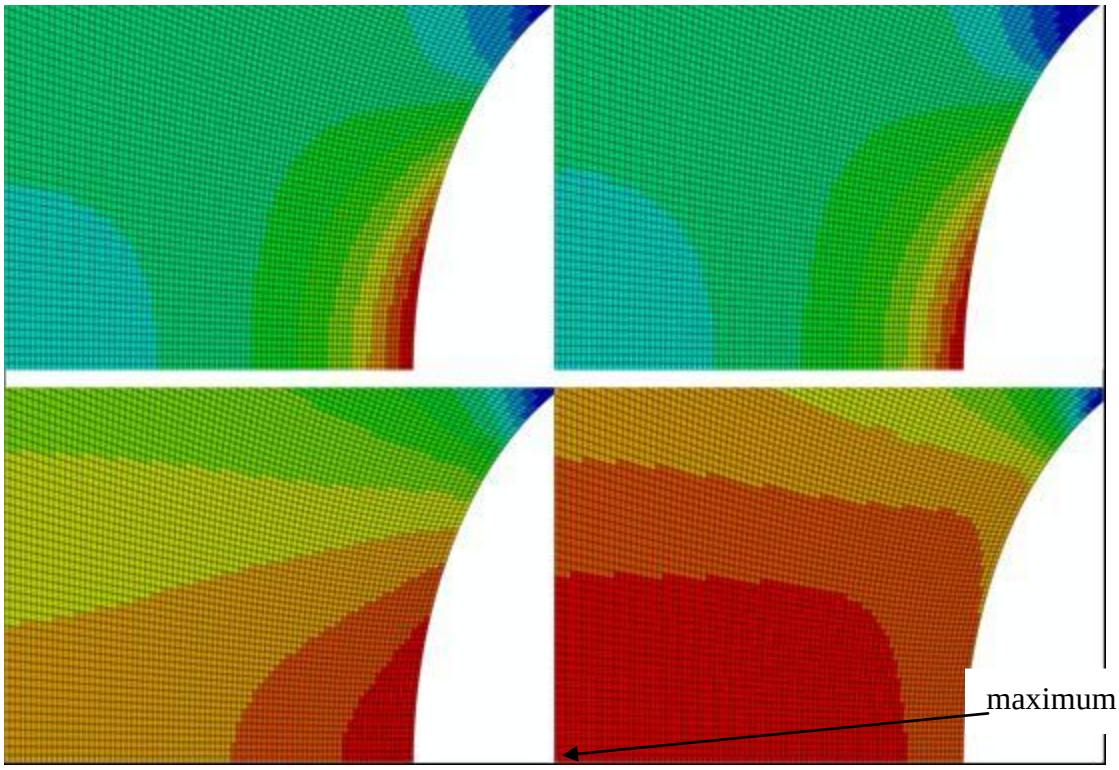



Figure 4.15 Contour plots of von Mises equivalent stress (top left), effective plastic strain (top right), pressure (bottom left), and stress triaxiality (bottom right) for the simulation of the notch Bridgeman tensile specimen with the uniform initial porosity distribution.

Like the other two cases, the uniform damage case exhibited a maximum von Mises stress, equivalent plastic strain, and pressure at the notch root, but the maximum stress triaxiality was at the direct center of the specimen. Therefore, if stress triaxiality was the governing factor in determining the damage evolution, then the damage near the center of the specimen would be elevated. So it is observed that in this model, stress triaxiality plays a secondary role in determining the growth of initial porosity. However, unlike the previous two cases, the damage, von Mises stress, equivalent plastic strain, and pressure all directly increase as the distance to the notch decreases. The predicted position of the first element failure for a uniform damage distribution is at the notch root.

## 4.4 Discussion

Several key observations can be drawn from the results presented in this chapter. The first observation is that using the DMG model to predict the damage evolution given the initial porosity distribution yields a result that follows the trends seen in the experimental data. Secondly, it was observed that the initial porosity distribution has no effect on the shape of the force versus strain curve for a notch Bridgeman tensile specimen. The only effect from the initial porosity distribution is on the damage evolution and the eventual first element failure of the specimen. That leads the third and final major observation of this chapter that the initial porosity distribution plays a major role in predicting the beginning of failure for the specimen.

In section 4.3.1, it was shown that the DMG model was capable of accurately predicting the major trends in the damage evolution when the initial porosity distribution is known. This was indicated by Figures 4.6 – 4.8, which show the side by side progression of damage in the experiment and simulations. The only two areas that the damage significantly evolved are the large pore and the axial flow line. In both the simulation and experiment, these two regions dominate the damage evolution for the notch Bridgeman specimen. That being said, it should also be noted that, while the model did accurately predict the two regions where damage would evolve most significantly, it did not predict the correct damage evolution within those regions. Specifically, the highest damage in the experiment is found in the pore, but it is very close to the damage level found in the top of the flow line. The simulation slightly over-predicted the damage at the pore and predicted elevated growth at the lower end of the flow line rather than the top of the flow line like the experiment. These small discrepancies between the experiment and the simulation notwithstanding, this simulation indicates that the DMG

model has the ability to effectively predict the evolution of damage in a part given the initial porosity distribution of that part.

The initial porosity distribution has no effect on the shape of the force versus strain response of the tensile specimen. The graph in Figure 4.11 clearly shows that the mechanical response for each of the three initial porosity distributions follows exactly the same curve. So, while the damage evolution varies greatly with the porosity distribution, the only effect that the porosity distribution has on the mechanical response is the point at which first element failure occurs. Section 4.3.2 details the differences between the damage evolution of the three simulations. The strain to first failure for the real distribution is 3.3%. Changing the same volume (area) of porosity into a random distribution yields a first element failure at 8.8% strain, an increase of 167% over the strain to first failure of the real distribution. If the porosity is averaged homogeneously, the strain to failure is 12.3%. That corresponds to an increase of 273% over the real distribution and a 40% increase over the random distribution. Therefore, the initial porosity distribution plays a vital role in simulating the mechanical response of a part near failure.

# CHAPTER V

# EXPLICIT SIMULATION RESULTS

## 5.1 Introduction

The results obtained from the ABAQUS Standard simulations predicted the trends in the experimental data, but were limited to the failure of the first element. In order to predict the component behavior beyond the first element failure, an explicit analysis is required. An explicit integration scheme is one in which the values are approximated using only known values from the previous step (Reddy, 2004 and Zienkiewicz, 1977). This simplification in the calculations, however, makes explicit schemes only conditionally stable. In this case, that means that the size of the time step is limited approximately to the time it takes an elastic wave to cross the smallest element dimension in the model.

## 5.2 Verification

With a given set of constants, and a sufficiently small time step, an implicit and explicit implementation of the same model should return the same material response for a given boundary value problem. In order to verify that the Umat and Vumat being used in ABAQUS were equivalent, the same simulation is run in both codes to verify that they return equivalent responses. Each simulation was a single 1 mm element loaded in uniaxial tension using the ABAQUS calculated time step size. For the explicit simulations in this chapter, double precision is always used. The results of those two simulations are compared to the experimental results in Figure 5.1. The explicit result

clearly shows a different response than the implicit implementation; both the slope of the elastic and elasto-plastic regimes are the same. The only observed deviation in the slopes of the stress strain curves found in Figure 5.1 is from approximately 0.3% to 1.0% true strain. It was shown in Figure 4.4 that the implicit solution shown in Figure 5.1 does not change with decreasing time step size and is therefore assumed to accurately represent the response of the DMG model.

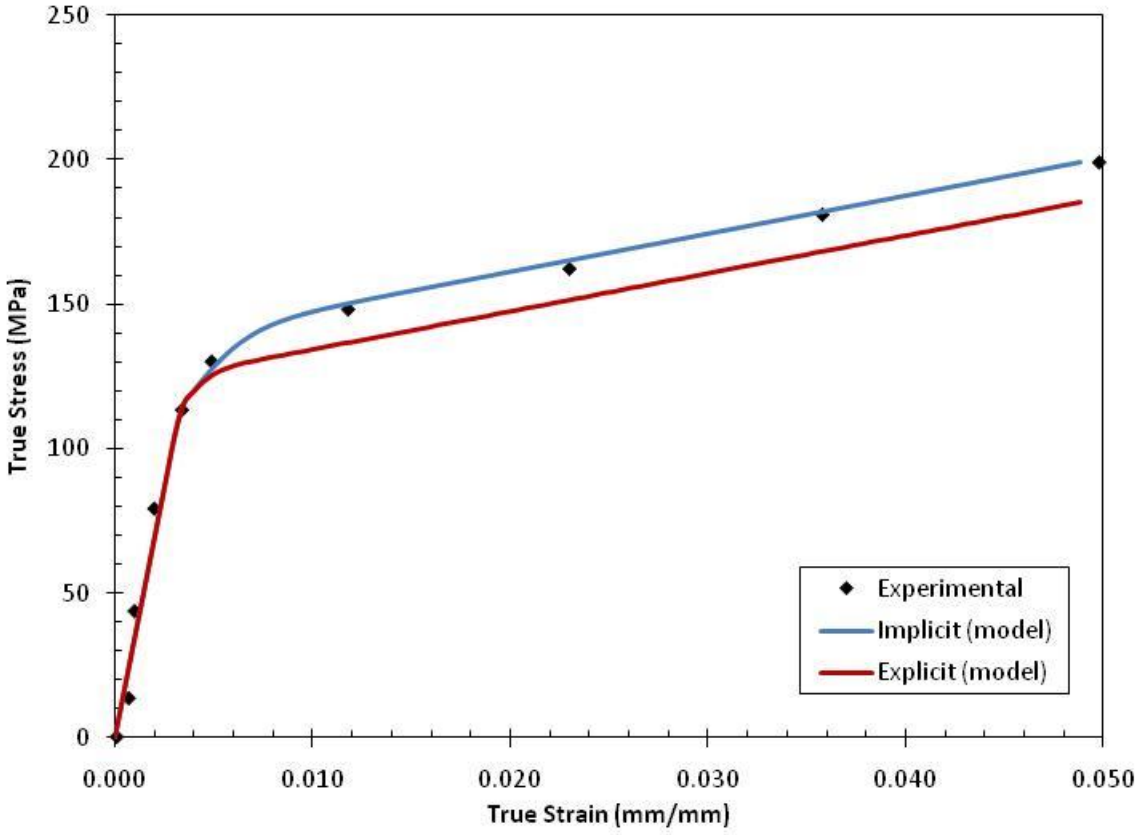



Figure 5.1 Comparison of the stress versus strain response of a 1mm cube as predicted by the implicit and explicit implementations of the DMG model.

Because the implicit solution uses an iterative scheme to minimize the error in the solution of the momentum balance equation, the implicit solution will be used as a standard for the remainder of this study. In this chapter, the word “error” will be used to

describe the difference between the implicit solution and whatever value is being analyzed for a given level of strain. Percent error is the error divided by the value of the implicit solution at that strain level.

In order to ascertain the source of the deviation, the values for all of the internal state variables were compared and it was found that they were all close in value except for the variables representing isotropic hardening. Figure 5.2 shows the plot of the isotropic hardening variable $\kappa$ for both the implicit and explicit implementation schemes. The difference between the total stress versus strain responses at any time after about 20 seconds is approximately 13.8 MPa, while the difference between the values of $\kappa$ for the implicit and explicit schemes is approximately 14 MPa. Therefore, the majority of the deviation can be attributed to this discrepancy in isotropic hardening.

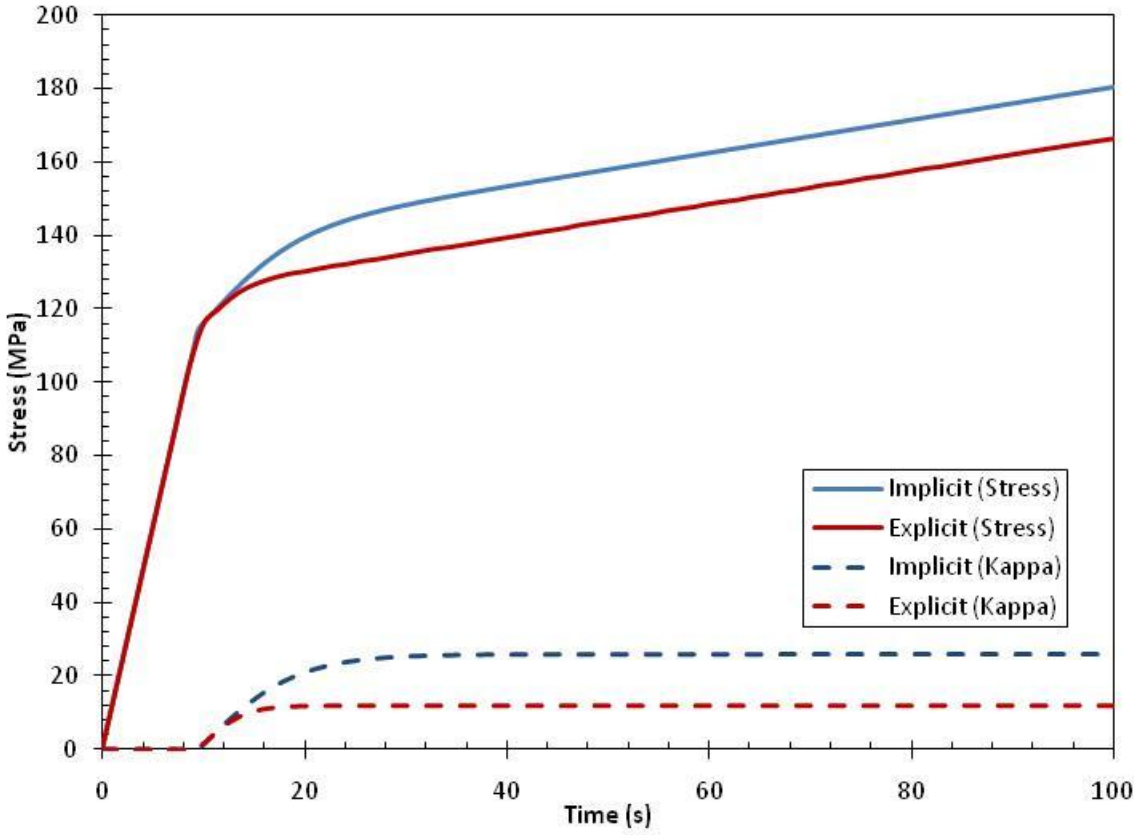



Figure 5.2 This illustrates clearly the difference in the evolution of the isotropic hardening variable kappa between the implicit and explicit schemes.

Once this was found, all of the plasticity and damage constants were zeroed out of the model and simulations rerun using both implicit and the explicit finite element codes (ABAQUS Standard and Explicit) in order to isolate the source of the deviation. The results were that the implicit and explicit solutions were identical. After this, the constants were systematically added back into the simulation until a deviation was observed. This process revealed that by setting the constant C1 equal to zero, all other constants could be retained with no adverse affects; specifically, with all constants added back into the model except C1, the implicit and explicit results are equivalent. Section 5.4 focuses on the effects of removing C1 from the simulation. However, removing C1 from the simulation removes the strain rate dependence of yield from the model, limiting the

functionality of the DMG model significantly. Therefore, some effort was spent on finding a way to preserve the use of all constants. So, another possible cause for the discrepancy between the implicit and explicit schemes is that the time step chosen by ABAQUS Explicit is not sufficiently small to accurately compute the response of the model. The remainder of this section will explore several methods used to check if and insure that the simulation was being run with a sufficiently small time step.

### 5.2.1 Reduce Time Step Size (Without Damage)

Increased integration accuracy of explicit integration schemes can be achieved by a reduction of the time step size. As such, simulations were run starting at the ABAQUS calculated time step, $\Delta t_0 = 1.117\ x\ 10^{-7}$, and lowering it to see if there is any improvement of simulation results. The time step was first reduced to $\Delta t_1 = 1.0\ x\ 10^{-7}$, an 10.5 % reduction in time step size. The one element simulation with time step size equal to $t_1$ took approximately 22 hours to complete on one processor. With this step size from simulation results were practically unchanged. Therefore, the time step was reduced further. However, without running any increments, for this set of constants, all time steps smaller than $\Delta t_1$ began returning a floating point error from ABAQUS.

### 5.2.2 Increase Element Size (Without Damage)

Because reducing the time step directly was not able to verify that the implicit and explicit solutions are equal, another method must then be found to verify that the implementations will converge to the same solution. According to Ling (2002), the size of the maximum stable time step for a problem is the maximum value of

$$\boldsymbol{t_s \leq a\sqrt{\frac{\rho}{K}} * f(\nu)} \qquad \textit{Equation 5.1}$$

for any element in the simulation. In Equation 5.1, a is the characteristic element size, $\rho$ is density, and K is the stiffness. $f(\nu)$ is a function of the Poisson's ratio and is dependent upon the type of element and whether a lumped or consistent mass matrix formulation is used. Therefore, enhanced accuracy may be achieved by increasing element size. The critical stable time step is the smallest stable time step for any element in the simulation. Since the minimum time step that can be used in ABAQUS, as stated in Section5.2.1, is larger than the time step required for a 1 mm element simulation to give accurate results, a series of simulations were run increasing the size of the element. In Equation 5.1, $\rho$, K, and $f(\nu)$ are material parameters and cannot be changed without changing the fundamental problem being solved. However, for a uniaxial tension test controlled by strain rate on a homogeneous cube, the size of the cube has no effect on the mechanical (stress versus strain) response. Therefore, increasing the size of the cube directly increases the maximum stable time step of the finite element problem without altering any material properties. Figure 5.3 shows the mechanical response of the series of simulations with increasing element size. It is immediately apparent that the stress versus strain response for the 25 mm element and the 100 mm element contain irregular behavior in the form of local bumps in the data.

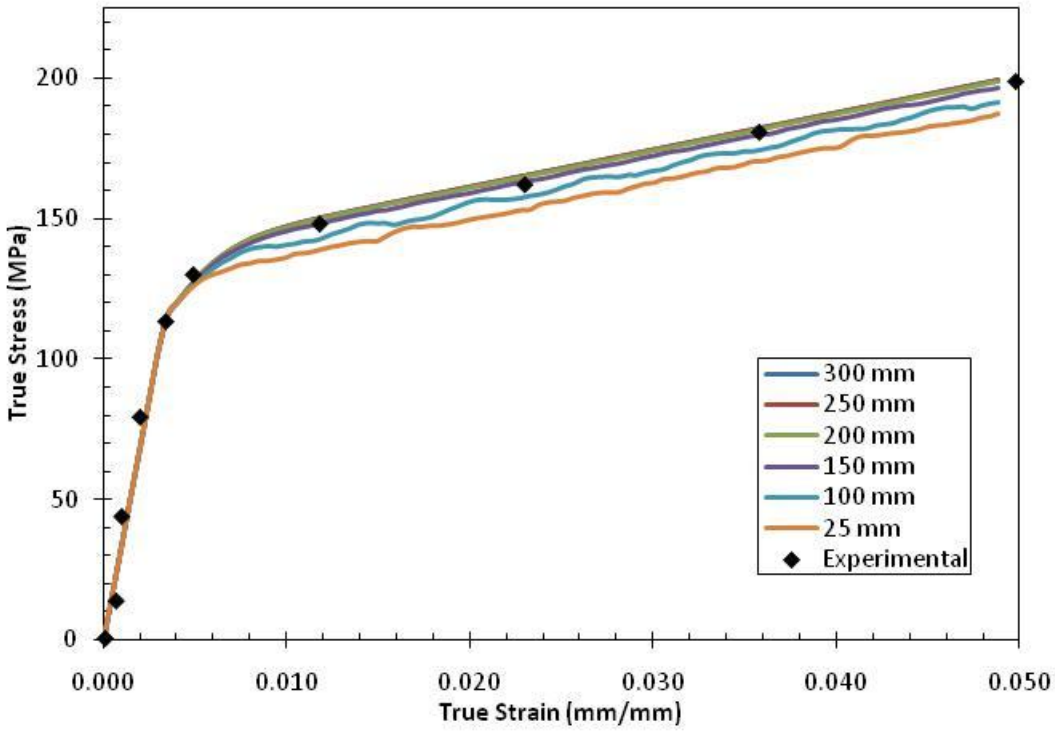



Figure 5.3 Stress versus strain responses of a series of simulations of increasing element size with a constant strain rate, boundary conditions, and material parameters.

From Figure 5.3 it can be seen that, as the size of the element increases under a constant time step and strain rate, the solution of the explicit problem approaches that of the implicit implementation scheme. For a time step of $1 \times 10^{-7}$ seconds and element sizes larger than 225 mm, the explicit results have converged to the same solution, which is approximately equal to the implicit solution. Figure 5.4 gives an idea of the rate at which the solutions converge as the element increases in size.

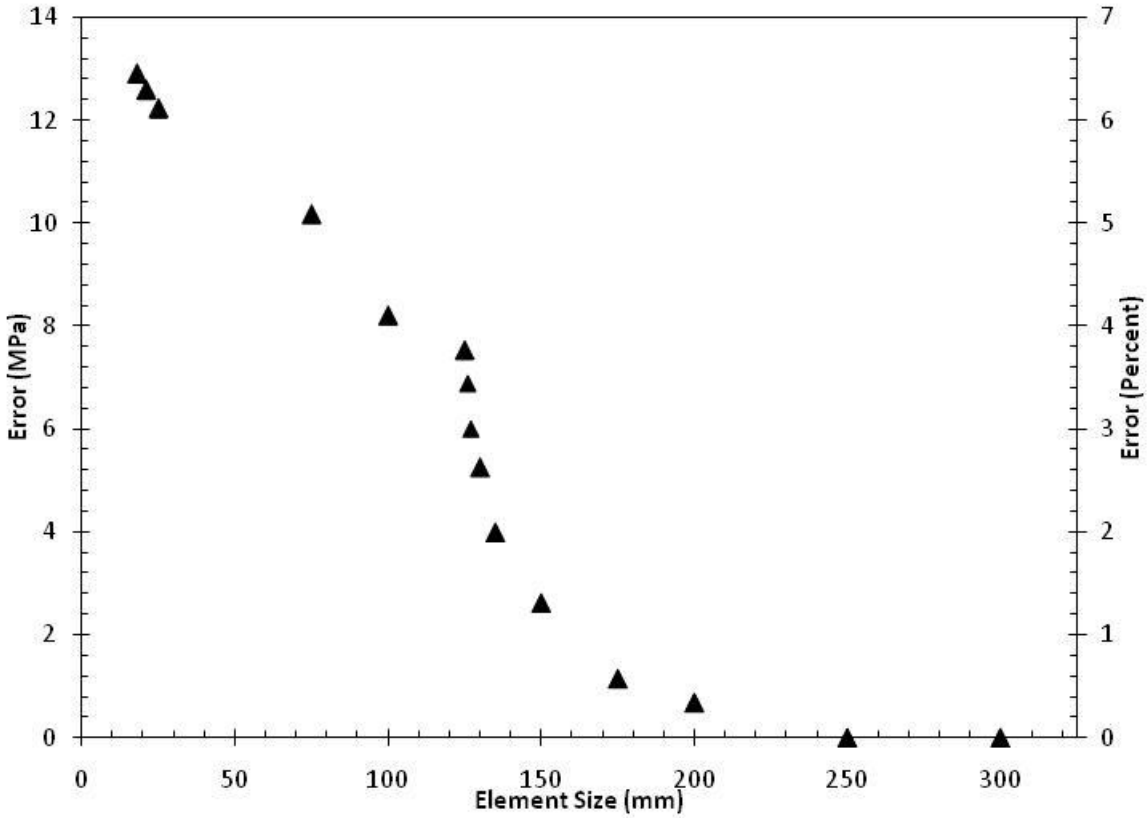



Figure 5.4 Plot showing the error (MPa and %) at the last step in each simulation.

Figure 5.4 shows on the left axis the error in stress for each simulation from the converged stress level at the final increment in the simulation; on the right axis, this error has been converted into percentage of the converged stress. For this series of simulations, the last increment corresponds to a true strain of 4.88% and a converged true stress of 199.4 MPa. It is stated above that the element size which corresponds to a fully converged solution is 225 mm; however, for engineering purposes, the error in the elements larger than 175 mm is approximately 0.5% or less and can therefore be sufficient. With elements larger than 125 mm, the results are convergent to the implicit solution. For elements smaller than 125 mm, however, the results are not smooth and seem to indicate some form of numerical noise. From this data, you can see that as the

element gets smaller (and therefore the stable time step), the error gets large. These results show that for a given time step size, increasing the element size allows the predicted mechanical response from the explicit finite element scheme to converge to the solution of the implicit finite element problem. It can be inferred from this result and the relationship stated in Equation 5.1 that, given a small enough time step size, the solutions of the implicit and explicit models will return the same results for any sized element. Now that it has been shown that the explicit model will converge to the true solution as the time step size tends to zero for the case with no damage, the damaged case will now be studied.

### 5.2.3 Increase Element Size (With Damage)

Because the damage evolution equations add nonlinearity to the approximation, the same process of increasing element size will be performed with a constant damage of 3% initial porosity to ensure that the results still converge. Elements are then increased in size until the solution converges. Figure 5.5 illustrates the mode of convergence of the simulations with damage.

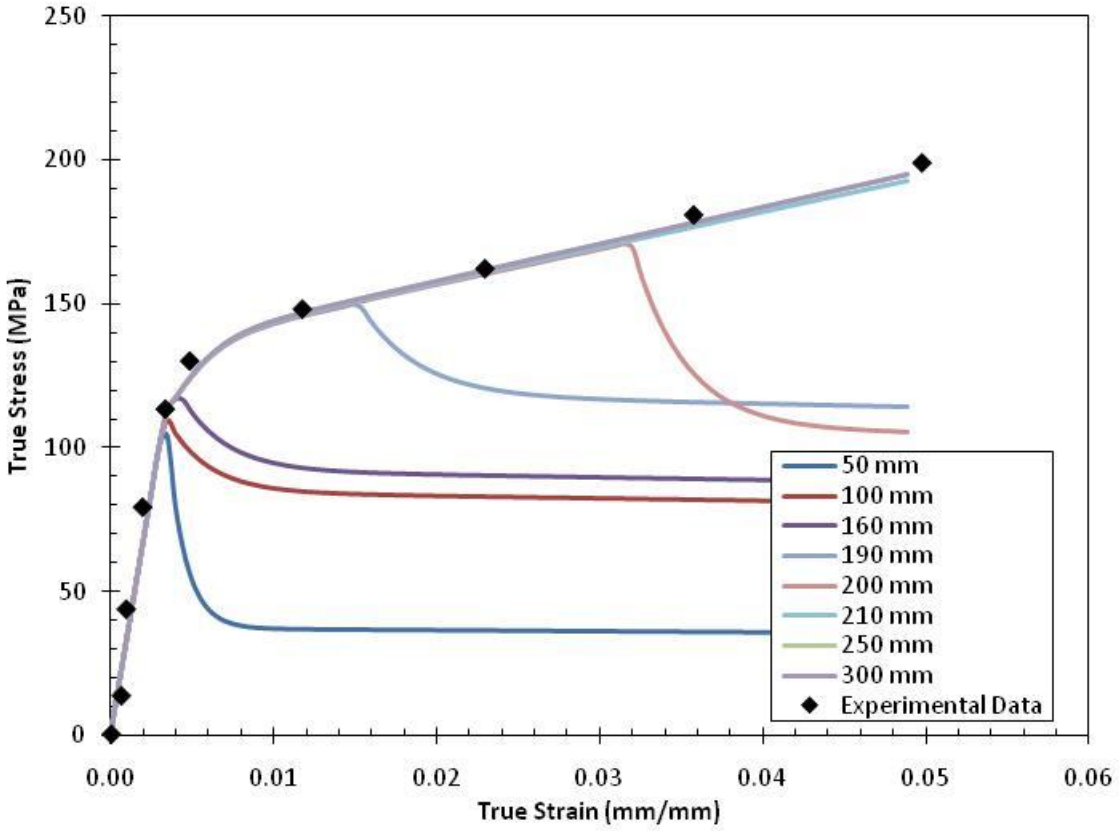



Figure 5.5 This figure shows the stress versus strain response of a series of explicit finite element simulations with an initial porosity of 3% and increasing in element size from 50 mm to 300 mm.

Figure 5.5 shows that for small element sizes, the simulation predicts significant softening in the stress strain response. As the element size increases, the fictitious softening diminishes and the stress strain response approaches the implicit solution. For elements that are 200 mm or smaller, the error from the converged solution is in the form of a severe drop in stress that seems to saturate at a different stress level for each size element. For elements larger than 200 mm, the solutions converge to the accurate solution in a similar manner to the undamaged simulations. This abrupt change in mode seems to indicate a shift from a fictitious softening or over-prediction of the radial return to a smooth, stable, and accurate response. Figure 5.6 shows the error in the maximum

stress for each simulation, and demonstrates that elements smaller than 200 mm have significant scatter. Figure 5.7 shows only the data for elements larger than 200 mm. Truncating the data to only include elements larger than 200 mm makes it easy to see that the data in Figure 5.7 is convergent to the accurate solution. At an element size of 250 mm, the solution is within .1 MPa of the converged solution to the problem. Comparing Figure 5.6 to Figure 5.4 above, it should be noted that the element size that corresponds to the limit of the convergent data is higher for the simulation containing non-zero damage. The "critical" element size for the simulations without damage is equal to approximately 125 mm and 210 mm for the simulations with damage. This indicates that for a given element size, a simulation with damage will require a smaller time step size for convergent results than a simulation without damage.

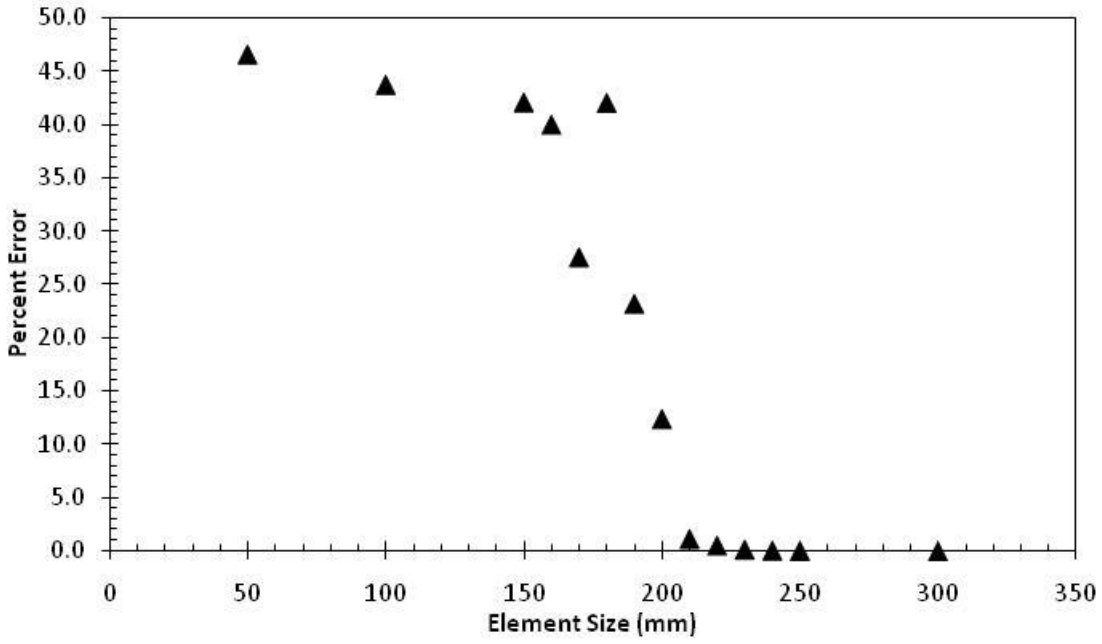



Figure 5.6 Plot showing the percent error between the stress at the last step in each simulation.

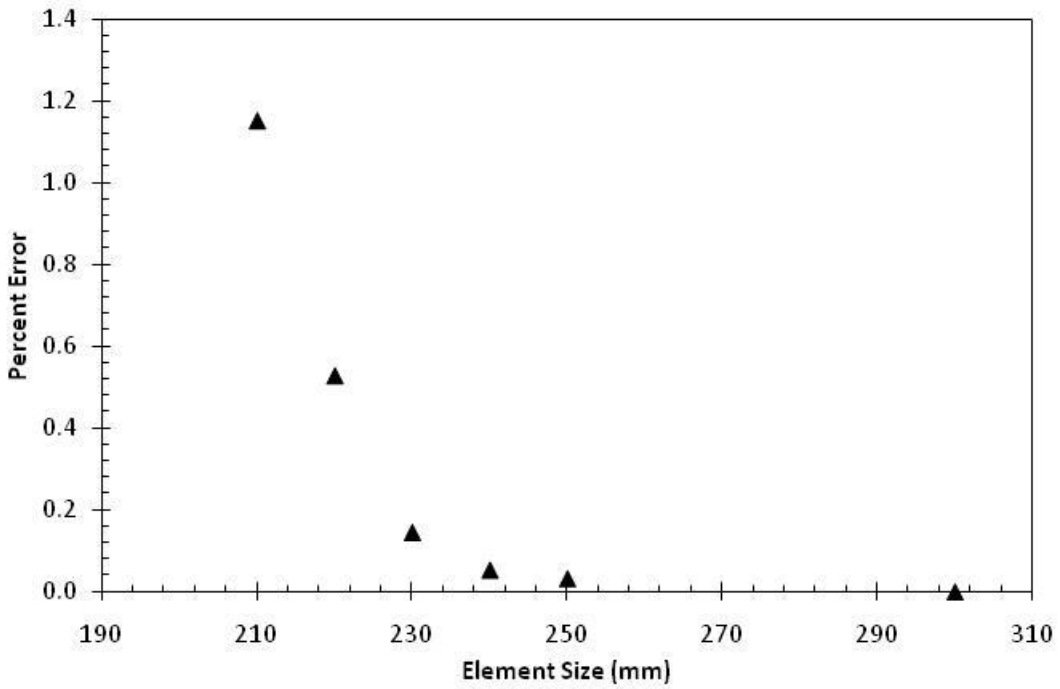



Figure 5.7 Truncated plot showing the percent error between the stress at the last step in each simulation. Each simulation contains 3% initial porosity, element sizes vary from 210 mm to 300 mm.

## 5.3 Fixed Mass Scaling

Because the required element size is too large for realistic problems, another method is required to make the models return similar results. Mass scaling will be studied. Equation 5.1 says that the stable time step is dependent on element size, material density, element stiffness, and Poisson's ratio. In the Section 5.2, it was shown that the explicit model gives accurate results given a very large element size or, potentially, a very small time step. Upon examination of Equation 5.1, the stable time step may be increased artificially by increasing the material density. The same change in time step could be accomplished by lowering the stiffness or possibly changing the Poisson's ratio, but alteration of these quantities would fundamentally change the behavior of the material, whereas artificially increasing the material may not significantly alter the

momentum balance for the case of quasi- static loading. As a check to verify that the mass scaling has only limited effect on the boundary value problem, the ratio of kinetic energy to total energy in the system will be considered. For a quasi static simulation, it is assumed that the kinetic energy is very close to zero, so in this study it will be checked to ensure that the kinetic energy is much smaller than the total energy of the system.

### 5.3.1 Mass Scaling Without Damage

As with the earlier sections, analysis will begin with the undamaged case and the damage will be added back in later. This section will study the effect of mass scaling on the explicit solution without the presence of damage. For this series of simulations, mass scaling is varied from no scaling, actual density, to a mass scaling factor of $1 \times 10^{15}$. The purpose of this range of values is to capture the entire range of responses related to artificially changing the mass of each individual element. Figure 5.8 shows the stress versus strain response for the range of scaling factors.

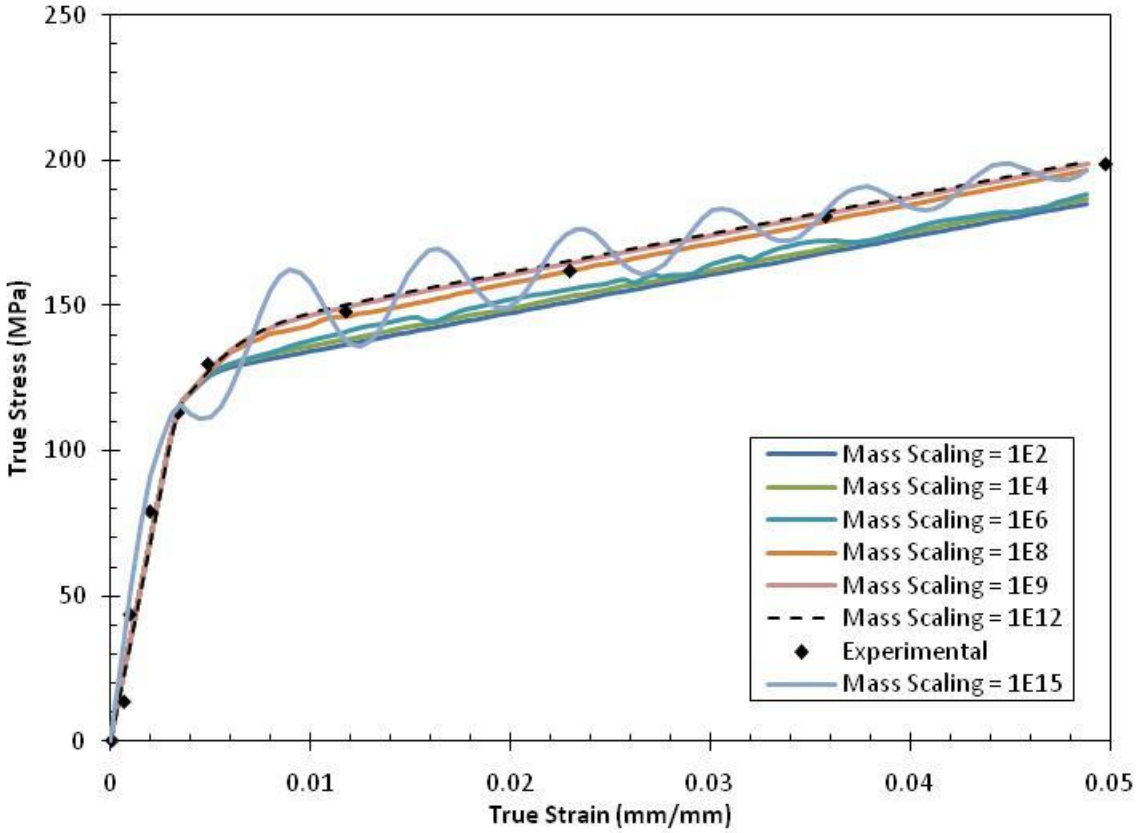



Figure 5.8 Stress versus strain responses for a series of simulations with increasing mass scaling factors. All simulations are run at the same strain rate and all elements are the same size.

Each of these simulations was run allowing ABAQUS to calculate the time step for each. As such, the time step is not constant for these simulations, but the element size (1 mm) and engineering strain rate ($3.5 \times 10^{-4}$ $s^{-1}$) are held constant. The explicit simulation with no mass scaling is approximately the same as the bottom curve on the figure with a mass scaling factor of 100. Increasing mass scaling from 10 to $1 \times 10^{8}$, the results tend to the implicit solution. As the mass scaling factor is increased from $1 \times 10^{9}$ to $1 \times 10^{13}$, there is very little change in the stress versus strain response of the element; all simulations in this range correspond closely to the converged solution from the element size study. When the mass scaling factor is raised further to $1 \times 10^{14}$, the

oscillating response seen in Figure 5.8 is observed. As the scaling is increased beyond this level, instability increases quickly. Therefore, for a 1mm, one element simulation with no damage, imposing a mass scaling factor between $1 \times 10^9$ and $1 \times 10^{13}$ causes the explicit and implicit implementations of the model to return similar stress versus strain responses. For quasi-static simulations, the kinetic energy should be negligible when compared to the total energy in the system. Introducing excessive mass scaling can artificially cause the kinetic energy to increase to levels that influence the results. Table 5.1 shows the kinetic energy, the total energy and the ratio of kinetic to total energy in each of the simulations. According to Table 5.1, quasi-static simulations with mass scaling less than 1E12 will have less than one percent of the total energy of the system in the form of kinetic energy. As the mass scaling factor increases from there, the amount of kinetic energy becomes significant and therefore could change the results of the simulation.

Table 5.1 Kinetic and total energy in each of the mass scaling simulations.

| Mass Scaling Factor | 1.00E+02 | 1.00E+04 | 1.00E+06 | 1.00E+08 | 1.00E+10 | 1.00E+12 | 1.00E+14 |
|---|---|---|---|---|---|---|---|
| Kinetic Energy (mJ) | 6.86E-15 | 6.86E-13 | 6.86E-11 | 6.86E-09 | 6.87E-07 | 8.46E-05 | 1.05E-02 |
| Total Energy (mJ) | 4.33E-03 | 4.33E-03 | 4.33E-03 | 4.33E-03 | 4.33E-03 | 4.27E-03 | 1.19E-02 |
| Ratio (KE / TE) | 1.60E-12 | 1.58E-10 | 1.58E-08 | 1.58E-06 | 1.59E-04 | 1.98E-02 | 8.82E-01 |

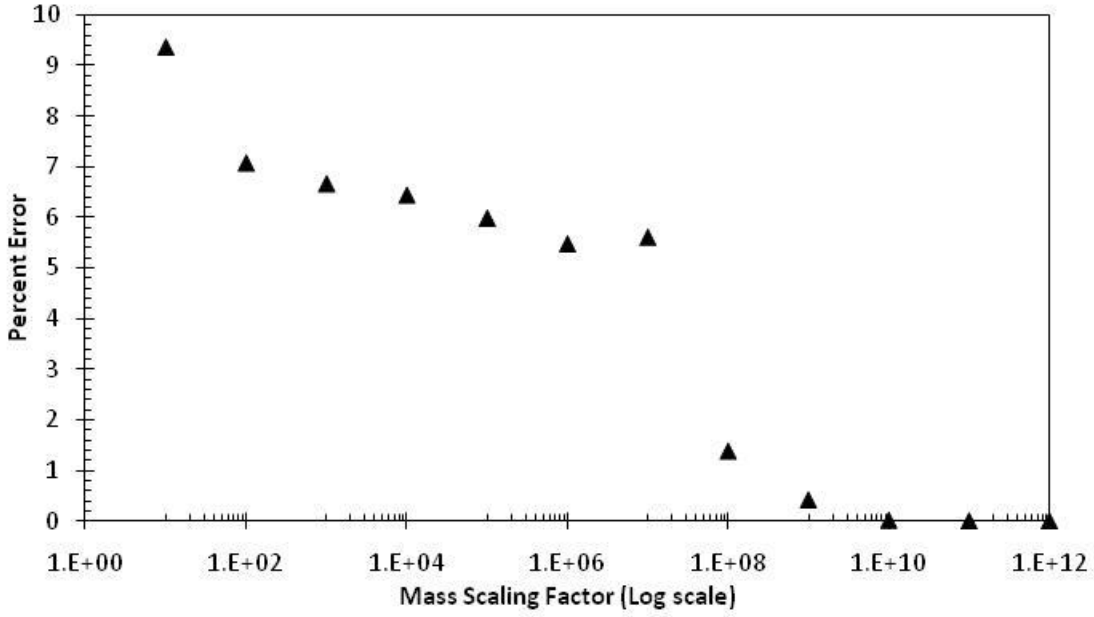



Figure 5.9 Plot showing the percent error of the true stress at 4.8% true strain for simulations with mass scaling factors ranging from 1 E 1 to 1 E 12. The error is measured between the stress in each simulation and the stress of the converged solution to this problem, 199.4 MPa.

Figure 5.9 shows the rate at which the solutions converge to the accurate solution with respect to the mass scaling factor. Like the data in Figure 5.6, Figure 5.9 contains two distinct regions of data. The first region is from a mass scaling factor of 1 (actual density) to a scaling factor of $1 \times 10^6$; the data in this region does notcorresponds to the implicit solution, rather it displays a numerical drift. The second region is found between mass scaling factors of $1 \times 10^7$ and $1 \times 10^{12}$ and is characterized by a plateau corresponding to the implicit solution. For a mass scaling factor greater than or equal to $1 \times 10^{13}$, the solution begins to show irregularity in the form of oscillations; when the scaling is high enough, the oscillations become unbounded and instability results.

### 5.3.2 Mass Scaling with Damage

The previous section showed that for the given set of constants, the explicit solution for a 1mm element with no damage will converge for a range of mass scaling factors; this section will determine the effect of adding in the nonlinearities associated with damage and alongside mass scaling in the element. For the data presented in Figures 5.10 and 5.11, the initial damage is 3%.

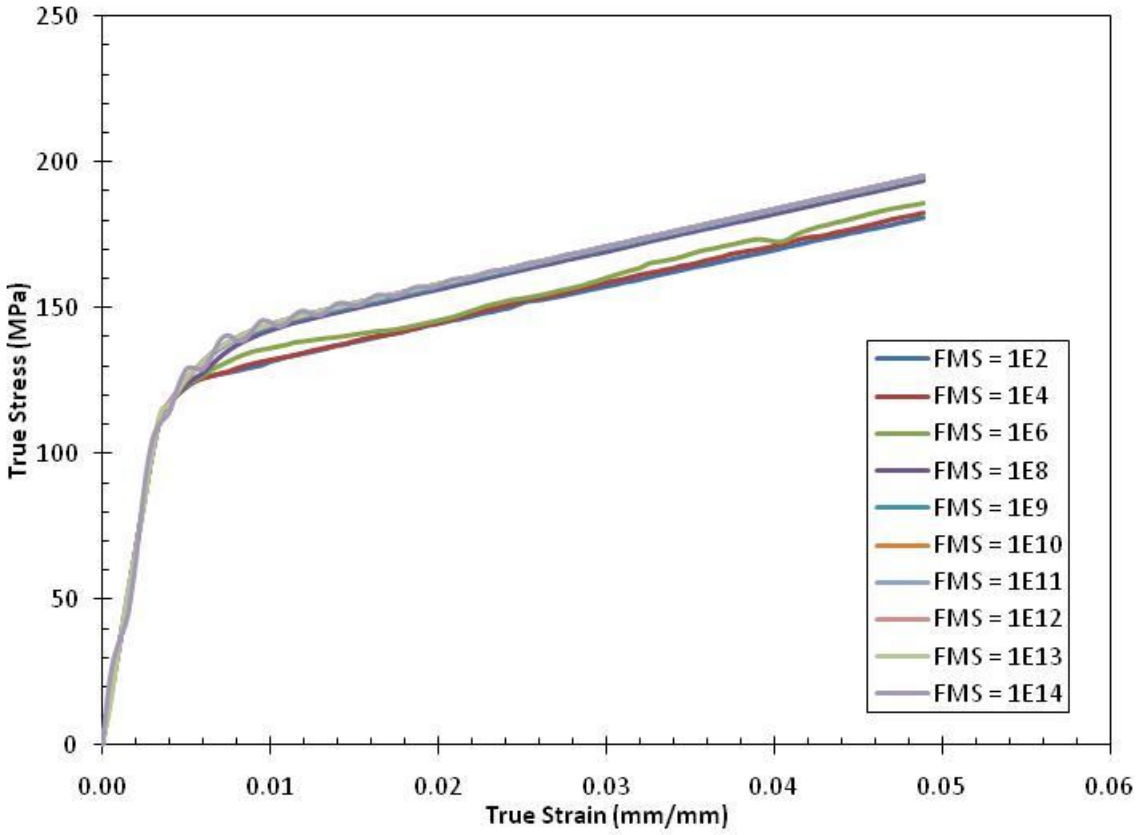



Figure 5.10 Stress versus strain responses for a series of simulations with increasing mass scaling factors and a non-zero initial porosity (3%). All simulations are run at the same strain rate and all elements are the same size.

Figure 5.10 shows the variation of mechanical response of a simulation with increased mass scaling. The trends seen in this figure are similar to those in Figure 5.8. Specifically, for the range of mass scaling from 1 to 1 x $10^6$, the stress versus strain

response tends towards the implicit result. For the range of 1 x $10^7$ to 1 x $10^{13}$, the solutions are approximately the same and equal to the implicit solution. For mass scaling factors greater than 1 x $10^{14}$, the solution begins to oscillate towards instability like in Figure 5.8.

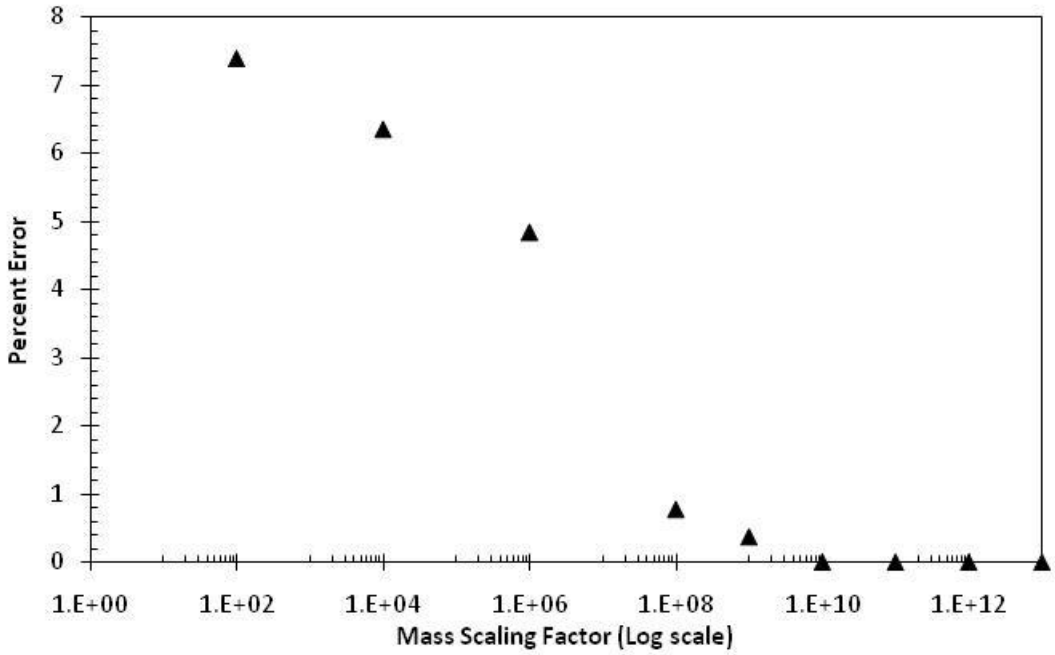



Figure 5.11 Plot showing the percent error of the true stress at 4.8% true strain for simulation with an initial porosity of 3% and mass scaling factors ranging from 1 E 1 to 1 E 13. The error is measured between the stress in each simulation and the stress of the converged solution to this problem, 195.1 MPa.

Figure 5.11 shows the percent error at the end of each simulation versus the mass scaling factor used. Comparing the data in Figure 5.11 and 5.9, similar trends are found in both. For the simulations of scaling factors less than 1 x $10^7$, the error is large and not predictably converging to any value. However, for the simulations containing a scaling factor between $1\ x\ 10^8$ and $1\ x\ 10^{13}$ the error converges smoothly to zero.

After showing that the explicit solution converges to the implicit solution for a given range of mass scaling factors with and without damage, simulations were run to determine what if any effect mass scaling had on the evolution of damage. The following simulations were run with a mass scaling factor of 1 x $10^9$ and 1 x $10^{10}$ because these values fall in the middle of the converged simulations for the data in Figures 5.9 and 5.11. Figure 5.12 shows that the elongation to failure was identical for the two values of mass scaling. Also, it shows that while the general trend in the experimental data is still captured by the simulation, the values for elongation to failure found in the explicit simulations are slightly higher than those found in Figure 3.3 from the implicit simulations. Therefore, it seems as though the full scale explicit simulations with mass scaling factors of $1\ x\ 10^9$ or $1\ x\ 10^{10}$ will predict similar damage evolutions as the implicit simulations but at higher strain levels.

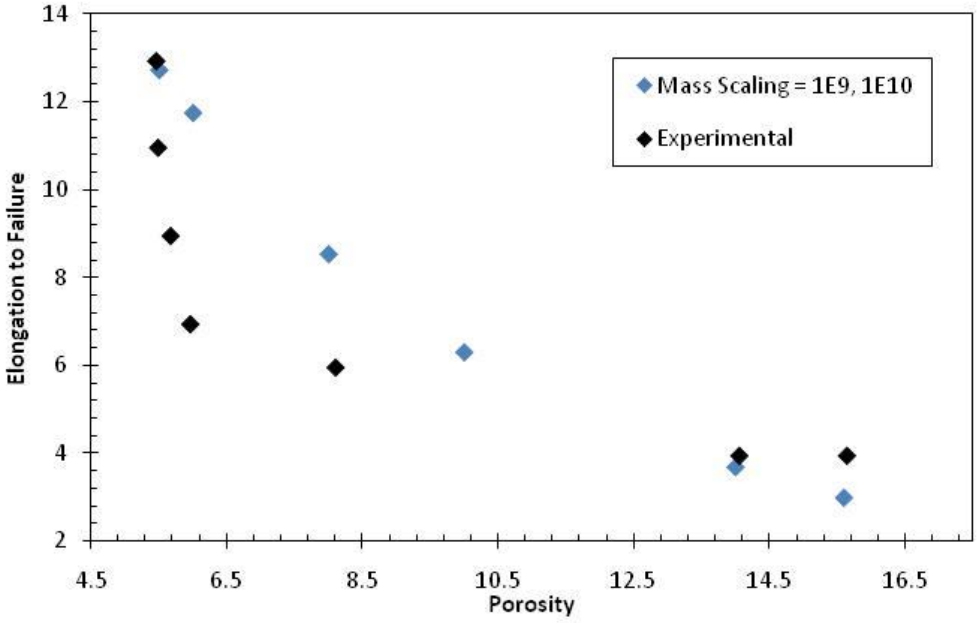



Figure 5.12 Plot comparing the elongation to failure of simulations run with various initial porosity levels and experimental data. The simulations were run on 1 mm elements with mass scaling factors of 1 E 9 and 1 E 10.

### 5.3.3 Notched Bridgeman Simulations

Once it had been shown that there is a range of mass scaling factors that will cause the explicit implementation to return a similar material response to the implicit implementation, simulations were performed of the full notch Bridgeman tensile specimen. The simulation of the notched Bridgeman tensile specimen was performed using mass scaling factors of $1\ x\ 10^6$, $1\ x\ 10^8$, $1\ x\ 10^{10}$. These values were chosen because they represent a transition from the region of scaling factors that do not closely match the implicit result to the region of factors that has converged to the implicit result. Each of the three simulations was then evaluated at the same strain levels as the implicit solution in Section 4.3.1.

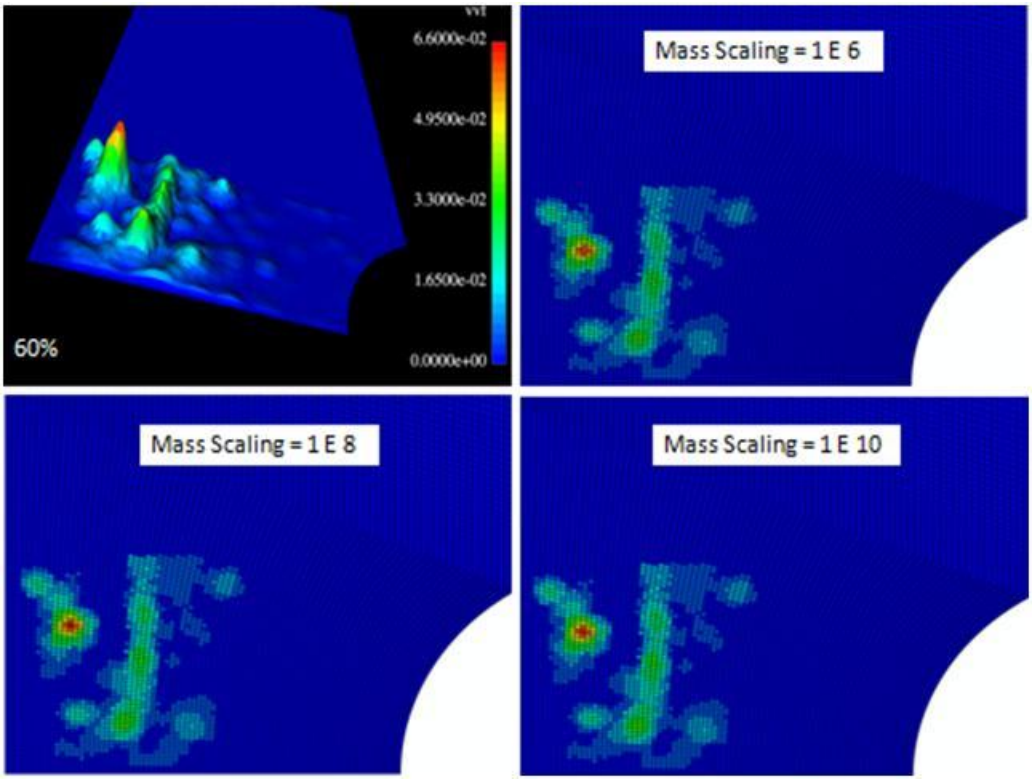
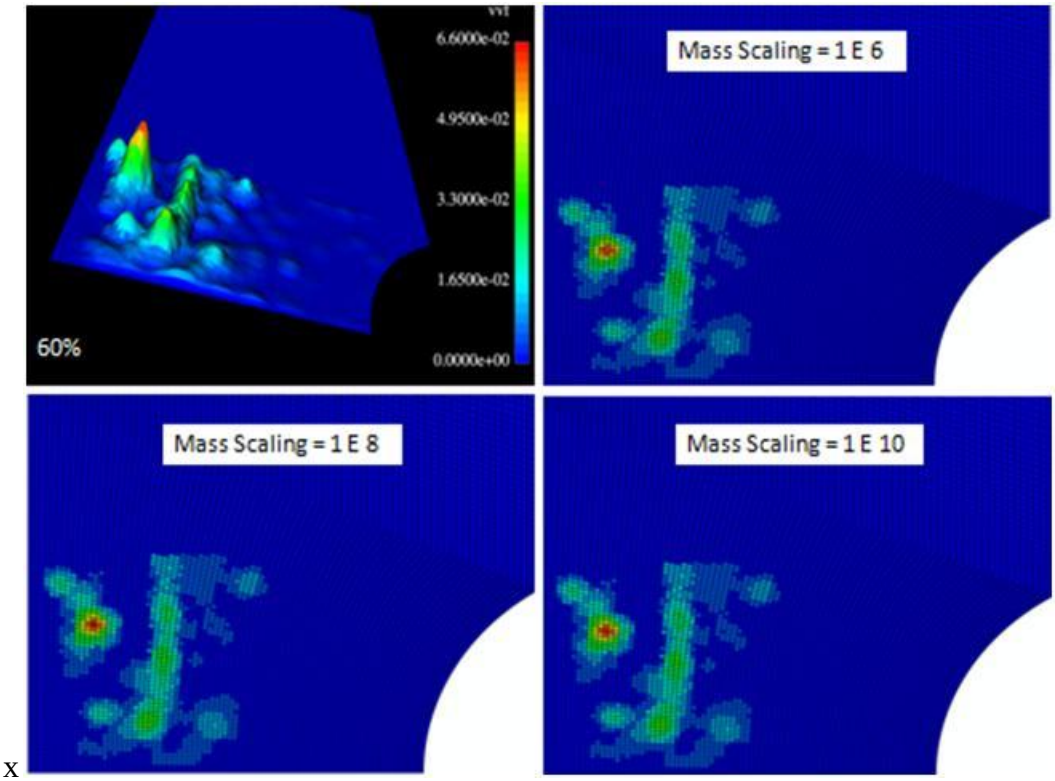



Figure 5.13 Damage levels at 60% of the failure load for the experiment ( top left) and the simulations with mass scaling factors of 1 E 6 (top right), 1 E 8 (bottom left), and 1 E 10 (bottom right). The damage contours were taken at 0.6% true strain.

Figure 5.13 shows the damage distribution at 60%; for this data, that corresponds to a true strain of 0.6%. It is readily observed that, for this strain level, all simulations yield nearly identical results. The main result from this figure is that at 60% of the projected total load, the damage has evolved very little. Figure 5.14 shows the evolution of damage through 87%, which corresponds to 2.1% true strain. It can be seen from this figure that the simulation using the scaling factor of $1\ x\ 10^6$ is not evolving much at all while the damage in the other two simulations very closely resemble that in the implicit solution. As with the implicit solution and the experiment, damage is evolving most significantly at the large pore and the flow line.

Figure 5.15 shows the evolution at 93%, a true strain of 2.8%. This figure shows much the same thing as Figure 5.14, the simulation using low mass scaling is not evolving damage and the other two simulations return results very similar to the implicit solution.




x

Figure 5.14 Damage levels at 87% of the failure load for the experiment ( top left) and the simulations with mass scaling factors of 1 E 6 (top right), 1 E 8 (bottom left), and 1 E 10 (bottom right). The damage contours were taken at 2.1% true strain.

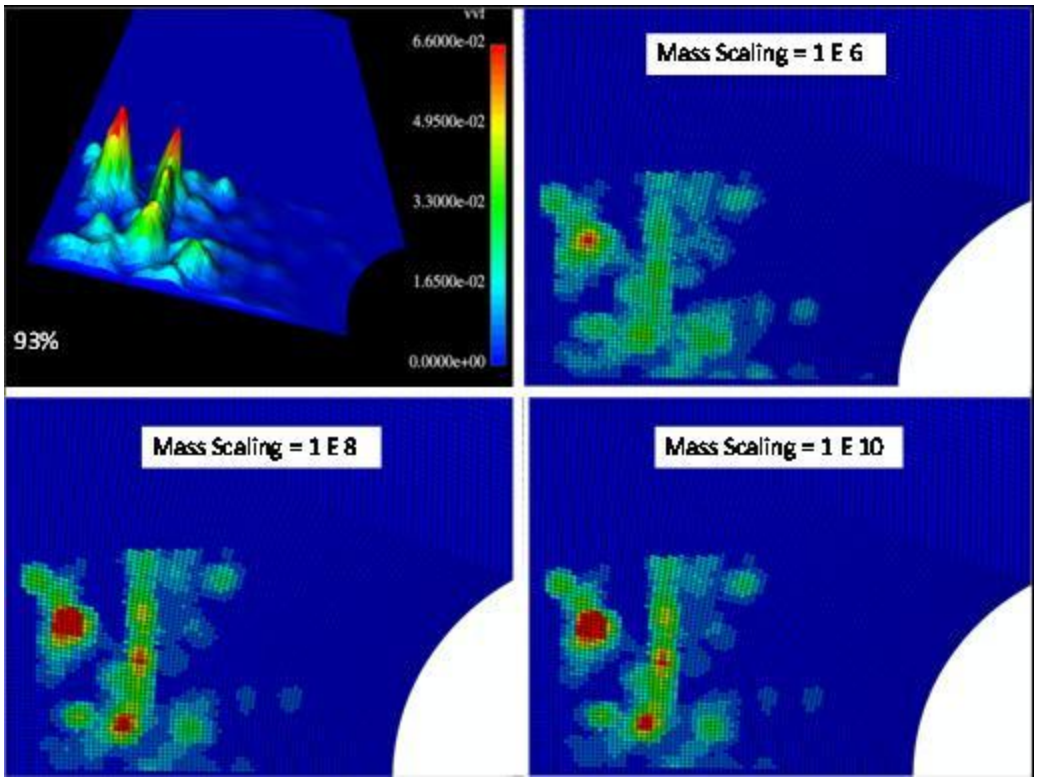



Figure 5.15 Damage levels at 93% of the failure load for the experiment ( top left) and the simulations with mass scaling factors of 1 E 6 (top right), 1 E 8 (bottom left), and 1 E 10 (bottom right). The damage contours were taken at 2.8% true strain.

Up to the point of Figure 5.15, the simulations with scaling factors of $1\ x\ 10^{8}$ and $1\ x\ 10^{10}$ predict the major trends in the evolution of damage. They both predict that the vast majority of the void growth will occur at the two locations which contain the highest initial void volumes, the large pore and the flow line.

The final step was to evaluate what each model predicted for complete failure. This poses a problem, however, for all of simulations; Figure 5.16 shows the damage distribution in the final step for the simulation with a mass scaling factor of $1\ x\ 10^{10}$. This simulation gets a little more information than the implicit simulation, but this simulation predicts first element failure at 4.6% true strain where the implicit simulation predicted first element failure at approximately 3.3% true strain. At this point in the

simulation, ABAQUS Explicit exited with an error. The same error occurred in the other two explicit simulations, but for those simulations, it occurred prior to the first element failure. When attempting to run the simulations with random and uniform damage distributions, the same error occurred again. Details concerning the state of the failed simulations can be found in Appendix A, but as of the writing of this work, the source of the error had not been identified.

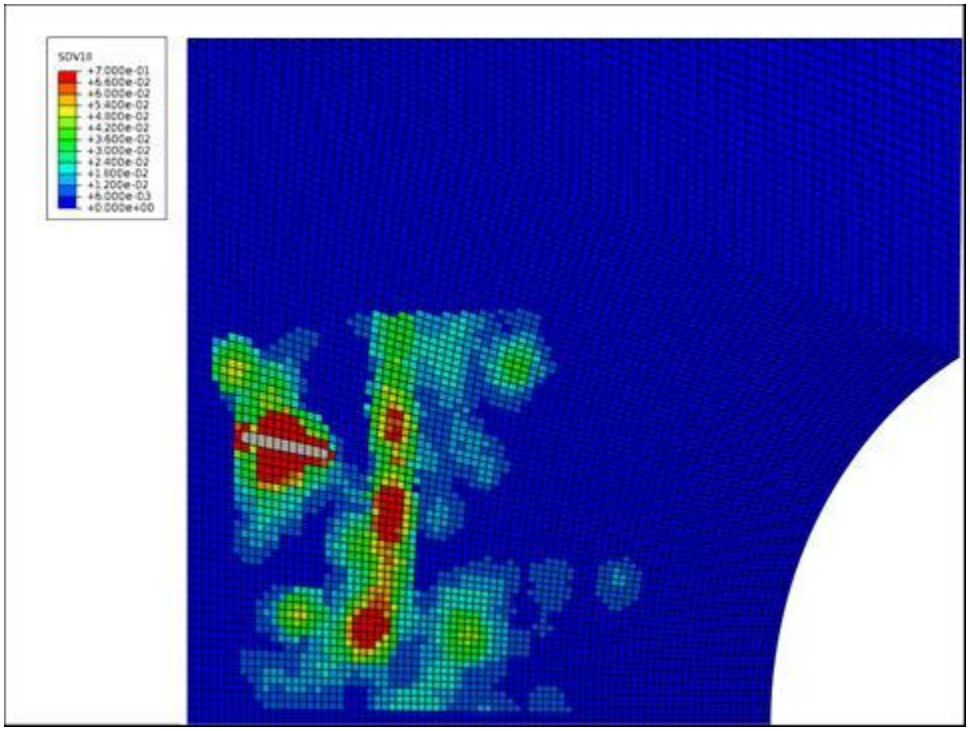



Figure 5.16 Damage contour at the final increment of the simulation of the notched Bridgeman tensile specimen applying a mass scaling factor of 1 E 10.

## 5.4 Without Rate Sensitivity on the Initiation of Yield

This section takes an alternative approach aimed at getting the implicit and explicit implementations of the DMG model to yield the same results. As discussed in the beginning of this chapter, it was observed that in the undamaged one element simulations, all of the discrepancy between the two implementations is introduced with

C1. C1 is the constant that, along with C2, determines the rate sensitivity of the initial yield point. Therefore, setting C1 equal to zero effectively removes the rate sensitivity of the initial yield point from the plasticity model.

### 5.4.1 One Element Simulations

The first step towards using this approach in a full specimen simulation is to compare the predicted material response for 1 element simulations using ABAQUS Standard and ABAQUS Explicit. Figure 5.17 below shows the stress versus strain response of the implicit and explicit simulations when C1 = 0. Upon inspection of the figure, it is easily observed that the two models return equivalent results. So, for a 1 mm element with no mass scaling, at a quasi- static loading rate, making C1 = 0 in the DMG model removes the discrepancy between the implicit and explicit solutions.

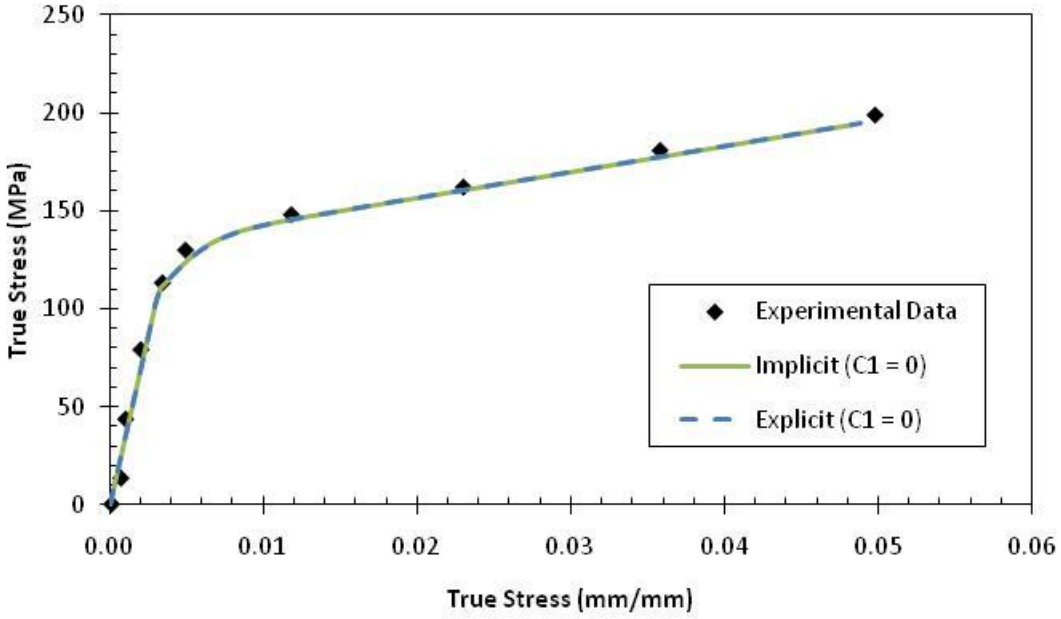



Figure 5.17 This plot shows the stress versus strain response for an implicit and explicit simulation with no damage and C1 = 0. The results are plotted over experimental data for a quasi static tensile test.

In order to run the full sized simulations in a more computationally efficient manner, mass scaling is often used. Therefore, simulations were run to see what effect, if any, that the mass scaling would have on the mechanical response of the simulation. Figure 5.18 shows the results of 1 element simulations run with an element size of 1 mm, strain rate of 3.5 x $10^{-4}$ $s^{-1}$, and mass scaling factors of 100, 10000, 1000000, and 100000000. It is clear from Figure 5.18 that mass scaling on the order of that shown in the figure has no effect on the shape of the stress versus strain response. The biggest difference between these simulations is the total run time; the simulations with mass scaling factors equal to 1E2, 1E4, 1E6, and 1E8 required 1.75 hours, 11 minutes, 1 minute, and 7 seconds, respectively.

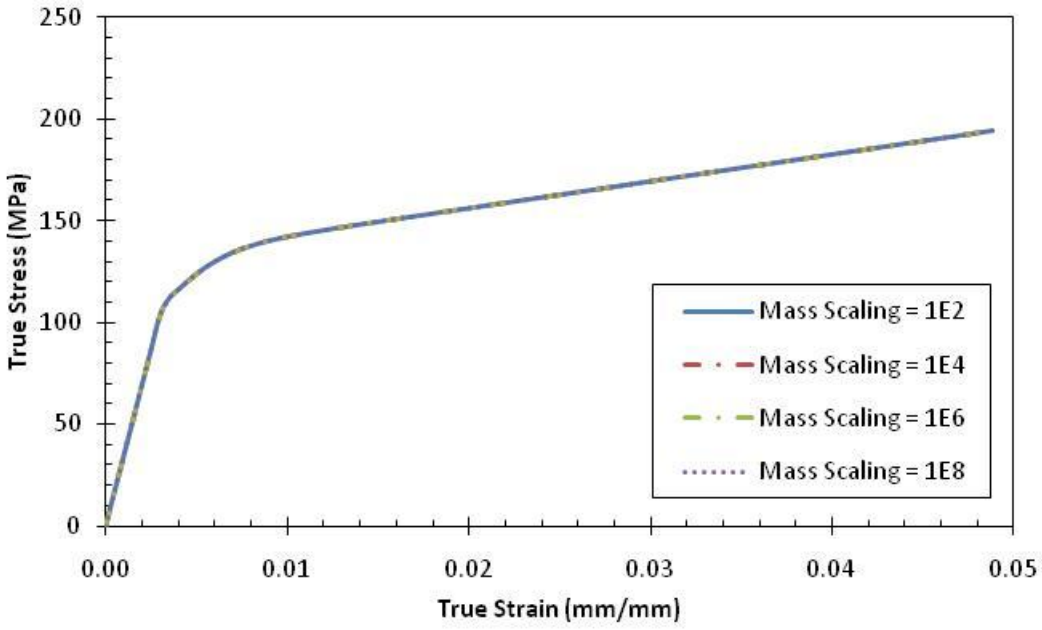



Figure 5.18 Plot showing the stress versus strain response for simulations run in ABAQUS Explicit on 1 mm elements and having a range of mass scaling factors from 100 to $1 \, x \, 10^8$.

So, now it has been shown that removing C1 from the model makes the implicit and explicit solutions equal in simulations containing no damage. The next step is then to verify that the same is true when damage is present in the element. Therefore, a simulation was run in the implicit and explicit code with 3% damage and the results compared. Those results can be found in Figure 5.19.

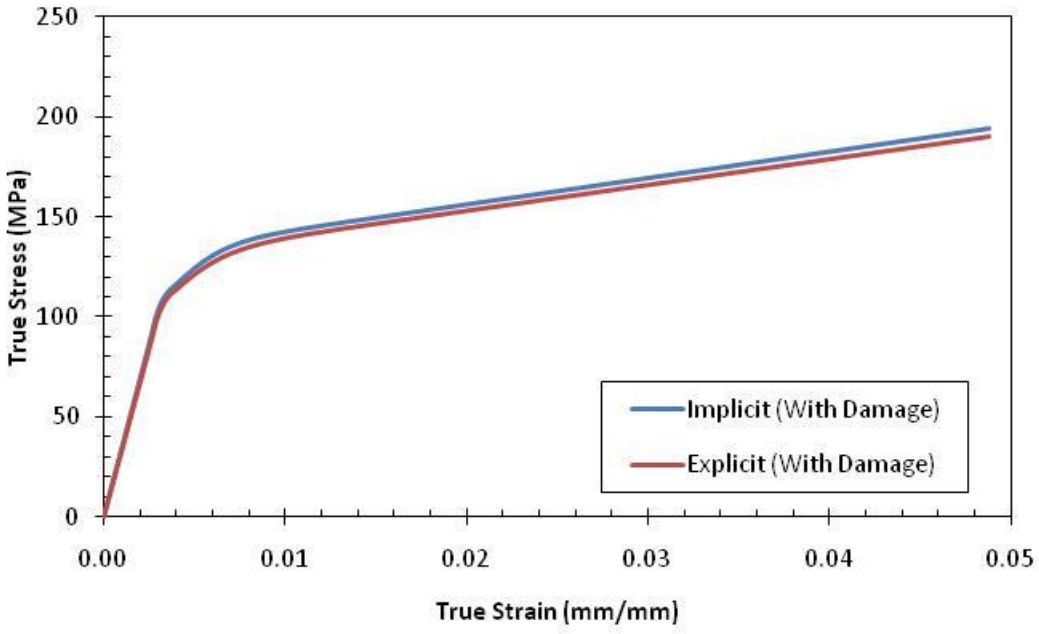



Figure 5.19 Stress versus strain results of simulations run with 3% initial porosity and C1 = 0.

The results found in Figure 5.19 show that the implicit and explicit schemes return similar results for the simulations run with 3% initial porosity. The results are not, however, identical. For the purposes of this study, the results are sufficiently accurate. Since this section has shown that removing C1 from the model will cause the implicit and explicit results to be equal or similar for a quasi static strain rate, the next step is to apply the material constants used here to the full notch Bridgeman tensile specimen.

### 5.4.2 Notched Bridgeman Simulations

Simulations of the notch Bridgeman tensile specimen were run with mass scaling factors of $1\ x\ 10^6$, $1\ x\ 10^8$, and $1\ x\ 10^{10}$ in order to verify that the mass scaling would cause minimal change in the damage evolution and mechanical response of the simulation. When analyzing the results of the three simulations with the real porosity distribution and different mass scaling factors, it was readily observed that the simulations were nearly identical; therefore, only results from the simulation with a mass scaling of $1\ x\ 10^6$ will be presented here. Figures 5.20 – 5.22 show the predicted damage evolution from these simulations side by side with the experimental damage evolution.

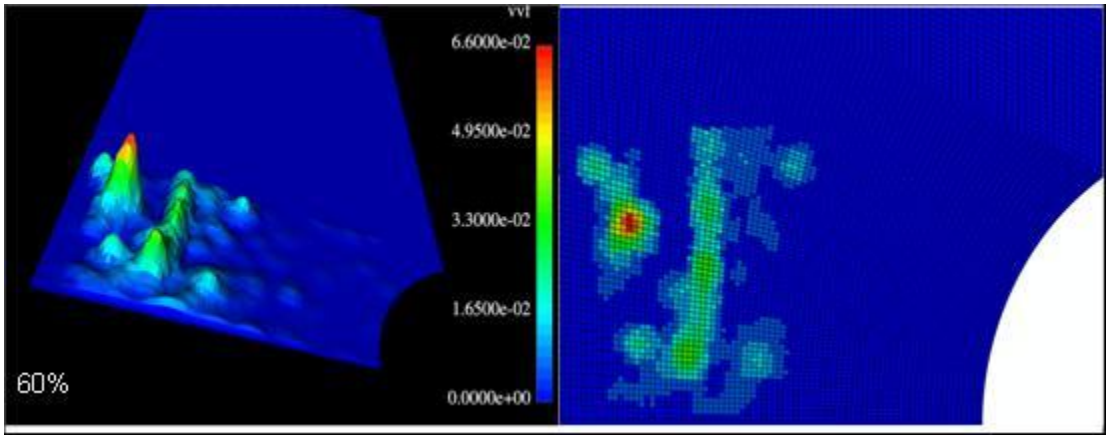



Figure 5.20 Damage contours at 0.6% true strain for the experiment (left) and simulation (right).

Figure 5.20 shows the damage contours at approximately 60% of the failure load, which occurs at 0.6% true strain. It is clear from this figure that the simulation accurately predicts the damage evolution to this point in the simulation. Comparing the results from this point in the explicit simulation to the implicit simulation discussed in Section 4.3.1, the evolution of damage is the same in both cases.

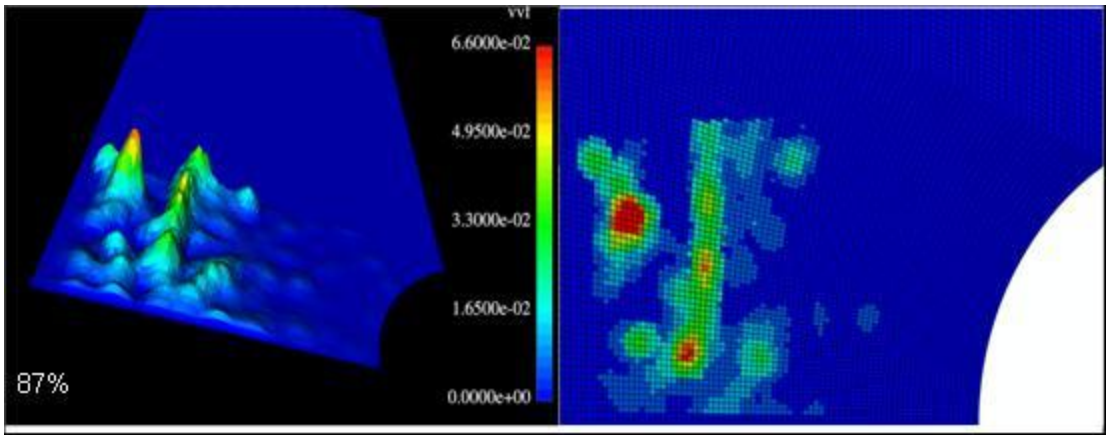



Figure 5.21 Damage contours at 2.1% true strain for the experiment (left) and simulation (right).

Figure 5.21 contains the damage contours at about 87% of failure, which corresponds to 2.1% true strain. Like the contours shown in Figure 5.20, the predicted damage evolution in the explicit simulation closely match both the trends seen in the data and the predicted response from the implicit simulation. Figure 5.22 shows the damage at 93% of failure or 2.8% true strain. The predicted damage evolution from the explicit simulation captures the trends in the experimental data. Comparing this result to the implicit solution shows only slight quantitative differences while maintaining the same qualitative properties.

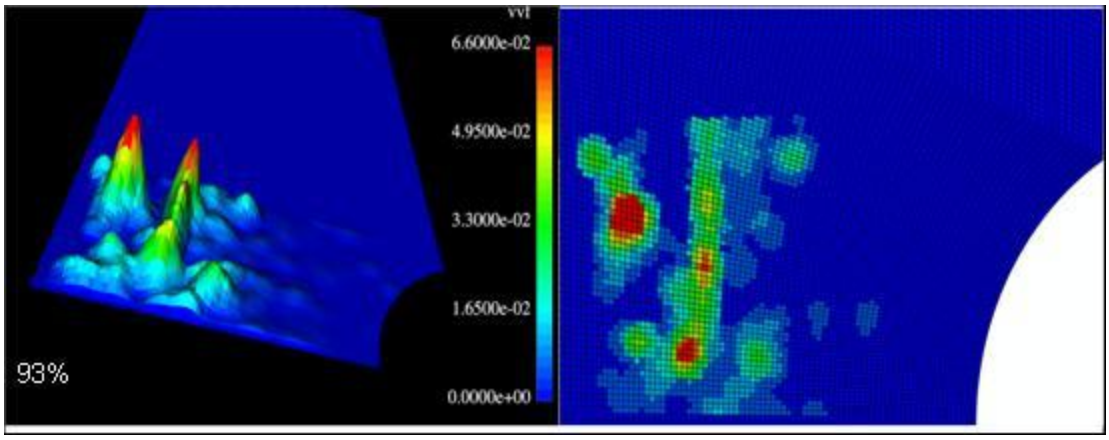



Figure 5.22 Damage contours at 2.8% true strain for the experiment (left) and simulation (right).

The final step in evaluating the explicit simulation of the notch Bridgeman tensile specimen without the influence of C1 is to analyze the damage state at the end of the simulation.

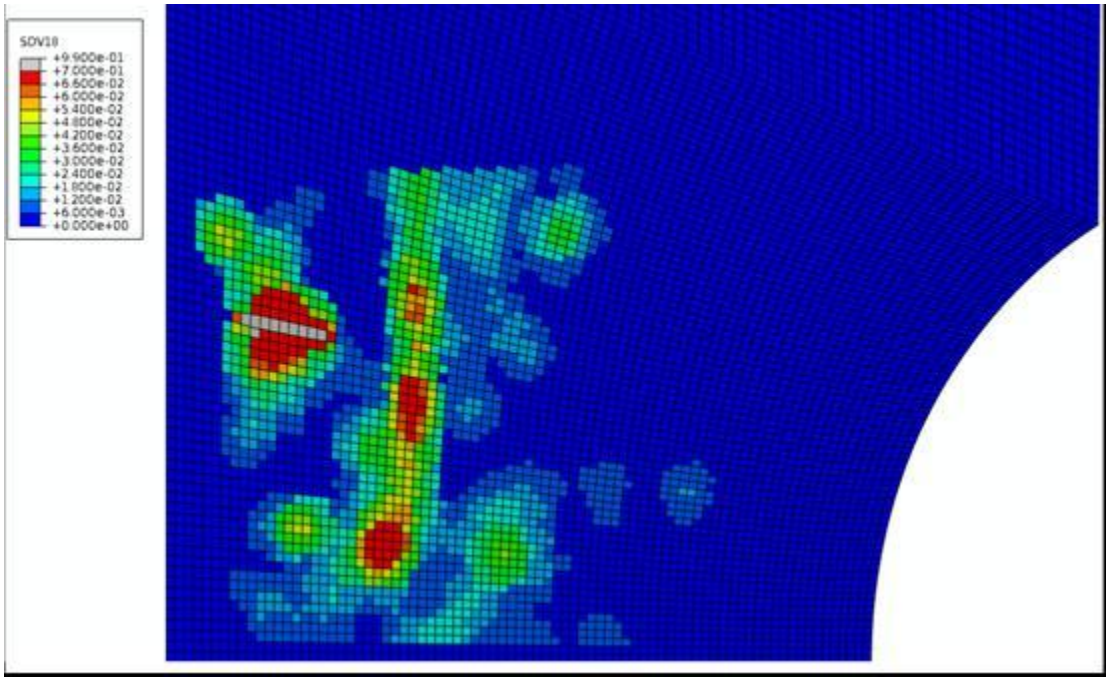



Figure 5.23 Damage contour at the final increment of the simulation of the notched Bridgeman tensile specimen with C1 = 0 and applying a mass scaling factor of 1 E 6.

Figure 5.23 shows a result much like Figure 5.16 in the mass scaling study; first element failure is located in the pore and it begins to grow perpendicular to the axis of loading. This figure also shows that the only significant damage evolution occurs either at the pore or at the flow line, a trend noticed in all of the simulations and the experiment. The primary difference between this simulation and the implicit simulation of the full specimen is that first element failure in this simulation occurred at a true strain of 4.01% and the first element failure for the implicit simulation occurred at a true strain of 3.2%. Therefore, other than predicting a slightly larger elongation to first element failure, removing C1 from the simulation does not inhibit the ability of the DMG model to predict damage evolution in a specimen at quasi-static strain rates.

The purpose for running the simulation in ABAQUS Explicit, however, was to evaluate the model's ability to predict damage evolution beyond the initial failed elements. To that end, this simulation was unable to achieve any results. At the point in the simulation from which Figure 5.23 was taken, the simulation exited and returned an error. The error that occurred is an illegal floating point error somewhere in the calculations. The same error occurred in the simulations with random and uniform damage distributions as well. Appendix A details the state of each of the simulations at the point at which this error occurs.

## 5.5 Discussion

The results presented in this chapter lead to several major observations. The first, and most significant, is that in order for the implicit and explicit implementation schemes to return the same solution for a quasi- static tensile simulation, the time step size must be exceedingly small for reasonably sized elements. Since running quasi- static simulations

with very small time steps leads to unreasonable computational cost ( 22 hours for 5% strain in a 1 element simulation), methods of making the two schemes were examined. Two such methods were found, mass scaling and removing C1. Both of these methods have problems associated with them, but allow for an accurate solution at much lower computational cost than lowering the time step size alone. Finally, it was shown in this chapter that it is possible for the simulations run in ABAQUS Explicit to predict damage evolution in a specimen in the same manner as ABAQUS Standard.

In order to have confidence that the simulations being run in any code are in any way accurate, it is necessary to validate the results. In this case, using ABAQUS Explicit a calculated time step size does not allow the results for this simulation to match the theoretical solution with any accuracy. However, changing the problem so that the only change is increasing the maximum stable time step by increasing the element size allows the explicit solution to approach the true solution. Increasing the element size is not an option in a real engineering problem, so mass scaling was used to increase the maximum stable time step size and C1 removed from the simulation to remove the source of the difference between the implicit and explicit solutions.

Mass scaling was shown to cause the mechanical response of a one element simulation to approach the implicit solution within a range of mass scaling factors. This method of making the implicit and explicit solutions equal changes the boundary value problem in that it artificially increases the density of the system. The change in the problem is minimal, however, due to the extremely low strain rate involved in this simulation. When using this method on the full notch Bridgeman tensile specimens, the predicted damage evolution was similar to that in the implicit solutions. The only

observed difference between the results is that the explicit simulation with mass scaling returned the same results as the implicit solution but at slightly higher strains.

The next method of making the implicit and explicit solutions equal involves setting the material parameter C1 equal to zero. This change makes the solution for the implicit and explicit implementation schemes the same, but it removes the strain rate sensitivity at initial yield from the DMG model. Because this simulation is quasi- static, this limitation does not affect the results, but it does severely limit the functionality of the DMG model. The full specimen simulations run using this method demonstrated the same damage evolution qualities as the mass scaling simulations without the drawback of higher strains. The results from these simulations were the same as those run using ABAQUS Standard. The purpose of running the simulations in ABAQUS Explicit was to see how the model predicted failure progression through the specimen, however, and no method found was able to capture this result. The next chapter contains suggestions for possible future work.

# CHAPTER VI

# CONCLUSIONS AND FUTURE WORK

The most significant result of this study is that the implicit implementation scheme of the DMG model is capable of predicting the trend of damage evolution in a part given the initial porosity distribution. Chapter 4 showed that, while the specific details about the damage were not captured perfectly, the damage evolution was highly dependent on the initial porosity distribution. The strain at first element failure also showed a very large dependence on the porosity distribution. Specifically, the first element failure for the real, random, and uniform distributions occurred at 3.3, 8.8, and 12.3% true strain, respectively. Therefore, if the porosity had been obtained in an average sense over the notched area of the specimen and randomly distributed for simulation purposes, the model would predict the first element failure 167% higher strain than for the real simulation; if the porosity was uniformly distributed, the predicted strain to failure would have been 273% greater.

With respect to the results presented in Chapter 5, the most significant observation is that damage evolution similar to that obtained with the implicit implementation can be found using an explicit scheme and a mass scaling factor of $1\ x\ 10^{10}$. However, the results show that this damage evolution is at a higher strain level. The first element failure of the explicit simulation with a mass scaling factor of $1\ x\ 10^{10}$ occurred at 4.6% true strain, 39% higher than was predicted by the implicit simulation. The data also indicated that for a constant strain rate test, the capability to impose rate

dependent yield can be removed and yield an accurate result. In terms of solving the specific boundary value and obtaining results past the point of first element failure, the explicit solution as currently implemented would not consistently predict a failure progression and without mass scaling, no full scale results could be obtained. It should also be noted that the ABAQUS defined stable time step is too large to return accurate results when used in simulating a quasi-static test.

Possible future work related to this study would focus around being able to obtain results on the damage evolution of a part beyond first element failure. First, since the ABAQUS calculated stable time step is too large for quasi-static application, it would be beneficial for the stable time step associated with this model to be calculated either outside of ABAQUS or within the user subroutine that defines the material model. Next, since the inaccuracy associated with the explicit results is only present if C1 is not equal to zero, a rate independent version of the model should be implemented to be used in quasi-static simulations that have necessity to use ABAQUS Explicit. Finally, in order to increase the computational efficiency of the user subroutine describing the DMG model, it should be thoroughly examined in order to remove or replace sections of the code that slow down computation time.

# REFERENCES


Allison PG. Structure-property relations for monotonic and fatigue loading conditions for a powder metal steel. PhD Dissertation. Mississippi State University, Mississippi State, MS. 2009.

Agarwal H, Gokhale AM, Graham S, Horstemeyer MF. Material Science and Engineering 2003; A341: 35-42.

Bammann DJ. Applied Mechanics Review 1990; 43: S312-S319.

Bammann DJ. Int. J. of Engineering Science 1984; 22: 1041-1053.

Bammann DJ, Aifantis EC. Nuclear Engineering and Design 1989; 116: 355-362.

Bammann DJ, Chiesa ML, Horstemeyer MF Weingarten LI. eds. N. Jones and T. Weirzbicki, Structural Crashworthiness and Failure, Elsevier Applied Science, 1993.

Bammann DJ, Chiesa ML, Johnson GC. Theoretical and Applied Mechanics 1996; 359-376.

Caceres CH. Scripta Metall. Metallurgica et Materalia 1995; 32: 1851-1856.

Caceres CH, Selling BI. Material Science and Engineering A 1996; 220: 109–116.

Cocks ACF, and Ashby MF. Progress in Material Science 1982; 27: 189-244.

Davison L, Stevens AK, Kipp ME. J. Mechanics and Physics of Solids 1977; 25: 11-28.

Eady JA, Smith DM. Materials Forum 1986; 9: 217.

Guo YB, Wen Q, Horstemeyer MF. Int. J. Mechanical Sciences 2005; 47: 1423-1441.

Hammi Y, Horstemeyer MF. Int. J. Plasticity 2007; 23: 1641-1678.

Herrera A, Kondic V. Proc. Int. Conf. Solidification and Cast Metals, The Metals Society, Sheffield, UK, 1977, p 460.

Horstemeyer MF. From Atoms to Autos: Part I" Monotonic Loading Conditions, Sandia National Lab 2001: 6.6-11.

Horstemeyer MF, et al. From Atoms to Autos: Designing a Mg Alloy Corvette Cradle by Employing Hierarchical Multiscale Microstructure-Property Models for Monotonic and Cyclic Loads. MSU.CAVS.CMD.2007-R0001. 2007.

Horstemeyer MF, Gohkale AM. Int. J. Solids and Structures 1999; 36: 6-11.

Horstemeyer MF, Lathrop J, Gohkale AM Dighe M. Theoretical and Applied Fracture Mechanics 2000; 33: 31-47.

Horstemeyer MF, Matalanis MM, Sieber AM, Botos ML. Int. J. Plasticity 2000; 16: 979-1015.

Jordan JB. Damage and Stress State Influence on Baushinger Effect in Aluminum Alloys. Masters Thesis. Mississippi State University, Mississippi State, MS. 2006.

Jordan JB, Horstemeyer MF, Solanki K, Xue Y. Mechanics of Materials 2007; 39: 920-931.

Lancaster P , Salkauskas K. Mathematics of Computation 1981; 87: 141-158.

Lee Choong Do. J. Material Science 2007; 42:10032-10039.

Lee Choong Do. Materials Science and Engineering A 2007; 575-580.

Lee CD, Shin KS. Acta Materialia 2007; 55: 4293-4303.

Ling X, Cherukuri HP. Computational Mechanics 2002; 29: 430-440.

McClintock FA. J. of Fracture 1968; 4: 103-130.

McLellan DL. J. Tet. Eval. 1980; 8: 170.

McVeigh C, Liu WK. Computer Methods Applied Mechanics Engineering 2008; 197: 3268-3290.

Nagtegaal JC. Computer Methods in Applied Engineering 1982; 33: 469-484.

Reddy JN. Introduction to Nonlinear Finite Element Analysis. New York, Oxford; 2004.

Simo JC, Taylor RL. Computer Methods Applied Mechanics Engineering 1985; 48: 101-118.

Simo JC, Taylor RL. Int. J. Numerical Methods in Engineering 1986; 22: 649-670.

Surappa MK, Blank EW, Jaquet JC. Scripta Metallurgica 1986; 20: 1281.

Tucker MT. Structure-property stress state dependent relationships under varying strain rates. PhD Dissertation. Mississippi State University, Mississippi State, MS. 2009.

Waters AM. Three-dimensional analysis of voids in am60b magnesium tensile bars using computed tomography imagery. PhD Thesis. John Hopkins University, Baltimore, MD, 2001.

Waters AM, et al. Materials Evaluation 2000; 58: 1221-1227.

Weiler JP, et al. Materials Science and Engineering A 2005; 395: 315-322.

# APPENDIX A

# CONDITIONS OF SIMULATION EXIT IN THE PRESENCE OF A FLOATING POINT ERROR

## A.1 Mass Scaling Simulations

### A.1.1 Sample Input File

```
*HEADING
**------------------------------------------------------------------------**
**------------------------------------------------------------------------**
**Parameters
*PARAMETER
**---------------------------------------**
  Period        = 180.
  Velocity      =   0.005
  Mass_Scaling  =   1E+10
**---------------------------------------**
**---------------------------------------**
** Material Parameters for VUMAT BCJ Model (vumat_dmg-52p.f)
**----------  Material Parameters
**----------  Constants from USCAR
**----------  Modified to fit experiments by dbh
  G         = 12810.
  a         = 1.0
  K         = 38440.
  b         = 0.
  Tmelt     = 5556.
**----------  BCJ Parameters
  C01       = 2.66
  C02       = 0.
  C03       = 92.82
  C04       = 47.93
  C05       = 1.000e-04
  C06       = 6.991e-07
  C07       = 1.929e+07
  C08       = 6.868e+03
  C09       = 1.577e+03
  C10       = 0.6931
  C11       = 6.529e-05
  C12       = 1.064e+06
  C13       = 14.8
  C14       = 6.911e-07
  C15       = 4.077e+04
  C16       = 1.024e+02
  C17       = 0.
  C18       = 0.
  C19       = 0.
  C20       = 0.
  Ca        = 1.883
  Cb        = 8.272e-03
**----------  Temperature Parameters
  InitTemp  = 297.
  HeatGen   = 0.
**----------  Microstructure Parameters
  VoidGrthn = 0.246
  InitVRad  = 2.0000e-4
  TorsCsta  = 1.
  NuclCsta  = 1.
  NuclCstc  = 1.
  NuclCoef  = 0.
  KIc       = 17.3
  PartSize  = 4.000e-04
  PartVF    = 0.07
  cd1       = 0.7
  cd2       = 1.
**----------  Grain Size Parameters
  Dcs0      = 20.
  Dcs       = 20.
```

```
  zz        = 0.0509
  InitVVF   = 0.01
**----------  Other BCJ Parameters
  C21       = 0.
  C22       = 0.
  C23       = 0.
  C24       = 0.
  C25       = 0.
  C26       = 0.
**----------  Temperature Parameters for Nucleation and Coalescence
  Tnucl     = 0.
  Tcoal     = 0.
**----------  Flag, Cacon and Elastic Modulus porosity exponent
  Flag      = 0.
  Cacon     = 10.
  Zeta      = 0.
**
** END PARAMETERS
**-----------------------------------------------------------------**
**-----------------------------------------------------------------**
**-----------------------------------------------------------------**
*NODE, INPUT=mesh5.nd
*INCLUDE,INPUT=mesh5.ns
*ELEMENT, INPUT=mesh5.el, TYPE=CAX4R, ELSET=ALL
**-----------------------------------------------------------------**
**-----------------------------------------------------------------**
*SOLID SECTION,ELSET=ALL,MATERIAL=AM60B
**HOURGLASS STIFFNESS
**49.248
**-----------------------------------------------------------------**
*MATERIAL,NAME=AM60B
*DENSITY
 1.8E-09
*USER MATERIAL,CONSTANTS=55
**-------------------------------------------------------------------------------------------
------------**
**-------------------------------------------------------------------------------------------
------------**
   <G>,     <a>,      <K>,      <b>,      <Tmelt>,       <C01>,      <C02>,       <C03>,
   <C04>,  <C05>,   <C06>,     <C07>,       <C08>,       <C09>,      <C10>,       <C11>,
   <C12>,  <C13>,   <C14>,     <C15>,       <C16>,       <C17>,      <C18>,       <C19>,
   <C20>,   <Ca>,    <Cb>,  <InitTemp>,  <HeatGen>, <VoidGrthn>, <InitVRad>,  <TorsCsta>,
 <NuclCsta>,  <NuclCstc>,  <NuclCoef>,    <KIc>,  <PartSize>,    <PartVF>,  <cd1>, <cd2>,
  <Dcs0>,  <Dcs>,    <zz>,   <InitVVF>,     <C21>,       <C22>,      <C23>,       <C24>,
      <C25>,       <C26>,     <Tnucl>,     <Tcoal>,      <Flag>,     <Cacon>,     <Zeta>
**-------------------------------------------------------------------------------------------
------------**
*DEPVAR
25
**-------------------------------------------------------------------------------------------
------------**
**-------------------------------------------------------------------------------------------
------------**
**-----------------------------------------------------------**
**--------   AMPLITUDE FOR CONSTANT STRAIN RATE   --------**
**-----------------------------------------------------------**
**AMPLITUDE,NAME=RATE,VALUE=ABSOLUTE,INPUT=disp.inp
**---------------------------------**
**--------    COMPACTION    ------**
**---------------------------------**
*BOUNDARY
LSIDE, 1,1,0.0
BSIDE, 2,2,0.0
**-----------------------------------------------------------**
**---------INITIAL CONDITIONS ADDITION--------------------**
**-----------------------------------------------------------**
*INITIAL CONDITIONS,TYPE=SOLUTION,INPUT=m5v1_final.dat
*RESTART,WRITE,NUMBER INTERVAL=100,TIME MARKS=NO,OVERLAY
**---------------------------------------**
```

```
*STEP,NLGEOM=YES
*DYNAMIC,EXPLICIT
,<Period>
**----------------------------------------**
*BULK VISCOSITY
0.06, 1.2
**----------------------------------------**
** Mass Scaling:
*Fixed Mass Scaling, factor=<Mass_Scaling>
**----------------------------------------**
**----------  TENSION  B.C.  -------------**
*BOUNDARY,TYPE=VELOCITY
TSIDE, 2, 2, <Velocity>
**----------------------**
**---  User Boundary Conditions for constant strain rate (vdisp.f)
**BOUNDARY,USER,TYPE=VELOCITY
**FACE4,1,1,
**----------------------**
**MONITOR,DOF=1,NODE=2
**----------------------------------------**
**----------------------------------------**
*OUTPUT, FIELD, NUMBER INTERVAL=100
*ELEMENT OUTPUT,ELSET=ALL
S, SDV, LE
*NODE OUTPUT
COORD, U, V, RF
**----------------------------------------**
*FILE OUTPUT,TIMEMARKS=YES,NUM=100
*NODE FILE
U,RF
*EL FILE
SDV,LE,S
*END STEP
**--------------------------------------------------------------------------**
**--------------------------------------------------------------------------**
**--------------------------------------------------------------------------**
```

## A.2 Sample Input File for Simulations with C1 = 0

```
*HEADING
**--------------------------------------------------------------------------**
**--------------------------------------------------------------------------**
**Parameters
*PARAMETER
**---------------------------------------**
  Period        = 180.
  Velocity      =   0.005
  Mass_Scaling  =   1E+10
**---------------------------------------**
**---------------------------------------**
** Material Parameters for VUMAT BCJ Model (vumat_dmg-52p.f)
**----------  Material Parameters
**----------  Constants from USCAR
**----------  Modified to fit experiments by dbh
  G         = 12810.
  a         = 1.0
  K         = 38440.
  b         = 0.
  Tmelt     = 5556.
**----------  BCJ Parameters
  C01       = 0.
  C02       = 0.
  C03       = 92.82
  C04       = 47.93
  C05       = 1.000e-04
```

```
  C06       = 6.991e-07
  C07       = 1.929e+07
  C08       = 6.868e+03
  C09       = 1.577e+03
  C10       = 0.6931
  C11       = 6.529e-05
  C12       = 1.064e+06
  C13       = 14.8
  C14       = 6.911e-07
  C15       = 4.077e+04
  C16       = 1.024e+02
  C17       = 0.
  C18       = 0.
  C19       = 0.
  C20       = 0.
  Ca        = 1.883
  Cb        = 8.272e-03
**----------  Temperature Parameters
  InitTemp  = 297.
  HeatGen   = 0.
**----------  Microstructure Parameters
  VoidGrthn = 0.246
  InitVRad  = 2.0000e-4
  TorsCsta  = 1.
  NuclCsta  = 1.
  NuclCstc  = 1.
  NuclCoef  = 0.
  KIc       = 17.3
  PartSize  = 4.000e-04
  PartVF    = 0.07
  cd1       = 0.7
  cd2       = 1.
**----------  Grain Size Parameters
  Dcs0      = 20.
  Dcs       = 20.
  zz        = 0.0509
  InitVVF   = 0.01
**----------  Other BCJ Parameters
  C21       = 0.
  C22       = 0.
  C23       = 0.
  C24       = 0.
  C25       = 0.
  C26       = 0.
**----------  Temperature Parameters for Nucleation and Coalescence
  Tnucl     = 0.
  Tcoal     = 0.
**----------  Flag, Cacon and Elastic Modulus porosity exponent
  Flag      = 0.
  Cacon     = 10.
  Zeta      = 0.
**
** END PARAMETERS
**-----------------------------------------------------------------**
**-----------------------------------------------------------------**
**-----------------------------------------------------------------**
*NODE, INPUT=mesh5.nd
*INCLUDE,INPUT=mesh5.ns
*ELEMENT, INPUT=mesh5.el, TYPE=CAX4R, ELSET=ALL
**-----------------------------------------------------------------**
**-----------------------------------------------------------------**
*SOLID SECTION,ELSET=ALL,MATERIAL=AM60B
**HOURGLASS STIFFNESS
**49.248
**-----------------------------------------------------------------**
*MATERIAL,NAME=AM60B
*DENSITY
 1.8E-09
*USER MATERIAL,CONSTANTS=55
```

```
**-------------------------------------------------------------------------------------------
------------**
**-------------------------------------------------------------------------------------------
------------**
   <G>,     <a>,      <K>,      <b>,      <Tmelt>,       <C01>,      <C02>,       <C03>,
   <C04>,  <C05>,   <C06>,     <C07>,       <C08>,       <C09>,      <C10>,       <C11>,
   <C12>,  <C13>,   <C14>,     <C15>,       <C16>,       <C17>,      <C18>,       <C19>,
   <C20>,   <Ca>,    <Cb>,  <InitTemp>,  <HeatGen>, <VoidGrthn>, <InitVRad>,  <TorsCsta>,
 <NuclCsta>,  <NuclCstc>,  <NuclCoef>,    <KIc>,  <PartSize>,    <PartVF>,  <cd1>, <cd2>,
  <Dcs0>,  <Dcs>,    <zz>,   <InitVVF>,     <C21>,       <C22>,      <C23>,       <C24>,
      <C25>,       <C26>,     <Tnucl>,     <Tcoal>,      <Flag>,     <Cacon>,     <Zeta>
**-------------------------------------------------------------------------------------------
------------**
*DEPVAR
25
**-------------------------------------------------------------------------------------------
------------**
**-------------------------------------------------------------------------------------------
------------**
**-----------------------------------------------------------**
**--------   AMPLITUDE FOR CONSTANT STRAIN RATE   --------**
**-----------------------------------------------------------**
**AMPLITUDE,NAME=RATE,VALUE=ABSOLUTE,INPUT=disp.inp
**---------------------------------**
**--------    COMPACTION    ------**
**---------------------------------**
*BOUNDARY
LSIDE, 1,1,0.0
BSIDE, 2,2,0.0
**-----------------------------------------------------------**
**---------INITIAL CONDITIONS ADDITION--------------------**
**-----------------------------------------------------------**
*INITIAL CONDITIONS,TYPE=SOLUTION,INPUT=m5v1_final.dat
*RESTART,WRITE,NUMBER INTERVAL=100,TIME MARKS=NO,OVERLAY
**---------------------------------------**
*STEP,NLGEOM=YES
*DYNAMIC,EXPLICIT
,<Period>
**---------------------------------------**
*BULK VISCOSITY
0.06, 1.2
**---------------------------------------**
** Mass Scaling:
*Fixed Mass Scaling, factor=<Mass_Scaling>
**---------------------------------------**
**----------  TENSION  B.C.  -------------**
*BOUNDARY,TYPE=VELOCITY
TSIDE, 2, 2, <Velocity>
**---------------------**
**---  User Boundary Conditions for constant strain rate (vdisp.f)
**BOUNDARY,USER,TYPE=VELOCITY
**FACE4,1,1,
**---------------------**
**MONITOR,DOF=1,NODE=2
**---------------------------------------**
**---------------------------------------**
*OUTPUT, FIELD, NUMBER INTERVAL=100
*ELEMENT OUTPUT,ELSET=ALL
S, SDV, LE
*NODE OUTPUT
COORD, U, V, RF
**---------------------------------------**
*FILE OUTPUT,TIMEMARKS=YES,NUM=100
*NODE FILE
U,RF
*EL FILE
SDV,LE,S
*END STEP
**-------------------------------------------------------------------**
```

## A.3 Condition at Simulation Failure

At the time of the failure of each of the full-scale explicit simulations, all of the ISVs were analyzed. None of the ISVs were changing drastically in magnitude or going to zero from a non-zero value. As of the writing of this work, no cause has been found for the floating point operation error found in these simulations.

# APPENDIX B

# OPTIMIZATION OF EXPLICIT NUMERICAL IMPLEMENTATION

Since a large portion of this project involves performing quasi static simulations over long periods using an explicit scheme, it is expected that the simulations will be very computationally expensive. Therefore, an effort was made to locate areas in the explicit implementation where the code could be made more computationally efficient. Two places were found where the same calculations were being made several times over each element each increment. The first of these calculations is located in the section of the code presented below. This section of code calculates the hardening and recovery terms for each increment.

VUMAT (original)

```
c stress state dependent material constants
        adj    = half*(one+tanh(cc19*(cc20-theta)))
        if(cc19.eq.zero) adj = one
        vtheta = cc1*exp(-cc2/theta)
        ytheta = cc3*exp(cc4/theta)*adj
        ftheta = cc5*exp(-cc6/theta)
c        if(dj2.eq.0)then
        if(dj2 .lt. 1.0D-08)then
          rd1=cc7*(1-ca*(4./27.))*exp(-cc8/theta)
          h1=(cc9-cc10*theta)*(1+ca*(4./27.))
          rs1    = cc11 * exp(-cc12/theta)
          rd2    = cc13*(1-ca*(4./27.))*exp(-cc14/theta)
          h2     = (cc15-cc16*theta)*(1+ca*(4./27.))
          rs2    = cc17 * exp(-cc18/theta)
          rd3    = cc21*(1-ca*(4./27.))*exp(-cc22/theta)
          h3     = (cc23-cc24*theta)*(1+ca*(4./27.))
          rs3    = cc25 * exp(-cc26/theta)
        else
          rd1    = cc7*(1-ca*(4./27.-dj3**2./dj2**3.)
     *               -cb*dj3/dj2**1.5)*exp(-cc8/theta)
          h1     = (cc9-cc10*theta)*(1+ca*(4./27.-dj3**2./dj2**3.)
     *                +cb*dj3/dj2**1.5)
          rs1    = cc11 * exp(-cc12/theta)
          rd2    = cc13*(1-ca*(4./27.-dj3**2./dj2**3.)
     *                -cb*dj3/dj2**1.5)*exp(-cc14/theta)
          h2     = (cc15-cc16*theta)*(1+ca*(4./27.-dj3**2./dj2**3.)
     *                 +cb*dj3/dj2**1.5)
          rs2    = cc17 * exp(-cc18/theta)
          rd3    = cc21*(1-ca*(4./27.-dj3**2./dj2**3.)
     *              -cb*dj3/dj2**1.5)*exp(-cc22/theta)
          h3     = (cc23-cc24*theta)*(1+ca*(4./27.-dj3**2./dj2**3.)
     *                 +cb*dj3/dj2**1.5)
          rs3    = cc25 * exp(-cc26/theta)
        endif
```

After the else statement in the code above, the calculations for rd1, h1, rd2, h2, rd3, and h3 all contain two of the same calculations. The terms containing ca and cb in each of these calculations will be the same for a given element on a single increment. Therefore the following substitution was made:

```
HRST1  = ca*(4./27.-dj3**2./dj2**3.)
HRST2  = cb*dj3/dj2**1.5
```

The resulting code after the change is in the table below.

VUMAT (updated)

```
c stress state dependent material constants
        adj    = half*(one+tanh(cc19*(cc20-theta)))
        if(cc19.eq.zero) adj = one
        vtheta = cc1*exp(-cc2/theta)
        ytheta = cc3*exp(cc4/theta)*adj
        ftheta = cc5*exp(-cc6/theta)
        if(dj2 .lt. 1.0D-08)then
          rd1=cc7*(1-ca*(4./27.))*exp(-cc8/theta)
          h1=(cc9-cc10*theta)*(1+ca*(4./27.))
          rs1    = cc11 * exp(-cc12/theta)
          rd2    = cc13*(1-ca*(4./27.))*exp(-cc14/theta)
          h2     = (cc15-cc16*theta)*(1+ca*(4./27.))
          rs2    = cc17 * exp(-cc18/theta)
          rd3    = cc21*(1-ca*(4./27.))*exp(-cc22/theta)
          h3     = (cc23-cc24*theta)*(1+ca*(4./27.))
          rs3    = cc25 * exp(-cc26/theta)
        else
          HRST1  = ca*(4./27.-dj3**2./dj2**3.)
          HRST2  = cb*dj3/dj2**1.5
          rd1    = cc7*(1-HRST1
     *               -HRST2)*exp(-cc8/theta)
          h1     = (cc9-cc10*theta)*(1+HRST1
     *                +HRST2)
          rs1    = cc11 * exp(-cc12/theta)
          rd2    = cc13*(1-HRST1
     *                -HRST2)*exp(-cc14/theta)
          h2     = (cc15-cc16*theta)*(1+HRST1
     *                 +HRST2)
          rs2    = cc17 * exp(-cc18/theta)
          rd3    = cc21*(1-HRST1
     *              -HRST2)*exp(-cc22/theta)
          h3     = (cc23-cc24*theta)*(1+HRST1
     *                 +HRST2)
          rs3    = cc25 * exp(-cc26/theta)
        endif
```

The second change involves a similar situation to the first. The term, $\left(\frac{dcs0}{dcs}\right)^{zz}$, was isolated in which there no possibility of change over the entire course of the simulation, but this term is calculated in 8 different places in the code. Therefore, this term was replaced with the constant, dcs1, and subsequently replaced throughout the code. These changes resulted in an increase in speed of about 11% for a 1 element simulation run on one processor.

# APPENDIX C

## UMAT REFERENCED IN THIS STUDY

```
       subroutine umat(stress,statev,ddsdde,sse,spd,scd,
     1   rpl,ddsddt,drplde,drpldt,
     2   stran,dstran,time,dtime,temp,dtemp,predef,dpred,cmname,
     3   ndi,nshr,ntens,nstatv,props,nprops,coords,drot,pnewdt,
     4   celent,dfgrd0,dfgrd1,noel,npt,layer,kspt,kstep,kinc)
       INCLUDE 'ABA_PARAM.INC'
c ************************************************
      character*8 cmname
      dimension stress(ntens),statev(nstatv),
     1 ddsdde(ntens,ntens),
     2 ddsddt(ntens),drplde(ntens),
     3 stran(ntens),dstran(ntens),time(2),predef(1),dpred(1),
     4 props(nprops),coords(3),drot(3,3),t(6),de(6),
     5 dfgrd0(3,3),dfgrd1(3,3),xi(6)
c
      data third  /.333333333333333/
      data twothd /.666666666666667/
      data con1   /.81649658092773/
      data pi     /3.1415927/
      data iparam1,iparam2,iparam3 / 0,0,0 /
c
c**********************************************************************
c
c     *  ntens = number of non-zero stress components (4 for 2d, 6 for 3d)
c     *  nshr  = number of non-zero shear components (1 for 2d, 3 for 3d)
c     *  ndi   = number of non-zero normal stresses (always 3)
c     *  nstatv = number of state variables (25)
c     *  nprops = number of material parameters (50)
c
c**********************************************************************
c
c     *  statev(1) = alpha-xx
c     *  statev(2) = alpha-yy
c     *  statev(3) = alpha-zz
c     *  statev(4) = alpha-xy
c     *  statev(5) = alpha-yz
c     *  statev(6) = alpha-zx
c     *  statev(7) = kappa
c     *  statev(8) = temperature
c     *  statev(9) = effective plastic strain
c     *  statev(10) = McClintock void growth(second phase pores)
c     *  statev(11) = rate of change of M porosity
c     *  statev(12) = stress triaxiality
c     *  statev(13) = nucleation
c     *  statev(14) = damage
c     *  statev(15) = nucleation rate
c     *  statev(16) = damage rate
c     *  statev(17) = nucleation from previous time step
c     *  statev(18) = Cocks-Ashby void growth(large pores)
c     *  statev(19) = rate of change of CA porosity
c     *  statev(20) = alpha-xx long range
c     *  statev(21) = alpha-yy long range
c     *  statev(22) = alpha-zz long range
c     *  statev(23) = alpha-xy long range
c     *  statev(24) = alpha-yz long range
c     *  statev(25) = alpha-zx long range

c
c**********************************************************************
c
c     *  props(1) thru props(5) are constants for Johnson/Bammann
c     *  formulas for shear and bulk moduli
```

```
c     *  props(1) = mu zero         , props(2) = a
c     *  props(3) = K zero          , props(4) = b
c     *  props(5) = T melt          , props(6) = C1
c     *  props(7) = C2              , props(8) = C3
c     *  props(9) = C4              , props(10)= C5
c     *  props(11)= C6              , props(12)= C7
c     *  props(13)= C8              , props(14)= C9
c     *  props(15)= C10             , props(16)= C11
c     *  props(17)= C12             , props(18)= C13
c     *  props(19)= C14             , props(20)= C15
c     *  props(21)= C16             , props(22)= C17
c     *  props(23)= C18             , props(24)= C19
c     *  props(25)= C20             , props(26)= CA
c     *  props(27)= CB
c     *  props(28)= initial temperature
c     *  props(29)= heat generation coefficient
c     *  props(30)= McClintock damage constant, n
c     *  props(31)= initial void radius
c     *  props(32)= torsional constant a in nucleation model
c     *  props(33)= tension/comp constant b in nucleation model
c     *  props(34)= triaxiality constant c in nucleation model
c     *  props(35)= coefficient constant in nucleation model
c     *  props(36)= fracture toughness, related to nucleation model
c     *  props(37)= ave size of particles, related to nucleation model
c     *  props(38)= particles vol fraction, related to nucleation model
c     *  props(39)= coalescence factor, D=nucleation*void volume*coal.
c     *  props(40)= coalescence factor, D=nucleation*void volume*coal.
c     *  props(41)= reference grain size or dendrite cell size,dcs0
c     *  props(42)= grain size or dendrite cell size of material,dcs
c     *  props(43)= grain size or dendrite cell size exponent,zz
c     *  props(44)= initial void volume fraction for CA void growth
c     *  props(45)= C21             , props(46)= C22
c     *  props(47)= C23             , props(48)= C24
c     *  props(49)= C25             , props(50)= C26
c     *  props(51)= nucleation temperature dependence
c     *  props(52)= coalescence temperature dependence
c     *  props(53)= flag to use vvfr4.dat file, 0=no, 1=yes
c     *  props(54)= flag to use random vvf generator, 0=no, 1=yes
c     *  props(55)= Cocks Ashby void growth constant
c*****************************************************************
c
c   * iparam1 = 0 for linear return (uses total strain in recovery)
c   *         = 1 for quadratic return (uses plastic strain in recovery)
c   *           (only linear return implemented in this version)
c
c   * iparam2 = 0 for Simo tangent stiffness matrix
c   *         = 1 for Lathrop tangent stiffness matrix
c   *           (only Simo stiffness implemented in this version)
c
c   * iparam3 = 0 for trial kappa = kappa(n)
c   *         = 1 for trial kappa = kappa(n + 1/2)
c
c*****************************************************************************
c---- initialize void volume fraction
c     this initization reads void volume fraction from the file
c     vvfr4.dat which is a text file in which each line contains
c     two entries:
c       element number, void volume fraction
c     the first line must be element 1, the next element 2 etc.
c     this file may be using the program mkvvf
c
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
```

```
c      if (props(53).eq.1)then
c        if(time(2).eq.0.0)then
c         write(6,'(a,i4)') "(initdmg.f) init umat, el = ",noel
c          open(11,file='/cavs/cmd/data1/users/dbh44/Kfrazier/m5b1/m5v1.csv',
c     *    status='OLD')
c          do i=1,noel
c            read(11,*)idum, vvf
c             vvf = 0.0
c          end do
c          close(11)
c          props(44) = vvf
c       end if
c      else
c       go to 15
c      end if
c      print *, "after initialization"
c---- initialize state variables
c     temperature is set first by *initial condition command in abaqus
c     or by props(28)
c 13   if (props(54).eq.1)then
c         if(time(2).eq.0.0)then
c            iseed=3
c            do i=1,noel
c               iseed = 2045*iseed + 1
c               iseed = iseed -((iseed/1048576)*1048576)
c               randx = real(iseed + 1)/1048577.0
c               vvf = (.06)*randx
c            end do
c            props(44) = vvf
c         end if
c      else
c         go to 15
c      end if
c
 15   if(time(2).eq.0.0)then
        statev(1) = 0.0
        statev(2) = 0.0
        statev(3) = 0.0
        statev(4) = 0.0
        statev(5) = 0.0
        statev(6) = 0.0
        statev(7) = 0.0
        statev(8) = props(28)
        statev(9) = 0.0
        statev(10) = pi*props(31)**2.0
        statev(11) = 0.0
        statev(12) = 0.0
        statev(13) = props(35)
        statev(14) = props(44)
        statev(15) = 0.0
        statev(16) = 0.0
        statev(17) = 0.0
        statev(18) = props(44)
        statev(19) = 0.0
        statev(20) = 0.0
        statev(21) = 0.0
        statev(22) = 0.0
        statev(23) = 0.0
        statev(24) = 0.0
        statev(25) = 0.0
         if(temp.eq.0.0)then
          if(props(28).eq.0.0)then
```

```
            write(*,*)' error - temperature is zero'
            stop
           else
            statev(8) = props(28)
           endif
          else
           statev(8) = temp
          endif
        endif
c
c**********************************************************************
c
      cc1  = props(6)
      cc2  = props(7)
      cc3  = props(8)
      cc4  = props(9)
      cc5  = props(10)
      cc6  = props(11)
      cc7  = props(12)
      cc8  = props(13)
      cc9  = props(14)
      cc10 = props(15)
      cc11 = props(16)
      cc12 = props(17)
      cc13 = props(18)
      cc14 = props(19)
      cc15 = props(20)
      cc16 = props(21)
      cc17 = props(22)
      cc18 = props(23)
      cc19 = props(24)
      cc20 = props(25)
      cc21 = props(45)
      cc22 = props(46)
      cc23 = props(47)
      cc24 = props(48)
      cc25 = props(49)
      cc26 = props(50)
      ca = props(26)
      cb = props(27)
      cd1 = props(39)
      cd2 = props(40)
      dcs0 = props(41)
      dcs = props(42)
      zz = props(43)
c
      htcp = props(29)
c
c---- g = shear modulus   twog = 2*g   blk = bulk modulus
c
      if(props(28).eq.0.)statev(8) = temp
      theta = statev(8)
      if (props(5).eq.0) then
      blk = props(3)
      g = props(1)
      else
      tratio = theta/props(5)
      tratio = min(tratio,0.9999)
      g = props(1)*(1.-tratio*exp(props(2)*(1.-1./tratio)))
      endif
      twog = 2.0 * g
      blk = props(3) - props(4)*tratio
```

```
c
c---- damage
      dam1=1.0-statev(14)
      dam2 = 1.0-min(1.0,dtime*statev(16)/dam1)
      phi1 = 1.0-statev(18)
c
c---- calculate pressure
c
      davg = third * (dstran(1) + dstran(2) + dstran(3))
      pold = third * (stress(1) + stress(2) + stress(3))
      p = pold*dam2 + dam1*blk * davg * 3.0
c
c---- check for melt
c
      if(theta.gt.props(5)) then
       do 35 i=1,ntens
       stress(i) = 0.
   35  statev(i) = 0.
       p = min(0.,p)
       stress(1) = p
       stress(2) = p
       stress(3) = p
       statev(7) = 0.
       statev(9) = 0.
       go to 315
       endif
c
c---- compute function evaluations
c       theta = temperature
c       ytheta = static yield stress
c       vtheta,ftheta = functions to define rate dependence of yield
c       h1,h2,h3 = plastic hardeing moduli
c       rs1,rs2,h3 = static recovery functions
c       rd1,rd2,h3 = dynamic recovery functions
c
c deviatoric stress
       ds11=stress(1)-pold
       ds22=stress(2)-pold
       ds33=stress(3)-pold
       ds12=stress(4)
       ds23=stress(5)
       ds13=stress(6)
c invariants of stress
       dj2=0.5*(ds11**2.+ds22**2.+ds33**2.+
     *     2*(ds12**2.+ds23**2.+ds13**2.))
       dj3=ds11*(ds22*ds33-ds23**2.)-ds22*(ds11*ds33-ds13**2.)
     *     +ds33*(ds22*ds11-ds12**2.)
c
c stress state dependent material constants
      adj    = 0.5*(1.+tanh(cc19*(cc20-theta)))
      if(cc19.eq.0.)adj=1.0
      vtheta = cc1*exp(-cc2/theta)
      ytheta = cc3*exp(cc4/theta)*adj
      ftheta = cc5*exp(-cc6/theta)
        if(dj2.eq.0)then
          rd1=cc7*(1-ca*(4./27.))*exp(-cc8/theta)
          h1=(cc9-cc10*theta)*(1+ca*(4./27.))
          rs1    = cc11 * exp(-cc12/theta)
          rd2    = cc13*(1-ca*(4./27.))*exp(-cc14/theta)
          h2     = (cc15-cc16*theta)*(1+ca*(4./27.))
          rs2    = cc17 * exp(-cc18/theta)
          rd3    = cc21*(1-ca*(4./27.))*exp(-cc22/theta)
```

```
          h3     = (cc23-cc24*theta)*(1+ca*(4./27.))
          rs3    = cc25 * exp(-cc26/theta)

        else
          rd1    = cc7*(1-ca*(4./27.-dj3**2./dj2**3.)
     *               -cb*dj3/dj2**1.5)*exp(-cc8/theta)
          h1     = (cc9-cc10*theta)*(1+ca*(4./27.-dj3**2./dj2**3.)
     *                +cb*dj3/dj2**1.5)
          rs1    = cc11 * exp(-cc12/theta)
          rd2    = cc13*(1-ca*(4./27.-dj3**2./dj2**3.)
     *                -cb*dj3/dj2**1.5)*exp(-cc14/theta)
          h2     = (cc15-cc16*theta)*(1+ca*(4./27.-dj3**2./dj2**3.)
     *                 +cb*dj3/dj2**1.5)
          rs2    = cc17 * exp(-cc18/theta)
          rd3    = cc21*(1-ca*(4./27.-dj3**2./dj2**3.)
     *              -cb*dj3/dj2**1.5)*exp(-cc22/theta)
          h3     = (cc23-cc24*theta)*(1+ca*(4./27.-dj3**2./dj2**3.)
     *                 +cb*dj3/dj2**1.5)
          rs3    = cc25 * exp(-cc26/theta)
        endif
c
c---- update alpha using abaqus rotation matrix
c
      if(rs1.ne.0. .or. rd1.ne.0. .or. h1.ne.0.)then
      if(ntens.eq.4)then
       term1 = drot(1,1)*statev(1) + drot(1,2)*statev(4)
       term2 = drot(1,1)*statev(4) + drot(1,2)*statev(2)
       term3 = drot(2,1)*statev(1) + drot(2,2)*statev(4)
       term4 = drot(2,1)*statev(4) + drot(2,2)*statev(2)
       term5 = statev(1) + statev(2)
       statev(1) = drot(1,1)*term1+drot(1,2)*term2-term5*drot(1,3)**2
       statev(2) = drot(2,1)*term3+drot(2,2)*term4-term5*drot(2,3)**2
       statev(3) = - (statev(1) + statev(2))
       statev(4) = drot(2,1)*term1+drot(2,2)*term2 -
     $             term5*drot(1,3)*drot(2,3)
      else
       term1=drot(1,1)*statev(1)+drot(1,2)*statev(4)+drot(1,3)*statev(6)
       term2=drot(1,1)*statev(4)+drot(1,2)*statev(2)+drot(1,3)*statev(5)
       term3=drot(1,1)*statev(6)+drot(1,2)*statev(5)+drot(1,3)*statev(3)
       term4=drot(2,1)*statev(1)+drot(2,2)*statev(4)+drot(2,3)*statev(6)
       term5=drot(2,1)*statev(4)+drot(2,2)*statev(2)+drot(2,3)*statev(5)
       term6=drot(2,1)*statev(6)+drot(2,2)*statev(5)+drot(2,3)*statev(3)
       term7=drot(3,1)*statev(1)+drot(3,2)*statev(4)+drot(3,3)*statev(6)
       term8=drot(3,1)*statev(4)+drot(3,2)*statev(2)+drot(3,3)*statev(5)
       term9=drot(3,1)*statev(6)+drot(3,2)*statev(5)+drot(3,3)*statev(3)
       statev(1)=term1*drot(1,1)+term2*drot(1,2)+term3*drot(1,3)
       statev(2)=term4*drot(2,1)+term5*drot(2,2)+term6*drot(2,3)
       statev(3)=term7*drot(3,1)+term8*drot(3,2)+term9*drot(3,3)
       statev(4)=term1*drot(2,1)+term2*drot(2,2)+term3*drot(2,3)
       statev(5)=term4*drot(3,1)+term5*drot(3,2)+term6*drot(3,3)
       statev(6)=term1*drot(3,1)+term2*drot(3,2)+term3*drot(3,3)
      endif
      endif
c---- update long range alpha using abaqus rotation matrix
c
      if(rs3.ne.0. .or. rd3.ne.0. .or. h3.ne.0.)then
      if(ntens.eq.4)then
       term1 = drot(1,1)*statev(20) + drot(1,2)*statev(23)
       term2 = drot(1,1)*statev(23) + drot(1,2)*statev(21)
       term3 = drot(2,1)*statev(20) + drot(2,2)*statev(23)
       term4 = drot(2,1)*statev(23) + drot(2,2)*statev(21)
       term5 = statev(20) + statev(21)
```

```
        statev(20) = drot(1,1)*term1+drot(1,2)*term2-term5*drot(1,3)**2
        statev(21) = drot(2,1)*term3+drot(2,2)*term4-term5*drot(2,3)**2
        statev(22) = - (statev(20) + statev(21))
        statev(23) = drot(2,1)*term1+drot(2,2)*term2 -
     $             term5*drot(1,3)*drot(2,3)
       else
        term1=drot(1,1)*statev(20)+drot(1,2)*statev(23)+drot(1,3)*
     *       statev(25)
        term2=drot(1,1)*statev(23)+drot(1,2)*statev(21)+drot(1,3)*
     *       statev(24)
        term3=drot(1,1)*statev(25)+drot(1,2)*statev(24)+drot(1,3)*
     *       statev(22)
        term4=drot(2,1)*statev(20)+drot(2,2)*statev(23)+drot(2,3)*
     *       statev(25)
        term5=drot(2,1)*statev(23)+drot(2,2)*statev(21)+drot(2,3)*
     *       statev(24)
        term6=drot(2,1)*statev(25)+drot(2,2)*statev(24)+drot(2,3)*
     *       statev(22)
        term7=drot(3,1)*statev(20)+drot(3,2)*statev(23)+drot(3,3)*
     *       statev(25)
        term8=drot(3,1)*statev(23)+drot(3,2)*statev(21)+drot(3,3)*
     *       statev(24)
        term9=drot(3,1)*statev(25)+drot(3,2)*statev(24)+drot(3,3)*
     *       statev(22)
        statev(20)=term1*drot(1,1)+term2*drot(1,2)+term3*drot(1,3)
        statev(21)=term4*drot(2,1)+term5*drot(2,2)+term6*drot(2,3)
        statev(22)=term7*drot(3,1)+term8*drot(3,2)+term9*drot(3,3)
        statev(23)=term1*drot(2,1)+term2*drot(2,2)+term3*drot(2,3)
        statev(24)=term4*drot(3,1)+term5*drot(3,2)+term6*drot(3,3)
        statev(25)=term1*drot(3,1)+term2*drot(3,2)+term3*drot(3,3)
       endif
       endif
c
c---- compute effective strain rate
c
       if(dtime.ne.0.0)then
        dum = 0.
        do 40 i=4,ntens
   40   dum = dum + dstran(i)**2
        ddd = sqrt(dstran(1)**2+dstran(2)**2+dstran(3)**2 +
     $       0.5*dum) * con1 / dtime
        else
        ddd = 0.0
       endif
c
c---- calculate trial alpha, kappa and yield radius
c
       alphaxx = statev(1)+statev(20)
       alphayy = statev(2)+statev(21)
       alphazz = statev(3)+statev(22)
       alphaxy = statev(4)+statev(23)
       alphayz = statev(5)+statev(24)
       alphazx = statev(6)+statev(25)
       alpm = con1 * sqrt(alphaxx**2 + alphayy**2 + alphazz**2+
     $       2.*(alphaxy**2 + alphayz**2 + alphazx**2))
       sto = dtime*rs1*alpm*(dcs0/dcs)**zz
       sto2 = dtime*rs2*(dcs0/dcs)**zz
        if(iparam1.eq.0)then
         sto  = dtime * (rs1+rd1*ddd+rs3+rd3*ddd)*alpm*(dcs0/dcs)**zz
         sto2 = dtime * (rs2+rd2*ddd)*(dcs0/dcs)**zz
        endif
       do 50 i=1,ntens
```

```
   50 statev(i) = statev(i) * (1.0 - sto)
      do 51 i=20,ntens+20-1
   51 statev(i) = statev(i) * (1.0 - sto)
       if(iparam3.eq.0)then
        trialk = statev(7)
       else
        trialk = (-1.0+sqrt(1.+2*sto2*(statev(7)+0.5*h2*ddd*dtime)))/
     $                max(1.e-30,sto2)
        endif
       statev(7) = statev(7) - sto2 *  trialk * trialk
       ak = (vtheta * log((ddd+sqrt(ddd**2+ftheta**2))/ftheta) +
     $     ytheta + statev(7))*dam1
c
c---- calculate trial elastic deviatoric stresses
c
      do 60 i=1,3
   60 stress(i) = dam2*(stress(i)-pold)+dam1*twog*(dstran(i)-davg)
      do 70 i=4,ntens
   70 stress(i) = dam2*stress(i) + dam1*g*dstran(i)
c
c---- compute xi (deviatoric stress - 2/3 alpha)
c
      do 80 i=1,ntens
   80 xi(i) = stress(i) - twothd * statev(i)
c
c---- compute (magnitude of xi) squared
c
      dum = 0.0
      do 85 i=4,ntens
   85 dum = dum + xi(i)**2
      ximag2 = xi(1)**2 + xi(2)**2 + xi(3)**2 + 2.*dum
c
c---- check for plasticity
c
      ak2 = ximag2 - twothd * ak * abs(ak)
      if(ak2 .le. 0.0 .or. ddd.eq.0.0) go to 300
c
c---- plasticity process begins here
c
      ximag = sqrt(ximag2)
c
c---- return trial stresses to yield surface, add pressure term
c      and update state variables
c
      if(iparam1.eq.0)then
      dgam = (ximag-con1*ak)/(dam1*twog+twothd*(dcs0/dcs)**zz
     *                   *(h1+h3+h2*dam1))
      endif
      dgam2 = dgam / ximag
c
      dsig = dam1*twog * dgam2
      do 90 i=1,ntens
   90 stress(i) = stress(i) - dsig * xi(i)
      stress(1) = stress(1) + p
      stress(2) = stress(2) + p
      stress(3) = stress(3) + p
c
      statev(7) = statev(7) + dgam * con1 * h2*(dcs0/dcs)**zz
      statev(7) = max(0.,statev(7))
c
      dalph = (h1+h3) * dgam2
      do 100 i=1,ntens
```

```
  100 statev(i) = statev(i) + dalph * xi(i)*(dcs0/dcs)**zz
c
c---- update plastic strain (for output purposes only)
c
      statev(9) = statev(9) + dgam * con1
c
c---- update temperature for adiabatic problems
c
      dum = 0.0
      do 110 i=4,ntens
  110 dum = dum + stress(i)*xi(i)
      statev(8) = statev(8) + htcp*dgam2*( stress(1)*xi(1) +
     $            stress(2)*xi(2) + stress(3)*xi(3) + 2.0*dum)
c
c---- update damage
c
      epsdot=dgam*con1/dtime
      sige=max(1.e-15,sqrt(0.5*((stress(1)-stress(2))**2.+
     *     (stress(2)-stress(3))**2.+(stress(3)-stress(1))**2.+
     *     6*(stress(4)**2.+stress(5)**2.+stress(6)**2.))))
c
c Cocks-Ashby large pore growth term
          cacon = abs(vtheta/ytheta)
          if(cacon.lt.props(55)) then
           cacon=props(55)*(8.0*exp(-0.00705*props(28)))
          endif
          Print*,'cacon=',cacon
          dterm=2*(2*cacon-1)/(2*cacon+1)
          arg = min(15.,p*dterm/sige)
         beta = sinh(max(0.,arg) )
         c90 = 1. + cacon
         psi = min(15.,beta*dtime*epsdot*c90)
         tmp = max(0.,(1.0+(phi1**c90-1.0)*exp(psi)))
         statev(18) = min((1.0-tmp**(1./c90)),.99)
c Cocks-Ashby void growth rate
         statev(19) = beta*epsdot*(1./(1.-statev(18))**(vtheta/ytheta)
     *                -(1.-statev(18)))
c McClintock form of void growth
          abc=3.**0.5/(2.*(1.-props(30)))*sinh(3.**0.5*
     *         0.5*(1.-props(30))*(2*p/sige+third))
          vrad=props(31)*exp(statev(9)*abc/con1)
          statev(10)=pi*vrad**2.0
          statev(11)=3.*statev(10)*abc*epsdot
c Nucleation of voids
c deviatoric stress
       ds11=stress(1)-p
       ds22=stress(2)-p
       ds33=stress(3)-p
       ds12=stress(4)
       ds23=stress(5)
       ds13=stress(6)
c invariants of stress
       di1=3.*p
       dj2=0.5*(ds11**2.+ds22**2.+ds33**2.+
     *     2*(ds12**2.+ds23**2.+ds13**2.))
       dj3=ds11*(ds22*ds33-ds23**2.)-ds22*(ds11*ds33-ds13**2.)
     *     +ds33*(ds22*ds11-ds12**2.)
         if(dj2.le.0.)then
         r1=0.
         r2=0.
         r3=0.
         else
```

```
          r1=(4./27.-dj3**2./dj2**3.)
          r2=dj3/dj2**(3./2.)
          r3=di1/(dj2**0.5)
          endif
           r3=abs(r3)
c         if(r3.lt.0)then
c            r3=-r3
c           endif
          zzz=(props(32)*r1+props(33)*r2+props(34)*r3)
          zzz=abs(zzz)
          zzzz=(props(37)**0.5/(props(36)*props(38)**third))*zzz
          statev(17)=statev(13)
          statev(13)=props(35)*exp(statev(9)*zzzz/con1)*
     *               exp(-props(51)/statev(8))
c added for nonmonotonic path sequences, statev(17) is old nucleation
          if(statev(13).lt.statev(17)) then
          statev(13) = abs(statev(17)+statev(13))
          endif
c Coalescence factor
          cf=(cd1+cd2*statev(13)*statev(10))*
     *          exp(props(52)*statev(8))
     *          *(dcs0/dcs)**zz
c Damage
      damage=cf*(statev(13)*statev(10)+statev(18))
      if(damage.gt.0.6) damage=.99
      statev(14)=min(damage,0.99)
c Nucleation Rate
          epsdot=abs(epsdot)
      statev(15)=zzzz*statev(13)*epsdot
c Damage Rate
      zsecond=cf*(statev(15)*statev(10)+statev(13)*statev(11)+
     *        statev(19))
      zthird=(statev(13)*statev(10)+statev(18))*
     *       cd2*(dcs0/dcs)**zz*exp(props(52)*statev(8))*
     *       (statev(15)*statev(10)+statev(13)*statev(11))
      statev(16)=zsecond+zthird
c
c Triaxiality
        statev(12)=p/sige
c
c---- form elastic-plastic constitutive matrix
c     if bbb=1 then conventional stiffness is calculated
c     otherwise simo stiffness is calculated
c
      if(iparam2.eq.0)then
      alpm2 = con1 * sqrt(alphaxx**2 + alphayy**2 + alphazz**2
     $        + 2.*(alphaxy**2.+alphayz**2.+alphazx**2.))
      bbb = (con1*(ak+h2*con1*dgam) + alpm2 - alpm) / ximag
      bbb=min(1.0,bbb)
      slope = h1+h3+h2-(rd1+rd3)*alpm2*alpm2-rd2*statev(7)*statev(7)
      slope = max((1.e-5*(h1+h2+h3)),slope)
      gamma = 1./(1.+slope/3.0/g) - (1.0-bbb)
      r = dam1*(twog * gamma / ximag**2)
      z1 = dam1*(blk + twothd * twog * bbb)
      z2 = dam1*(blk - third * twog * bbb)
      z3 = dam1*(g * bbb)
      do 140 i=1,ntens
      do 140 j=1,ntens
  140 ddsdde(i,j) = - r * xi(i) * xi(j)
      ddsdde(1,1) = z1 + ddsdde(1,1)
      ddsdde(2,2) = z1 + ddsdde(2,2)
      ddsdde(3,3) = z1 + ddsdde(3,3)
```

```
      ddsdde(1,2) = z2 + ddsdde(1,2)
      ddsdde(1,3) = z2 + ddsdde(1,3)
      ddsdde(2,1) = z2 + ddsdde(2,1)
      ddsdde(3,1) = z2 + ddsdde(3,1)
      ddsdde(2,3) = z2 + ddsdde(2,3)
      ddsdde(3,2) = z2 + ddsdde(3,2)
      do 130 i=4,ntens
  130 ddsdde(i,i) = z3 + ddsdde(i,i)
      endif
 2000 format(' debug',5e13.5/4e13.5)
      go to 400
c
c---- elastic process begins here
c
  300 continue
      do 310 i=1,3
  310 stress(i) = stress(i) + p
      dgam=0.
c
c---- form elastic stiffness matrix
c
  315 continue
      do 320 i=1,ntens
      do 320 j=1,ntens
  320 ddsdde(i,j) = 0.0
      z1 = dam1*(blk + twothd * twog)
      z2 = dam1*(blk - third * twog)
      ddsdde(1,1) = z1
      ddsdde(2,2) = z1
      ddsdde(3,3) = z1
      ddsdde(1,2) = z2
      ddsdde(1,3) = z2
      ddsdde(2,1) = z2
      ddsdde(3,1) = z2
      ddsdde(2,3) = z2
      ddsdde(3,2) = z2
      do 330 i=4,ntens
  330 ddsdde(i,i) = dam1*g
c
c---- clean up
c
  400 continue
      statev(7) = max(statev(7),0.0)
c
      return
      end
```

# APPENDIX D

# VUMAT REFERENCED IN THIS STUDY

```
      subroutine vumat (
C Read only -
     *     nblock, ndir, nshr, nstatev, nfieldv, nprops, lanneal,
     *     stepTime, totalTime, dt, cmname, coordMp, charLength,
     *     props, density, strainInc, relSpinInc,
     *     tempOld, stretchOld, defgradOld, fieldOld,
     *     stressOld, stateOld, enerInternOld, enerInelasOld,
     *     tempNew, stretchNew, defgradNew, fieldNew,
C Write only -
     *     stressNew, stateNew, enerInternNew, enerInelasNew )
C
      include 'vaba_param.inc'
C
      dimension coordMp(nblock,*), charLength(nblock), props(nprops),
     1     density(nblock), strainInc(nblock,ndir+nshr),
     2     relSpinInc(nblock,nshr), tempOld(nblock),
     3     stretchOld(nblock,ndir+nshr),
     4     defgradOld(nblock,ndir+nshr+nshr),
     5     fieldOld(nblock,nfieldv), stressOld(nblock,ndir+nshr),
     6     stateOld(nblock,nstatev), enerInternOld(nblock),
     7     enerInelasOld(nblock), tempNew(nblock),
     8     stretchNew(nblock,ndir+nshr),
     9     defgradNew(nblock,ndir+nshr+nshr),
     1     fieldNew(nblock,nfieldv),
     2     stressNew(nblock,ndir+nshr), stateNew(nblock,nstatev),
     3     enerInternNew(nblock), enerInelasNew(nblock)
      dimension xi(6)
C
      character*80 cmname
      parameter ( zero = 0.d0, one = 1.d0, two = 2.d0, three = 3.d0,
     *     third = one / three, half = 0.5d0, twothds = two / three,
     *     op5 = 1.5d0 )

      data con1   /.81649658092773/
      data pi     /3.1415927/
      data iparam1,iparam2,iparam3 / 0,0,0 /
c
c**********************************************************************
c
c     *  ntens = number of non-zero stress components (4 for 2d, 6 for 3d)
c     *  nshr  = number of non-zero shear components (1 for 2d, 3 for 3d)
c     *  ndi   = number of non-zero normal stresses (always 3)
c     *  nstatv = number of state variables (25)
c     *  nprops = number of material parameters (50)
c
c**********************************************************************
c
c     *  statev(1) = alpha-xx
c     *  statev(2) = alpha-yy
c     *  statev(3) = alpha-zz
c     *  statev(4) = alpha-xy
c     *  statev(5) = alpha-yz
c     *  statev(6) = alpha-zx
c     *  statev(7) = kappa
c     *  statev(8) = temperature
c     *  statev(9) = effective plastic strain
c     *  statev(10) = McClintock void growth (second phase pores)
c     *  statev(11) = rate of change of M porosity
c     *  statev(12) = stress triaxiality
c     *  statev(13) = nucleation
c     *  statev(14) = damage
c     *  statev(15) = nucleation rate
```

```
c     *  statev(16) = damage rate
c     *  statev(17) = nucleation from previous time step
c     *  statev(18) = Cocks-Ashby void growth(large pores)
c     *  statev(19) = rate of change of CA porosity
c     *  statev(20) = alpha-xx long range
c     *  statev(21) = alpha-yy long range
c     *  statev(22) = alpha-zz long range
c     *  statev(23) = alpha-xy long range
c     *  statev(24) = alpha-yz long range
c     *  statev(25) = alpha-zx long range
c
c**********************************************************************
c
c     *  props(1) thru props(5) are constants for Johnson/Bammann
c     *  formulas for shear and bulk moduli
c     *  props(1) = mu zero         , props(2) = a
c     *  props(3) = K zero          , props(4) = b
c     *  props(5) = T melt          , props(6) = C1
c     *  props(7) = C2              , props(8) = C3
c     *  props(9) = C4              , props(10)= C5
c     *  props(11)= C6              , props(12)= C7
c     *  props(13)= C8              , props(14)= C9
c     *  props(15)= C10             , props(16)= C11
c     *  props(17)= C12             , props(18)= C13
c     *  props(19)= C14             , props(20)= C15
c     *  props(21)= C16             , props(22)= C17
c     *  props(23)= C18             , props(24)= C19
c     *  props(25)= C20             , props(26)= CA
c     *  props(27)= CB
c     *  props(28)= initial temperature
c     *  props(29)= heat generation coefficient
c     *  props(30)= McClintock damage constant, n
c     *  props(31)= initial void radius
c     *  props(32)= torsional constant a in nucleation model
c     *  props(33)= tension/comp constant b in nucleation model
c     *  props(34)= triaxiality constant c in nucleation model
c     *  props(35)= coefficient constant in nucleation model
c     *  props(36)= fracture toughness, related to nucleation model
c     *  props(37)= ave size of particles, related to nucleation model
c     *  props(38)= particles vol fraction, related to nucleation model
c     *  props(39)= coalescence factor, D=nucleation*void volume*coal.
c     *  props(40)= coalescence factor, D=nucleation*void volume*coal.
c     *  props(41)= reference grain size or dendrite cell size,dcs0
c     *  props(42)= grain size or dendrite cell size of material,dcs
c     *  props(43)= grain size or dendrite cell size exponent,zz
c     *  props(44)= initial void volume fraction for CA void growth
c     *  props(45)= C21             , props(46)= C22
c     *  props(47)= C23             , props(48)= C24
c     *  props(49)= C25             , props(50)= C26
c     *  props(51)= nucleation temperature dependence
c     *  props(52)= coalescence temperature dependence
c     *  props(53)= flag to use vvfr4.dat file, 0=no, 1=yes
c     *  props(54)= Cacon
c     *  props(55)= elastic modulus - porosity exponent
c**********************************************************************
c
c   * iparam1 = 0 for linear return (uses total strain in recovery)
c   *         = 1 for quadratic return (uses plastic strain in recovery)
c   *           (only linear return implemented in this version)
c
c   * iparam2 = 0 for Simo tangent stiffness matrix
c   *         = 1 for Lathrop tangent stiffness matrix
```

```
c   *           (only Simo stiffness implemented in this version)
c
c   * iparam3 = 0 for trial kappa = kappa(n)
c   *         = 1 for trial kappa = kappa(n + 1/2)
c
c*********************************************************************
c---- initialize void volume fraction
c     this initization reads void volume fraction from the file
c     vvfr4.dat which is a text file in which each lineabaqus analysis
user=vumat_dmg-55p_edit.f input=RND.inp job=RND double cpus=4 parallel=domain
domains=4 contains
c     two entries:
c       element number, void volume fraction
c     the first line must be element 1, the next element 2 etc.
c     this file may be using the program mkvvf
c
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
c*********************************************************************
c
      ntens = ndir+nshr
c
c-------  BCJ parameters
c
c      set material constants
c
      cc1  = props(6)
      cc2  = props(7)
      cc3  = props(8)
      cc4  = props(9)
      cc5  = props(10)
      cc6  = props(11)
      cc7  = props(12)
      cc8  = props(13)
      cc9  = props(14)
      cc10 = props(15)
      cc11 = props(16)
      cc12 = props(17)
      cc13 = props(18)
      cc14 = props(19)
      cc15 = props(20)
      cc16 = props(21)
      cc17 = props(22)
      cc18 = props(23)
      cc19 = props(24)
      cc20 = props(25)
c
      htcp = props(29)
c
      cc21 = props(45)
      cc22 = props(46)
      cc23 = props(47)
      cc24 = props(48)
      cc25 = props(49)
      cc26 = props(50)
c
      ca   = props(26)
      cb   = props(27)
      cd1  = props(39)
      cd2  = props(40)
      dcs0 = props(41)
      dcs  = props(42)
```

```
      zz   = props(43)
      zeta = props(55)
c   Add this term throughout
      dcs1 = (dcs0 / dcs)**zz
c
c-----------------------------------------------------------------------
c     If stepTime equals to zero, assume the material pure elastic
c     and use initial elastic modulus
c
      if ( TotalTime .eq. zero ) then
        do k = 1, nblock
c
c--------- initialize state variables
          if (props(54).eq.one)then
            randx = rand(0)/(1-2*rand(0))
            vvf = props(44)*randx
          endif
         stateOld(k,8)  = props(28)
         stateOld(k,10) = pi*props(31)**two
         stateOld(k,13) = props(35)
         stateOld(k,14) = props(44)
         stateOld(k,18) = props(44)
c--------- temperature is set first by *initial condition command in abaqus
c     or by props(28)
          stateOld(k,8)  = props(28)
          if(tempOld(k) .eq. zero) then
            if(props(28) .eq. zero) then
              write(*,*)' error - temperature is zero'
              stop
            else
              stateOld(k,8) = props(28)
            endif
          else
            stateOld(k,8) = tempOld(k)
          endif
          if(props(28).eq.zero) stateOld(k,8) = tempOld(k)
        end do
      end if
      if ( StepTime .eq. zero ) then
        do k = 1, nblock
c--------- g = shear modulus   twog = 2*g   blk = bulk modulus
          g   = props(1)
          blk = props(3)
c---------- Trial stress
          blkg  = blk-twothds*g
          trace = strainInc(k,1) + strainInc(k,2) + strainInc(k,3)
          blkgt = blkg*trace
          do i=1,ntens
            stressNew(k,i) = stressOld(k,i) + twog * strainInc(k,i)
          end do
          stressNew(k,1) = stressNew(k,1)
     *                   + twog * strainInc(k,1) + blkgt
          stressNew(k,2) = stressNew(k,2)
     *                   + twog * strainInc(k,2) + blkgt
          stressNew(k,3) = stressNew(k,3)
     *                   + twog * strainInc(k,3) + blkgt
        end do
        return
      end if
c-----------------------------------------------------------------------
c
      do 100 k = 1, nblock
```

```
c
c--------- temperature
        if(props(28).eq.zero) stateOld(k,8) = tempOld(k)
        theta = stateOld(k,8)
c--------- state variables
        do i=1,nstatev
           stateNew(k,i) = stateOld(k,i)
        end do
c-------- damage
c        stateOld(k,14) = zero
        dam1 = one-stateOld(k,14)
        dam2 = one-min(one,dt*stateOld(k,16)/dam1)
        phi1 = one-stateOld(k,18)
c
c--------- g = shear modulus   twog = 2*g   blk = bulk modulus
        if (props(5).eq.zero) then
           tratio = zero
           g   = props(1)
        else
           tratio = theta/props(5)
           tratio = min(tratio,0.9999)
           g = props(1)*(one-tratio*exp(props(2)*(one-one/tratio)))
        endif
        twoga = two * g
        blka  = props(3) - props(4)*tratio
        twog  = (twoga * dam1**zeta)
        blk   = (blka * dam1**zeta)
c
c------ calculate pressure
        davg = third*(strainInc(k,1) + strainInc(k,2) + strainInc(k,3))
        pold = third*(stressOld(k,1) + stressOld(k,2) + stressOld(k,3))
        p = pold*dam2 + dam1*blk * davg * three
c
c---- check for melt
c
        if(theta.gt.props(5)) then
          do i=1,ntens
            stressNew(k,i) = zero
            stateNew(k,i)  = zero
          end do
          p = min(zero,p)
          stressNew(k,1) = p
          stressNew(k,2) = p
          stressNew(k,3) = p
          stateNew(k,7)  = zero
          stateNew(k,9)  = zero
          go to 200
        endif
c
c---- compute function evaluations
c       theta = temperature
c       ytheta = static yield stress
c       vtheta,ftheta = functions to define rate dependence of yield
c       h1,h2 = plastic hardeing moduli
c       rs1,rs2 = static recovery functions
c       rd1,rd2 = dynamic recovery functions
c
c deviatoric stress
        ds11 = stressOld(k,1)-pold
        ds22 = stressOld(k,2)-pold
        ds33 = stressOld(k,3)-pold
        ds12 = stressOld(k,4)
```

```
        ds23 = stressOld(k,5)
        ds13 = stressOld(k,6)
c invariants of stress
        dj2 = half*(ds11**two+ds22**two+ds33**two
     *        + two*(ds12**two+ds23**two+ds13**two))
        dj3 = ds11*(ds22*ds33-ds23**two)-ds22*(ds11*ds33-ds13**two)
     *        + ds33*(ds22*ds11-ds12**two)
c
c stress state dependent material constants
        adj    = half*(one+tanh(cc19*(cc20-theta)))
        if(cc19.eq.zero) adj = one
        vtheta = cc1*exp(-cc2/theta)
        ytheta = cc3*exp(cc4/theta)*adj
        ftheta = cc5*exp(-cc6/theta)
c        if(dj2.eq.0)then
c        if(dj2 .lt. 1.0D-08)then
c          rd1=cc7*(1-ca*(4./27.))*exp(-cc8/theta)
c          h1=(cc9-cc10*theta)*(1+ca*(4./27.))
c          rs1    = cc11 * exp(-cc12/theta)
c          rd2    = cc13*(1-ca*(4./27.))*exp(-cc14/theta)
c          h2     = (cc15-cc16*theta)*(1+ca*(4./27.))
c          rs2    = cc17 * exp(-cc18/theta)
c          rd3    = cc21*(1-ca*(4./27.))*exp(-cc22/theta)
c          h3     = (cc23-cc24*theta)*(1+ca*(4./27.))
c          rs3    = cc25 * exp(-cc26/theta)
c        else
c          rd1    = cc7*(1-ca*(4./27.-dj3**2./dj2**3.)
c     *                -cb*dj3/dj2**1.5)*exp(-cc8/theta)
c          h1     = (cc9-cc10*theta)*(1+ca*(4./27.-dj3**2./dj2**3.)
c     *                 +cb*dj3/dj2**1.5)
c          rs1    = cc11 * exp(-cc12/theta)
c          rd2    = cc13*(1-ca*(4./27.-dj3**2./dj2**3.)
c     *                -cb*dj3/dj2**1.5)*exp(-cc14/theta)
c          h2     = (cc15-cc16*theta)*(1+ca*(4./27.-dj3**2./dj2**3.)
c     *                 +cb*dj3/dj2**1.5)
c          rs2    = cc17 * exp(-cc18/theta)
c          rd3    = cc21*(1-ca*(4./27.-dj3**2./dj2**3.)
c     *              -cb*dj3/dj2**1.5)*exp(-cc22/theta)
c          h3     = (cc23-cc24*theta)*(1+ca*(4./27.-dj3**2./dj2**3.)
c     *                 +cb*dj3/dj2**1.5)
c          rs3    = cc25 * exp(-cc26/theta)
c         endif
        if(dj2 .lt. 1.0D-08)then
          rd1=cc7*(1-ca*(4./27.))*exp(-cc8/theta)
          h1=(cc9-cc10*theta)*(1+ca*(4./27.))
          rs1    = cc11 * exp(-cc12/theta)
          rd2    = cc13*(1-ca*(4./27.))*exp(-cc14/theta)
          h2     = (cc15-cc16*theta)*(1+ca*(4./27.))
          rs2    = cc17 * exp(-cc18/theta)
          rd3    = cc21*(1-ca*(4./27.))*exp(-cc22/theta)
          h3     = (cc23-cc24*theta)*(1+ca*(4./27.))
          rs3    = cc25 * exp(-cc26/theta)
        else
         HRST1  = ca*(4./27.-dj3**2./dj2**3.)
         HRST2  = cb*dj3/dj2**1.5
          rd1    = cc7*(1-HRST1
     *               -HRST2)*exp(-cc8/theta)
          h1     = (cc9-cc10*theta)*(1+HRST1
     *                +HRST2)
          rs1    = cc11 * exp(-cc12/theta)
          rd2    = cc13*(1-HRST1
     *                -HRST2)*exp(-cc14/theta)
```

```
          h2     = (cc15-cc16*theta)*(1+HRST1
     *                 +HRST2)
          rs2    = cc17 * exp(-cc18/theta)
          rd3    = cc21*(1-HRST1
     *              -HRST2)*exp(-cc22/theta)
          h3     = (cc23-cc24*theta)*(1+HRST1
     *                 +HRST2)
          rs3    = cc25 * exp(-cc26/theta)
        endif
c
c---- compute effective strain rate
c
        if(dt .ne. zero)then
          dum = zero
          do i=4,ntens
            dum = dum + strainInc(k,i)**two
          end do
          ddd = sqrt(strainInc(k,1)**two+strainInc(k,2)**two
     *             + strainInc(k,3)**two + half*dum) * con1 / dt
        else
          ddd = zero
        endif
c
c---- calculate trial alpha, kappa and yield radius
c
        alphaxx = stateOld(k,1)+stateOld(k,20)
        alphayy = stateOld(k,2)+stateOld(k,21)
        alphazz = stateOld(k,3)+stateOld(k,22)
        alphaxy = stateOld(k,4)+stateOld(k,23)
        alphayz = stateOld(k,5)+stateOld(k,24)
        alphazx = stateOld(k,6)+stateOld(k,25)
        alpm = con1 * sqrt(alphaxx**two + alphayy**two + alphazz**two
     *        + two*(alphaxy**two + alphayz**two + alphazx**two))
c        sto = dt*rs1*alpm*(dcs0/dcs)**zz
c        sto2 = dt*rs2*(dcs0/dcs)**zz
        sto = dt*rs1*alpm*dcs1
        sto2 = dt*rs2*dcs1
        if(iparam1.eq.0)then
c          sto  = dt * (rs1+rd1*ddd+rs3+rd3*ddd)*alpm*(dcs0/dcs)**zz
c          sto2 = dt * (rs2+rd2*ddd)*(dcs0/dcs)**zz
          sto  = dt * (rs1+rd1*ddd+rs3+rd3*ddd)*alpm*dcs1
          sto2 = dt * (rs2+rd2*ddd)*dcs1
        endif
        do i=1,ntens
          stateNew(k,i)    = stateOld(k,i) * (one - sto)
          stateNew(k,i+19) = stateOld(k,i+19) * (one - sto)
        end do
        if(iparam3.eq.0)then
          trialk = stateOld(k,7)
        else
     *    trialk = (-one+sqrt(one+two*sto2*(stateOld(k,7)
     *                        + half*h2*ddd*dt)))/ max(1.e-30,sto2)
        endif
        stateNew(k,7) = stateOld(k,7) - sto2 *  trialk * trialk
        ak = (vtheta * log((ddd+sqrt(ddd**two+ftheta**two))/ftheta)
     *     + ytheta + stateNew(k,7))*dam1
c
c---- calculate trial elastic deviatoric stresses
c
        do i=1,3
          stressNew(k,i) = dam2*(stressOld(k,i)-pold)
     *                   + dam1*twog*(strainInc(k,i)-davg)
```

```
        end do
        do i=4,ntens
          stressNew(k,i) = dam2*stressOld(k,i) + dam1*g*strainInc(k,i)
        end do
c
c---- compute xi (deviatoric stress - 2/3 alpha)
c
        do i=1,ntens
          xi(i) = stressNew(k,i) - twothds * stateNew(k,i)
        end do
c
c---- compute (magnitude of xi) squared
c
        dum = zero
        do i=4,ntens
          dum = dum + xi(i)**two
        end do
        ximag2 = xi(1)**two + xi(2)**two + xi(3)**two + two*dum
c
c---- check for plasticity
c
        ak2 = ximag2 - twothds * ak * abs(ak)
        if(ak2.le.zero .or. ddd.eq.zero) then
c
c---- elastic process begins here
          do i=1,3
            stressNew(k,i) = stressNew(k,i) + p
          end do
          go to 200
        end if
c
c---- plasticity process begins here
c
        ximag = sqrt(ximag2)
c
c---- return trial stresses to yield surface, add pressure term
c      and update state variables
c
        if(iparam1.eq.0)then
c        dgam = (ximag-con1*ak)/(dam1*twog+twothds*(dcs0/dcs)**zz
c     *                     *(h1+h2*dam1))
        dgam = (ximag-con1*ak)/(dam1*twog+twothds*dcs1
     *                    *(h1+h2*dam1))
        endif
        dgam2 = dgam / ximag
c
        dsig = dam1*twog * dgam2
        do i=1,ntens
          stressNew(k,i) = stressNew(k,i) - dsig * xi(i)
        end do
        stressNew(k,1) = stressNew(k,1) + p
        stressNew(k,2) = stressNew(k,2) + p
        stressNew(k,3) = stressNew(k,3) + p
c
        stateNew(k,7) = stateNew(k,7) + dgam * con1 * h2*(dcs0/dcs)**zz
        stateNew(k,7) = max(zero,stateNew(k,7))
c
        dalph = (h1+h3) * dgam2
        do i=1,ntens
c          stateNew(k,i) = stateNew(k,i) + dalph * xi(i)*(dcs0/dcs)**zz
          stateNew(k,i) = stateNew(k,i) + dalph * xi(i)*dcs1
        end do
```

```
c
c---- update plastic strain (for output purposes only)
c
      stateNew(k,9) = stateOld(k,9) + dgam * con1
c
c---- update temperature for adiabatic problems
c
      dum = zero
      do i=4,ntens
        dum = dum + stressNew(k,i)*xi(i)
      end do
      stateNew(k,8) = stateOld(k,8)
     *              + htcp*dgam2*( stressNew(k,1)*xi(1)
     *              + stressNew(k,2)*xi(2) + stressNew(k,3)*xi(3)
     *              + two*dum)
c---- update damage
c
      epsdot = dgam*con1/dt
      sigeqv = (stressNew(k,1)-stressNew(k,2))**two
     *       + (stressNew(k,2)-stressNew(k,3))**two
     *       + (stressNew(k,3)-stressNew(k,1))**two
     *       + 6.0*(stressNew(k,4)**two+stressNew(k,5)**two
     *                                 +stressNew(k,6)**two)
      sigeqv = sqrt(half*sigeqv)
      sige = max(1.e-15,sigeqv)
c
c Cocks-Ashby large pore growth term
         cacon = abs(vtheta/ytheta)
c         if(cacon.lt.props(54))  cacon=props(54)
         if(cacon.lt.props(54)) then
          cacon=props(54)*(8.0*exp(-0.00705*props(28)))
         end if
         dterm=two*(two*cacon-1)/(two*cacon+1)
         arg = min(15.,p*dterm/sige)
        beta = sinh(max(zero,arg) )
        c90 = one + cacon
        psi = min(15.,beta*dt*epsdot*c90)
        tmp = max(zero,(one+(phi1**c90-one)*exp(psi)))
        stateNew(k,18) = min((one-tmp**(one/c90)),.99)
c Cocks-Ashby void growth rate
         vy = vtheta/ytheta
        stateNew(k,19) = beta*epsdot*(one/(one-stateNew(k,18))**vy
     *                 -(one-stateNew(k,18)))
c McClintock form of void growth
         sqtth = three**half
         abc   = sqtth/(two*(one-props(30)))
     *         * sinh(sqtth*half*(one-props(30))*(two*p/sige+third))
         if(props(31).gt.zero) then
            vrad  = props(31)*exp(stateNew(k,9)*abc/con1)
         else
            vrad  = zero
         end if
         stateNew(k,10) = pi*vrad**two
         stateNew(k,11) = three*stateNew(k,10)*abc*epsdot
c Nucleation of voids
c deviatoric stress
         ds11 = stressNew(k,1)-p
         ds22 = stressNew(k,2)-p
         ds33 = stressNew(k,3)-p
         ds12 = stressNew(k,4)
         ds23 = stressNew(k,5)
         ds13 = stressNew(k,6)
```

```
c invariants of stress
         di1 = three*p
         dj2 = half*(ds11**two+ds22**two+ds33**two
     *       + two*(ds12**two+ds23**two+ds13**two))
         dj3 = ds11*(ds22*ds33-ds23**two)-ds22*(ds11*ds33-ds13**two)
     *       + ds33*(ds22*ds11-ds12**two)
         if(dj2.le.zero) then
           r1 = zero
           r2 = zero
           r3 = zero
         else
           r1 = (4./27.-dj3**two/dj2**3.)
           r2 = dj3/dj2**(op5)
           r3 = di1/(dj2**half)
         endif
         r3 = abs(r3)
         zzz  = (props(32)*r1+props(33)*r2+props(34)*r3)
         zzz  = abs(zzz)
         zzzz = (props(37)**half/(props(36)*props(38)**third))*zzz
         stateNew(k,17) = stateOld(k,13)
         stateNew(k,13) = props(35)*exp(stateNew(k,9)*zzzz/con1)
     *                  * exp(-props(51)/stateNew(k,8))
c added for nonmonotonic path sequences, statev(17) is old nucleation
         if(stateNew(k,13).lt.stateNew(k,17)) then
           stateNew(k,13) = abs(stateNew(k,17)+stateNew(k,13))
         endif
c Coalescence factor
c         cf = (cd1+cd2*stateNew(k,13)*stateNew(k,10))*
c     *          * exp(props(52)*stateNew(k,8))*(dcs0/dcs)**zz
         cf = (cd1+cd2*stateNew(k,13)*stateNew(k,10))*
     *          * exp(props(52)*stateNew(k,8))*dcs1
c Damage
         damage=cf*(stateNew(k,13)*stateNew(k,10)+stateNew(k,18))
         if(damage.gt.0.6) damage = .99
         stateNew(k,14) = min(damage,0.99)
c        stateNew(k,14) = zero
c Nucleation Rate
         epsdot = abs(epsdot)
         stateNew(k,15)=zzzz*stateNew(k,13)*epsdot
c Damage Rate
         zsecond = cf*(stateNew(k,15)*stateNew(k,10)
     *                +stateNew(k,13)*stateNew(k,11)
     *                +stateNew(k,19))
c         zthird = (stateNew(k,13)*stateNew(k,10)+stateNew(k,18))
c     *          * cd2*(dcs0/dcs)**zz*exp(props(52)*stateNew(k,8))
c     *          * (stateNew(k,15)*stateNew(k,10)
c     *            +stateNew(k,13)*stateNew(k,11))
         zthird = (stateNew(k,13)*stateNew(k,10)+stateNew(k,18))
     *          * cd2*dcs1*exp(props(52)*stateNew(k,8))
     *          * (stateNew(k,15)*stateNew(k,10)
     *            +stateNew(k,13)*stateNew(k,11))
         stateNew(k,16)=zsecond+zthird
c Triaxiality
         stateNew(k,12) = p/sige
c
  200 continue
c
      stateNew(k,7) = max(stateNew(k,7),zero)
c
 100  continue
      return
      end
```