Patrick Höschele

An Additive Approach to Finite Anisotropic Plasticity using Logarithmic Strains

Masterarbeit

zur Erlangung des akademischen Grades
eines Diplom–Ingenieurs

Institut für Festigkeitslehre
Technische Universität Graz
Fakultät für Maschinenbau und Wirtschaftswissenschaften

Studienrichtung:
Wirtschaftsingenieurwesen-Maschinenbau

Betreuer: Assoc. Prof. Dipl.-Ing. Dr.techn. Manfred Ulz
Beurteiler: Assoc. Prof. Dipl.-Ing. Dr.techn. Manfred Ulz

Deutsche Fassung:
Beschluss der Curricula-Kommission für Bachelor-, Master- und Diplomstudien vom 10.11.2008
Genehmigung des Senates am 1.12.2008

EIDESSTATTLICHE ERKLÄRUNG

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Graz, am …………………………… ………………………………………………..
(Unterschrift)

Englische Fassung:

STATUTORY DECLARATION

I declare that I have authored this thesis independently, that I have not used other than the declared sources / resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

…………………………… ………………………………………………..
date (signature)

Vorwort

An dieser Stelle danke ich Assoc. Prof. Dipl.-Ing. Dr.techn. Manfred Ulz für die tatkräftige Unterstützung während der Erstellung dieser Masterarbeit. Die wertvollen Ratschläge und interessanten Diskussionen bei allen anfallenden Frage- und Problemstellungen waren eine große Hilfe.

Mein Dank gilt auch Univ.-Prof. Dipl.-Math.techn. Dr.-Ing. Thomas Hochrainer für die Möglichkeit zur Erstellung meiner Masterarbeit am Institut für Festigkeitslehre.

Des Weiteren möchte ich mich bei allen Mitarbeiterinnen und Mitarbeitern des Instituts für das gute Arbeitsklima und die tolle Zusammenarbeit bedanken.

Ich bedanke mich herzlich bei meinen Eltern, die mir dieses Studium ermöglicht und mich in jeder Hinsicht unterstützt haben. Ich bedanke mich auch bei meinen Freunden, die mich während meines Studiums begleitet haben.

Besonderer Dank gilt zuletzt Annika Eitler, die mich während der Erstellung dieser Arbeit stets unterstützte, motivierte und immer ein offenes Ohr für mich hatte.

Patrick Höschele, Graz, April 2019

Abstract

This thesis presents a model to describe anisotropic, rate-independent plasticity in the logarithmic Lagrangian space for finite strains. The strain measure is decomposed into an elastic and plastic part as proposed by Green & Naghdi. The well known structures within the small strain theory can be applied by defining the strains in logarithmic strain space. Constitutive equations for the defined internal variables are derived in the logarithmic Lagrangian space. A return-mapping scheme as the implicit integration method is used as the iteration algorithm for local plasticity. A calculation specification transforms the internal variables from the logarithmic Lagrangian strain space into the Lagrangian strain space. The anisotropic material behavior in the plastic domain is modeled by material symmetry groups with the help of a constant fourth-order Hill tensor. The solution algorithm is implemented in an object oriented finite element program called "soofeaM" (software for object-oriented finite element analysis in Matlab) provided by the Institute of Strength of Materials at Graz University of Technology. Representative benchmark simulations are calculated, in order to demonstrate the performance of the proposed model and computational implementation. 8-node trilinear hexahedral elements are used in the examples to discretize the calculation domain. The simulation results show quadratic convergence and plausible mechanical behavior. Further investigations can be conducted with more appropriate element types for plasticity as the set of constitutive equations is independent on the element type.

Kurzfassung

Diese Arbeit präsentiert ein Model zur Beschreibung von anisotroper, dehnratenunabhängiger Plastizität im logarithmischen Verzerrungsraum in der Materialkonfiguration bei großen Verformungen. Das Verzerrungsmaß wird in einen elastischen und plastischen Teil zerlegt, wie von Green & Naghdi vorgeschlagen. Die bekannten Strukturen aus der Verzerrungstheorie kleiner Verformungen können dabei durch die Definition im logarithmischen Verzerrungsraum angewandt werden. Beschreibungsgleichungen für die definierten internen Variablen werden im logarithmischen Verzerrungsraum hergeleitet. Ein "return-mapping" Schema in Form einer impliziten Integration fungiert als Iterationsalgorithmus für lokale Plastizität. Eine Rechenvorschrift transformiert die internen Variablen vom logarithmischen Verzerrungsraum in den Verzerrungsraum der Materialkonfiguration. Das anisotrope Materialverhalten im plastischen Bereich wird mit Materialsymmetriegruppen mit Hilfe eines konstanten Hilltensors vierter Ordnung modelliert. Der Lösungsalgorithmus ist in ein vom Institut für Festigkeitslehre der Technischen Universität Graz bereitgestelltem objektorientiertem Finite Elemente Programm implementiert, welches sich "soofeaM" (software for object-oriented finite element analysis in Matlab) nennt. Repräsentative Benchmark Simulationen wurden berechnet, um die Leistung des vorgeschlagenen Modells bzw. der rechnerischen Implementierung zu demonstrieren. Lineare 8-Knoten Hexaeder Elemente wurden in diesen Beispielen benutzt, um das Berechnungsgebiet zu diskretisieren. Die Simulationsergebnisse zeigen quadratische Konvergenz und plausibles mechanisches Verhalten. Weitere Untersuchungen können mit für Plastizität besser geeigneten Elementtypen durchgeführt werden, da die Beschreibungsgleichungen unabhängig vom Elementtyp sind.

Contents

1. Introduction 2

1.1. Motivation . . . 2

1.2. Approach and Objectives . . . 3

2. Continuum Mechanics 4

2.1. Motion . . . 4

2.2. Deformation . . . 5

2.2.1. Deformation Gradient . . . 5

2.2.2. Strain . . . 5

2.2.3. Polar Decomposition . . . 6

2.3. Stress . . . 7

2.4. Principle of Virtual Work . . . 9

2.4.1. Equilibrium Equations . . . 9

2.4.2. Linearisation of the Equilibrium Equations . . . 12

3. Basics of Finite Element Method 15

3.1. Element Formulation . . . 15

3.2. Newton Algorithm . . . 17

4. Anisotropic Plasticity Material Model 20

4.1. Basics for the established Material Model . . . 20

4.1.1. Basics of Isotropic Plasticity . . . 20

4.1.2. Logarithmic Strain and additive Lagrangian approach to finite plasticity . . . 22

4.1.3. Calculation specification for transformations from Logarithmic Lagrangian space to Lagrangian space . . . 23

4.2. Constitutive Equations for an Additive Formulation of Plasticity . . . 25

4.2.1. Strain-energy Function . . . 25

4.2.2. Anisotropic Yield Criterion and Dissipation Inequality . . . 25

4.3. Solution Algorithm for rate-independent Plasticity . . . 28

4.3.1. Local Iteration Scheme . . . 29

4.3.2. Tangent Modulus . . . 31

4.3.3. Summary of the Solution Algorithm . . . 32

5. Verification 33

5.1. Convergence Test . . . 34

5.2. Rectangular Strip under Tension and Compression . . . 36

5.2.1. Model of Computation . . . 36

5.2.2. Results - Compression Load . . . 38

5.2.3. Results - Tension Load . . . 46

5.3. Drawing of a Thin Circular Plate 53
5.3.1. Model of Computation 53
5.3.2. Results 55

6. Conclusion 63

A. Derivations 66

B. Fourth-order tensor: Inversion 69

C. Second-order tensor: Spectral decomposition 70

D. Matlab Code 71

Bibliography 83

List of Figures

1.1. Deep drawing process. 1 - punch, 2 - blank holder, 3 - die, 4 - die mounting, 5 - ground plate, 6 - ejector (Cf. Dietrich [5]) 3
1.2. Earing of a rolled metal sheet after deep drawing. Left: experiment, Right: simulation (Cf. Singh et al. [28]) 3

2.1. Deformation gradient and nonlinear mapping φ between original and current configuration 4
2.2. Polar decomposition of the deformation gradient 7
2.3. Traction vector $\mathbf{t}$ acting on the cut area da 7
2.4. Elemental tetrahedron, exemplary only $\mathbf{t}_1$ is shown 8
2.5. Equilibrium of forces for a general deformable body 9
2.6. Different stress measurements in Lagrangian and Eulerian configuration 12

3.1. Principle illustration of (a) Cartesian and (b) natural coordinates . . 15
3.2. Virtual displacement 17
3.3. Newton Algorithm 19

4.1. Von Mises yield surface (Cf. Rösler et al. [24]) 21
4.2. Maximal dissipation at plastic deformation (Cf. Rösler et al. [24]) . . 21
4.3. Approach for the solution algorithm for rate-independent plasticity in logarithmic Lagrangian strain space (Cf. Ulz [33]) 28

5.1. Convergence Test. Considered 8-node hexahedral trilinear solid element in initial $t = 0$ and deformed state t. 34
5.2. Rectangular strip under tension and compression. Geometry of the considered calculation domain. 36
5.3. Rectangular strip under tension and compression. Mesh with the considered nodes A and B. 37
5.4. Rectangular strip under compression. Deformed meshes and displacements as shown in color bar for a) isotropic $\varrho_1 = \sqrt{3}$, b) anisotropic $\varrho_2 = 2\sqrt{3}$ and c) anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior. First column: top view for global iteration step $i = 100$. Second column: top view for $i = 200$. Third column: top view for $i = 300$. 38
5.5. Rectangular strip under compression. Displacements v in y-direction over compression displacement. Nodes A and B are represented by curves without marker and with cross. Solid line represents isotropic $\varrho_1 = \sqrt{3}$, dashed-dotted anisotropic $\varrho_2 = 2\sqrt{3}$ and dashed anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior. 39

5.6. Rectangular strip under compression. Deformed meshes and Von Mises stress as shown in color bar for a) isotropic $\varrho_1 = \sqrt{3}$, b) anisotropic $\varrho_2 = 2\sqrt{3}$ and c) anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior. First column: top view for global iteration step $i = 100$. Second column: top view for $i = 200$. Third column: top view for $i = 300$. 40

5.7. Rectangular strip under compression. Von Mises stress over compression displacement. Nodes A and B are represented by curves without marker and with cross. Solid line represents isotropic $\varrho_1 = \sqrt{3}$, dashed-dotted anisotropic $\varrho_2 = 2\sqrt{3}$ and dashed anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior. 41

5.8. Rectangular strip under compression. Deformed meshes and equivalent plastic strain as shown in color bar for a) isotropic $\varrho_1 = \sqrt{3}$, b) anisotropic $\varrho_2 = 2\sqrt{3}$ and c) anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior. First column: top view for global iteration step $i = 100$. Second column: top view for $i = 200$. Third column: top view for $i = 300$. 42

5.9. Rectangular strip under compression. Equivalent plastic strain over compression displacement. Nodes A and B are represented by curves without marker and with cross. Solid line represents isotropic $\varrho_1 = \sqrt{3}$, dashed-dotted anisotropic $\varrho_2 = 2\sqrt{3}$ and dashed anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior. 43

5.10. Rectangular strip under tension. Deformed meshes and displacements as shown in color bar for a) isotropic $\varrho_1 = \sqrt{3}$, b) anisotropic $\varrho_2 = 2\sqrt{3}$ and c) anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior. First column: top view for global iteration step $i = 100$. Second column: top view for $i = 200$. Third column: top view for $i = 300$. 46

5.11. Rectangular strip under tension. Displacements v in y-direction over tension displacement. Nodes A and B are represented by curves without marker and with cross. Solid line represents isotropic $\varrho_1 = \sqrt{3}$, dashed-dotted anisotropic $\varrho_2 = 2\sqrt{3}$ and dashed anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior. 47

5.12. Rectangular strip under tension. Deformed meshes and Von Mises stress as shown in color bar for a) isotropic $\varrho_1 = \sqrt{3}$, b) anisotropic $\varrho_2 = 2\sqrt{3}$ and c) anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior. First column: top view for global iteration step $i = 100$. Second column: top view for $i = 200$. Third column: top view for $i = 300$. 48

5.13. Rectangular strip under tension. Von Mises stress over tension displacement. Nodes A and B are represented by curves without marker and with cross. Solid line represents isotropic $\varrho_1 = \sqrt{3}$, dashed-dotted anisotropic $\varrho_2 = 2\sqrt{3}$ and dashed anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior. 49

5.14. Rectangular strip under tension. Deformed meshes and equivalent plastic strain as shown in color bar for a) isotropic $\varrho_1 = \sqrt{3}$, b) anisotropic $\varrho_2 = 2\sqrt{3}$ and c) anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior. First column: top view for global iteration step $i = 100$. Second column: top view for $i = 200$. Third column: top view for $i = 300$. 50

5.15. Rectangular strip under tension. Equivalent plastic strain over tension displacement. Nodes A and B are represented by curves without marker and with cross. Solid line represents isotropic $\varrho_1 = \sqrt{3}$, dashed-dotted anisotropic $\varrho_2 = 2\sqrt{3}$ and dashed anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior. 51

5.16. Drawing of a thin circular plate. Geometry of the calculation domain. 53

5.17. Drawing of a thin circular plate. Mesh with the considered nodes. . . 54

5.18. Drawing of a thin circular plate. Deformed meshes and displacement as shown in color bar for a) isotropic $\varrho_1 = \sqrt{3}$, b) anisotropic $\varrho_2 = 2\sqrt{3}$ and c) anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior. First column: top view for global iteration step $i = 250$. Second column: top view for $i = 500$. Third column: top view for $i = 750$. 56

5.19. Drawing of a thin circular plate. Displacement u in x-direction, v in y-direction and radial displacement |r| over drawing displacement. Nodes 11, 5 and 21 are represented by curves without marker, with cross and with rhombus. Solid line represents isotropic $\varrho_1 = \sqrt{3}$, dashed-dotted anisotropic $\varrho_2 = 2\sqrt{3}$ and dashed anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior. 57

5.20. Drawing of a thin circular plate. Von Mises stress over drawing displacement. Nodes 11, 5 and 21 are represented by curves without marker, with cross and with rhombus. Solid line represents isotropic $\varrho_1 = \sqrt{3}$, dashed-dotted anisotropic $\varrho_2 = 2\sqrt{3}$ and dashed anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior. 58

5.21. Drawing of a thin circular plate. Deformed mesh and Von Mises Stress as shown in color bar for a) isotropic $\varrho_1 = \sqrt{3}$, b) anisotropic $\varrho_2 = 2\sqrt{3}$ and c) anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior. First column: top view for global iteration step $i = 250$. Second column: top view for $i = 500$. Third column: top view for $i = 750$. 59

5.22. Drawing of a thin circular plate. Equivalent plastic strain over drawing displacement. Nodes 11, 5 and 21 are represented by curves without marker, with cross and with rhombus. Solid line represents isotropic $\varrho_1 = \sqrt{3}$, dashed-dotted anisotropic $\varrho_2 = 2\sqrt{3}$ and dashed anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior. 60

5.23. Drawing of a thin circular plate. Deformed mesh and equivalent plastic strain shown in color bar for a) isotropic $\varrho_1 = \sqrt{3}$, b) anisotropic $\varrho_2 = 2\sqrt{3}$ and c) anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior. First column: top view for global iteration step $i = 250$. Second column: $i = 500$. Third column: $i = 750$. 61

D.1. Matlab code. Schematic illustration and pseudo-code of used functions. 71

List of Tables

5.1. Convergence Test. x -, y- and z - coordinates of the considered 8-node hexahedral trilinear solid element. 34

5.2. Convergence Test. Material properties for the considered 8-node hexahedral trilinear solid element. 35

5.3. Convergence Test. Convergence of the unbalanced energy for the considered 8-node hexahedral trilinear solid element. Timestep t 1, 3, 5, 7 and 10. Global iteration counter i. 35

5.4. Rectangular strip under tension and compression. Material properties for the model of computation. 37

5.5. Rectangular strip under compression. Convergence of the unbalanced energy for a) isotropic $\varrho_1 = \sqrt{3}$, b) anisotropic $\varrho_2 = 2\sqrt{3}$ and c) anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior. Timestep $t = \{200, 250, 300\}$. Global iteration counter i. 44

5.6. Rectangular strip under tension. Convergence of the unbalanced energy for a) isotropic $\varrho_1 = \sqrt{3}$, b) anisotropic $\varrho_2 = 2\sqrt{3}$ and c) anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior. Timestep $t = \{200, 250, 300\}$. Global iteration counter i. 52

5.7. Drawing of a thin circular plate. x - and y- coordinates of the considered nodes. 54

5.8. Drawing of a thin circular plate. Material properties for the model of computation. 55

5.9. Drawing of a thin circular plate. Convergence of the unbalanced energy for a) isotropic $\varrho_1 = \sqrt{3}$, b) anisotropic $\varrho_2 = 2\sqrt{3}$ and c) anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior. Timestep $t = \{650, 700, 750\}$. Global iteration counter i. 62

1. Introduction

In state of the art finite element simulations elastic-plastic problems with large deformations can be found frequently. Different approaches exist to describe an appropriate material behavior. The locally multiplicative approach for finite plasticity is widely spread. In this case the deformation gradient is decomposed into an elastic and a plastic part. The elastic part describes rigid body motions. The reversible, elastic response vanishes by unloading the body under consideration. The plastic part depicts the irreversible deformation of the body under consideration which corresponds to the movement of dislocations in terms of the crystallography of materials. An intermediate state is introduced within this approach (see for instance Eidel & Gruttmann [6] or Menzel & Steinmann [16]). Problems emerge for a multiplicative decomposition in the case of anisotropic material behavior due to the intermediate state and the formulation of constitutive equations within this framework. This evidence is discussed for rate-independent plasticity for instance in Eidel & Gruttmann [6], Menzel & Steinmann [16] and Sansour et al. [25].

In contrast to that Green & Naghdi [7] proposed an additive approach to finite plasticity where the strain tensor is decomposed into an elastic and plastic part. This concept is employed merely as a phenomenological framework and was investigated for anisotropic rate-independent plasticity for instance by Miehe [17], Papadopoulos & Lu [22], Miehe et al. [19], Löblein et al. [12], Schröder et al. [26] and Ulz [32, 33]. A comparison of the multiplicative and additive approach to finite plasticity was conducted by Miehe & Apel [18].

1.1. Motivation

Different material machining processes cause anisotropic behavior of the processed material. Rolling and deep drawing are prominent examples. Especially deep drawing induces complex stress states due to the anisotropic material parameters of thin rolled metal sheets (see for instance Dietrich [5], Singh et al. [28] and Tikhovskiy et al. [30]). In deep drawing process a metal sheet is pressed into a die by a punch and undergoes large plastic deformations (see Figure 1.1). Uneven rims may arise during deep drawing due to the anisotropy of the metal sheets. This is illustrated in Figure 1.2 and is usually referred to as earing. The process induces inhomogeneous distribution of the mechanical properties and wall thickness. The stress state of the deformed end product and the extent of earing are of great interest for the manufacturer. The introduction of internal stresses through the machining process can influence the product lifetime and should therefore be taken into account. Knowledge about earing allows a prior modification of the process to optimize production and reduce defects.

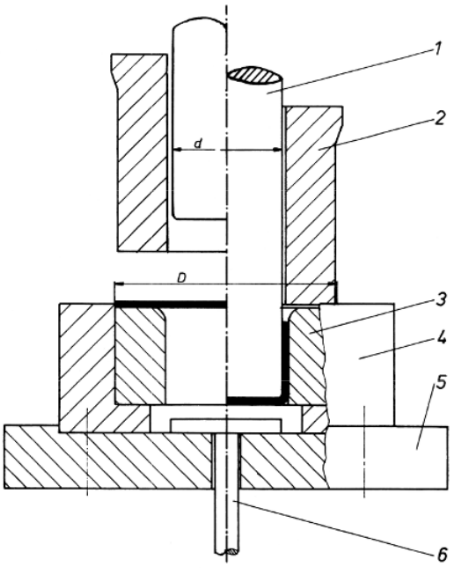


Figure 1.1. Deep drawing process. 1 - punch, 2 - blank holder, 3 - die, 4 - die mounting, 5 - ground plate, 6 - ejector (Cf. Dietrich [5])

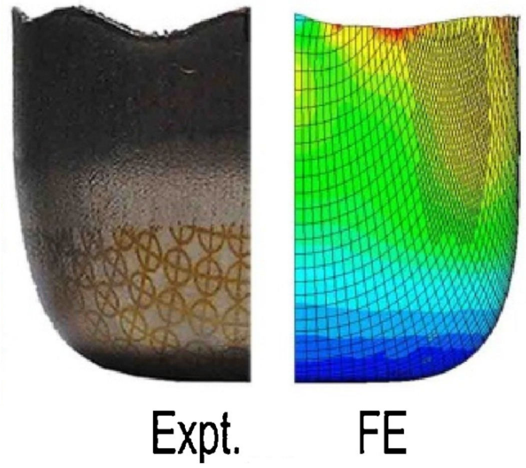


Figure 1.2. Earing of a rolled metal sheet after deep drawing. Left: experiment, Right: simulation (Cf. Singh et al. [28])

1.2. Approach and Objectives

A material model capable of modeling anisotropic effects was derived for rate-independent material behavior in terms of an additive approach of finite plasticity as proposed by Green & Naghdi [7]. This approach was chosen to circumvent the problems which occur within a multiplicative approach by introducing an intermediate state. Firstly the necessary basics of continuum mechanics and finite element method are outlined. Proceeding from an additive decomposition of the strain tensor into an elastic and a plastic part, local constitutive equations were deduced within the logarithmic Lagrangian strain space. By defining the strains in a logarithmic strain space the well known relationships within the small strain theory can be applied. Anisotropic effects were taken into account by implementing orthotropic material symmetry groups in terms of a constant fourth-order Hill tensor. The derived constitutive equations were implemented in an object oriented finite element program called ”soofeaM” (software for object-oriented finite element analysis in Matlab) provided by the Institute of Strength of Materials at Graz University of Technology and was used to calculate simple numerical benchmark examples.

2. Continuum Mechanics

This chapter describes the basic principles of continuum mechanics. In continuum mechanics materials are modeled as continuous bodies which conform to fundamental conservation laws like the conservation of mass or momentum. The basic derivations below can be found in diverse literature and follow the quotations found in Bonet & Wood [3] for instance.

2.1. Motion

In continuum mechanics a body is represented as an assemblage of material points whose positions in the initial state $t = 0$ can be described with a vector $\mathbf{X}$ with respect to a global coordinate system. The configuration of the undeformed body is called Lagrangian or material configuration. By applying body and surface loads on the body it deforms accordingly at time $t \neq 0$ and the material points move to new positions. This can be described by the vector $\mathbf{x}$ with respect to a global Cartesian basis. The configuration of the deformed body at time t is named Eulerian or current configuration. The relationship between both configurations is illustrated in Figure 2.1.

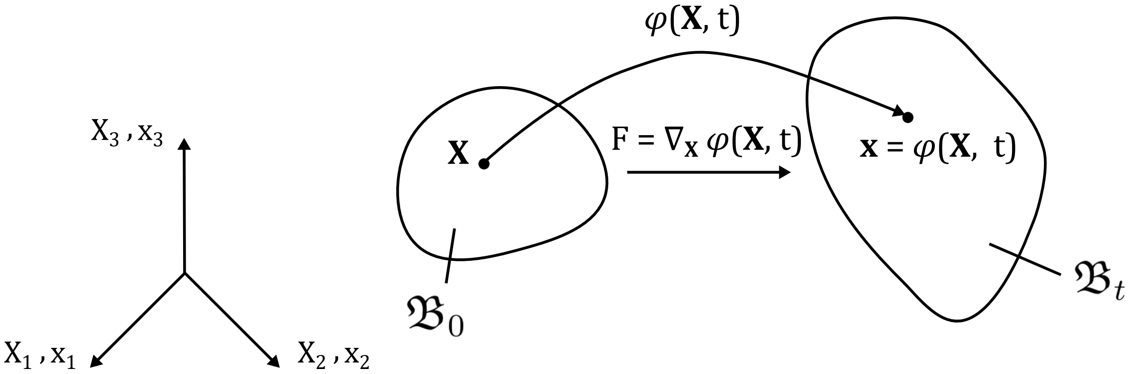


Figure 2.1. Deformation gradient and nonlinear mapping φ between original and current configuration

In Figure 2.1 $\mathfrak{B}_0$ describes the Lagrangian configuration and is bounded by $\partial\mathfrak{B}_0$. The Eulerian configuration is noted as $\mathfrak{B}_t$ which is bounded by $\partial\mathfrak{B}_t$. The nonlinear mapping φ maps the material point $\mathbf{X} \in \mathfrak{B}_0$ at time $t = 0$ onto the position $\mathbf{x} \in \mathfrak{B}_t$ at time t. This relation is shown in Equation 2.1.

$$\mathbf{x} = \varphi(\mathbf{X}, t) \tag{2.1}$$

2.2. Deformation

2.2.1. Deformation Gradient

The local deformation gradient $\mathbf{F}$ is defined according to Equation 2.3 and describes a mapping of the tangent vectors of the Lagrangian configuration onto the tangent vectors of the Eulerian configuration. This context is described by Equation 2.2. The deformation gradient $\mathbf{F}$ plays an important part in the definition of strain measures as it reveals as the central description of kinematics. It can be used to describe the relative position of two neighboring material points before and after deformation. Furthermore the deformation gradient $\mathbf{F}$ is specified as a two-point tensor which exposes the aforementioned behavior.

$$d\mathbf{x} = \mathbf{F} d\mathbf{X} \tag{2.2}$$

$$\mathbf{F} = \nabla_{\mathbf{X}} \varphi(\mathbf{X}, t) \tag{2.3}$$

2.2.2. Strain

Deformation can be measured by using a strain measure. In finite strain theory the strain is defined as the change of the scalar product of two infinitesimal vectors $\mathrm{d}\mathbf{X}_1$ and $\mathrm{d}\mathbf{X}_2$. This context is shown in Equation 2.4

$$\mathrm{d}\mathbf{x}_1 \cdot \mathrm{d}\mathbf{x}_2 = \mathrm{d}\mathbf{X}_1 \cdot \mathbf{C}\mathrm{d}\mathbf{X}_2 \tag{2.4}$$

where $\mathbf{C}$ denotes the right Cauchy-Green deformation tensor. The right Cauchy-Green deformation tensor $\mathbf{C}$ operates on the material configuration and is defined in terms of the deformation gradient $\mathbf{F}$ as noted in Equation 2.5.

$$\mathbf{C} = \mathbf{F}^T \mathbf{F} \tag{2.5}$$

The change in the scalar product of two infinitesimal vectors can now be found in terms of the Green Lagrange strain tensor $\mathbf{E}$ which by itself is defined by the right Cauchy-Green tensor $\mathbf{C}$ (see Equation 2.6 and Equation 2.7). In Equation 2.7 the tensor $\mathbf{I}$ denotes the unit tensor.

$$\frac{1}{2}(\mathrm{d}\mathbf{x}_1 \cdot \mathrm{d}\mathbf{x}_2 - \mathrm{d}\mathbf{X}_1 \cdot \mathrm{d}\mathbf{X}_2) = \mathrm{d}\mathbf{X}_1 \cdot \mathbf{E}\mathrm{d}\mathbf{X}_2 \tag{2.6}$$

$$\mathbf{E} = \frac{1}{2}(\mathbf{C} - \mathbf{I}) \tag{2.7}$$

2.2.3. Polar Decomposition

The deformation gradient $\mathbf{F}$ can be split into stretch and rotation components which is called decomposition. Two procedures can be applied to decompose the deformation gradient $\mathbf{F}$. In the first procedure the deformation gradient $\mathbf{F}$ is made up of the rotation tensor $\mathbf{R}$ and the stretch tensor $\mathbf{U}$:

$$\mathbf{F} = \mathbf{R}\mathbf{U} \tag{2.8}$$

This procedure can be interpreted as a stretch in the Lagrangian configuration and a following rotation into the Eulerian configuration. The right Cauchy-Green tensor $\mathbf{C}$ can be rewritten by applying Equation 2.8:

$$\mathbf{C} = \mathbf{F}^T\mathbf{F} = \mathbf{U}^T\mathbf{R}^T\mathbf{R}\mathbf{U} \tag{2.9}$$

The rotational tensor $\mathbf{R}$ can be assembled as an orthogonal tensor, therefore $\mathbf{R}^T\mathbf{R} = \mathbf{I}$. If $\mathbf{U}$ is chosen to be a symmetric tensor the right Cauchy-Green tensor $\mathbf{C}$ can be rewritten as in Equation 2.10.

$$\mathbf{C} = \mathbf{U}\mathbf{U} = \mathbf{U}^2 \tag{2.10}$$

In order to determine the stretch tensor $\mathbf{U}$, the principle directions of the right Cauchy-Green tensor $\mathbf{C}$ have to be evaluated. Those are denoted by the eigenvectors $\mathbf{N}_\alpha$ with their corresponding eigenvalues λ_α^2 for $\alpha = \{1, 2, 3\}$. The spectral decomposition of $\mathbf{C}$ and the stretch tensor $\mathbf{U}$ can then be written as:

$$\mathbf{C} = \sum_{\alpha=1}^{3} \lambda_\alpha^2 \mathbf{N}_\alpha \otimes \mathbf{N}_\alpha \tag{2.11}$$

$$\mathbf{U} = \sum_{\alpha=1}^{3} \lambda_\alpha \mathbf{N}_\alpha \otimes \mathbf{N}_\alpha \tag{2.12}$$

In the second procedure the deformation gradient $\mathbf{F}$ is made up of the stretch tensor $\mathbf{V}$ and rotation tensor $\mathbf{R}$:

$$\mathbf{F} = \mathbf{V}\mathbf{R} \tag{2.13}$$

This procedure can be interpreted as a rotation into the Eulerian configuration and then a stretch. The rotation tensor $\mathbf{R}$ can be calculated as $\mathbf{R} = \mathbf{F}\mathbf{U}^{-1}$. The stretch tensor $\mathbf{V}$ can then be obtained by combining Equation 2.8 and Equation 2.13:

$$\mathbf{V} = \mathbf{R}\mathbf{U}\mathbf{R}^T \tag{2.14}$$

The described methods and the interpretation of those are illustrated in Figure 2.2. The deformation gradient defines the direct linkage between the non-deformed and deformed state of the body under consideration.

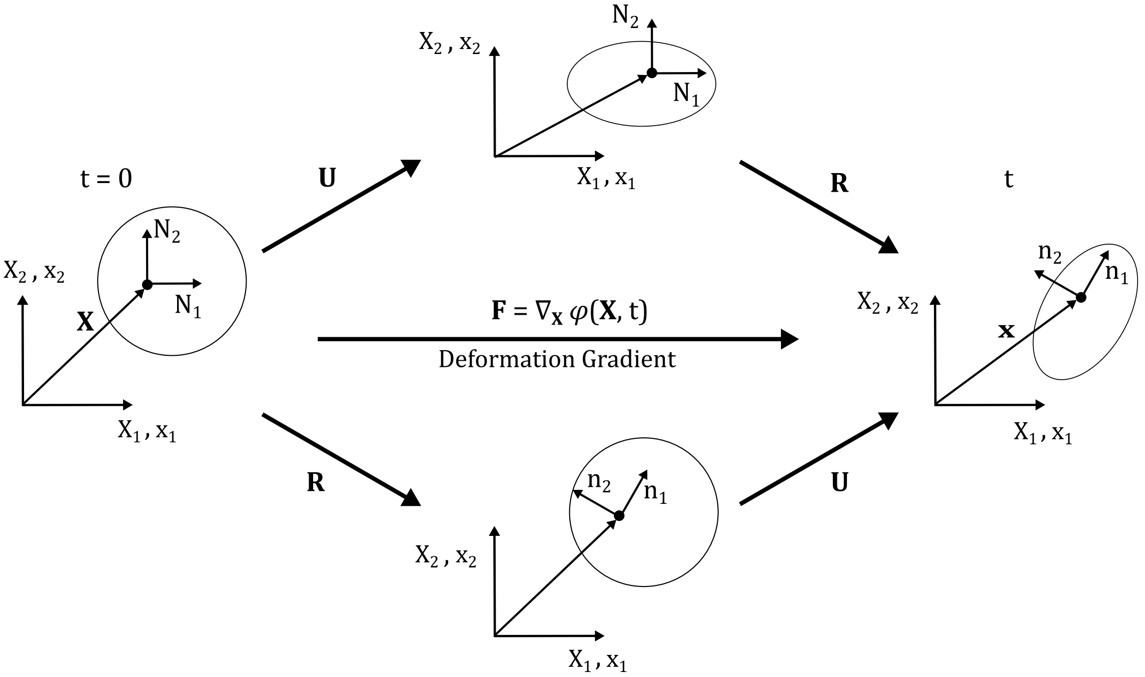


Figure 2.2. Polar decomposition of the deformation gradient

2.3. Stress

A deformation of a body generates internal forces which can be expressed in terms of a stress measurement. Considering a general deformable body which is cut into two halves the surface traction vector $\mathbf{t}$ can be introduced which acts on a cut area da. This context is visualized in Figure 2.3.

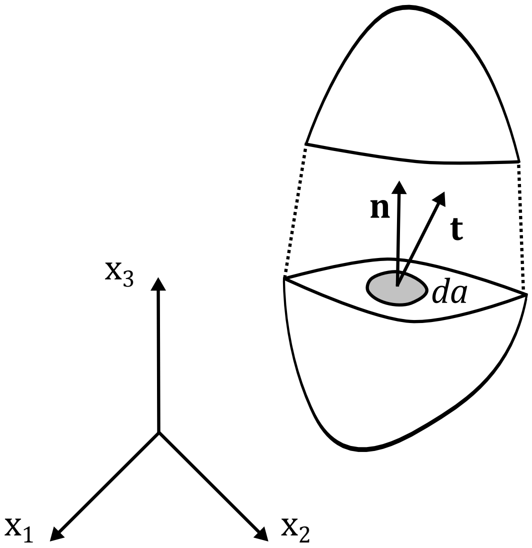


Figure 2.3. Traction vector $\mathbf{t}$ acting on the cut area da

The surface traction vector $\mathbf{t}$ can be expressed in terms of a limiting value for a material point $\mathbf{x}$ located in a cut area Δa which is loaded by the force vector $\Delta\mathbf{f}$:

$$\mathbf{t}(\mathbf{x}, t) = \lim_{\Delta a \to 0} \frac{\Delta \mathbf{f}}{\Delta a} = \frac{d\mathbf{f}}{da} \tag{2.15}$$

Consider an elemental tetrahedron shown in Figure 2.4 which is loaded by a body force $\mathbf{f}$ and is kept in mechanical equilibrium by the surface traction vectors $\mathbf{t}_i$ for $i = \{1, 2, 3\}$ and $\mathbf{t}_n$. These vectors can be expressed in their Cartesian parts by introducing the stress components σ_{ij} which are defined in the Cartesian planes da_i:

$$\mathbf{t}_i = \sigma_{1i}\mathbf{e}_1 + \sigma_{2i}\mathbf{e}_2 + \sigma_{3i}\mathbf{e}_3 \tag{2.16}$$

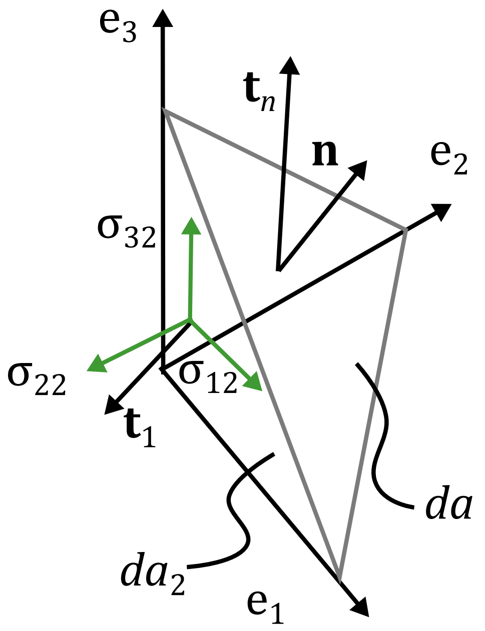


Figure 2.4. Elemental tetrahedron, exemplary only $\mathbf{t}_1$ is shown

The equilibrium on the tetrahedron leads to the following equation:

$$\mathbf{t}_n \, da + \sum_{i=1}^{3} \mathbf{t}_i \, da_i + \mathbf{f} dv = \mathbf{0} \tag{2.17}$$

Equation 2.17 can be simplified by dividing by da and taking into account that the plane areas da_i can be expressed as a projection of da as

$$da_i = \mathbf{n} \cdot \mathbf{e}_i \, da. \tag{2.18}$$

In the case of $dv \to 0$ Equation 2.17 can be rewritten as:

$$\mathbf{t}_n = \sum_{i=1}^{3} \mathbf{t}_i (\mathbf{n} \cdot \mathbf{e}_i) \tag{2.19}$$

$$\mathbf{t}_n = \sum_{i,j=1}^{3} \sigma_{ji} \mathbf{e}_j (\mathbf{n} \cdot \mathbf{e}_i) \tag{2.20}$$

$$\mathbf{t}_n = \sum_{i,j=1}^{3} \sigma_{ji}\mathbf{e}_j \otimes \mathbf{e}_i\mathbf{n} \tag{2.21}$$

$$\mathbf{t}_n = \boldsymbol{\sigma}\mathbf{n} \tag{2.22}$$

The Cauchy stress tensor $\boldsymbol{\sigma}$ can be found in Equation 2.22. It is a symmetric tensor due to the duality of shear stresses and shows an objective behavior which means that the Cauchy stress tensor remains unaltered by rigid body motions.

2.4. Principle of Virtual Work

2.4.1. Equilibrium Equations

In Figure 2.5 a general deformable body is shown which is defined by its volume v with the boundary ∂v and is loaded by a body force $\mathbf{f}$ per unit volume and a traction force $\mathbf{t}$ per unit area. In the case of a static problem the sum of all forces acting on the deformable body has to vanish. This yields Equation 2.23.

$$\int_{\partial v} \mathbf{t} da + \int_v \mathbf{f} dv = 0 \tag{2.23}$$

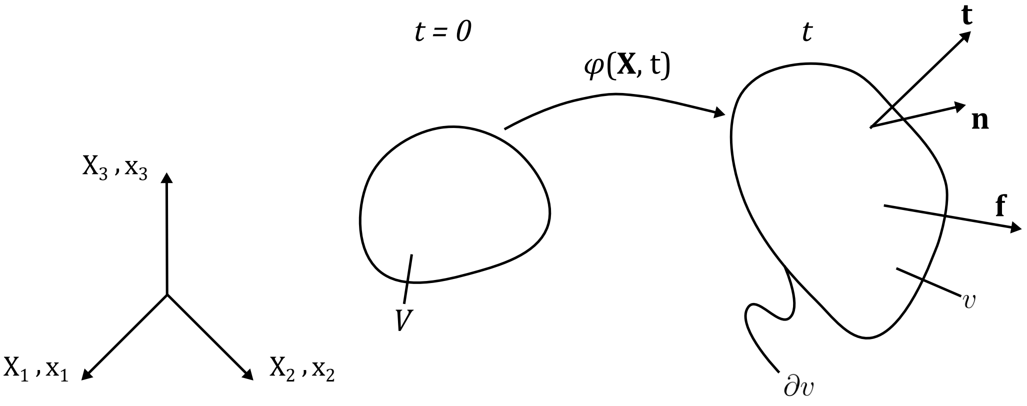


Figure 2.5. Equilibrium of forces for a general deformable body

Using the Cauchy theorem for the traction vector $\mathbf{t}$ from Equation 2.22 leads to an expression in terms of the Cauchy stresses:

$$\int_{\partial v} \boldsymbol{\sigma}\mathbf{n} da + \int_v \mathbf{f} dv = 0 \tag{2.24}$$

The area integral in Equation 2.24 can be transformed into a volume integral by the use of the Gauss theorem (see Equation 2.25):

$$\int_v \mathrm{div}\mathbf{S} dv = \int_{\partial v} \mathbf{S}\mathbf{n} da \tag{2.25}$$

$$\int_v (\mathrm{div}\boldsymbol{\sigma} + \mathbf{f}) dv = 0 \tag{2.26}$$

Equation 2.26 shows the integral formula of the local equilibrium equation in the spatial configuration and has to be applicable for any enclosed region dv which leads to

$$\mathrm{div}\boldsymbol{\sigma} + \mathbf{f} = 0. \tag{2.27}$$

By introducing a virtual velocity $\delta\mathbf{v}$ the virtual work δW can be derived as shown in Equation 2.28.

$$\delta W = \int_v (\mathrm{div}\boldsymbol{\sigma} + \mathbf{f}) \cdot \delta\mathbf{v} dv = 0 \tag{2.28}$$

Equation 2.28 can be rewritten by the use of the divergence theorem (see Equation 2.29) and Gauss theorem (see Equation 2.30). Equation 2.31 shows that the virtual work δW can be divided into an internal part δW_{int} which describes the virtual work due to the internal forces and an external part δW_{ext} due to the external loads.

$$\mathrm{div}(\boldsymbol{\sigma}\delta\mathbf{v}) = (\mathrm{div}\boldsymbol{\sigma}) \cdot \delta\mathbf{v} + \boldsymbol{\sigma} : grad\delta\mathbf{v} \tag{2.29}$$

$$\int_v \mathrm{div}\boldsymbol{\sigma} \cdot \delta\mathbf{v} dv = \int_{\partial v} \boldsymbol{\sigma}\mathbf{n} \cdot \delta\mathbf{v} da = \int_{\partial v} \mathbf{t} \cdot \delta\mathbf{v} da \tag{2.30}$$

$$\underbrace{-\int_v \boldsymbol{\sigma} : grad\delta\mathbf{v} dv}_{\delta W_{int}} + \underbrace{\int_v \mathbf{f} \cdot \delta\mathbf{v} dv + \int_{\partial v} \mathbf{t} \cdot \delta\mathbf{v} da}_{\delta W_{ext}} = 0 \tag{2.31}$$

The gradient of the virtual velocity $\delta\mathbf{v}$ is defined as the virtual velocity gradient $\delta\mathbf{l}$. Introducing this relation into Equation 2.31 yields:

$$\underbrace{-\int_v \boldsymbol{\sigma} : \delta\mathbf{l} dv}_{\delta W_{int}} + \underbrace{\int_v \mathbf{f} \cdot \delta\mathbf{v} dv + \int_{\partial v} \mathbf{t} \cdot \delta\mathbf{v} da}_{\delta W_{ext}} = 0 \tag{2.32}$$

The virtual velocity gradient can be divided into the symmetric virtual rate of deformation $\delta\mathbf{d}$ and the antisymmetric virtual spin tensor $\delta\mathbf{w}$. Due to the symmetry of $\boldsymbol{\sigma}$ Equation 2.32 can be rewritten in terms of the symmetric virtual rate of deformation:

$$\underbrace{-\int_v \boldsymbol{\sigma} : \delta\mathbf{d} dv}_{\delta W_{int}} + \underbrace{\int_v \mathbf{f} \cdot \delta\mathbf{v} dv + \int_{\partial v} \mathbf{t} \cdot \delta\mathbf{v} da}_{\delta W_{ext}} = 0 \tag{2.33}$$

The stress can be expressed in other definitions than the Cauchy stress tensor $\boldsymbol{\sigma}$. In the Lagrangian configuration the second Piola Kirchhoff stress tensor $\mathbf{S}$ is conveniently used and should therefore be derived.

The velocity gradient tensor $\mathbf{l}$ is defined in terms of the deformation gradient $\mathbf{F}$ as:

$$\mathbf{l} = \dot{\mathbf{F}}\mathbf{F}^{-1} \tag{2.34}$$

Further the material strain rate tensor $\dot{\mathbf{E}}$ is given as:

$$\dot{\mathbf{E}} = \frac{1}{2}\dot{\mathbf{C}} = \frac{1}{2}(\dot{\mathbf{F}}^T\mathbf{F} + \mathbf{F}^T\dot{\mathbf{F}}) \tag{2.35}$$

The time derivative of Equation 2.4 becomes with the above stated equations and the relations $\mathrm{d}\mathbf{X}_i = \mathbf{F}^{-1}\mathrm{d}\mathbf{x}_i$:

$$\frac{d}{dt}(\mathrm{d}\mathbf{x}_1 \cdot \mathrm{d}\mathbf{x}_2) = \mathrm{d}\mathbf{X}_1 \cdot \dot{\mathbf{C}}\mathrm{d}\mathbf{X}_2 = 2\mathrm{d}\mathbf{X}_1 \cdot \dot{\mathbf{E}}\mathrm{d}\mathbf{X}_2 = \mathrm{d}\mathbf{x}_1 \cdot (\mathbf{F}^{-T}\dot{\mathbf{E}}\mathbf{F}^{-1})\mathrm{d}\mathbf{x}_2 \tag{2.36}$$

In Equation 2.36 the rate of deformation tensor can be found as:

$$\mathbf{d} = \mathbf{F}^{-T}\dot{\mathbf{E}}\mathbf{F}^{-1} \tag{2.37}$$

The virtual rate of deformation $\delta\mathbf{d}$ can now be expressed in the form of:

$$\delta\mathbf{d} = \mathbf{F}^{-T}\delta\dot{\mathbf{E}}\mathbf{F}^{-1} \tag{2.38}$$

Introducing Equation 2.38 into Equation 2.31 results in an alternative expression of the internal virtual work δW_{int} in terms of the second Piola Kirchhoff stress tensor $\mathbf{S}$ as shown in Equation 2.42.

$$\delta W_{int} = \int_v \boldsymbol{\sigma} : \delta\mathbf{d}dv \tag{2.39}$$

$$\delta W_{int} = \int_V J\boldsymbol{\sigma} : \mathbf{F}^{-T}\delta\dot{\mathbf{E}}\mathbf{F}^{-1}dV \tag{2.40}$$

$$\delta W_{int} = \int_V \underbrace{J\mathbf{F}^{-1}\boldsymbol{\sigma}\mathbf{F}^{-T}}_{\mathbf{S}} : \delta\dot{\mathbf{E}}dV \tag{2.41}$$

$$\delta W_{int} = \int_V \mathbf{S} : \delta\dot{\mathbf{E}}dV \tag{2.42}$$

$$\mathbf{S} = J\mathbf{F}^{-1}\boldsymbol{\sigma}\mathbf{F}^{-T} \tag{2.43}$$

The different stress measurements can be interpreted in different ways. The Cauchy stresses $\boldsymbol{\sigma}$ can be interpreted as the current force per unit of deformed area and can therefore also be referred to as true stresses. The second Piola Kirchhoff stress tensor $\mathbf{S}$ can be interpreted as a material force per unit of undeformed area. The material force can be read as a push back of the spatial force. The aforementioned relations are illustrated in Figure 2.6.

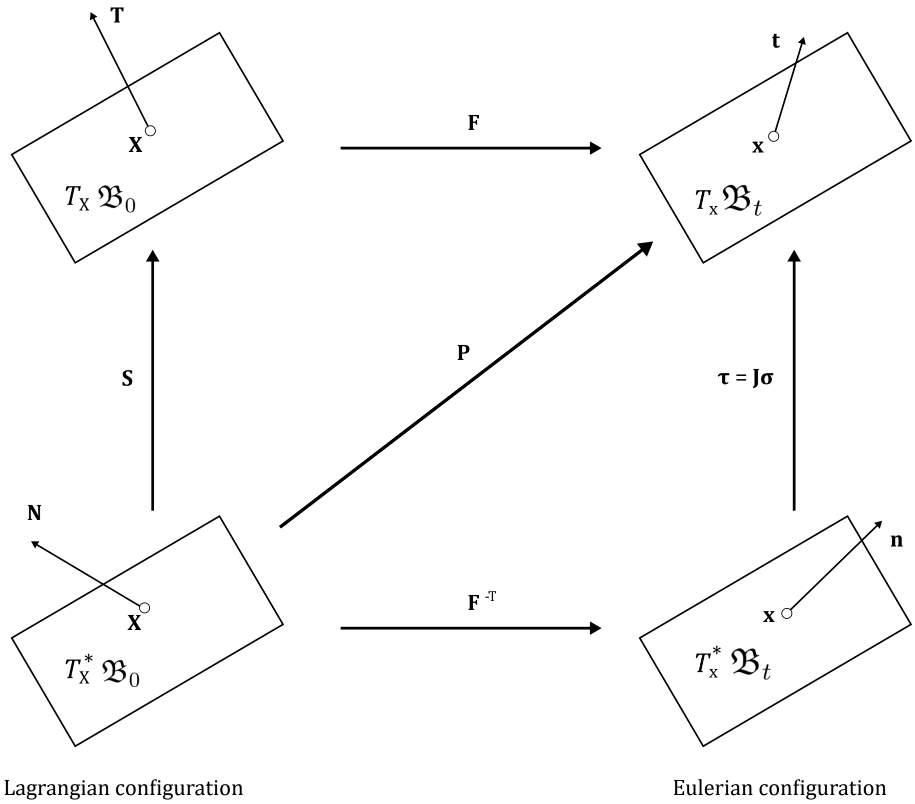


Figure 2.6. Different stress measurements in Lagrangian and Eulerian configuration

2.4.2. Linearisation of the Equilibrium Equations

The equilibrium of a deformable body is described with Equation 2.31 and has to be discretized to perform a linearization. The discretization of the equilibrium is denoted in Equation 2.44. The external load is predetermined which means that the virtual work of external forces δW_{ext} is already known and no linearisation of this term has to be performed. In Equation 2.45 $D\delta W_{int}[\mathbf{u}]$ delineates the directional derivative of δW_{int} in the direction of the displacement vector $\mathbf{u}$.

$$\delta W_{int}^{n+1} = \delta W_{ext}^{n+1} \tag{2.44}$$

$$D\delta W_{int}^{n}[\mathbf{u}] = \delta W_{ext}^{n+1} - \delta W_{int}^{n} \tag{2.45}$$

Taking Equation 2.42 into account and performing the directional derivative leads to Equation 2.46 which can also be expressed in terms of the index notation as shown in Equation 2.47.

$$D\delta W_{int}^{n}[\mathbf{u}] = \int_V (D\mathbf{S}[\mathbf{u}] : \delta\dot{\mathbf{E}} + \mathbf{S} : D\delta\dot{\mathbf{E}}[\mathbf{u}])dV \tag{2.46}$$

$$D\delta W_{int}^{n}[\mathbf{u}] = \int_V (DS_{ij}[\mathbf{u}]\delta\dot{E}_{ij} + S_{ij}D\delta\dot{E}_{ij}[\mathbf{u}])dV \tag{2.47}$$

As shown in Equation 2.47 the derivative of the Green Lagrange Strain tensor $\dot{\mathbf{E}}$ must be rewritten in alternative expressions in terms of its directional derivative $D...[\mathbf{u}]$ (see Equation 2.49) and its variation $\delta[...]$ (see Equation 2.48).

$$\delta\dot{E}_{ij} = \frac{1}{2}\delta\dot{C}_{ij} = \frac{1}{2}(\delta\dot{F}_{mi}F_{mj} + F_{ni}\delta\dot{F}_{nj}) \tag{2.48}$$

$$D\dot{E}_{kl}[\mathbf{u}] = \frac{1}{2}(D\dot{F}_{rk}[\mathbf{u}]F_{rl} + F_{sk}D\dot{F}_{sl}[\mathbf{u}]) \tag{2.49}$$

The directional derivative of the variation of the Green Lagrange Strain tensor $\dot{\mathbf{E}}$ can be evaluated by using the two equations above which leads to Equation 2.50.

$$D\delta\dot{E}_{ij}[\mathbf{u}] = \frac{1}{2}(D\delta\dot{F}_{mi}[\mathbf{u}]F_{mj} + \delta\dot{F}_{mi}DF_{mj}[\mathbf{u}] + DF_{ni}[\mathbf{u}]\delta\dot{F}_{nj} + F_{ni}D\delta\dot{F}_{nj}[\mathbf{u}]) \tag{2.50}$$

This equation simplifies by implying that the virtual quantities $\delta[...]$ remain constant in an incremental displacement $\mathbf{u}$. Those terms vanish by applying a linearisation which leads to Equation 2.51.

$$D\delta\dot{E}_{ij}[\mathbf{u}] = \frac{1}{2}(\delta\dot{F}_{mi}DF_{mj}[\mathbf{u}] + DF_{li}[\mathbf{u}]\delta\dot{F}_{lj}) \tag{2.51}$$

A linearisation of the second Piola Kirchhoff stress tensor $\mathbf{S}$ gives Equation 2.52 by applying the chain rule. This causality can be explained by the fact that the second Piola Kirchhoff stress tensor $\mathbf{S}$ can be expressed as a function of the Green Lagrange Strain tensor $\mathbf{E}$. In this equation $\mathbb{C}$ represents the material/Lagrangian tangent moduli or Lagrangian elasticity tensor which can be calculated by the derivative of $\mathbf{S}$ with respect to $\mathbf{E}$.

$$D\delta S_{ij}[\mathbf{u}] = \frac{\partial S_{ij}}{\partial E_{kl}}DE_{kl}[\mathbf{u}] = \mathbb{C}_{ijkl}DE_{kl}[\mathbf{u}] \tag{2.52}$$

Inserting Equation 2.52 into Equation 2.47 leads to:

$$D\delta W^n_{int}[\mathbf{u}] = \underbrace{\int_V \mathbb{C}_{ijkl}DE_{kl}[\mathbf{u}]\delta\dot{E}_{ij}dV}_{*} + \underbrace{\int_V S_{ij}D\delta\dot{E}_{ij}[\mathbf{u}]dV}_{**}\,. \tag{2.53}$$

The first term of Equation 2.53 (*) can be further simplified by using Equation 2.48 and Equation 2.49 and the fact that the Lagrangian elasticity tensor $\mathbb{C}$ has minor symmetries which implies $\mathbb{C}_{ijkl} = \mathbb{C}_{jikl} = \mathbb{C}_{ijlk}$. This leads to Equation 2.56.

$$(*) \qquad \int_V \mathbb{C}_{ijkl}\underbrace{\frac{1}{2}(DF_{rk}[\mathbf{u}]F_{rl} + F_{sk}DF_{sl}[\mathbf{u}])}_{DE_{kl}[\mathbf{u}]}\underbrace{\frac{1}{2}(\delta\dot{F}_{mi}F_{mj} + F_{li}\delta\dot{F}_{lj})}_{\delta\dot{E}_{ij}[\mathbf{u}]}dV \tag{2.54}$$

$$\begin{aligned}(*) \qquad \int_V \frac{1}{4}\mathbb{C}_{ijkl}(&DF_{rk}[\mathbf{u}]F_{rl}\delta\dot{F}_{mi}F_{mj} + DF_{rk}[\mathbf{u}]F_{rl}F_{li}\delta\dot{F}_{lj} + F_{sk}DF_{sl}[\mathbf{u}]\delta\dot{F}_{mi}F_{mj} \\ &+ F_{sk}DF_{sl}[\mathbf{u}]F_{li}\delta\dot{F}_{lj})dV\end{aligned} \tag{2.55}$$

$$(*) \qquad \int_V \mathbb{C}_{ijkl} DF_{rk}[\mathbf{u}] F_{rl} \delta \dot{F}_{mi} F_{mj} dV \tag{2.56}$$

The second term (**) of Equation 2.53 can be rewritten by using Equation 2.51 and the symmetry of the second Piola Kirchhoff stress tensor **S** which results in Equation 2.59.

$$(**) \qquad \int_V S_{ij} D\delta \dot{E}_{ij}[\mathbf{u}] dV \tag{2.57}$$

$$(**) \qquad \int_V \frac{1}{2} S_{ij} (\delta \dot{F}_{mi} DF_{mj}[\mathbf{u}] + DF_{li}[\mathbf{u}] \delta \dot{F}_{lj}) dV \tag{2.58}$$

$$(**) \qquad \int_V S_{ij} \delta \dot{F}_{mi} DF_{mj}[\mathbf{u}] dV \tag{2.59}$$

Equation 2.47 can now be rewritten by using the relations from above:

$$\underbrace{\int_V \mathbb{C}_{ijkl} DF_{rk}[\mathbf{u}] F_{rl} \delta \dot{F}_{mi} F_{mj} dV}_{Lagrangian\ tangent} + \underbrace{\int_V S_{ij} \delta \dot{F}_{mi} DF_{mj}[\mathbf{u}] dV}_{geometrical\ tangent} = \delta W_{ext}^{n+1} - \delta W_{int}^{n} \tag{2.60}$$

3. Basics of Finite Element Method

The finite element method is a widely used numerical calculation technique which is particulary deployed in mechanical calculations. In the literature a wide variety of contributions can be obtained. The basic descriptions below can be found in Bathe [1] for instance.

3.1. Element Formulation

In the finite element analysis the body under consideration is discretized by a number of finite elements. By doing this the problem will be idealized through an assemblage of these finite elements which consist of a number of nodes connecting those elements on the element boundaries. By solving the equilibrium equations from chapter 2 the displacements of the according nodes can be calculated. This can then be used to obtain the stresses within a finite element.
In an isoparametric approach the coordinates x, y and z and the displacements u, v and w of the element can be described by interpolations in the natural coordinates of the element by using interpolation functions. The natural coordinate system of the element consists of the variables r, s and t which vary from -1 to +1. The interpolation functions h_i of the element node i must be unity at the node i and vanish at all other nodes of the element. An illustration of the principle of natural coordinates can be found in Figure 3.1.

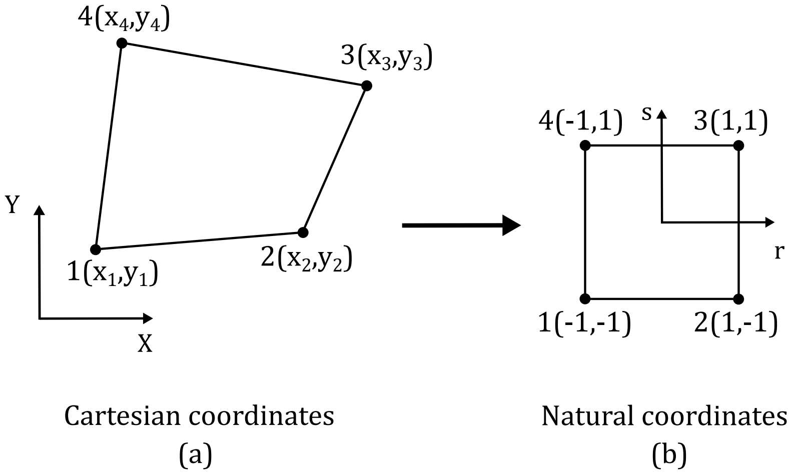


Figure 3.1. Principle illustration of (a) Cartesian and (b) natural coordinates

Interpolation functions can be found by using Lagrange polynomials (see Equation 3.2) for the different degrees of freedom. By introducing these polynomials the shape functions can be rewritten as shown in Equation 3.1.

$$h(r,s,t) = l_i^n(r) l_j^o(s) l_k^p(t) \tag{3.1}$$

$$l_i^n(r) = \prod_{\substack{k=0 \\ k \neq i}}^{n} \frac{x - x_k}{x_i - x_k} \tag{3.2}$$

By using the interpolation functions h_i the element coordinates x, y and z and element displacements u, v and w can be expressed in terms of the corresponding nodal values which is stated in Equation 3.3 and Equation 3.4.

$$x = \sum_{i=1}^{n} h_i(r,s,t)\hat{x}_i \qquad y = \sum_{i=1}^{n} h_i(r,s,t)\hat{y}_i \qquad z = \sum_{i=1}^{n} h_i(r,s,t)\hat{z}_i \tag{3.3}$$

$$u = \sum_{i=1}^{n} h_i(r,s,t)\hat{u}_i \qquad v = \sum_{i=1}^{n} h_i(r,s,t)\hat{v}_i \qquad w = \sum_{i=1}^{n} h_i(r,s,t)\hat{w}_i \tag{3.4}$$

By using the above stated relations the element displacement $\mathbf{u}$ can be expressed as a function of the nodal displacements $\hat{\mathbf{u}}$. The relation between the nodal and the element displacements itself is given by the displacement interpolation matrix $\mathbf{H}$ which includes the interpolation functions of the different degrees of freedom from Equation 3.3 and Equation 3.4 (see Equation 3.5).

$$\mathbf{u} = \mathbf{H}(r,s,t)\hat{\mathbf{u}} \tag{3.5}$$

The strain-displacement matrix $\mathbf{B}$ can be derived with the definition of the deformation gradient $\mathbf{F}$:

$$\mathbf{F} = \frac{\partial \mathbf{x}}{\partial \mathbf{X}} = \frac{\partial (\mathbf{X} + \mathbf{u})}{\partial \mathbf{X}} = 1 + \frac{\partial \mathbf{u}}{\partial \mathbf{X}} = 1 + \frac{\partial \mathbf{r}}{\partial \mathbf{X}} \frac{\partial \mathbf{u}}{\partial \mathbf{r}} = 1 + \underbrace{\mathbf{J}^{-1} \frac{\partial \mathbf{H}(r,s,t)}{\partial r}}_{\mathbf{B}} \hat{\mathbf{u}} \tag{3.6}$$

The deformation gradient $\mathbf{F}$ can be expressed by the use of the strain-displacement matrix $\mathbf{B}$. The virtual variation and the directional derivative of the deformation gradient is shown in Equation 3.7 and Equation 3.8.

$$\delta F_{ij} = B_{im} \delta \hat{u}_{mj} \qquad \delta \mathbf{F} = \mathbf{B} \delta \hat{\mathbf{u}} \tag{3.7}$$

$$DF_{ij}[\mathbf{u}] = B_{im} \hat{u}_{mj} \qquad D\mathbf{F}[\mathbf{u}] = \mathbf{B}\hat{\mathbf{u}} \tag{3.8}$$

In Equation 3.7 the term $\delta \mathbf{F} = \frac{\partial \delta \mathbf{u}}{\partial \mathbf{X}}$ represents the virtual deformation gradient. The virtual displacement $\delta \mathbf{u}$ is a test function which means that it has to be constant within one increment. An illustration of this relation can be found in Figure 3.2.

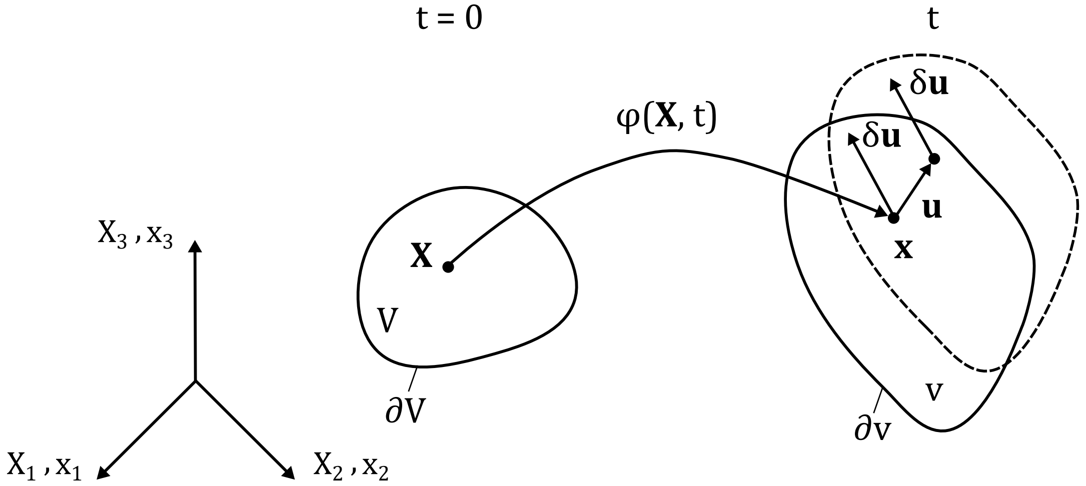


Figure 3.2. Virtual displacement

3.2. Newton Algorithm

The necessary steps of calculation are executed on element level and have to be assembled to global values. On the element level integrations are performed on the natural coordinates where new integration constants have to be taken into account. Equation 3.9 illustrates this relation where $\mathbf{J}$ is the Jacobian matrix of the shape functions, V the volume of the element in the Cartesian coordinates and $\bar{V}$ the volume of the element in natural coordinates.

$$\mathrm{d}V = \det(\mathbf{J})\mathrm{d}\bar{V} \tag{3.9}$$

The internal virtual energy δW_{int} from Equation 2.42 can be rewritten by using the aforementioned relations:

$$\begin{aligned}
\delta W_{int} &= \int_{\bar{V}} S_{ij}\delta E_{ij}\det(\mathbf{J})\mathrm{d}\bar{V} \\
&= \int_{\bar{V}} S_{ij}\frac{1}{2}(\delta F_{mi}F_{mj} + F_{ni}\delta F_{nj})\det(\mathbf{J})\mathrm{d}\bar{V} \\
&= \int_{\bar{V}} \delta F_{mi}S_{ij}F_{mj}\det(\mathbf{J})\mathrm{d}\bar{V} \\
&= \int_{\bar{V}} \delta\hat{u}_{si}B_{ms}S_{ij}F_{mj}\det(\mathbf{J})\mathrm{d}\bar{V}
\end{aligned} \tag{3.10}$$

The external virtual energy is calculated by the product of the external load F_{ext} and the virtual displacement δu as stated in Equation 3.11.

$$\delta W_{ext} = \delta\hat{u}_{ms}F^{ext}_{sm} \tag{3.11}$$

In subsection 2.4.2 the linearization of the equilibrium equations was derived. Equation 2.60 can be rewritten by using Equation 3.7 and Equation 3.8 which leads to Equation 3.13. This equation can be expressed in a compact manner as Equation 3.14.

$$\begin{aligned}&\int_{\bar{V}} \delta\hat{u}_{ni} B_{mn} F_{mj} \mathbb{C}_{ijkl} F_{rl} B_{rq} \hat{u}_{qk} det(\mathbf{J}) d\bar{V} + \int_{\bar{V}} \delta\hat{u}_{li} B_{ml} S_{ij} B_{mn} \hat{u}_{nj} det(\mathbf{J}) d\bar{V} \\ &= \delta\hat{u}_{ms} F^{ext}_{sm} - \int_{\bar{V}} \delta\hat{u}_{si} B_{ms} S_{ij} F_{mj} det(\mathbf{J}) d\bar{V}\end{aligned} \tag{3.12}$$

$$\begin{aligned}&\delta\hat{u}_{ni} \underbrace{\int_{\bar{V}} B_{mn} F_{mj} \mathbb{C}_{ijkl} F_{rl} B_{rq} \hat{u}_{qk} + B_{ml} S_{ij} B_{mn} \hat{u}_{nj} det(\mathbf{J}) d\bar{V}}_{D\mathbf{F}_{int}[\hat{\mathbf{u}}]} \\ &= \delta\hat{u}_{ms} F^{ext}_{sm} - \delta\hat{u}_{si} \underbrace{\int_{\bar{V}} B_{ms} S_{ij} F_{mj} det(\mathbf{J}) d\bar{V}}_{\mathbf{F}_{int}}\end{aligned} \tag{3.13}$$

$$D\mathbf{F}_{int}[\hat{\mathbf{u}}] = \mathbf{F}_{ext} - \mathbf{F}_{int} \tag{3.14}$$

The Newton iteration algorithm can be derived by the discretization of the linearized equilibrium equations. Therefore the notation from Equation 3.15 is used where i stands for the i-th iteration of the Newton algorithm and n for the n-th load prescription.

$$[\star]^i_n \qquad \begin{array}{l} n \ldots n\text{-th load prescription} \\ i \ldots i\text{-th iteration of Newton algorithm} \end{array} \tag{3.15}$$

The discretized equation for the Newton algorithm is shown in Equation 3.16. In a non-linear analysis the external load is often applied in n load prescriptions to receive better convergence. In static analysis a so called pseudo time is introduced and the external load is discretized in terms of this time and increases in general linearly over the whole time step.

$$D\mathbf{F}_{int}[\hat{\mathbf{u}}^{i+1}_{n+1}] = (\mathbf{F}_{ext})_{n+1} - (\mathbf{F}_{int})^i_{n+1} \tag{3.16}$$

The Newton algorithm is used until a converged state defined by a tolerance value or an abort criterium is reached. Convergence can be assessed by calculating the unbalanced energy W_{unb} (see Equation 3.17). The unbalanced energy is the scalar product of the difference of the displacements between two Newton iteration steps and the residual force vector $\mathbf{\Psi}$. The residual force vector is defined as the unbalanced force in a calculation step and is stated in Equation 3.18.

$$(W_{unb})^{i+1}_{n+1} = (\hat{\mathbf{u}}^{i+1}_{n+1} - \hat{\mathbf{u}}^{i}_{n+1}) \cdot \mathbf{\Psi}^{i+1}_{n+1} < tolerance \tag{3.17}$$

$$\mathbf{\Psi}^{i+1}_{n+1} = (\mathbf{F}_{ext})_{n+1} - (\mathbf{F}_{int})^{i+1}_{n+1} \tag{3.18}$$

The principle of the Newton algorithm is illustrated in Figure 3.3.

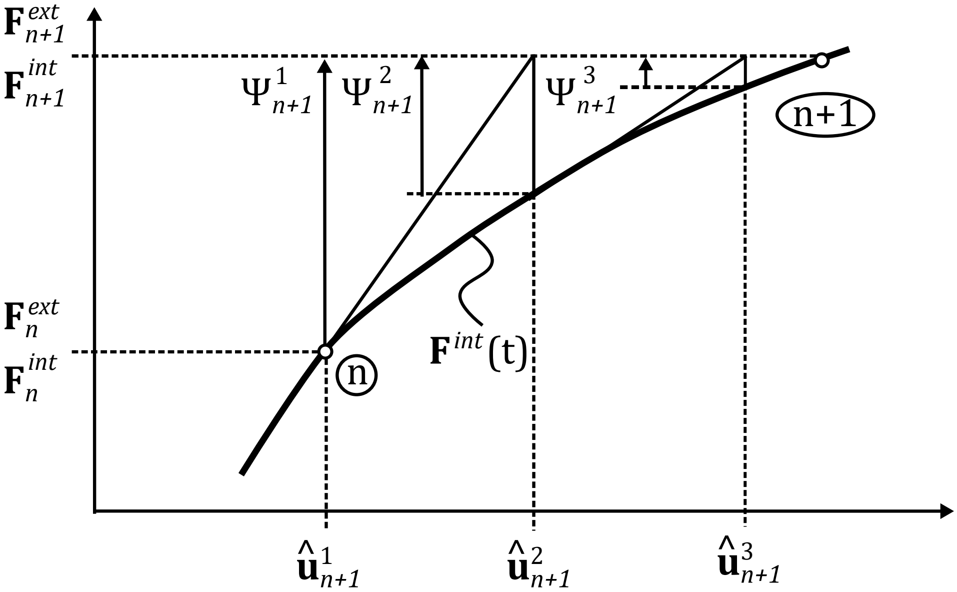


Figure 3.3. Newton Algorithm

4. Anisotropic Plasticity Material Model

4.1. Basics for the established Material Model

This section describes the necessary basics for the derivation of an anisotropic plasticity material model in logarithmic Lagrangian space. Firstly the basics of plasticity and the additive approach of plasticity is presented. Secondly the transformation from the logarithmic Lagrangian space back to the Lagrangian space is derived.

4.1.1. Basics of Isotropic Plasticity

In the following basics of plasticity is described. For further information see for instance Hashiguchi & Yamakawa [8]. The deformation of a deformable body is differentiated in a reversible part which can be described by elastic material constitutive equations (for example a hyperelastic material law) and an irreversible part which can be described by plastic material constitutive equations. The irreversible part of the deformation remains after unloading the body whereas the reversible part vanishes. An applied force typically causes an elastic material response which transitions to an elastic-plastic material response by increasing the applied load to a certain point which is called yielding. Due to hardening the yielding stress increases for a plastic deformation when the load escalates. A yield criterion has to be defined to distinguish between an elastic and a plastic deformation. In a multiaxial stress state an equivalent stress model has to be set up. A well known model is the von Mises yield criterion (see Mises [21]) which is shown in Equation 4.1. The equivalent stress σ_v is calculated by the use of the principal stresses. Comparing the equivalent stress with a yield stress leads to a yield criterion stated in Equation 4.2, where σ_v describes the equivalent stress and σ_y the initial yield stress. The von Mises yield criterion can be illustrated with a yield surface as shown in Figure 4.1.

$$\sigma_v = \sqrt{\frac{1}{2}[(\sigma_{11} - \sigma_{22})^2 + (\sigma_{22} - \sigma_{33})^2 + (\sigma_{33} - \sigma_{11})^2]} \tag{4.1}$$

$$\sigma_v^2 - \sigma_y^2 = 0 \tag{4.2}$$

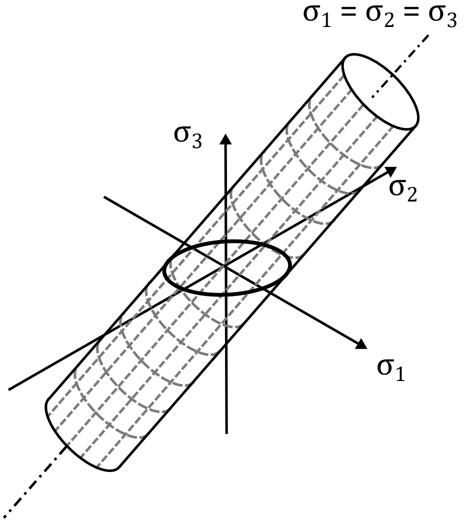


Figure 4.1. Von Mises yield surface (Cf. Rösler et al. [24])

The yield criterion determines whether a deformation is purely elastic or elastic-plastic. Additionally to the yield criterion a set of constitutive equations is needed to describe the plastic deformation. When a deformable body reaches plastic behavior no distinct relation between strain and stress can be obtained. The increase of the plastic strain can be formulated in an incremental formulation with the use of Drucker's postulate which states that the deformation energy must not obtain negative values. Moreover the dissipated energy for a plastic deformation which can be written as $W^{pl} = \sigma_{ij}\varepsilon_{ij}^{pl}$ gains a maximal value. This is attained by maximizing the projection of the stress σ_{ij} onto the plastic strain ε_{ij}^{pl} which is illustrated in Figure 4.2. In the figure $\boldsymbol{\sigma}_1$ has a smaller dissipation energy than $\boldsymbol{\sigma}_2$. The postulated constraints lead to a basic formulation for plastic behavior as shown in Equation 4.3 where $d\lambda$ describes the plastic consistency parameter and F the yield surface.

$$d\varepsilon_{ij}^{pl} = d\lambda \frac{\partial F}{\partial \sigma_{ij}} \tag{4.3}$$

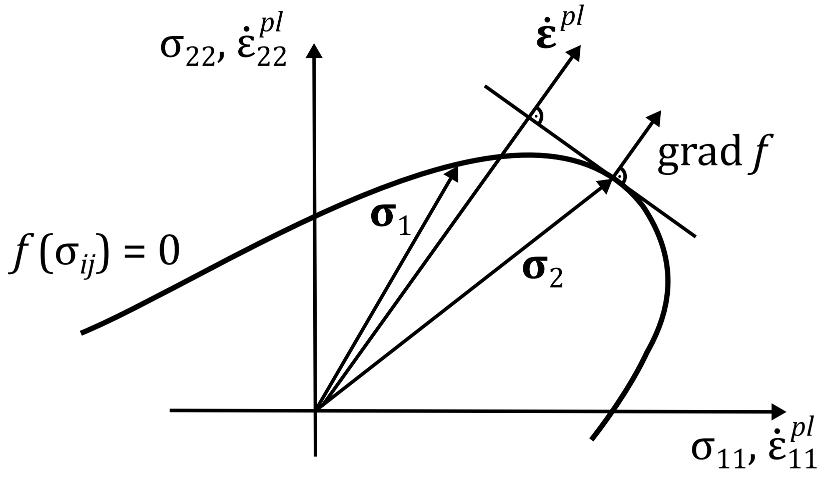


Figure 4.2. Maximal dissipation at plastic deformation (Cf. Rösler et al. [24])

At last a hardening law has to be provided describing the change of the yield surface for increasing plastic deformations. There are several types of hardening laws to describe these changes. A simple model is described by the isotropic hardening where the shape of the yield surface remains the same but the diameter increases. In the case of a Von Mises yield criterion it follows $\sigma_v^2 - f(\sigma_y, k) = 0$ where k describes the isotropic hardening parameter which can be a constant value or a function.

4.1.2. Logarithmic Strain and additive Lagrangian approach to finite plasticity

The later described material model uses a logarithmic strain measure. This measure shows advantages in finite deformation calculations. Heinrich Hencky introduced a fully three-dimensional logarithmic elastic law in 1928 [9]. A historical overview of the field of logarithmic strain measures in nonlinear elasticity can be found in Martin et al. [15]. In Henckys article two conditions were postulated:

> 1) “First we require that the applied work is converted fully into elastic energy, which is released without loss after the loading is removed.”
> 2) “Second [. . .]: if we apply a second loading to an already deformed body, it must not be possible to obtain the first loading from the deformations resulting from the additional loading.”

This shows Henckys idea about an idealized material with an ideally elastic behavior. This approach leads to a distinction between purely elastic and plastic deformations. The second condition implies a law of superposition for the Cauchy stress tensor $\boldsymbol{\sigma}$ which means that it can be expressed in the form of $\boldsymbol{\sigma}(\mathbf{V}_1 \cdot \mathbf{V}_2) = \boldsymbol{\sigma}(\mathbf{V}_1) + \boldsymbol{\sigma}(\mathbf{V}_2)$. Hencky showed that these two postulated conditions can only be accomplished when the Cauchy stress tensor is given by

$$\boldsymbol{\sigma} = 2\mu \mathrm{dev}(\log\mathbf{V}) + \kappa[\mathrm{tr}(\log\mathbf{V})] \cdot \mathbf{1} \tag{4.4}$$

where $\mathbf{V} = \sqrt{\mathbf{F}\mathbf{F}^T}$ denotes the left stretch tensor, μ and κ are material parameters and with the deviatoric part of a matrix $\mathrm{dev}\mathbf{X} = \mathbf{X} - \frac{\mathrm{tr}(\mathbf{X})}{3} \cdot \mathbf{1}$.
With Henckys idea an elastic strain measure $\mathbf{E}^e$ in the logarithmic Lagrangian space can be defined which implies an additive decomposition:

$$\mathbf{E}^e := \mathbf{E} - \mathbf{E}^p. \tag{4.5}$$

In Equation 4.5 $\mathbf{E}^p$ describes the plastic logarithmic Lagrangian strain which must be described in terms of a constitutive equation. The total logarithmic Lagrangian strain $\mathbf{E}$ can be defined with the use of the right Cauchy-Green deformation tensor $\mathbf{C}$ according to the class of Seth-Hill measures of strain as :

$$\mathbf{E} := \frac{1}{2}\ln[\mathbf{C}]. \tag{4.6}$$

Further annotations can be found in Seth [27] and Hill [10]. When the principle directions of the deformation gradients are preserved an additive nature of strains due to the deformation gradients can be achieved and the logarithmic strain tensor can be interpreted as a sum of infinitesimal engineering strain tensors [11].
Remark: $\ln[\mathbf{A}]$ of a symmetric, positive definite tensor $\mathbf{A}$ can be calculated by a diagonilization of the tensor. This leads to $\ln[\mathbf{A}] = \mathbf{V}\ln[\mathbf{D}]\mathbf{V}^T$ where $\mathbf{D}$ denotes a diagonal matrix with the eigenvalues of $\mathbf{A}$ with the corresponding eigenvectors in form of a matrix $\mathbf{V}$.

4.1.3. Calculation specification for transformations from Logarithmic Lagrangian space to Lagrangian space

A calculation specification for a transformation from the logarithmic stress $\mathbf{T}$ and modulus $\mathbb{E}^{ep}$ to the Lagrangian stress $\mathbf{S}$ and modulus $\mathbb{C}^{ep}$ can be obtained in accordance to Miehe and Lambrecht [20]. The derivations of the transformation tensors can be done according to Appendix C. The Lagrangian stress and the corresponding elasticity modulus can be calculated with the internal energy U defined in the logarithmic Lagrangian space as

$$\mathbf{S} := 2\partial_{\mathbf{C}} U(\mathbf{E}) \qquad \text{and} \qquad \mathbb{C}^{ep} := 4\partial_{\mathbf{CC}} U(\mathbf{E}). \tag{4.7}$$

A spectral decomposition of the tensor $\mathbf{C}$ and the total logarithmic Lagrangian strain $\mathbf{E}$ yields for

$$\mathbf{C} = \sum_{a}^{3} \lambda_a \mathbf{N}_a \otimes \mathbf{N}_a \qquad \text{and} \qquad \mathbf{E}(\mathbf{C}) = \sum_{a}^{3} e_a \mathbf{N}_a \otimes \mathbf{N}_a \tag{4.8}$$

with the eigenvalues λ_a and e_a of the tensors $\mathbf{C}$ and $\mathbf{E}$. Both tensors possess the same eigenvectors $\mathbf{N}_a$ as they are co-axial. The eigenvectors are combined to the eigenvalue bases $\mathbf{M}_a$ with the corresponding eigenvalues λ_a:

$$\mathbf{M}_a := \mathbf{N}_a \otimes \mathbf{N}_a \qquad \text{and} \qquad e_a = \frac{1}{2}\ln[\lambda_a] \tag{4.9}$$

This leads to the symmetric fourth- and sixth-order Lagrangian transformation tensors

$$\mathbb{P} := 2\partial_{\mathbf{C}}\mathbf{E} = \sum_{a}^{3} d_a \mathbf{M}_a \otimes \mathbf{M}_a + \sum_{a}^{3}\sum_{b \neq a}^{3} \vartheta_{ab} \mathbf{G}_{ab} \tag{4.10}$$

and

$$\begin{aligned} \mathbb{L} :&= 4\partial^2_{\mathbf{CC}}\mathbf{E} \\ &= \sum_{a}^{3} f_a \mathbf{M}_a \otimes \mathbf{M}_a \otimes \mathbf{M}_a + \sum_{a}^{3}\sum_{b \neq a}^{3} \xi_{ab}(\mathbf{H}_{bab} + \mathbf{H}_{bba} + \mathbf{H}_{abb}) + \sum_{a}^{3}\sum_{b \neq a}^{3}\sum_{c \neq a, c \neq b}^{3} \eta \mathbf{H}_{abc} \end{aligned} \tag{4.11}$$

where the following coefficients are defined:

$$d_a := \lambda_a^{-1} \qquad \text{and} \qquad f_a := -2\lambda_a^{-2}. \tag{4.12}$$

Additionally, introduce:

$$\begin{aligned}
(\mathbf{G}_{ab})^{ijkl} &:= (\mathbf{M}_a)^{ik}(\mathbf{M}_b)^{jl} + (\mathbf{M}_a)^{il}(\mathbf{M}_b)^{jk} \\
(\mathbf{H}_{abc})^{ijklmn} &:= (\mathbf{M}_a)^{ik}(\mathbf{M}_b)^{jm}(\mathbf{M}_c)^{ln} + (\mathbf{M}_a)^{ik}(\mathbf{M}_b)^{jn}(\mathbf{M}_c)^{lm} \\
&+ (\mathbf{M}_a)^{il}(\mathbf{M}_b)^{jm}(\mathbf{M}_c)^{kn} + (\mathbf{M}_a)^{il}(\mathbf{M}_b)^{jn}(\mathbf{M}_c)^{km} \\
&+ (\mathbf{M}_a)^{jl}(\mathbf{M}_b)^{im}(\mathbf{M}_c)^{kn} + (\mathbf{M}_a)^{jl}(\mathbf{M}_b)^{in}(\mathbf{M}_c)^{km} \\
&+ (\mathbf{M}_a)^{jk}(\mathbf{M}_b)^{im}(\mathbf{M}_c)^{ln} + (\mathbf{M}_a)^{jk}(\mathbf{M}_b)^{in}(\mathbf{M}_c)^{lm}.
\end{aligned} \tag{4.13}$$

For three different eigenvalues $\lambda_a \neq \lambda_b \neq \lambda_c$ set:

$$\begin{aligned}
\vartheta_{ab} &:= (e_a - e_b)/(\lambda_a - \lambda_b) \\
\xi_{ab} &:= (\vartheta_{ab} - \frac{1}{2}d_b)/(\lambda_a - \lambda_b) \\
\eta &:= \sum_{a}^{3}\sum_{b\neq a}^{3}\sum_{c\neq a,c\neq b}^{3} e_a/[2(\lambda_a - \lambda_b)(\lambda_a - \lambda_c)].
\end{aligned} \tag{4.14}$$

Use the rules of L'Hospital to define the coefficients for two different eigenvalues $\lambda_a = \lambda_b \neq \lambda_c$:

$$\lim_{\lambda_b \to \lambda_a} \vartheta_{ab} = \frac{1}{2}d_a \qquad \lim_{\lambda_b \to \lambda_a} \xi_{ab} = \frac{1}{8}f_a \qquad \lim_{\substack{\lambda_b \to \lambda_a \\ \lambda_c \neq \lambda_a}} \eta = \xi_{ca} \tag{4.15}$$

For equal eigenvalues $\lambda_a = \lambda_b = \lambda_c$ follows additionally to Equation 4.15:

$$\lim_{\substack{\lambda_b \to \lambda_a \\ \lambda_c \to \lambda_a}} \eta = \frac{1}{8}f_a. \tag{4.16}$$

The Second Piola Kirchhoff stress tensor $\mathbf{S}$ is power conjugated to the rate of the right Cauchy strain tensor in the form of $\mathbf{S} : \frac{1}{2}\dot{\mathbf{C}}$. The logarithmic stress $\mathbf{T}$ is power conjugated to the rate of the total logarithmic strain tensor in the form $\mathbf{T} : \dot{\mathbf{E}}$. This yields the local stress power:

$$\mathcal{P} = \mathbf{S} : \frac{1}{2}\dot{\mathbf{C}} = \mathbf{T} : \dot{\mathbf{E}}. \tag{4.17}$$

The sensitivity of the total logarithmic Lagrangian strain $\mathbf{E}$ in accordance to the change of the deformation leads to:

$$\dot{\mathbf{E}} = \mathbb{P} : \frac{1}{2}\dot{\mathbf{C}} \qquad \text{and} \qquad \dot{\mathbb{P}} = \mathbb{L} : \frac{1}{2}\dot{\mathbf{C}}. \tag{4.18}$$

Inserting Equation 4.18 into Equation 4.17 yields:

$$\mathbf{S} := \mathbf{T} : \mathbb{P} \qquad \text{and} \qquad \mathbb{C}^{ep} = \mathbb{P} : \mathbb{E}^{ep} : \mathbb{P} + \mathbf{T} : \mathbb{L}. \tag{4.19}$$

4.2. Constitutive Equations for an Additive Formulation of Plasticity

In this section the constitutive equations for an additive formulation of plasticity in the logarithmic Lagrangian space are specified for the rate-independent case. Due to the logarithmic strain measures the same structure as found in the geometrically linear theory can be applied. Firstly a thermodynamic potential will be defined. Moreover internal variables and conjugated forces are introduced.

4.2.1. Strain-energy Function

In elasticity the deformation energy can be described in terms of a strain-energy function. The stored energy is therein a function of the deformation of the body. A widely used approach is the St. Venant-Kirchhoff model which is illustrated in Equation 4.20. Further discussion can be found for instance in Hashiguchi & Yamakawa [8].

$$\Psi = \frac{1}{2}\lambda(I_{\mathbf{E}})^2 + \mu II_{\mathbf{E}} \tag{4.20}$$

The parameters λ and μ describe the Lamé constants and $I_{\mathbf{E}} = \mathrm{tr}[\mathbf{E}]$ denotes the first and $II_{\mathbf{E}} = \mathbf{E} : \mathbf{E}$ the second invariant of the Green Lagrange strain tensor $\mathbf{E}$. In case of the logarithmic Lagrangian space structures like in the small strain theory can be used. This will be applied to the later deduced material model. Additional remarks about logarithmic strain-energy functions can be found in Bruhns et al. [4] and references therein.

Assume the stored free energy function Ψ per unit reference volume as a thermodynamic potential as:

$$\Psi = \frac{1}{2}\lambda(I_{\mathbf{E}^e})^2 + \mu II_{\mathbf{E}^e} + \frac{1}{2}h\xi^2. \tag{4.21}$$

The internal scalar variable ξ models hardening where h is the isotropic hardening parameter. The chosen thermodynamic potential is suitable to describe the plastic behavior of metals. The plastic strains $\mathbf{E}^P$ are typically large compared to the elastic ones and therefore a potential function of the second-order elastic logarithmic Lagrangian tensor $\mathbf{E}^e$ as a quadratic scalar-valued function can be used to model the free energy Ψ at a certain material point.

4.2.2. Anisotropic Yield Criterion and Dissipation Inequality

A set of strain-like internal variables can be defined as $\mathcal{I} = \{\mathbf{E}^P, \xi\}$. Similarly stress-like internal forces $\mathcal{F} = \{\mathbf{T}^P, \zeta\}$ can be declared as (see also Ulz [33]):

$$\mathbf{T}^P := -\partial_{\mathbf{E}^P}\psi \tag{4.22}$$

$$\zeta := -\partial_{\xi}\psi. \tag{4.23}$$

The time derivative of ψ yields:

$$\dot{\psi} = \partial_{\mathbf{E}}\psi : \dot{\mathbf{E}} + \partial_{\mathbf{E}^P}\psi \cdot \dot{\mathbf{E}}^P + \partial_{\xi}\psi \cdot \dot{\xi} \tag{4.24}$$

An elastic domain has to be defined to describe a yield criterion. For this purpose a level set function $\phi(\mathbf{T}^P, \zeta; \mathbf{E}^P, \xi)$ and the initial yield stress $c > 0$, which can be defined as $c = \sqrt{2/3}\, y_0$, are used which yields:

$$\mathbb{E} = \{(\mathbf{T}^P, \zeta; \mathbf{E}^P, \xi) \in \mathbb{R}^n \,|\, \phi\,(\mathbf{T}^P, \zeta; \mathbf{E}^P, \xi) \leq c\}. \tag{4.25}$$

The level set function ϕ defines the shape of the elastic domain and has to fulfill certain boundary conditions. Firstly it has to be convex which results from Drucker's postulate. Secondly it has to be positively homogeneous of degree one which is $\phi\,(t\mathcal{F}_1, t\mathcal{F}_2) = t\phi\,(\mathcal{F}_1, \mathcal{F}_2)$ for $t > 0$. Thirdly it is zero at the origin $\phi\,(0) = 0$ and is always positive $\phi\,(\mathbf{T}^P, \zeta; \mathbf{E}^P, \xi) \geq 0$.

The Clausius-Planck inequality is a frequently used formulation of the second law of thermodynamics, which has to be satisfied:

$$\mathcal{D} = \mathbf{T} : \dot{\mathbf{E}} - \dot{\psi} \geq 0 \tag{4.26}$$

where the quantity $\mathcal{D}$ describes the internal dissipation, which is a non-decreasing function in time [31]. With the help of Equation 4.24 it follows:

$$\mathcal{D} = (\mathbf{T} - \frac{\partial \psi}{\partial \mathbf{E}}) : \dot{\mathbf{E}} - \frac{\partial \psi}{\partial \mathbf{E}^P}\dot{\mathbf{E}}^{\mathbf{P}} - \frac{\partial \psi}{\partial \xi} \cdot \dot{\xi} \geq 0. \tag{4.27}$$

By invoking standard arguments, this equation is valid for all processes. An elastic deformation ($\dot{\mathbf{E}}^P = \mathbf{0}$ and $\dot{\xi} = 0$) gives:

$$\mathbf{T} = \frac{\partial \psi}{\partial \mathbf{E}}. \tag{4.28}$$

Furthermore, with $\psi = \psi(\mathbf{E}^e)$ and $\mathbf{E}^e = \mathbf{E} - \mathbf{E}^P$ we find $\frac{\partial \psi}{\partial \mathbf{E}} = \frac{\partial \psi}{\partial \mathbf{E}^e}\frac{\partial \mathbf{E}^e}{\partial \mathbf{E}} = \frac{\partial \psi}{\partial \mathbf{E}^e}$ and $\frac{\partial \psi}{\partial \mathbf{E}^P} = \frac{\partial \psi}{\partial \mathbf{E}^e}\frac{\partial \mathbf{E}^e}{\partial \mathbf{E}^P} = -\frac{\partial \psi}{\partial \mathbf{E}^e}$. Therefore, we can write $\frac{\partial \psi}{\partial \mathbf{E}} = \frac{\partial \psi}{\partial \mathbf{E}^e} = -\frac{\partial \psi}{\partial \mathbf{E}^P}$.

Hence, the reduced Clausius-Planck inequality may be written as:

$$\mathcal{D} = \underbrace{-\frac{\partial \psi}{\partial \mathbf{E}^P}}_{\mathbf{T}^P} : \dot{\mathbf{E}}^{\mathbf{P}} \underbrace{-\frac{\partial \psi}{\partial \xi}}_{\zeta} \cdot \dot{\xi} \tag{4.29}$$

The principle of maximum of plastic dissipation is used to define an optimization problem:

$$\mathcal{D} = \max_{\mathbf{T}^P, \zeta \in \mathbb{E}} \{\mathbf{T}^P : \dot{\mathbf{E}}^P + \zeta \cdot \dot{\xi}\} \tag{4.30}$$

with the side condition $\phi \leq c$ which leads to the following Lagrange functions:

$$\mathcal{L} = \mathbf{T}^P : \dot{\mathbf{E}}^P + \zeta \cdot \dot{\xi} - \lambda[\phi - c]. \tag{4.31}$$

The parameter λ represents the Lagrange multiplier and is found as the plastic consistency parameter. The solution of the equation system with Equation 4.31 and the side condition $\phi - c = 0$ yields the normality rules of plasticity in this case. This solution defines the evolution of the plastic parameters and the hardening.
Correlating the level set function ϕ with the internal dissipation $\mathcal{D}$ by inserting Equation 4.31 into Equation 4.29 and taking into account the positive homogeneity of degree one of ϕ yields:

$$\mathcal{D} = \lambda\phi \geq 0. \tag{4.32}$$

λ has to be positive and therefore the second law of thermodynamics is a priori met.
A level set function ϕ has to be defined to calculate the solution algorithm. In this case an anisotropic plasticity should be modeled. The model of Mandel [14] with a yield criterion $\phi(\mathbf{T}^P, \zeta) - c = 0$ leads to:

$$\phi = \underbrace{||\mathbf{T}^P||_{\mathbb{H}}}_{\phi^e} + \underbrace{\sqrt{\frac{2}{3}}\zeta}_{\phi^h} \tag{4.33}$$

with $||\mathbf{T}^P||_{\mathbb{H}} = \sqrt{\mathbf{T}^P : \mathbb{H} : \mathbf{T}^P}$. The terms ϕ^e and ϕ^h denote the hyperelastic contribution and the isotropic hardening, respectively. The constant fourth-order Hill tensor $\mathbb{H}$ with major and minor symmetries $\mathbb{H}_{IJKL} = \mathbb{H}_{KLIJ} = \mathbb{H}_{JIKL} = \mathbb{H}_{IJLK}$ can be used to model different anisotropic behavior. A special case is defined by $\mathbb{H} = \mathbb{I} - 1/3 \cdot \mathbf{I} \otimes \mathbf{I}$ with the fourth-order identity tensor $\mathbb{I}_{IJKL} = 1/2 \cdot (\mathbf{I}_{IK}\mathbf{I}_{JL} + \mathbf{I}_{IL}\mathbf{I}_{JK})$. In that case Equation 4.33 defines a classical Huber-Von Mises yield function. Additionally the constant limit c is set to $c = \sqrt{2/3} \cdot y_0$.
Anisotropy can be modeled by defining material symmetry groups by extending an isotropic tensor function with an invariant structural tensor. The structural tensor has to be invariant to the according material symmetry group. In Boehler [2] and Spencer [29] classic representation theorems can be found. Anisotropy and its type are determined by the material symmetry group which is described by the structural tensor $\mathbf{M}$. Focusing on orthotropic tensor functions the structural tensor $\mathbf{M}$ can be set according to Zheng [34] as a symmetric tensor $\mathbf{M} = \mathbf{a}_1 \otimes \mathbf{a}_1 - \mathbf{a}_2 \otimes \mathbf{a}_2$ where the vectors $\mathbf{a}_i$ with $i = 1, 2, 3$ and $\delta_{ij} = \mathbf{a}_i \cdot \mathbf{a}_j$ describe an orthonormal base.
With Miehe et al. [19] the basis can be rewritten and the fourth-order structural tensor $\mathbb{M}$ can be derived by describing the anisotropic response in terms of its invariants and calculating the second derivative of a constitutive function with respect to its arguments:

$$\begin{aligned} \mathbb{M} = {} & \alpha_1 \mathbf{M}_{11} \otimes \mathbf{M}_{11} + \alpha_2 \mathbf{M}_{22} \otimes \mathbf{M}_{22} + \alpha_3 \mathbf{M}_{33} \otimes \mathbf{M}_{33} \\ & + 2\alpha_4 sym[\mathbf{M}_{11} \otimes \mathbf{M}_{22}] + 2\alpha_5 sym[\mathbf{M}_{22} \otimes \mathbf{M}_{33}] + 2\alpha_6 sym[\mathbf{M}_{11} \otimes \mathbf{M}_{33}] \\ & + 2\alpha_7 \mathbf{M}_{12} \otimes \mathbf{M}_{21} + 2\alpha_8 \mathbf{M}_{23} \otimes \mathbf{M}_{32} + 2\alpha_9 \mathbf{M}_{13} \otimes \mathbf{M}_{31} \end{aligned} \tag{4.34}$$

with the abbreviation $sym[\mathbf{M}_{ii} \otimes \mathbf{M}_{jj}] = 1/2(\mathbf{M}_{ii} \otimes \mathbf{M}_{jj} + \mathbf{M}_{jj} \otimes \mathbf{M}_{ii})$. The fourth-order structural tensor $\mathbb{M}$ has minor and major symmetries. By setting $\mathbb{H} = \mathbb{M}$ in Equation 4.33 orthotropic anisotropy can be achieved.

The material model has independent material parameters which can be calculated with the three tension modes y_{11}, y_{22} and y_{33} and the three shear modes y_{12}, y_{23} and y_{13}:

$$\begin{aligned} \alpha_1 &= \frac{2}{3}\frac{y_0^2}{y_{11}^2} \qquad \alpha_2 = \frac{2}{3}\frac{y_0^2}{y_{22}^2} \qquad \alpha_3 = \frac{2}{3}\frac{y_0^2}{y_{33}^2} \\ \alpha_7 &= \frac{1}{3}\frac{y_0^2}{y_{12}^2} \qquad \alpha_8 = \frac{1}{3}\frac{y_0^2}{y_{23}^2} \qquad \alpha_9 = \frac{1}{3}\frac{y_0^2}{y_{13}^2} \end{aligned} \tag{4.35}$$

The introduction of the deviatoric property of the fourth-order Hill tensor $\mathbb{H} : \mathbf{I} = \mathbf{0}$ can be achieved by arrogating incompressibility which yields:

$$\alpha_4 = \frac{1}{2}(\alpha_3 - \alpha_1 - \alpha_2) \quad \alpha_5 = \frac{1}{2}(\alpha_1 - \alpha_2 - \alpha_3) \quad \alpha_6 = \frac{1}{2}(\alpha_2 - \alpha_3 - \alpha_1) \tag{4.36}$$

Isotropic behavior can be accomplished by setting $y_{11} = y_{22} = y_{33} = y_0$ and $y_{12} = y_{23} = y_{13} = y_0/\sqrt{3}$.

4.3. Solution Algorithm for rate-independent Plasticity

In this section a solution algorithm for anisotropic rate-independent plasticity in the logarithmic Lagrangian strain space is established. A framework in the large-strain scope of the Lagrangian strain space is defined which will be moved to the small-strain scope in the logarithmic Lagrangian strain space where the constitutive equations are formulated. The results within the logarithmic framework can finally be transformed back to the Lagrangian strain space.

The right Cauchy-Green deformation tensor $\mathbf{C}$ can be computed with the deformation gradient $\mathbf{F}$ in Lagrangian space. Move the resulting deformation tensor to logarithmic Lagrangian space by calculating the total logarithmic strain $\mathbf{E}$. Define a set of internal variables constituted of the plastic logarithmic Lagrangian strain tensor $\mathbf{E}^P$ and a hardening parameter ξ. A set of constitutive equations is used to calculate the logarithmic Lagrangian stress tensor $\mathbf{T}^P$ and the appropriate elastic-plastic logarithmic Lagrangian tangent modulus $\mathbb{E}^{ep}$. The additive structure of strains in the logarithmic Lagrangian space allows the adaption of the standard constitutive functions from the small strain theory. A calculation specification for the transformation between both strain spaces is used to calculate the associated Second Piola Kirchhoff stress $\mathbf{S}$ in the Lagrangian space and the elastic-plastic Lagrangian tangent modulus $\mathbb{C}^{ep}$.

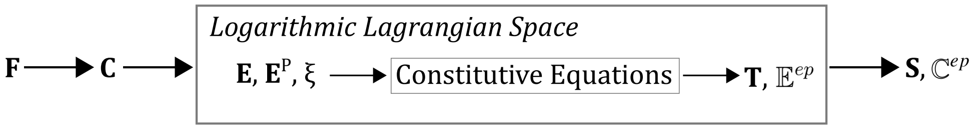


Figure 4.3. Approach for the solution algorithm for rate-independent plasticity in logarithmic Lagrangian strain space (Cf. Ulz [33])

4.3.1. Local Iteration Scheme

The solution of the optimization problem from Equation 4.30 and Equation 4.31 yields the evolution equations for the internal variables:

$$\begin{aligned} \dot{\mathbf{E}}^P &= \lambda \partial_{\mathbf{T}^P} \phi \\ \dot{\xi} &= \lambda \partial_{\zeta} \phi. \end{aligned} \tag{4.37}$$

This formulation can be rewritten in terms of on an integration in the time interval $[t_n, t_{n+1}]$:

$$\begin{aligned} \mathbf{E}^P_{n+1} &= \mathbf{E}^P_n + \int_{t^n}^{t^{n+1}} \lambda \partial_{\mathbf{T}^P} \phi \\ \xi_{n+1} &= \xi_n + \int_{t^n}^{t^{n+1}} \lambda \partial_{\zeta} \phi \end{aligned} \tag{4.38}$$

The obtained time integral formulations can be solved by introducing either an implicit or explicit numerical integration. In an implicit integration scheme the unknown quantities are computed with items at the time step t_{n+1} whereas an explicit scheme uses the solution from the time step t_n to calculate the unknown quantities:

$$\mathcal{I}_{n+1} = \mathcal{I}_n + \lambda \Delta t \begin{cases} \partial_{\mathcal{F}} \phi_{n+1} & \text{implicit} \\ \partial_{\mathcal{F}} \phi_n & \text{explicit} \end{cases} \tag{4.39}$$

Introduce a dimensionless multiplier $\gamma^P = \lambda \Delta t$ with $\Delta t = t_{n+1} - t_n$ which is called the plastic multiplier in the rate-independent formulation and use an implicit integration scheme for Equation 4.38:

$$\begin{aligned} \mathbf{E}^P_{n+1} &= \mathbf{E}^P_n + \gamma^P \partial_{\mathbf{T}^P} \phi \\ \xi_{n+1} &= \xi_n + \gamma^P \partial_{\zeta} \phi \end{aligned} \tag{4.40}$$

The Karush-Kuhn-Tucker loading conditions and the positively homogeneous level set function ϕ limit the plastic multiplier γ^P in the rate-independent formulation:

$$\mathcal{K}_{n+1} := \{\gamma^P \geq 0, \phi_{n+1} \leq c, \gamma^P [\phi_{n+1} - c] = 0\} \tag{4.41}$$

Introducing residuals that vanish in the solution point in terms of Equation 4.40 yields:

$$\begin{aligned} \mathbf{R}_{\mathbf{E}^P} &= -\mathbf{E}^P_{n+1} + \mathbf{E}^P_n + \gamma^P \partial_{\mathbf{T}^P} \phi_{n+1} \\ R_{\xi} &= -\xi_{n+1} + \xi_n + \gamma^P \partial_{\zeta} \phi_{n+1} \\ R_f &= \phi_{n+1} - c. \end{aligned} \tag{4.42}$$

A linearization of the residuals from Equation 4.42 with respect to $\mathbf{E}^P$, ξ and γ^P leads to:

$$\begin{gathered}
0 = \mathbf{R}_{\mathbf{E}^P} - \Delta\mathbf{E}^P + \gamma^P \partial^2_{\mathbf{T}^P\mathbf{T}^P}\phi_{n+1}\partial_{\mathbf{E}^P}\mathbf{T}^P\Delta\mathbf{E}^P + \partial_{\mathbf{T}^P}\phi_{n+1}\Delta\gamma^P \\
0 = R_\xi - \Delta\xi + \gamma^P\partial^2_{\zeta\zeta}\phi_{n+1}\partial_\xi\zeta\Delta\xi + \partial_\zeta\phi_{n+1}\Delta\gamma^P \\
0 = R_f + \partial_{\mathbf{T}^P}\phi_{n+1}\partial_{\mathbf{E}^P}\mathbf{T}^P\Delta\mathbf{E}^P + \partial_\zeta\phi_{n+1}\partial_\xi\zeta\Delta\xi.
\end{gathered} \tag{4.43}$$

The terms for $\Delta\mathbf{E}^P$, $\Delta\xi$ and $\Delta\gamma^P$ have to be found with the help of Equation 4.43 to establish a local iteration scheme:

$$\begin{gathered}
\Delta\mathbf{E}^P = \frac{\mathbf{R}_{\mathbf{E}^P} + \partial_{\mathbf{T}^P}\phi_{n+1}\Delta\gamma^P}{\mathbb{I} + \gamma^P\partial^2_{\mathbf{T}^P\mathbf{T}^P}\phi_{n+1}\partial^2_{\mathbf{E}^P\mathbf{E}^P}U_{n+1}} := \frac{\mathbf{A}}{\mathbb{B}} \\
\Delta\xi = \frac{R_\xi + \partial_\zeta\phi_{n+1}\Delta\gamma^P}{1 + \gamma^P\partial^2_{\zeta\zeta}\phi_{n+1}\partial^2_{\xi\xi}U_{n+1}} := \frac{a}{b}.
\end{gathered} \tag{4.44}$$

In Equation 4.44 $\Delta\gamma^P$ is to be found. Inserting $\Delta\mathbf{E}^P$ and $\Delta\xi$ from Equation 4.44 into Equation 4.43 yields:

$$\Delta\gamma^P = \frac{c - \phi_{n+1} - \partial_{\mathbf{T}^P}\phi_{n+1}\partial_{\mathbf{E}^P}\mathbf{T}^P\mathbf{R}_{\mathbf{E}^P}\mathbb{B}^{-1} - \partial_\zeta\phi_{n+1}\partial_\xi\zeta R_\xi b^{-1}}{\partial_{\mathbf{T}^P}\phi_{n+1}\partial_{\mathbf{E}^P}\mathbf{T}^P\partial_{\mathbf{T}^P}\phi_{n+1}\mathbb{B}^{-1} + \partial_\zeta\phi_{n+1}\partial_\xi\zeta\partial_\zeta\phi_{n+1}b^{-1}}. \tag{4.45}$$

By applying $\partial_{\mathbf{E}^P}\mathbf{T}^P = -\partial^2_{\mathbf{E}^P\mathbf{E}^P}\psi$ and $\partial_\xi\zeta = -\partial^2_{\xi\xi}\psi$ the constitutive equation for $\Delta\gamma^P$ follows as:

$$\Delta\gamma^P = \frac{\phi_{n+1} - c + \partial_{\mathbf{T}^P}\phi_{n+1}\partial^2_{\mathbf{E}^P\mathbf{E}^P}\psi_{n+1}\mathbf{R}_{\mathbf{E}^P}\mathbb{B}^{-1} + \partial_\zeta\phi_{n+1}\partial^2_{\xi\xi}\psi_{n+1}R_\xi b^{-1}}{\partial_{\mathbf{T}^P}\phi_{n+1}\partial^2_{\mathbf{E}^P\mathbf{E}^P}\psi_{n+1}\partial_{\mathbf{T}^P}\phi_{n+1}\mathbb{B}^{-1} + \partial_\zeta\phi_{n+1}\partial^2_{\xi\xi}\psi_{n+1}\partial_\zeta\phi_{n+1}b^{-1}} \tag{4.46}$$

The constitutive equations for $\Delta\mathbf{E}^P$ and $\Delta\xi$ can be calculated by using the formulation for $\Delta\gamma^P$. The obtained system can the be solved with a numerical iteration scheme like the Newton-Raphson scheme.
The derivatives of the internal variables with respect to the logarithmic Lagrangian strain $\mathbf{E}$ and the plastic multiplier γ^P for the rate-independent case follows as:

$$\partial_{\gamma^P}\mathbf{E}^P = \partial_{\mathbf{T}^P}\phi_{n+1} + \gamma^P\partial^2_{\mathbf{T}^P\mathbf{T}^P}\phi_{n+1}\cdot\partial_{\mathbf{E}^P}\mathbf{T}^P\cdot\partial_{\gamma^P}\mathbf{E}^P \tag{4.47}$$

$$\partial_{\gamma^P}\xi = \partial_\zeta\phi_{n+1} + \gamma^P\partial^2_{\zeta\zeta}\phi_{n+1}\cdot\partial_\xi\zeta\cdot\partial_{\gamma^P}\xi. \tag{4.48}$$

Rewriting the equations with the help of Equation 4.22 and Equation 4.23 yields:

$$\partial_{\gamma^P}\mathbf{E}^P = (\mathbb{I} + \gamma^P\partial^2_{\mathbf{T}^P\mathbf{T}^P}\phi_{n+1}\cdot\partial^2_{\mathbf{E}^P\mathbf{E}^P}\psi_{n+1})^{-1}\partial_{\mathbf{T}^P}\phi_{n+1} \tag{4.49}$$

$$\partial_{\gamma^P}\xi = (1 + \gamma^P\partial^2_{\zeta\zeta}\phi_{n+1}\cdot\partial^2_{\xi\xi}\psi_{n+1})^{-1}\partial_\zeta\phi_{n+1}. \tag{4.50}$$

4.3.2. Tangent Modulus

The logarithmic stress $\mathbf{T}$ and the logarithmic elastic-plastic tangent modulus $\mathbb{E}^{ep}$ in the logarithmic Lagrangian space are defined in terms of the Helmholtz free energy ψ as:

$$\mathbf{T}_{n+1} := \mathrm{d}_{\mathbf{E}}\psi(\mathbf{E}_{n+1}) \qquad \text{and} \qquad \mathbb{E}^{ep}_{n+1} := \mathrm{d}^2_{\mathbf{EE}}\psi(\mathbf{E}_{n+1}) \tag{4.51}$$

The minimum principle of the internal energy ψ with respect to the plastic multiplier γ^P yields $\partial_{\gamma^P}\psi = 0$. Using the chain rule for the derivatives from Equation 4.51 leads to:

$$\mathbf{T}_{n+1} := \partial_{\mathbf{E}}\psi + \underbrace{\partial_{\gamma^P}\psi}_{=0}\,\partial_{\mathbf{E}}\gamma^P \qquad \text{and} \qquad \mathbb{E}^{ep}_{n+1} := \underbrace{\partial^2_{\mathbf{EE}}\psi}_{elastic} + \underbrace{\partial^2_{\mathbf{E}\gamma^P}\psi\partial_{\mathbf{E}}\gamma^P}_{softening} \tag{4.52}$$

By introducing a factor β which is used as a calculation switch Equation 4.52 can be rewritten as:

$$\mathbb{E}^{ep} = \mathbb{E}^{ep}_{elastic} + \beta\mathbb{E}^{ep}_{softening} \qquad \text{with} \qquad \beta = \begin{cases} 1 & \text{for} \quad \gamma^P > 0 \\ 0 & \text{else} \end{cases} \tag{4.53}$$

where $\beta = 1$ for $\gamma^P > 0$. This factor is solely introduced for numerical reasons to switch off the softening part in case of no additional plastification. The single terms from Equation 4.53 can be expressed in a different way by using the positive homogeneity of the level set function $\bullet \cdot \partial_{\bullet}\phi = \phi$ and $\bullet \cdot \partial^2_{\bullet\bullet}\phi = 0$ with $(\bullet) = \{\mathbf{T}^P, \zeta\}$ and with the help of $\mathbf{T}^P \cdot \partial_{\mathbf{E}}\mathbf{E}^P = \mathbf{0}$, $\zeta \cdot \partial_{\mathbf{E}}\xi = \mathbf{0}$ and Equation 4.22. The elastic part of the elastic-plastic tangent modulus yields:

$$\mathbb{E}^{ep}_{elastic} = \partial^2_{\mathbf{EE}}\psi + \partial^2_{\mathbf{EE}}\psi\partial_{\mathbf{E}}\mathbf{E}^P + \partial^2_{\mathbf{E}\xi}\psi\partial_{\mathbf{E}}\xi. \tag{4.54}$$

This equation can be expanded:

$$\mathbb{E}^{ep}_{elastic} = \partial^2_{\mathbf{EE}}\psi - \frac{\gamma^P\partial^2_{\mathbf{EE}^P}\psi}{\mathbb{I} + \gamma^P\partial^2_{\mathbf{T}^P\mathbf{T}^P}\phi\partial^2_{\mathbf{E}^P\mathbf{E}^P}\psi}\partial^2_{\mathbf{T}^P\mathbf{T}^P}\phi\partial^2_{\mathbf{E}^P\mathbf{E}}\psi + \partial_{\zeta\zeta}\phi \tag{4.55}$$

The softening part of the elastic-plastic tangent modulus yields:

$$\mathbb{E}^{ep}_{softening} = \partial^2_{\mathbf{EE}}\psi\partial_{\gamma}\mathbf{E}^P\partial_{\mathbf{E}}\gamma + \partial^2_{\mathbf{E}\xi}\psi\partial_{\gamma}\xi\partial_{\mathbf{E}}\gamma. \tag{4.56}$$

By applying the known formulations Equation 4.56 can be rearranged as:

$$\begin{aligned} \mathbb{E}^{ep}_{softening} &= \partial_{\mathbf{E}}\phi \otimes \partial_{\mathbf{E}}\phi(\partial\gamma^P\phi)^{-1} \\ &= \frac{\partial_{\mathbf{E}}\phi \otimes \partial_{\mathbf{E}}\phi}{-\partial_{\mathbf{T}^P}\phi\partial^2_{\mathbf{E}^P\mathbf{E}^P}\psi\mathbb{B}^{-1}\partial_{\mathbf{T}^P}\phi - \partial_{\zeta}\phi\partial^2_{\mathbf{E}^P\mathbf{E}^P}\psi b^{-1}\partial_{\zeta}\phi} \\ &= \frac{(-\partial^2_{\mathbf{EE}^P}\psi\mathbb{B}^{-1}\partial_{\mathbf{T}^P}\phi) \otimes (-\partial^2_{\mathbf{EE}^P}\psi\mathbb{B}^{-1}\partial_{\mathbf{T}^P}\phi)}{-\partial_{\mathbf{T}^P}\phi\partial^2_{\mathbf{E}^P\mathbf{E}^P}\psi\mathbb{B}^{-1}\partial_{\mathbf{T}^P}\phi - \partial_{\zeta}\phi\partial^2_{\mathbf{E}^P\mathbf{E}^P}\psi b^{-1}\partial_{\zeta}\phi}. \end{aligned} \tag{4.57}$$

4.3.3. Summary of the Solution Algorithm

This section summarizes the deployed solution algorithm for a rate-independent elasto-plastic material in logarithmic Lagrangian strain space. In every calculation step of the global iteration (**) the plastic multiplier is initialized with $\gamma^P = 0$ and the initial values of the internal variables are set as $(\mathbf{E}^P)^{i=0}_{n+1} = (\mathbf{E}^P)_n$ and $\xi^{i=0}_{n+1} = \xi_n$ where i denotes the iteration step of the local iteration (*) and n the iteration step of the global iteration (**). If the Karush-Kuhn-Tucker conditions are not violated an elastic response occurs and the local plasticity algorithm (*) has to be skipped and the algorithm continues with step 7. The local iteration scheme (*) is running till a maximum number of iterations is reached or the term $|\phi_{n+1} - c| <$ tol falls below a user-defined tolerance value tol. Similarly the global iteration scheme (**) is running till a maximum number of iterations is reached or the unbalanced energy falls below a user-defined tolerance value $|\text{unbalanced energy}| < \text{tol}_{\text{unb}}$.

1. Calculate deformation gradient $\mathbf{F}$:

   $$\mathbf{F} = \frac{\partial \mathbf{x}}{\partial \mathbf{X}}$$

2. Calculate right Cauchy-Green deformation tensor $\mathbf{C}$:

   $$\mathbf{C} = \mathbf{F}^T\mathbf{F}$$

3. Calculate total logarithmic Lagrangean strain $\mathbf{E}$:

   $$\mathbf{E} = \tfrac{1}{2}\ln\mathbf{C}$$

4. Calculate local internal variables for every integration point:

   $$(\mathbf{E}^P)^{i+1}_{n+1} = (\mathbf{E}^P)^i_n + (\Delta\mathbf{E}^P)^i$$

   $$\xi^{i+1}_{n+1} = \xi^i_n + \Delta\xi^i \qquad (*) \qquad\qquad (**)$$

   $$(\gamma^P)^{i+1}_{n+1} = (\gamma^P)^i_n + (\Delta\gamma^P)^i$$

5. Calculate elastic logarithmic Lagrangean strain $\mathbf{E}^e$:

   $$\mathbf{E}^e = \mathbf{E} - (\mathbf{E}^P)^{i+1}_{n+1}$$

6. Set $\beta = 0$ in Equation 4.53 if $\gamma^P < 0$

7. Calculate $\mathbf{T}^P$, $\mathbb{E}^{ep}$

8. Transform to $\mathbf{S}$, $\mathbb{C}^{ep}$:

   $$\mathbf{S} = \mathbf{T}^P : \mathbb{P}$$

   $$\mathbb{C}^{ep} = \mathbb{P} : \mathbb{E}^{ep} : \mathbb{P} + \mathbf{T}^P : \mathbb{L}$$

5. Verification

Numerical simulations by means of the established constitutive equations are presented in this chapter. The objective is to show a set of easy examples which function as a benchmark. The material model is implemented in an already existing framework programmed in Matlab which is called "soofeaM" (software for object-oriented finite element analysis in Matlab) and was provided by the Institute of Strength of Materials at Graz University of Technology. The used framework represents an object oriented finite element program with the possibility of an implementation of new material models. The numerical calculations are performed with 8-node trilinear hexahedral solid elements on a computer with Windows 10 (Intel(R) Core(TM) i7-8700K CPU @ 3.70 GHz, 64-bit) and the numerical integration is implemented as Gauß-quadrature. An implicit time integration scheme is used for the following examples. The calculation output results are the nodal displacements $\mathbf{u}$, the Von Mises Stress S and the equivalent plastic strain $PEEQ$ which can be derived with the help of a definition as found in Lubliner [13] for instance, which is $\kappa = \int \sqrt{\frac{2}{3}\,\dot{\mathbf{E}}^P : \dot{\mathbf{E}}^P}$. This expression results in $PEEQ = \sqrt{\frac{2}{3}\,\mathbf{E}^P : \mathbf{E}^P}$ within a considered time interval of $[t_0, t_n]$ by defining $\mathbf{E}^P(t_0) = \mathbf{0}$.

The numerical simulation examples contain firstly a single element test to check the quadratic convergence of the derived material model and the correct implementation of the equations. Secondly the simulation results for a rectangular strip under tension and compression are shown. Thirdly the drawing of a thin circular plate is simulated which represents a simplification of a deep drawing process. For every numerical example the material parameters are varied to gain the difference between isotropic and anisotropic material behavior.

The simulation model has some limitations. For the element definition 8-node trilinear hexahedral elements were used. It is well known that the near incompressibility condition of plastic deformation in metals requires special care of the used element type. Rice et al. [23] summarizes possible problems that can occur by choosing a non feasible element formulation or mesh layout. This results in the fully plastic range in much too stiff a response. A point-wise incompressible behavior of the mesh leads to a limit-load at a certain deformation in the plastic regime. If this constraint is not ensured an unlimited deformation without further load increase occurs. In this work the focus solely lies on the numerical implementation of the plasticity algorithm and it was therefore refrained from the implementation of a remedy to this incompressibility constraint.

5.1. Convergence Test

A convergence test with a single element was simulated to verify the functionality of the implemented material model with its local and global Newton algorithm. A load was applied to one face of a 8-node hexahedral trilinear solid element. Figure 5.1 yields the described situation. Therein $t = 0$ and t describe the initial and deformed state, respectively. The load was applied within 10 iteration steps.

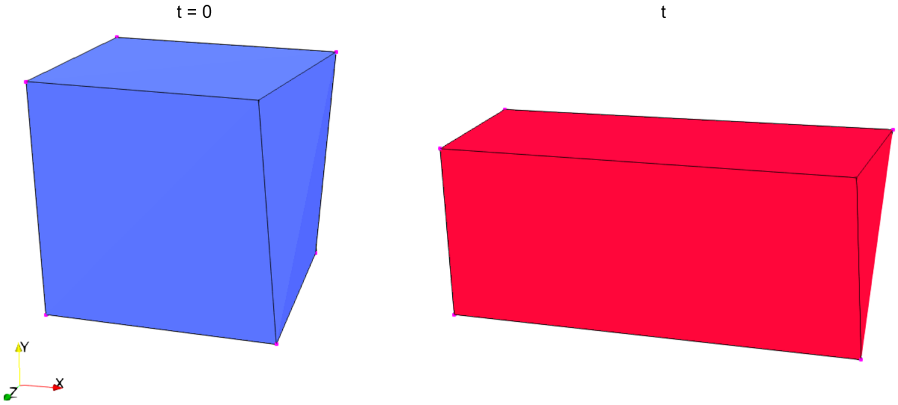


Figure 5.1. Convergence Test. Considered 8-node hexahedral trilinear solid element in initial $t = 0$ and deformed state t.

The coordinates of the 8 nodes of the hexahedral trilinear solid element are denoted in Table 5.1. The element was chosen to be not perfectly rectangular to check the sensitivity of the implemented model.

Table 5.1. Convergence Test. x -, y- and z - coordinates of the considered 8-node hexahedral trilinear solid element.

Node ID	x - coordinate	y - coordinate	z - coordinate
1	1.06	0.98	0.00
2	0.00	1.02	0.00
3	0.00	0.00	0.00
4	0.97	0.00	0.00
5	0.98	1.01	1.05
6	0.00	1.03	0.99
7	0.00	0.00	0.98
8	1.03	0.00	1.01

Table 5.2 yields the properties used for the calculation of the convergence test. Isotropic hardening was applied.

Table 5.2. Convergence Test. Material properties for the considered 8-node hexahedral trilinear solid element.

Parameter	Symbol	Value	Unit
Young's Modulus	E	210	GPa
Poisson's ratio	ν	0.3	-
Initial yield stress	y_0	500	MPa
Linear isotropic hardening	h	129.24	MPa

The Newton algorithm shows quadratic convergence for the unbalanced energy when the solution algorithm is programmed correctly. Table 5.3 yields that the implemented algorithm converges in a quadratic manner. This behavior implies the correct implementation of code.

Table 5.3. Convergence Test. Convergence of the unbalanced energy for the considered 8-node hexahedral trilinear solid element. Timestep t 1, 3, 5, 7 and 10. Global iteration counter i.

	Unbalanced energy for timestep				
i/t	1	3	5	7	10
1	1.5e+03	1.3e+03	1.2e+03	1.1e+03	9.6e+02
2	1.2e+02	9.9e+01	8.0e+01	6.5e+01	5.0e+01
3	4.5e-02	3.1e-02	2.0e-02	1.4e-02	8.0e-03
4	3.9e-03	1.5e-03	7.1e-04	4.5e-04	3.4e-04
5	3.6e-10	2.2e-11	5.8e-11	2.5e-10	9.2e-10
6	1.1e-22	5.0e-25	8.8e-24	3.6e-22	1.4e-20

5.2. Rectangular Strip under Tension and Compression

This section shows a thin rectangular strip under a tension and compression load. The example represents a simple numerical benchmark and is based on calculations in Ulz [33]. This model of computation is used to analyze the capability of the derived material model. Different material configurations are examined. The simulation is displacement driven. The investigations embraces three different material configurations in the two load cases tension and compression.

5.2.1. Model of Computation

The rectangular strip has the dimension of 10 × 10 mm and a thickness of 1.0 mm. A displacement of $u = 3\,\mathrm{mm}$ with increments of $\Delta u = 0.01\,\mathrm{mm}$ is applied on the right edge of the strip within 300 global iteration steps. The dimensions of the model of computation can be taken from Figure 5.2. The orthotropic axis $\mathbf{a}_i$ with $i = \{1, 2, 3\}$ are rotated by an angle of 30° to the global coordinate system (see Figure 5.2). Buckling is prevented by supporting the bottom of the strip. Equal loads are applied on the geometry and the simulation outputs displacement, Von Mises stress and the equivalent plastic strain are compared for different material configurations.

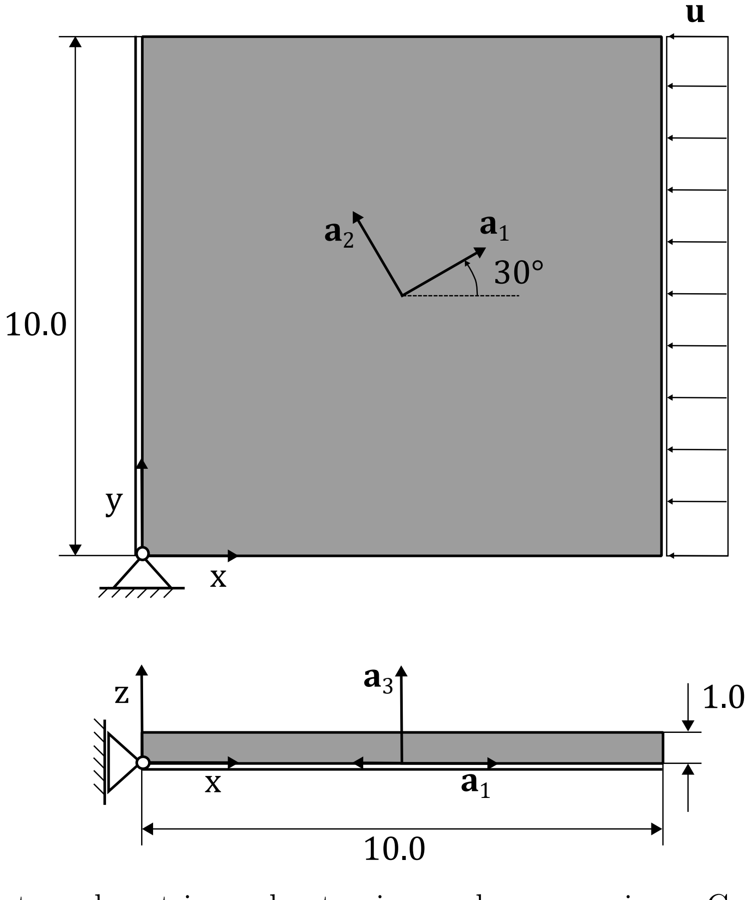


Figure 5.2. Rectangular strip under tension and compression. Geometry of the considered calculation domain.

The geometry is discretized by 7x7 8-node trilinear hexahedral solid elements with a total number of 128 nodes. The thickness direction of the thin strip is modelled by one element row. Figure 5.3 yields the mesh and the nodes under consideration. The nodes A and B have the coordinates (10.0 | 10.0 | 0.0) and (10.0 | 0.0 | 0.0).

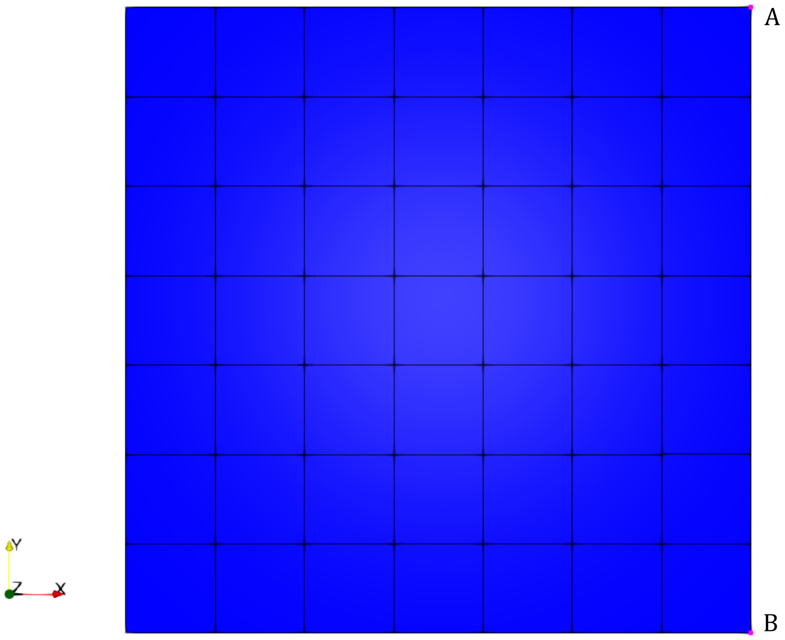


Figure 5.3. Rectangular strip under tension and compression. Mesh with the considered nodes A and B.

Table 5.4 lists the basic material properties for the numerical example. Isotropic material behavior is achieved by setting the parameters $y_{11} = y_{22} = y_{33} = y_0$ and $y_{12} = y_{23} = y_{13} = y_0/\sqrt{3}$. A parameter ϱ is introduced to distinguish between different material behavior and is defined as the ratio of the normal and shear stresses as $\varrho := y_{ii}/y_{ji}$. The isotropic case is associated with $\varrho_1 = \sqrt{3}$. The other two calculated sets are defined by $\varrho_2 = 2\sqrt{3}$ and $\varrho_3 = 0.5\sqrt{3}$ and represent the orthotropic material response respectively.

Table 5.4. Rectangular strip under tension and compression. Material properties for the model of computation.

Parameter	Symbol	Value	Unit
Young's Modulus	E	206.9	GPa
Poisson's ratio	ν	0.29	-
Initial yield stress	y_0	450	MPa
Linear isotropic hardening	h	129.24	MPa

5.2.2. Results - Compression Load

Displacements

The deformed mesh and displacement of the nodes of the geometry can be taken from Figure 5.4. The result of the isotropic material ϱ_1 shows an increased height of the body under consideration. The mesh deforms without shear modes. The maximum displacement can be seen at the top right node, which is denoted with A (see Figure 5.3). The other two material configurations deform with shear modes in different directions accordingly. For the case ϱ_2 the body stretches in y - direction and shows a shear movement towards negative y - direction. In the case of ϱ_3 the shear movement points to positive y - direction.

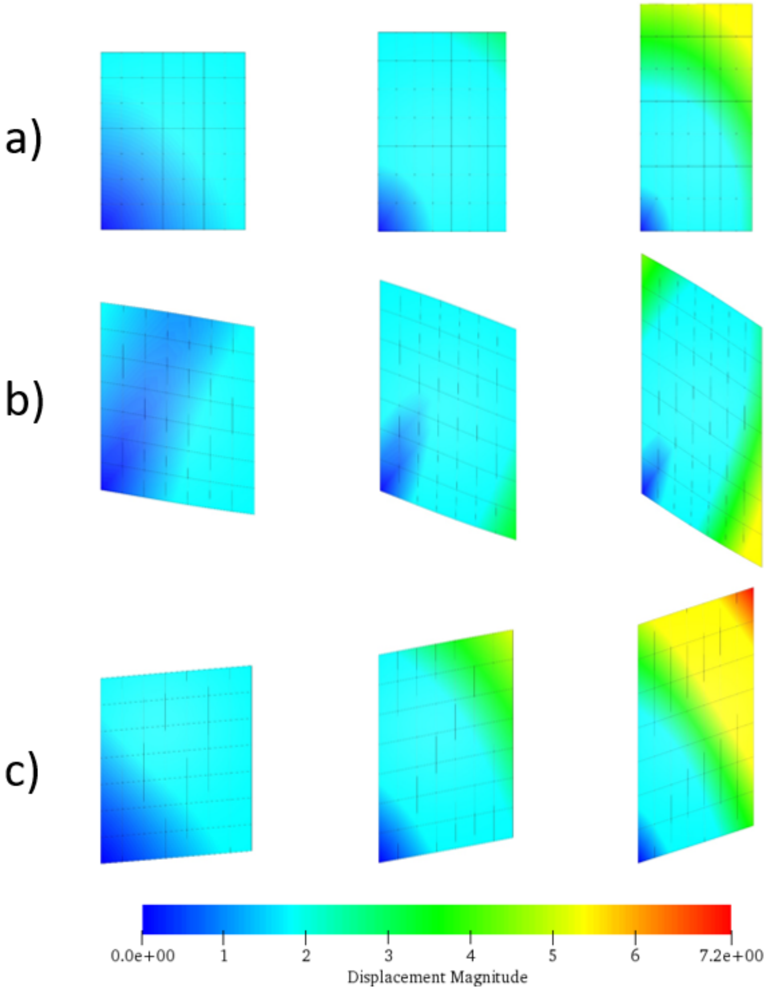


Figure 5.4. Rectangular strip under compression. Deformed meshes and displacements as shown in color bar for a) isotropic $\varrho_1 = \sqrt{3}$, b) anisotropic $\varrho_2 = 2\sqrt{3}$ and c) anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior. First column: top view for global iteration step $i = 100$. Second column: top view for $i = 200$. Third column: top view for $i = 300$.

The displacement v in y - direction of the two reference nodes A and B for the different material configurations can be obtained from Figure 5.5. In the isotropic case ϱ_1 node B remains in the same y - coordinate. In the other two cases the nodes A and B move in y - direction according to their softer material orientation. In the case of ϱ_2 the material is softer in the direction of the $\mathbf{a}_2$ axis. For ϱ_3 the softer material axis is represented by $\mathbf{a}_1$.

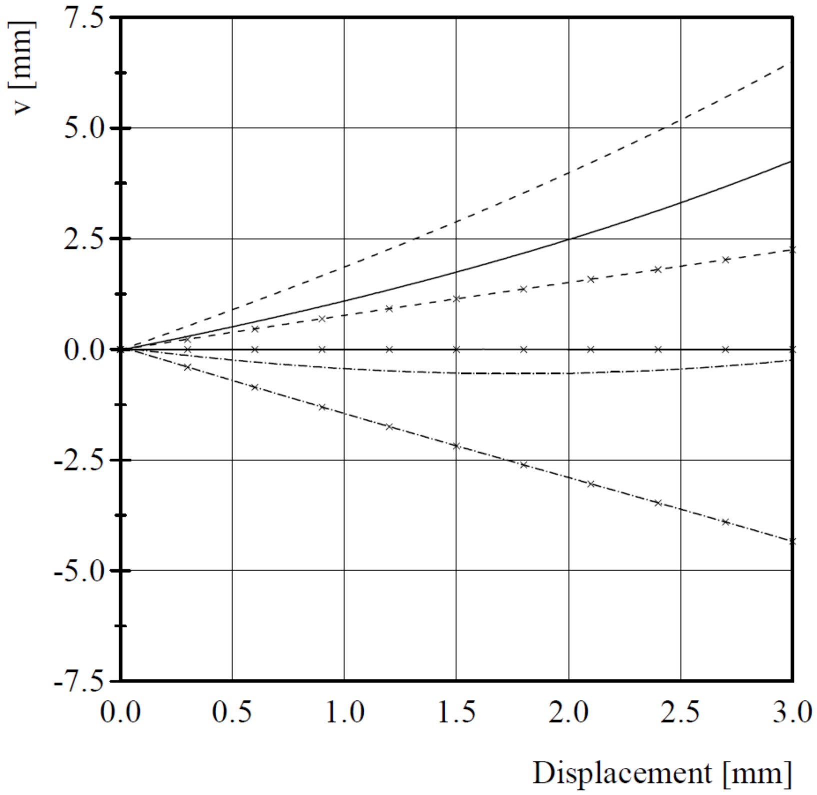


Figure 5.5. Rectangular strip under compression. Displacements v in y-direction over compression displacement. Nodes A and B are represented by curves without marker and with cross. Solid line represents isotropic $\varrho_1 = \sqrt{3}$, dashed-dotted anisotropic $\varrho_2 = 2\sqrt{3}$ and dashed anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior.

Von Mises Stress

The deformed mesh and Von Mises Stress can be obtained from Figure 5.6. In the case of isotropy ϱ_1 the stresses are uniformly distributed. For ϱ_2 the stress shows a symmetric distribution. The stress has its maximum at the bottom left and top right node. This corresponds to the material orientation and direction of soft mechanical properties. In the case of ϱ_3 a symmetric stress distribution in the opposite direction can be found. Maximum Von Mises stresses can be found in the top left and bottom right node. A detailed representation of the Von Mises stress for the nodes A and B can be obtained from Figure 5.7. The case ϱ_2 shows higher magnitudes in the Von Mises stress.

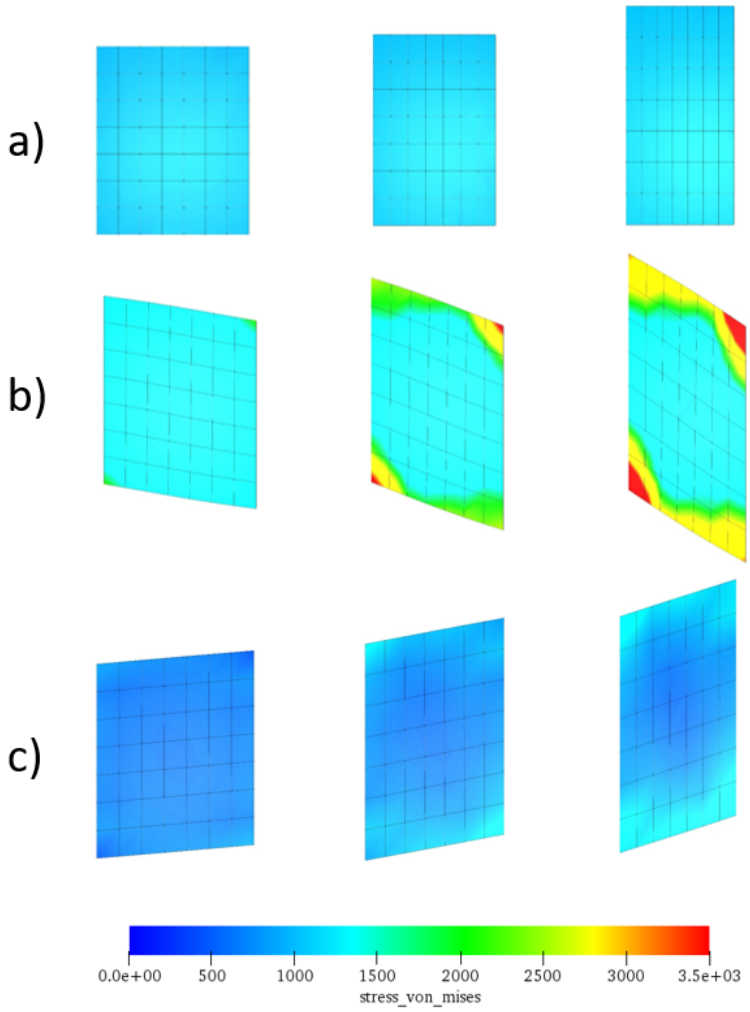


Figure 5.6. Rectangular strip under compression. Deformed meshes and Von Mises stress as shown in color bar for a) isotropic $\varrho_1 = \sqrt{3}$, b) anisotropic $\varrho_2 = 2\sqrt{3}$ and c) anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior. First column: top view for global iteration step $i = 100$. Second column: top view for $i = 200$. Third column: top view for $i = 300$.

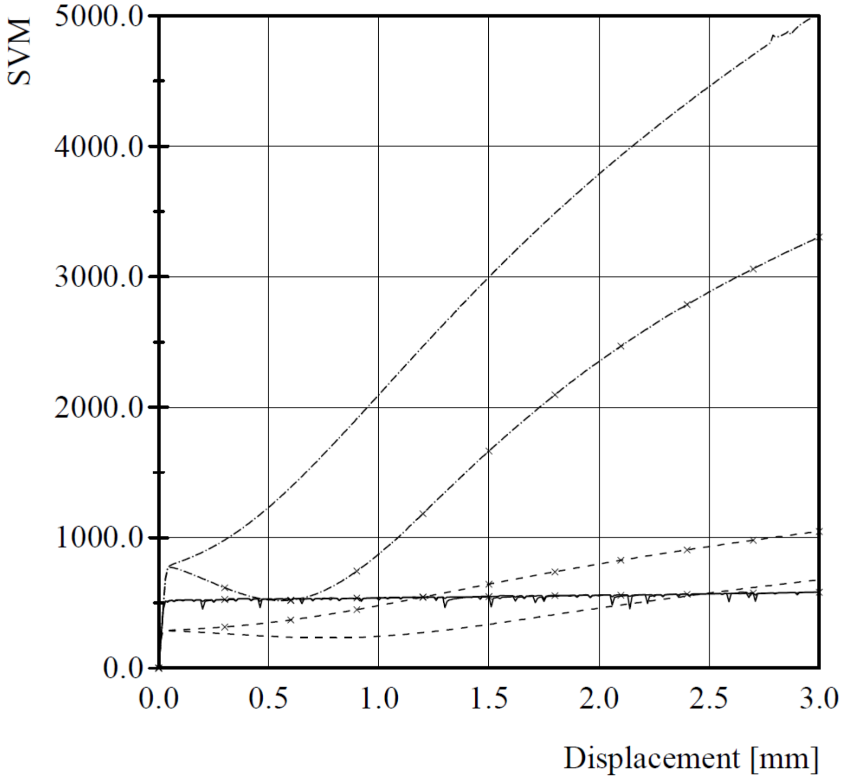


Figure 5.7. Rectangular strip under compression. Von Mises stress over compression displacement. Nodes A and B are represented by curves without marker and with cross. Solid line represents isotropic $\varrho_1 = \sqrt{3}$, dashed-dotted anisotropic $\varrho_2 = 2\sqrt{3}$ and dashed anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior.

Equivalent Plastic Strain - PEEQ

The deformed mesh and equivalent plastic strain can be obtained from Figure 5.8. The results are similarly distributed as the Von Mises stress. The symmetry axis for ϱ_2 and ϱ_3 equal those of the stress distribution. For the isotropic case ϱ_1 an uniformly distributed equivalent plastic strain can be obtained. Detailed information about the equivalent plastic strain for the nodes A and B yields Figure 5.9.

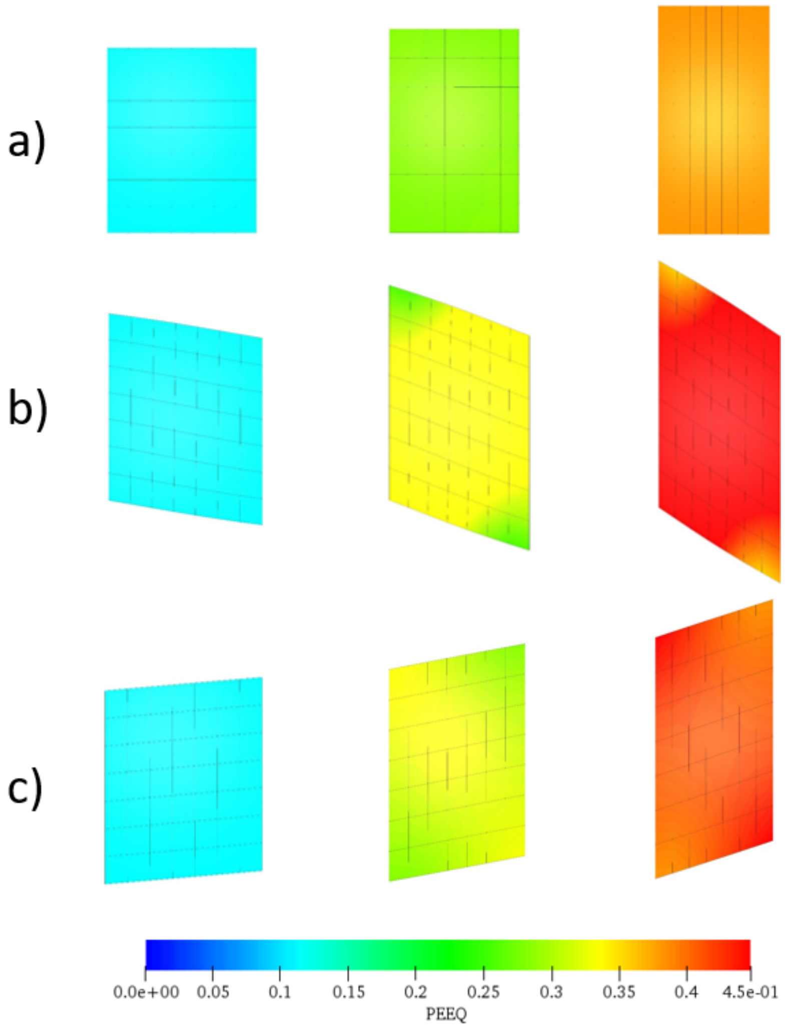


Figure 5.8. Rectangular strip under compression. Deformed meshes and equivalent plastic strain as shown in color bar for a) isotropic $\varrho_1 = \sqrt{3}$, b) anisotropic $\varrho_2 = 2\sqrt{3}$ and c) anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior. First column: top view for global iteration step $i = 100$. Second column: top view for $i = 200$. Third column: top view for $i = 300$.

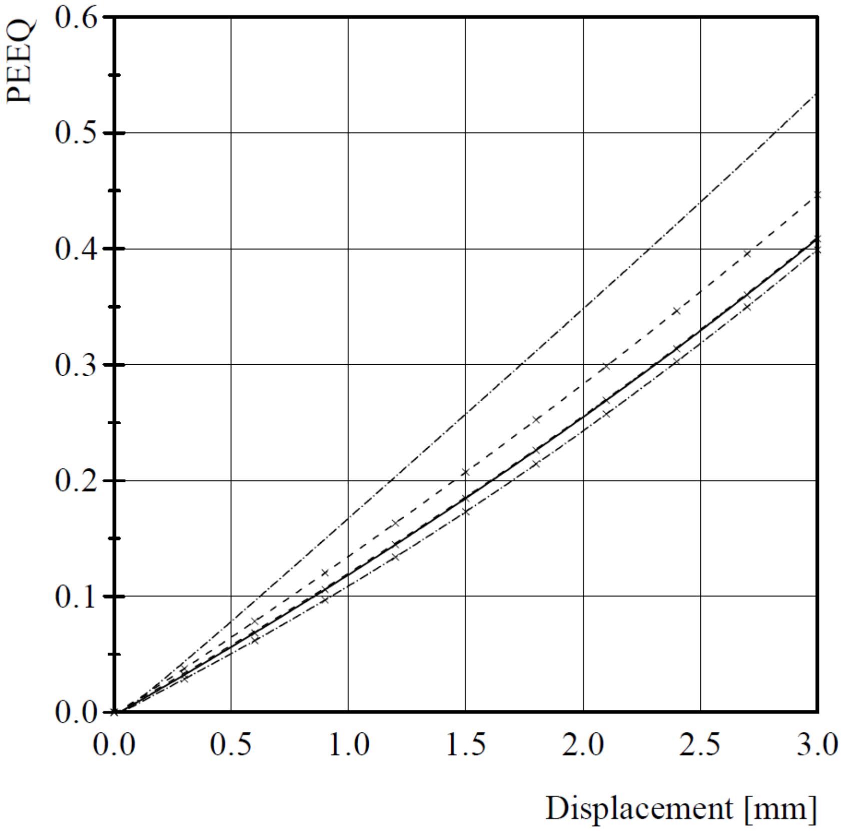


Figure 5.9. Rectangular strip under compression. Equivalent plastic strain over compression displacement. Nodes A and B are represented by curves without marker and with cross. Solid line represents isotropic $\varrho_1 = \sqrt{3}$, dashed-dotted anisotropic $\varrho_2 = 2\sqrt{3}$ and dashed anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior.

Table 5.5 shows the convergence of the unbalanced energy for the calculations with different material behavior. The table shows the unbalanced energy for timesteps $t = \{200, 250, 300\}$ over the iteration counter i.

Table 5.5. Rectangular strip under compression. Convergence of the unbalanced energy for a) isotropic $\varrho_1 = \sqrt{3}$, b) anisotropic $\varrho_2 = 2\sqrt{3}$ and c) anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior. Timestep $t = \{200, 250, 300\}$. Global iteration counter i.

	a) $\varrho_1 = \sqrt{3}$			b) $\varrho_2 = 2\sqrt{3}$			c) $\varrho_3 = 0.5\sqrt{3}$		
i/t	200	250	300	200	250	300	200	250	300
1	4.6e+02	5.1e+02	5.7e+02	4.6e+02	5.1e+02	5.8e+02	4.6e+02	5.1e+02	5.7e+02
2	9.8e+00	1.1e+01	1.3e+01	1.3e+01	1.4e+01	1.5e+01	1.1e+01	1.2e+01	1.4e+01
3	4.8e-03	5.6e-03	6.7e-03	1.0e-01	9.4e-02	8.6e-02	2.6e-02	2.9e-02	3.2e-02
4	7.0e-10	9.6e-10	1.1e-09	4.7e-03	2.4e-03	7.2e-03	1.6e-03	1.6e-03	4.5e-04
5	2.3e-23	3.9e-12	3.5e-23	1.6e-03	1.2e-03	3.3e-04	8.7e-04	9.5e-04	3.5e-04
6		2.3e-12		3.1e-05	5.0e-05	2.2e-04	7.3e-05	1.5e-04	7.8e-05
7		6.2e-13		5.5e-07	3.4e-07	2.3e-06	3.1e-07	2.2e-07	1.3e-07
8		5.3e-15		2.4e-09	6.9e-09	5.4e-09	5.1e-18	4.8e-18	1.2e-10
9		4.5e-18		4.5e-12	3.2e-10	1.3e-11	1.2e-23	1.4e-23	2.8e-13
10		1.9e-22		1.2e-15	2.8e-13	3.9e-15			3.9e-17
11				1.0e-23	3.1e-15	1.3e-23			2.0e-23
12					7.9e-23				

5.2.3. Results - Tension Load

Displacements

The deformed mesh and displacement for the different material configurations is shown in Figure 5.10. For the isotropic case ϱ_1 the body under consideration reduces its height without shear modes. The maximum displacement yields the point A. The other two material configurations deform with shear modes. Those shear modes yield the opposite behavior as obtained for the compression load. For ϱ_2 the body deforms in y - direction and shows a shear movement in positive y - direction. In the case ϱ_3 the shear movement changes orientation to the negative y - direction.

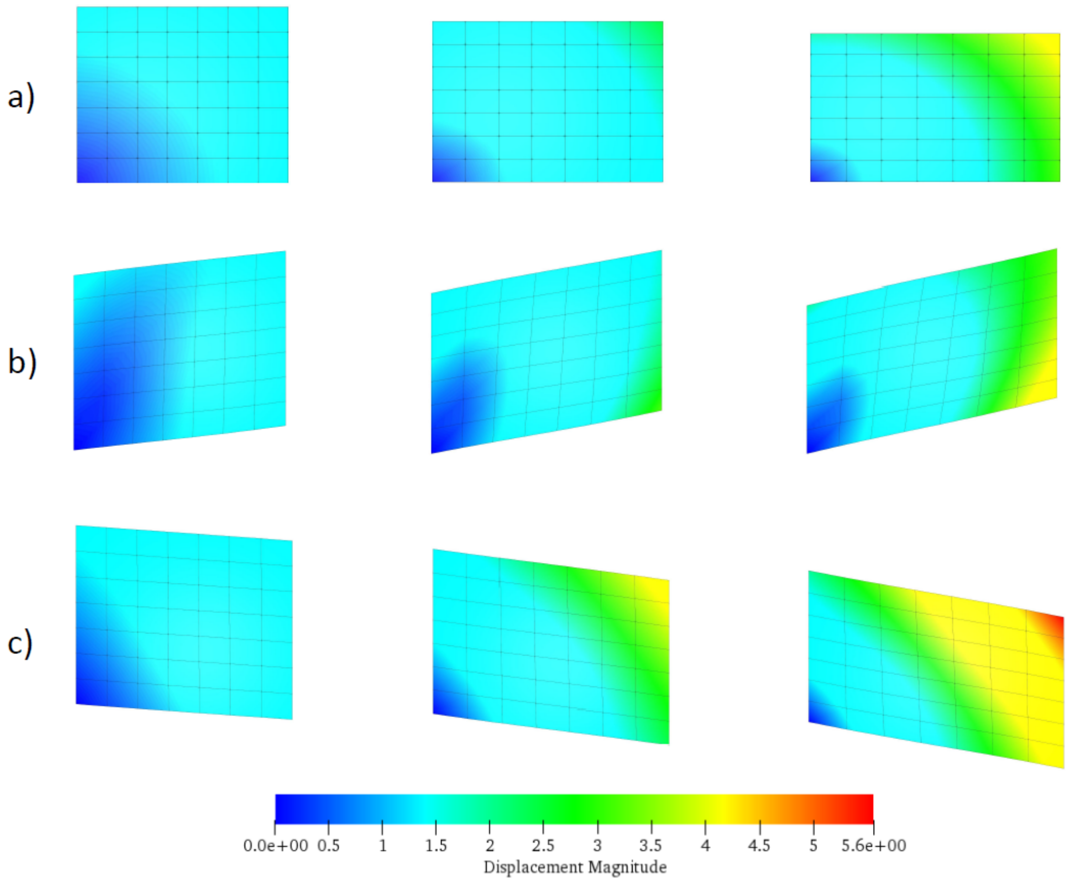


Figure 5.10. Rectangular strip under tension. Deformed meshes and displacements as shown in color bar for a) isotropic $\varrho_1 = \sqrt{3}$, b) anisotropic $\varrho_2 = 2\sqrt{3}$ and c) anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior. First column: top view for global iteration step $i = 100$. Second column: top view for $i = 200$. Third column: top view for $i = 300$.

Figure 5.11 shows the displacement v in y - direction of the two reference nodes A and B for the different material configurations.

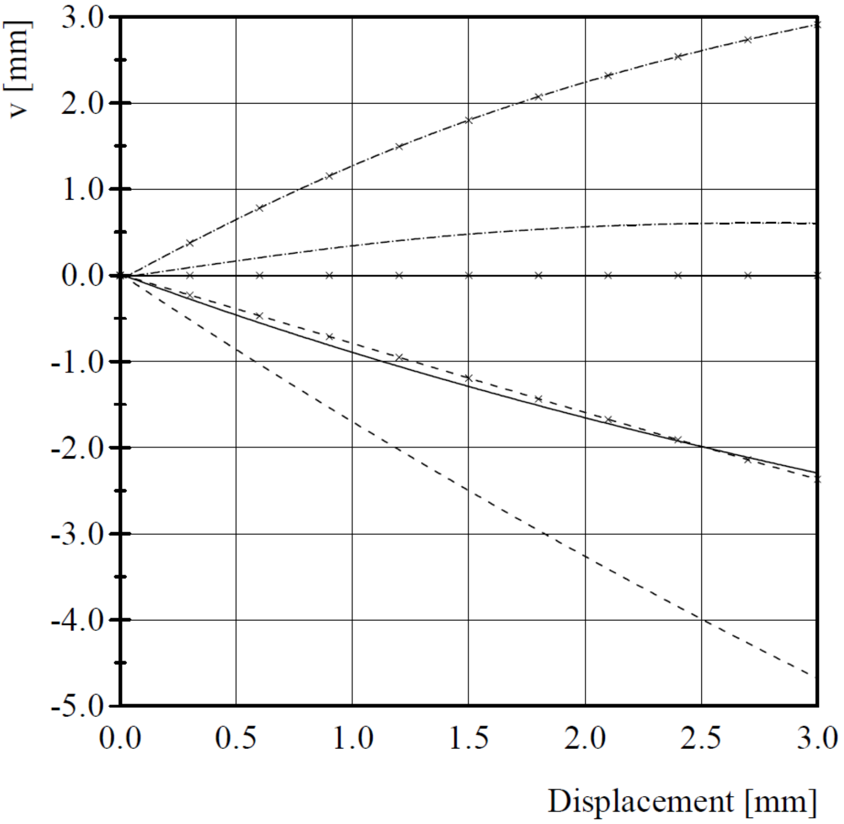


Figure 5.11. Rectangular strip under tension. Displacements v in y-direction over tension displacement. Nodes A and B are represented by curves without marker and with cross. Solid line represents isotropic $\varrho_1 = \sqrt{3}$, dashed-dotted anisotropic $\varrho_2 = 2\sqrt{3}$ and dashed anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior.

Von Mises Stress

The deformed mesh and Von Mises stress distribution are shown in Figure 5.12. The stress is equally distributed in the case of isotropic material behavior ϱ_1. The case ϱ_2 shows a symmetric distribution and maximal stresses in the top left and bottom right (node B) corner. This corresponds to the soft material axis and shows a different trend as for the compression load. This behavior was expected. For the case ϱ_3 a symmetric distribution can be obtained. The maximum stress can be obtained at the bottom left and top right (node A) corner. A detailed representation of the Von Mises stress for the nodes A and B yields Figure 5.13.

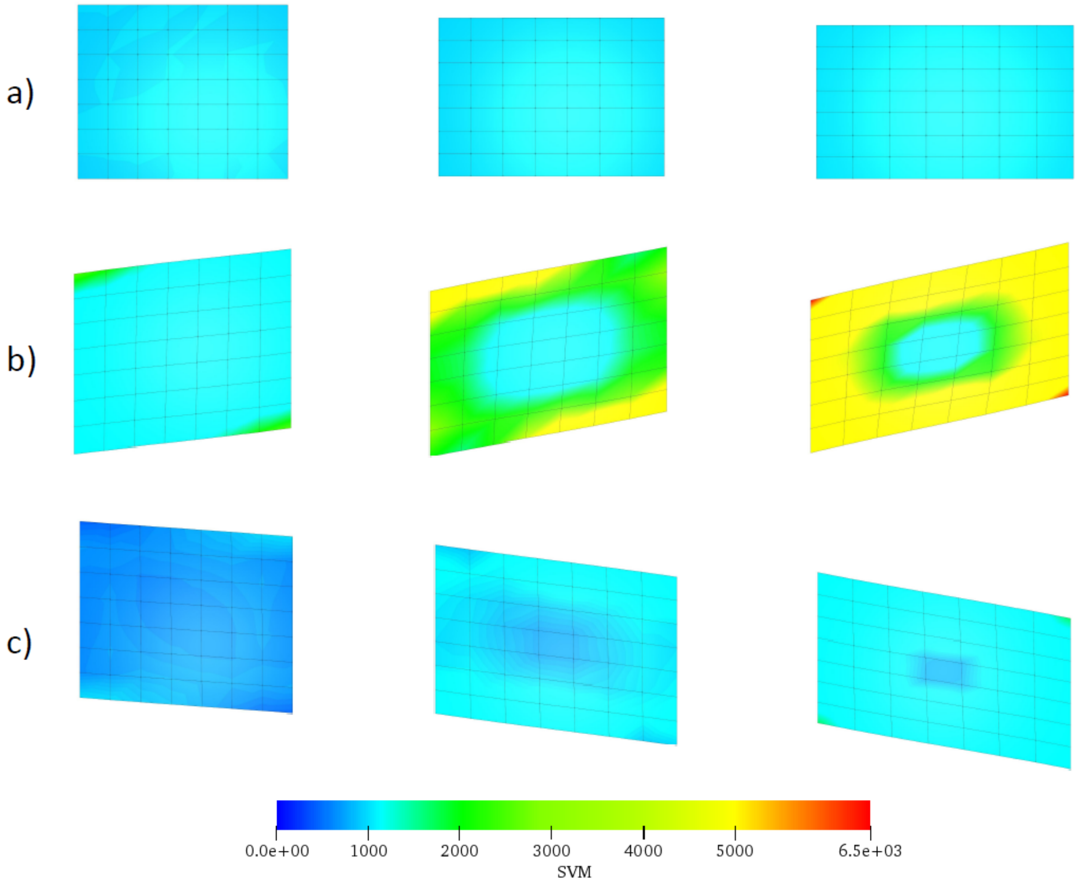


Figure 5.12. Rectangular strip under tension. Deformed meshes and Von Mises stress as shown in color bar for a) isotropic $\varrho_1 = \sqrt{3}$, b) anisotropic $\varrho_2 = 2\sqrt{3}$ and c) anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior. First column: top view for global iteration step $i = 100$. Second column: top view for $i = 200$. Third column: top view for $i = 300$.

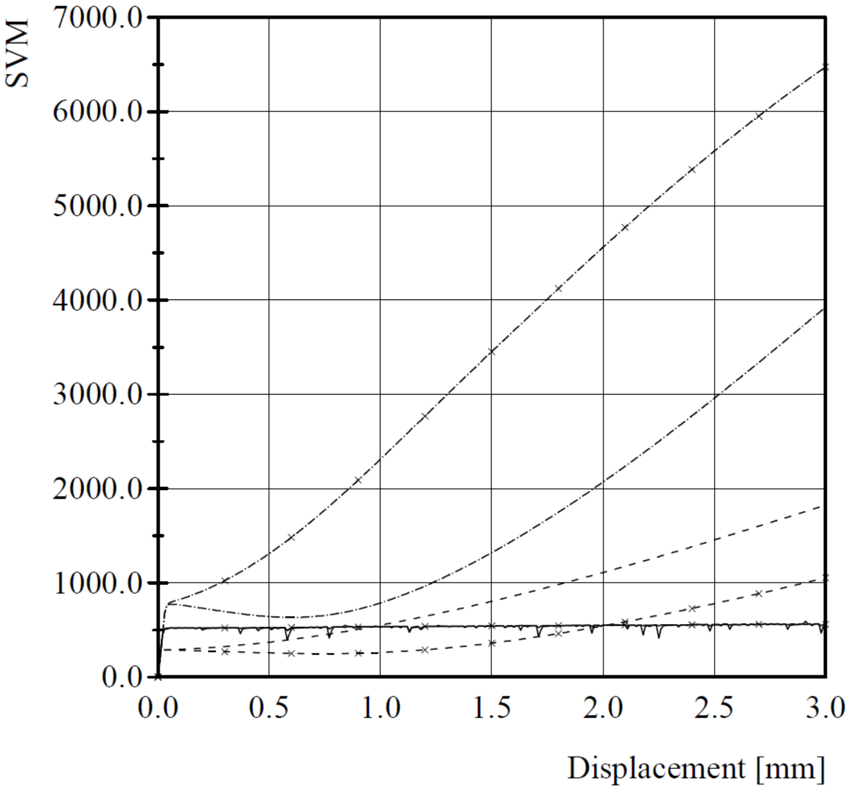


Figure 5.13. Rectangular strip under tension. Von Mises stress over tension displacement. Nodes A and B are represented by curves without marker and with cross. Solid line represents isotropic $\varrho_1 = \sqrt{3}$, dashed-dotted anisotropic $\varrho_2 = 2\sqrt{3}$ and dashed anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior.

Equivalent Plastic Strain - PEEQ

Figure 5.14 illustrates the deformed mesh and equivalent plastic strain. The obtained distributions are similar to those of the Von Mises stress. The symmetry axis for ϱ_2 and ϱ_3 remain the same. For the isotropic case ϱ_1 an equally distributed equivalent plastic strain results. Figure 5.15 yields detailed information about the equivalent plastic strain for the nodes A and B.

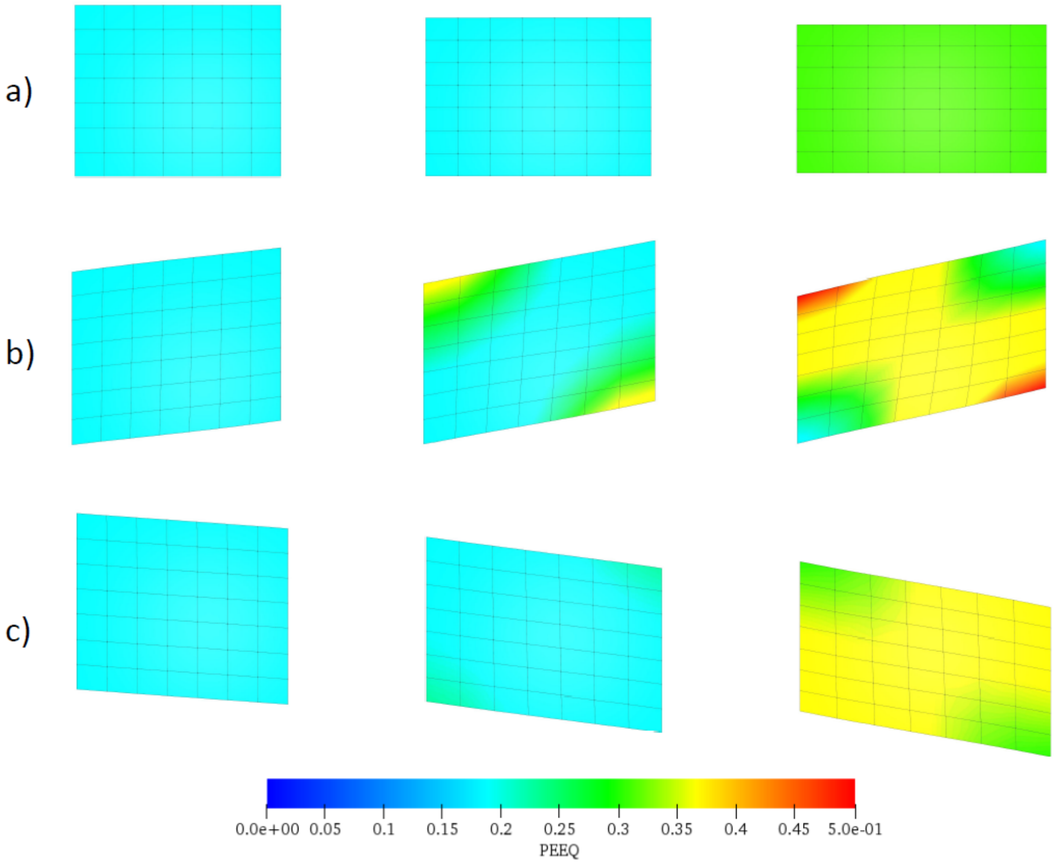


Figure 5.14. Rectangular strip under tension. Deformed meshes and equivalent plastic strain as shown in color bar for a) isotropic $\varrho_1 = \sqrt{3}$, b) anisotropic $\varrho_2 = 2\sqrt{3}$ and c) anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior. First column: top view for global iteration step $i = 100$. Second column: top view for $i = 200$. Third column: top view for $i = 300$.

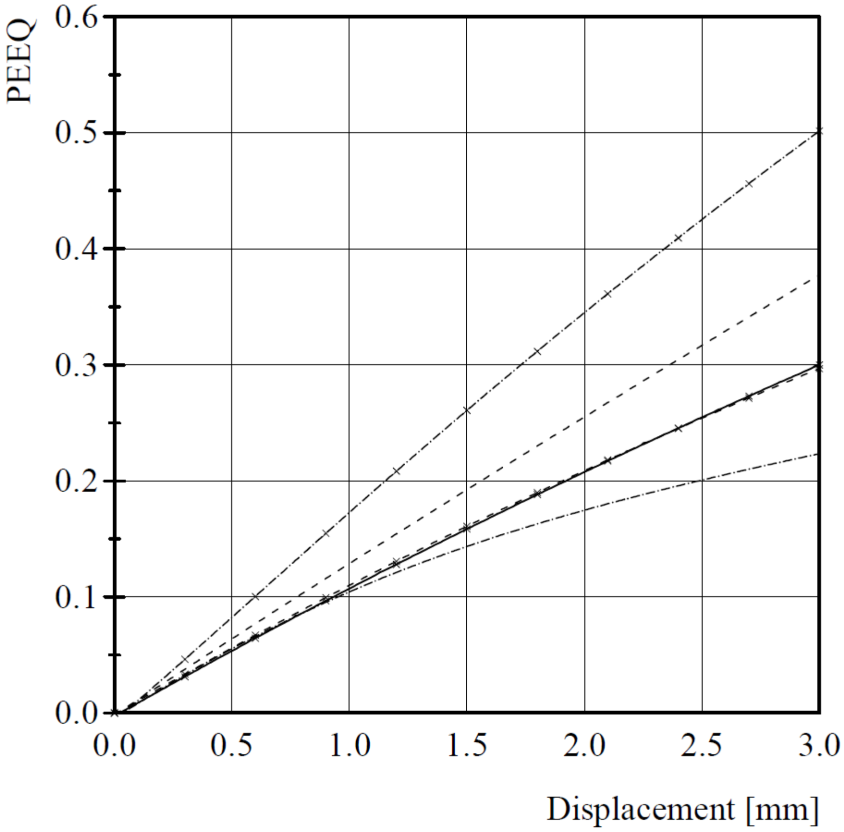


Figure 5.15. Rectangular strip under tension. Equivalent plastic strain over tension displacement. Nodes A and B are represented by curves without marker and with cross. Solid line represents isotropic $\varrho_1 = \sqrt{3}$, dashed-dotted anisotropic $\varrho_2 = 2\sqrt{3}$ and dashed anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior.

Table 5.6 yields the convergence of the unbalanced energy of the calculation algorithm. The table shows the unbalanced energy for the time steps $t = \{200, 250, 300\}$ with the global iteration counts i.

Table 5.6. Rectangular strip under tension. Convergence of the unbalanced energy for a) isotropic $\varrho_1 = \sqrt{3}$, b) anisotropic $\varrho_2 = 2\sqrt{3}$ and c) anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior. Timestep $t = \{200, 250, 300\}$. Global iteration counter i.

	a) $\varrho_1 = \sqrt{3}$			b) $\varrho_2 = 2\sqrt{3}$			c) $\varrho_3 = 0.5\sqrt{3}$		
i/t	200	250	300	200	250	300	200	250	300
1	2.6e+02	2.5e+02	2.4e+02	2.6e+02	2.5e+02	2.5e+02	2.6e+02	2.5e+02	2.4e+02
2	2.6e+00	3.5e+00	3.4e+00	5.9e+00	5.1e+00	4.4e+00	5.9e+00	5.4e+00	4.8e+00
3	2.4e-01	4.3e-03	4.6e-03	7.6e-02	6.0e-02	4.9e-02	3.2e-02	3.3e-02	3.1e-02
4	7.3e-02	4.8e-04	1.3e-03	8.8e-03	1.0e-02	9.7e-03	2.3e-06	6.7e-06	7.2e-05
5	1.1e-02	5.5e-05	5.0e-04	1.2e-02	1.6e-02	1.6e-02	1.2e-06	1.3e-06	1.8e-04
6	1.2e-04	6.4e-07	7.9e-06	2.3e-04	2.5e-04	1.9e-04	8.1e-07	4.6e-07	2.2e-05
7	1.4e-07	8.8e-09	7.3e-09	2.3e-06	3.9e-06	2.6e-06	2.0e-08	4.4e-09	3.9e-08
8	9.9e-18	2.7e-19	6.4e-23	8.3e-08	5.5e-07	4.8e-15	9.7e-11	5.9e-10	5.9e-10
9	1.1e-23	1.0e-23		1.3e-08	7.1e-10	6.5e-18	2.6e-12	7.5e-13	1.7e-12
10				1.6e-10	1.9e-12	1.6e-19	5.7e-14	1.2e-23	1.2e-23
11				5.8e-14	1.2e-15	7.7e-23	2.4e-16		
12				1.5e-23	1.4e-23		2.1e-19		
13					1.4e-23		1.4e-21		

5.3. Drawing of a Thin Circular Plate

This example shows the drawing of a thin circular plate which is a simple benchmark example to examine isotropic elastic-anisotropic plastic responses. A variation of the material parameters shows the capability of anisotropic material behavior for the derived material model. Therefore equal loads are applied on the geometry and the simulation outputs displacement, Von Mises stress and the equivalent plastic strain are compared. Similar investigations were performed for instance by Papadopulus & Lu [22], Miehe et al. [19], Löblein et al. [12] or Ulz [33].

5.3.1. Model of Computation

The circular plate measures a radius of 400 mm with a hole of 200 mm in diameter. The thickness of the plate is 10 mm. The calculation domain is simplified by the symmetry of the geometry. Therefore the quarter of the examined circular plate is modeled by introducing boundary conditions on the cutting edges as shown in Figure 5.16. A displacement $u = 75\,\mathrm{mm}$ is applied in radial direction to simulate a drawing process. The displacements is applied in displacement increments of $\Delta u = 0.1\,\mathrm{mm}$ within 750 global iteration steps. The orthotropic axis a_i with $i = \{1, 2, 3\}$ equals the global coordinate system. Figure 5.16 summarizes the aforementioned scope conditions. Buckling is prevented by supporting the bottom of the plate.

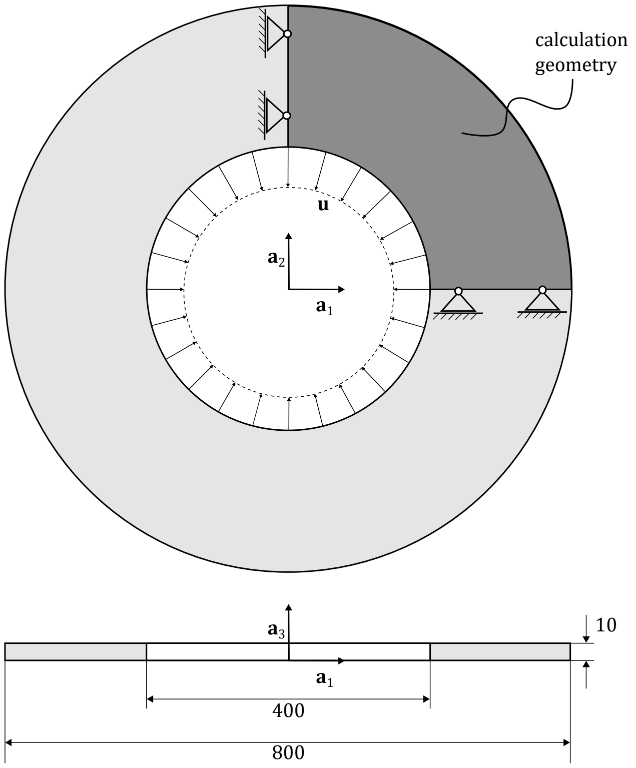


Figure 5.16. Drawing of a thin circular plate. Geometry of the calculation domain.

The mesh consists of 10x10 8-node trilinear hexahedral solid elements with a total number of 242 nodes. The discretization of the thickness direction of the plate is realized with one element row. The mesh can be obtained from Figure 5.17 where the considered nodes for the evaluation of the simulation results are labeled with the associated node ID. The x - and y - coordinates of the considered nodes can be obtained from Table 5.7.

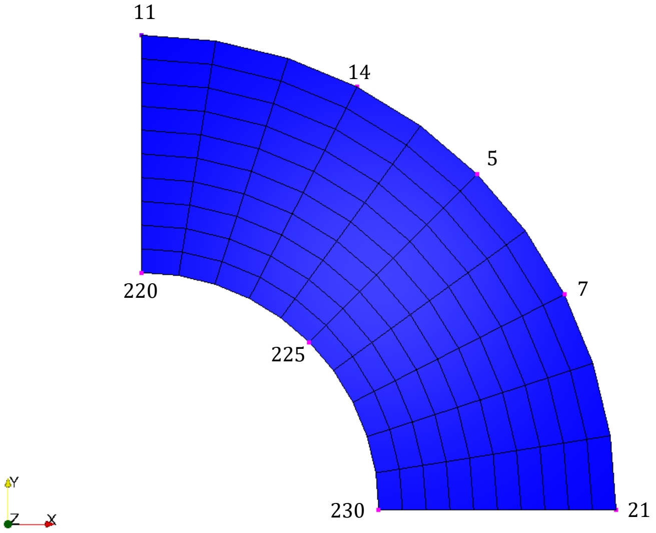


Figure 5.17. Drawing of a thin circular plate. Mesh with the considered nodes.

Table 5.7. Drawing of a thin circular plate. x - and y- coordinates of the considered nodes.

Node ID	x - coordinate	y - coordinate
220	0	200
225	$200/\sqrt{2}$	$200/\sqrt{2}$
230	200	0
11	0	400
14	181.6	356.4
5	$400/\sqrt{2}$	$400/\sqrt{2}$
7	356.4	181.6
21	400	0

In Table 5.8 the basic set of material properties is denoted. The first calculated set is the isotropic case which is associated with $\varrho_1 = \sqrt{3}$. The orthotropic material response for the other two simulation sets are defined by setting $\varrho_2 = 2\sqrt{3}$ and $\varrho_3 = 0.5\sqrt{3}$.

Table 5.8. Drawing of a thin circular plate. Material properties for the model of computation.

Parameter	Symbol	Value	Unit
Young's Modulus	E	210	GPa
Poisson's ratio	ν	0.3	-
Initial yield stress	y_0	500	MPa
Linear isotropic hardening	h	129.24	MPa

5.3.2. Results

Displacement

The deformed meshes and displacement field for the various material definitions differ significantly from each other. Figure 5.18 shows the direction independence in the isotropic case ϱ_1. In the case of ϱ_2 the result yields a minimal displacement in the direction of maximum shear stress at 45 degree. The result for ϱ_3 shows minimal displacement in the maximal normal stress direction associated with the vectors $\mathbf{a}_1$ and $\mathbf{a}_2$. Additionally the outer ring of the body deforms wavelike. This behavior is also called earing.

Detailed results for the x -, y - and radial displacement of the nodes 11, 5 and 21 from Figure 5.17 can be obtained from Figure 5.19. The radial displacement shows for the isotropic case ϱ_1 the same result for the 3 nodes. In the case of ϱ_2 the nodes 11 and 21 show the same radial displacement. Node 5 has the lowest displacement. This correlates to the aforementioned symmetry to the 45 degree shear stress maximum. The result for ϱ_3 yields the same radial displacement for the nodes 11 and 21. Node 5 has the highest displacement. This correlates to the results obtained in Figure 5.18.

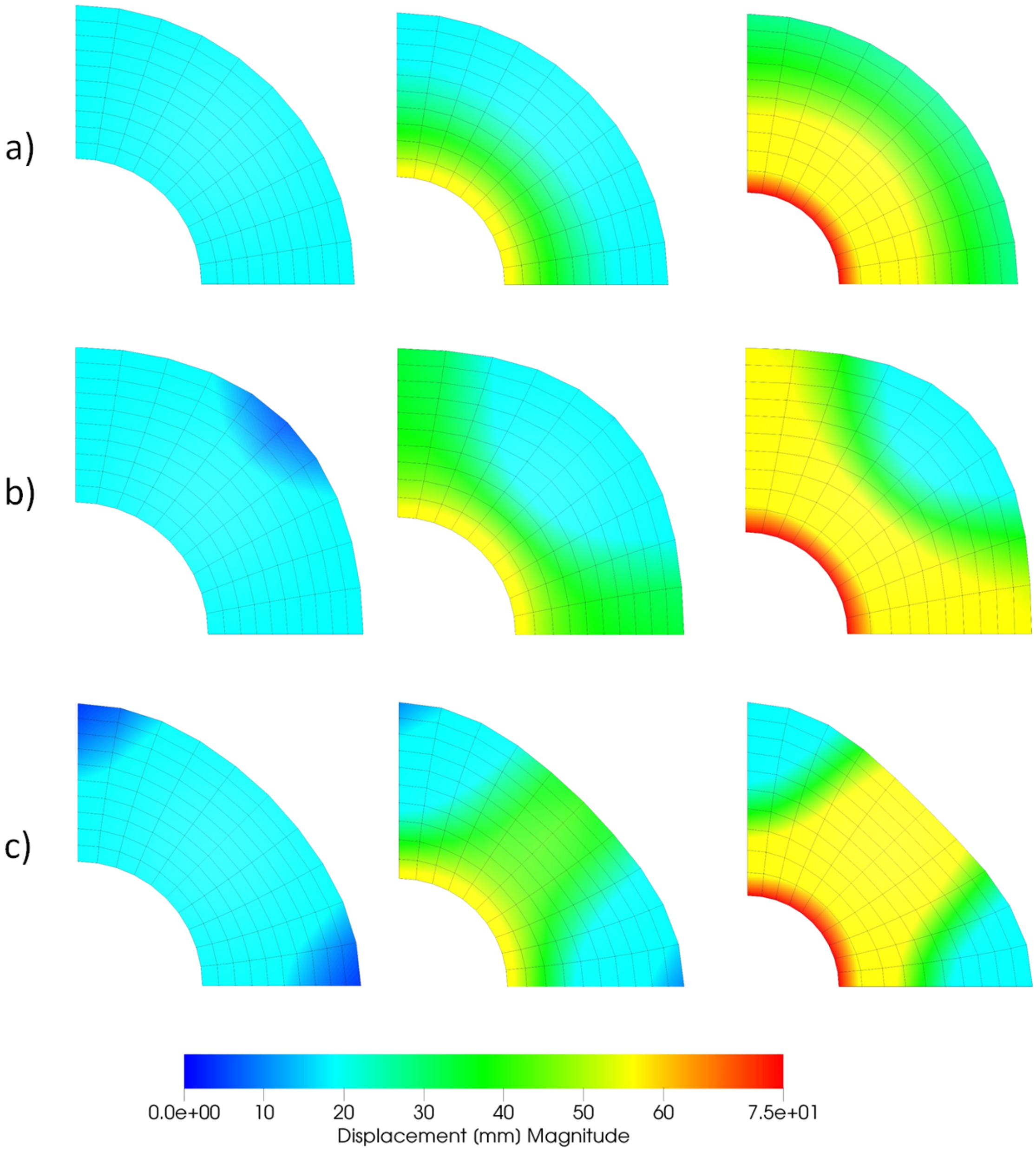


Figure 5.18. Drawing of a thin circular plate. Deformed meshes and displacement as shown in color bar for a) isotropic $\varrho_1 = \sqrt{3}$, b) anisotropic $\varrho_2 = 2\sqrt{3}$ and c) anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior. First column: top view for global iteration step $i = 250$. Second column: top view for $i = 500$. Third column: top view for $i = 750$.

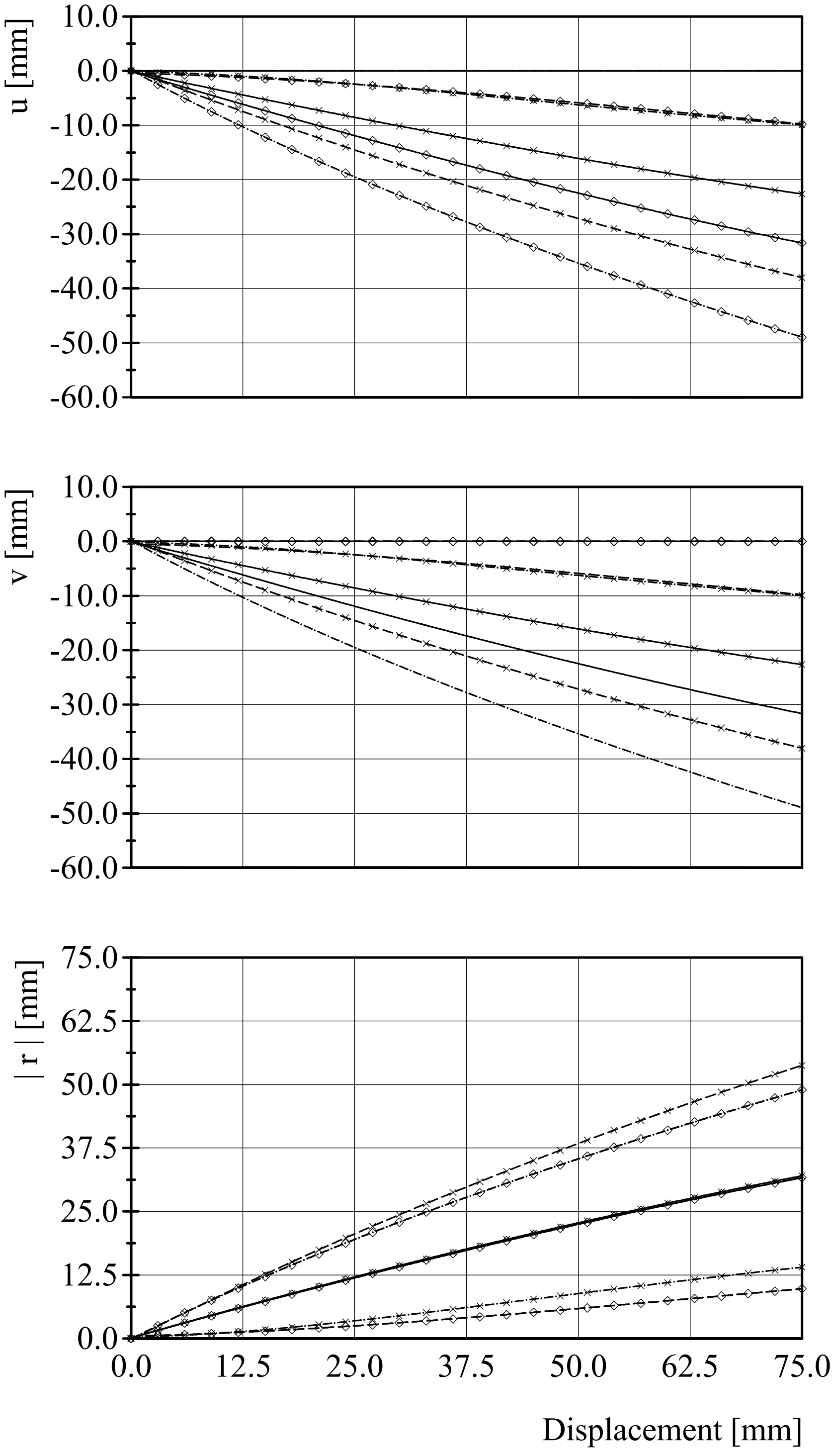


Figure 5.19. Drawing of a thin circular plate. Displacement u in x-direction, v in y-direction and radial displacement |r| over drawing displacement. Nodes 11, 5 and 21 are represented by curves without marker, with cross and with rhombus. Solid line represents isotropic $\varrho_1 = \sqrt{3}$, dashed-dotted anisotropic $\varrho_2 = 2\sqrt{3}$ and dashed anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior.

Von Mises Stress

The deformed mesh and Von Mises stress can be obtained from Figure 5.21. In the case of isotropy ϱ_1 no preferred directions can be obtained. The results of ϱ_2 and ϱ_3 show symmetric behavior to the 45 degree axis of maximum shear stresses. The Von Mises stress for ϱ_2 has its maximum on the inner ring under 45 degrees. The case ϱ_3 yields maximal values on the inner ring for the direction $\mathbf{a}_1$ and $\mathbf{a}_2$. Detailed results for the nodes 11, 5 and 21 can be obtained from Figure 5.20. The results show that no symmetry can be achieved for the corresponding nodes.

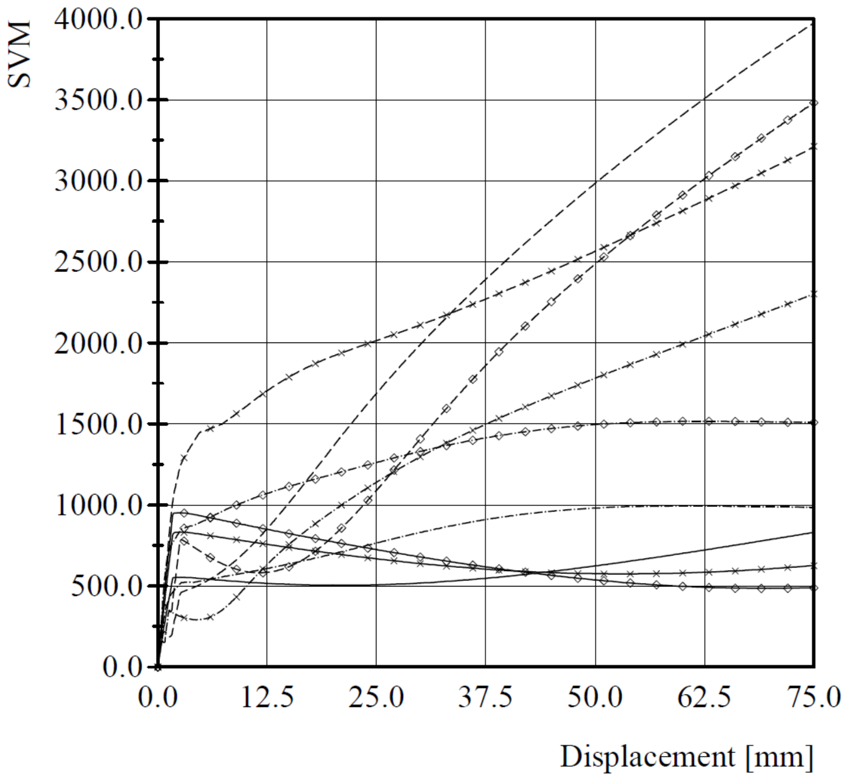


Figure 5.20. Drawing of a thin circular plate. Von Mises stress over drawing displacement. Nodes 11, 5 and 21 are represented by curves without marker, with cross and with rhombus. Solid line represents isotropic $\varrho_1 = \sqrt{3}$, dashed-dotted anisotropic $\varrho_2 = 2\sqrt{3}$ and dashed anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior.

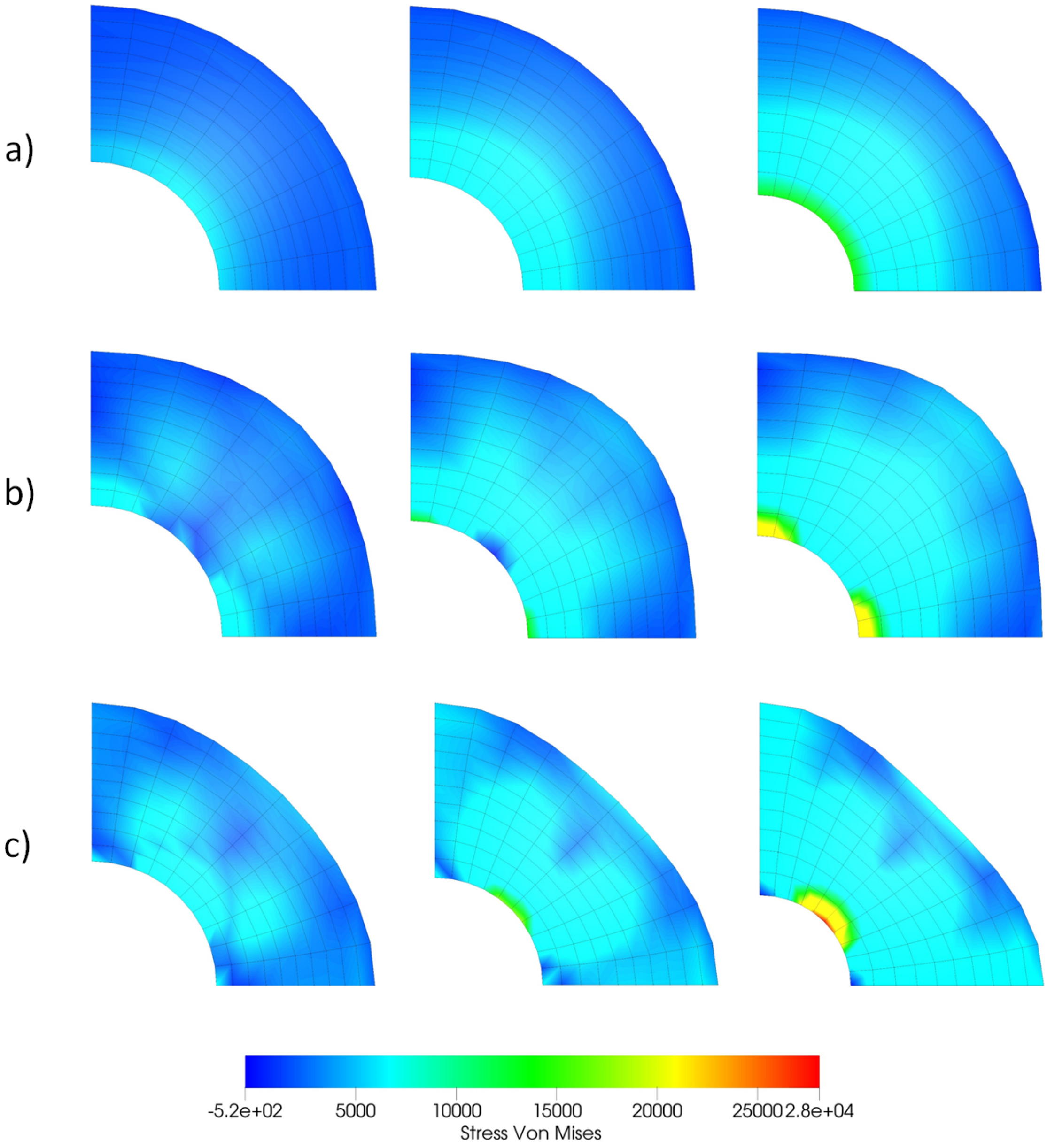


Figure 5.21. Drawing of a thin circular plate. Deformed mesh and Von Mises Stress as shown in color bar for a) isotropic $\varrho_1 = \sqrt{3}$, b) anisotropic $\varrho_2 = 2\sqrt{3}$ and c) anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior. First column: top view for global iteration step $i = 250$. Second column: top view for $i = 500$. Third column: top view for $i = 750$.

Equivalent Plastic Strain - PEEQ
The deformed mesh and equivalent plastic strain can be obtained from Figure 5.23. The same effects as for the Von Mises Stress can be found. The maximal values are on the inner ring of the circle and correlate to the associated maxima for Von Mises stress. Detailed results for the nodes 11, 5 and 21 can be obtained from Figure 5.22. In the case of ϱ_2 the material is softer in the 45 degrees direction. Therefore the maximal values can be obtained at node 225. For ϱ_3 the material is softer in the direction of $\mathbf{a}_1$ and $\mathbf{a}_2$. Hence the maximal values can be found at the nodes 220 and 230.

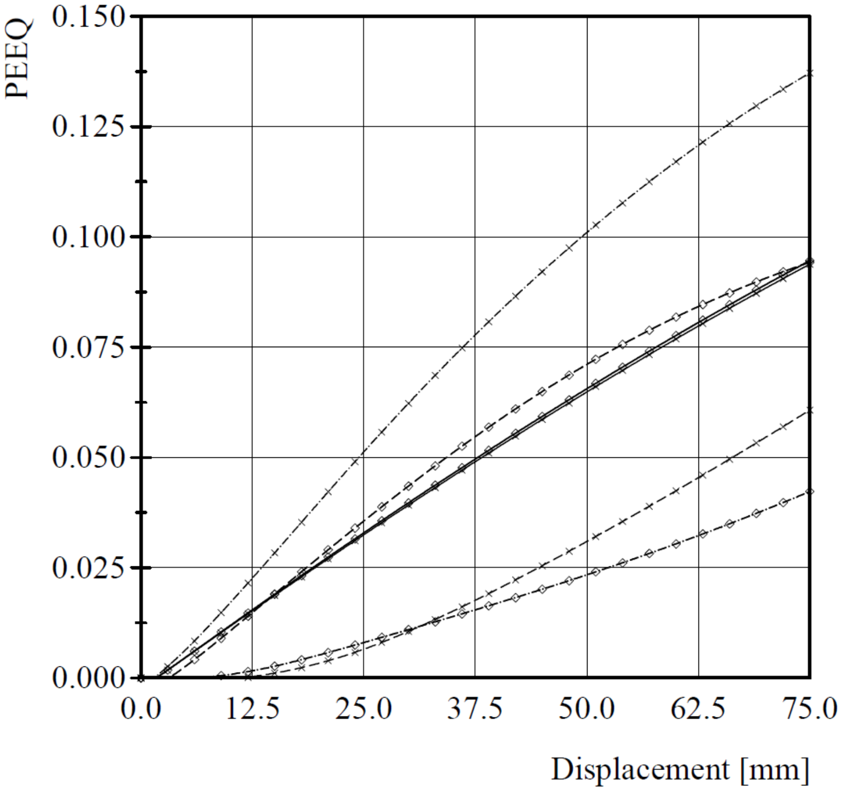


Figure 5.22. Drawing of a thin circular plate. Equivalent plastic strain over drawing displacement. Nodes 11, 5 and 21 are represented by curves without marker, with cross and with rhombus. Solid line represents isotropic $\varrho_1 = \sqrt{3}$, dashed-dotted anisotropic $\varrho_2 = 2\sqrt{3}$ and dashed anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior.

The convergence of the unbalanced energy for the calculations of the different material behaviors can be taken from Table 5.9. The unbalanced energy for the timesteps $t = \{650, 700, 750\}$ are presented over the global iterations i. The calculation procedure shows good convergence for all three cases.

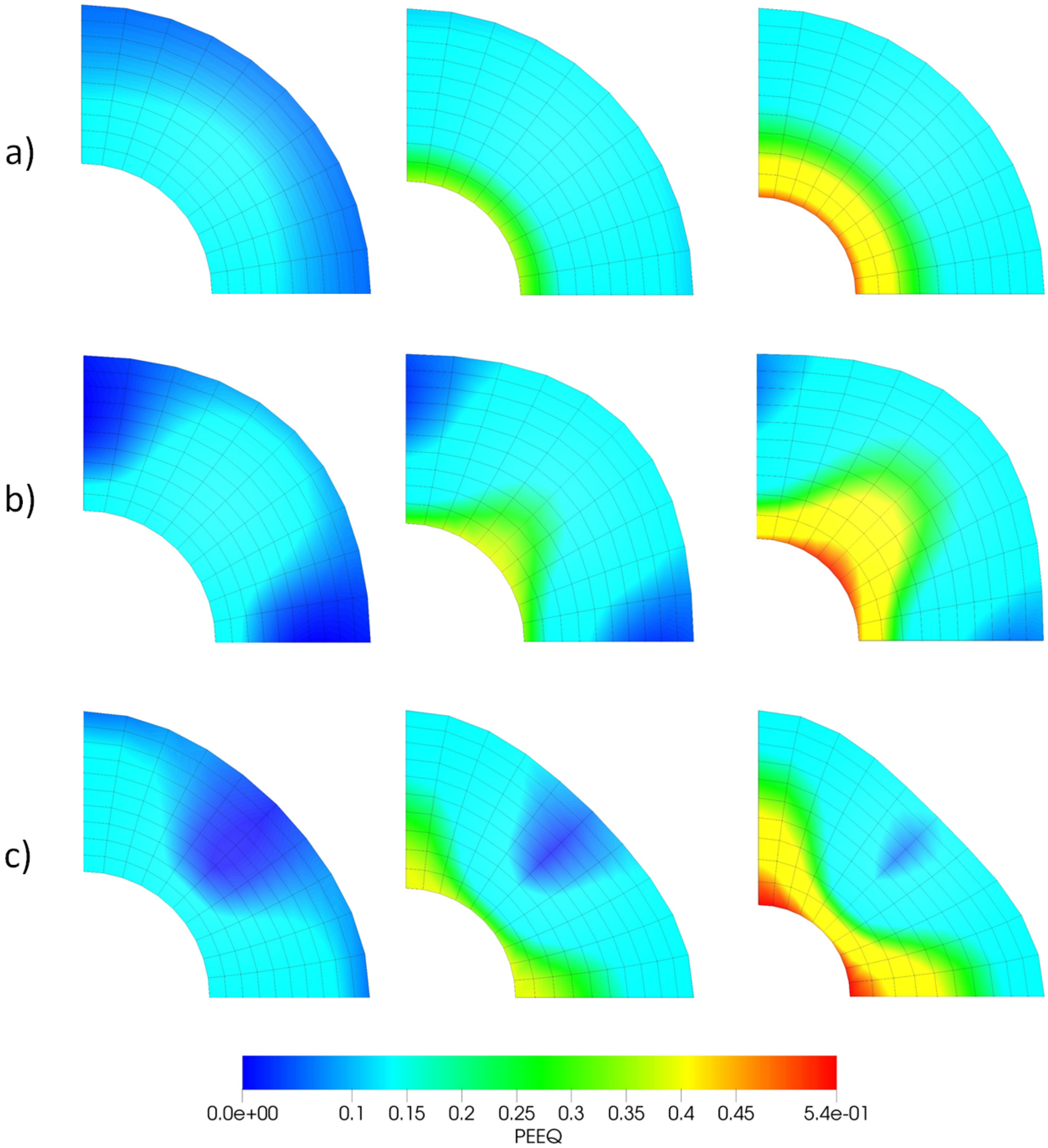


Figure 5.23. Drawing of a thin circular plate. Deformed mesh and equivalent plastic strain shown in color bar for a) isotropic $\varrho_1 = \sqrt{3}$, b) anisotropic $\varrho_2 = 2\sqrt{3}$ and c) anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior. First column: top view for global iteration step $i = 250$. Second column: $i = 500$. Third column: $i = 750$.

Table 5.9. Drawing of a thin circular plate. Convergence of the unbalanced energy for a) isotropic $\varrho_1 = \sqrt{3}$, b) anisotropic $\varrho_2 = 2\sqrt{3}$ and c) anisotropic $\varrho_3 = 0.5\sqrt{3}$ material behavior. Timestep $t = \{650, 700, 750\}$. Global iteration counter i.

	a) $\varrho_1 = \sqrt{3}$			b) $\varrho_2 = 2\sqrt{3}$			c) $\varrho_3 = 0.5\sqrt{3}$		
i/t	650	700	750	650	700	750	650	700	750
1	5.7e+05	5.6e+05	5.4e+05	5.8e+05	5.8e+05	5.6e+05	5.7e+05	5.5e+05	5.3e+05
2	1.0e+03	9.9e+02	9.4e+02	2.3e+03	2.2e+03	2.1e+03	2.2e+03	2.1e+03	2.0e+03
3	1.2e-03	1.0e-03	8.6e-04	4.5e+02	4.5e+02	3.7e+02	4.0e+02	3.5e+02	3.1e+02
4	9.6e-15	3.7e-09	4.0e-08	1.2e+02	1.2e+02	1.0e+02	1.1e+02	9.8e+01	8.1e+01
5	2.8e-17	1.4e-09	2.7e-08	1.4e+01	1.3e+01	1.3e+01	1.1e+01	1.5e+01	8.2e+00
6	8.6e-18	5.0e-10	1.1e-08	6.9e-02	6.7e-02	8.1e-02	4.2e-02	7.8e-02	1.5e-01
7	1.3e-18	2.4e-14	2.6e-12	3.3e-13	3.1e-13	7.9e-13	1.7e-13	4.5e-13	4.4e-05
8	5.7e-19	1.1e-18	5.6e-19	6.0e-19	5.7e-19	5.8e-19	5.8e-19	5.4e-19	3.6e-16
9		6.5e-19							5.0e-19

6. Conclusion

A local iteration scheme for finite, anisotropic plasticity in the logarithmic Lagrangian strain space was derived. The strain measure is decomposed into an elastic and plastic part as proposed by Green & Naghdi. This approach allows the usage of model structures identical to those of the small strain theory. A strain-energy function is defined in terms of the elastic logarithmic Lagrangian strain and a dissipation potential. Dissipation is introduced by plastic deformation with isotropic hardening. Internal variables are defined in the logarithmic Lagrangian strain space. A transformation of the internal variables from logarithmic Lagrangian back to Lagrangian strain space is introduced to map back the internal variables to the large-strain scope. Anisotropy is modeled by the introduction of material symmetry groups. A constant fourth-order Hill tensor describes initial anisotropy.

An 8-node hexahedral trilinear element was used to calculate representative numerical simulations. The convergence test in section 5.1 demonstrates the convergence behavior of the computational implementation for a single element. The implemented Newton algorithm converges in a quadratic manner. This result can be interpreted as a correct implementation of the proposed model. A rectangular strip under compression and tension load in section 5.2 shows the capability of anisotropic behavior of the implemented model. Different yield stresses for the geometry axis generate an anisotropic material configuration. The numerical simulation results demonstrate a plausible tendency for the deformation behavior and show the demanded anisotropy. Nevertheless the used element type is too stiff to describe plastic deformations. This issue can be addressed by using another element type or refinement of the finite element mesh. The focus within this work laid on the correct implementation of the proposed model. A deep drawing process of a thin rolled circular plate is simulated in section 5.3 by simplifying the problem and using the symmetry of the calculation domain. Different yield stresses for the geometry axis yield differing deformation behavior. Isotropic material leads to a consistent radial deformation on the inner and outer ring of the body under consideration. The numerical simulations for anisotropic material reveal a different deformation behavior for the inner and outer ring. The "earing" phenomena appears and demonstrates the utility of the implemented model. In this case the aforementioned issue of too stiff element definitions leads to non-prominent "earing".

In future the proposed model has to be tested with other element definitions that are not as stiff as the herein demonstrated element type. This can be done for instance by using 8-node assumed-additively enhanced hexahedral elements. In further consequence the implemented model can be expanded by rate-dependent material behavior by replacing the constant Lagrange multiplier with a constitutive equation.

Summarizing a material description in the logarithmic Lagrangian strain-space is presented which yields a simpler structure compared to the multiplicative approach. A drawback is the numerical computation-intensive transformation from the logarithmic Lagrangian strain space back to Lagrangian strain-space which is outweighed by the advantages of the demonstrated approach.

Appendix A. Derivations

This appendix reveals different derivations which are used in the established constitutive equations from chapter 4. The internal variables $\mathbf{E}^P$ and ξ are conjugated to the internal forces $\mathbf{T}^P$ and ζ. The logarithmic Lagrangian strain tensor $\mathbf{E}^e$ is defined as $\mathbf{E}^e = \mathbf{E} - \mathbf{E}^P$ where $\mathbf{E}$ and $\mathbf{E}^P$ describe the total and plastic logarithmic Lagrangian strain.
The strain-energy function is defined as:

$$\psi = \frac{\lambda}{2}(I_{\mathbf{E}^e})^2 + \mu II_{\mathbf{E}^e} + \frac{1}{2}h\xi^2$$

where $I_{\mathbf{E}^e} = \mathrm{tr}(\mathbf{E}^e)$ and $II_{\mathbf{E}^e} = \mathbf{E}^e : \mathbf{E}^e$ describe the first and second invariant of the elastic logarithmic Lagrangian strain tensor $\mathbf{E}^e$ and λ, μ denote the Lamé parameters. Furthermore h describes the isotropic hardening parameter.
The level set function ϕ is defined as:

$$\phi = ||\mathbf{T}^P||_{\mathbb{H}} + \sqrt{\frac{2}{2}}\zeta$$

with $|| \bullet ||_{\mathbb{H}} = \sqrt{(\bullet) : \mathbb{H} : (\bullet)}$ where $\mathbb{H}$ describes a Hill function with the material symmetry groups used to define the initial anisotropy axis. The constant fourth-order Hill tensor $\mathbb{H}$ has minor and major symmetries yielding $\mathbb{H}_{IJKL} = \mathbb{H}_{KLIJ} = \mathbb{H}_{JIKL} = \mathbb{H}_{IJLK}$.

1.) Derivation of $\partial_{\mathbf{T}^P}\phi$:

$$\partial_{\mathbf{T}^P}\phi = \frac{\partial\phi}{\partial\mathbf{T}^P} = \frac{\partial(||\mathbf{T}^P||_{\mathbb{H}} + \sqrt{\frac{2}{3}}\zeta)}{\partial\mathbf{T}^P} = \frac{\partial(\sqrt{\mathbf{T}^P : \mathbb{H} : \mathbf{T}^P} + \sqrt{\frac{2}{3}}\zeta)}{\partial\mathbf{T}^P}$$

$$= \frac{\partial(\sqrt{T^P_{ij}\mathbb{H}_{ijkl}T^P_{kl}} + \sqrt{\frac{2}{3}}\zeta)}{\partial T^P_{mn}} = \frac{\delta_{im}\delta_{jn}\mathbb{H}_{ijkl}T^P_{kl} + T^P_{ij}\mathbb{H}_{ijkl}\delta_{km}\delta_{ln}}{2\sqrt{T^P_{ij}\mathbb{H}_{ijkl}T^P_{kl}}}$$

$$= \frac{2\mathbb{H}_{ijkl}T^P_{kl}}{2\sqrt{T^P_{ij}\mathbb{H}_{ijkl}T^P_{kl}}} = \frac{\mathbb{H} : \mathbf{T}^P}{||\mathbf{T}^P||_{\mathbb{H}}} \tag{A.1}$$

2.) Derivation of $\partial^2_{\mathbf{T}^P\mathbf{T}^P}\phi$:

$$\partial^2_{\mathbf{T}^P\mathbf{T}^P}\phi = \frac{\partial^2\phi}{\partial\mathbf{T}^P\partial\mathbf{T}^P} = \frac{\partial}{\partial\mathbf{T}^P}\frac{\mathbb{H}:\mathbf{T}^P}{||\mathbf{T}^P||_{\mathbb{H}}} = \frac{\partial}{\partial T^P_{qr}}\frac{\mathbb{H}_{ijkl}T^P_{kl}}{\sqrt{T^P_{mn}\mathbb{H}_{mnst}T^P_{st}}}$$

$$= \frac{\mathbb{H}_{ijkl}\delta_{kq}\delta_{lr}\sqrt{T^P_{mn}\mathbb{H}_{mnst}T^P_{st}} - \frac{\mathbb{H}_{mnst}T^P_{st}}{\sqrt{T^P_{mn}\mathbb{H}_{mnst}T^P_{st}}}\mathbb{H}_{ijkl}T^P_{kl}}{(\sqrt{T^P_{mn}\mathbb{H}_{mnst}T^P_{st}})^2}$$

$$= \frac{\mathbb{H}_{ijqr}\sqrt{T^P_{mn}\mathbb{H}_{mnst}T^P_{st}}}{(\sqrt{T^P_{mn}\mathbb{H}_{mnst}T^P_{st}})^2} - \frac{\mathbb{H}_{mnst}T^P_{st}\mathbb{H}_{ijkl}T^P_{kl}}{(\sqrt{T^P_{mn}\mathbb{H}_{mnst}T^P_{st}})^3}$$

$$= \frac{\mathbb{H}_{ijqr}}{\sqrt{T^P_{mn}\mathbb{H}_{mnst}T^P_{st}}} - \frac{\mathbb{H}_{mnst}T^P_{st}\mathbb{H}_{ijkl}T^P_{kl}}{(\sqrt{T^P_{mn}\mathbb{H}_{mnst}T^P_{st}})^3}$$

$$= \frac{\mathbb{H}}{||\mathbf{T}^P||_{\mathbb{H}}} - \frac{[\mathbb{H}:\mathbf{T}^P]\otimes[\mathbb{H}:\mathbf{T}^P]}{||\mathbf{T}^P||^3_{\mathbb{H}}} \tag{A.2}$$

3.) Derivation of $\partial_\zeta\phi$:

$$\partial_\zeta\phi = \frac{\partial\phi}{\partial\zeta} = \frac{\partial(||\mathbf{T}^P||_{\mathbb{H}} + \sqrt{\frac{2}{3}}\zeta)}{\partial\zeta} = \sqrt{\frac{2}{3}} \tag{A.3}$$

4.) Derivation of $\partial^2_{\zeta\zeta}\phi$:

$$\partial^2_{\zeta\zeta}\phi = \frac{\partial^2\phi}{\partial\zeta\partial\zeta} = \frac{\partial\sqrt{\frac{2}{3}}}{\partial\zeta} = 0 \tag{A.4}$$

5.) Derivation of ζ:

$$\zeta = -\partial_\xi\psi = -\frac{\partial\psi}{\partial\xi} = -h\xi \tag{A.5}$$

6.) Derivation of $\partial^2_{\xi\xi}\psi$:

$$\partial^2_{\xi\xi}\psi = \frac{\partial^2\psi}{\partial\xi\partial\xi} = h \tag{A.6}$$

7.) Derivation of $\mathbf{T}^P$:

$$\mathbf{T}^P = \partial_{\mathbf{E}}\psi = \frac{\partial\psi}{\partial\mathbf{E}} = \frac{\partial(\frac{\lambda}{2}(I_{\mathbf{E}^e})^2 + \mu II_{\mathbf{E}^e} + \frac{1}{2}h\xi^2)}{\partial\mathbf{E}}$$

$$= \frac{\partial(\frac{\lambda}{2}E^2_{ii} + \mu(E_{ij}E_{ij}) + \frac{1}{2}h\xi^2)}{\partial E_{kl}} = \lambda E_{ii}\delta_{ik}\delta_{il} + \mu(\delta_{ik}\delta_{jl}E_{ij} + \delta_{ik}\delta_{jl}E_{ij})$$

$$= \lambda\mathrm{tr}(\mathbf{E}^e)\mathbf{1} + 2\mu\mathbf{E}^e \tag{A.7}$$

8.) Derivation of $\partial^2_{\mathbf{E}^P\mathbf{E}^P}\psi$:

$$\partial^2_{\mathbf{E}^P\mathbf{E}^P}\psi = \partial^2_{\mathbf{E}\mathbf{E}}\psi = \frac{\partial^2\psi}{\partial\mathbf{E}\partial\mathbf{E}}$$

$$= \frac{\partial(\lambda \mathrm{tr}(\mathbf{E}^e)\mathbf{1} + 2\mu\mathbf{E}^e)}{\partial\mathbf{E}} = \frac{\partial(\lambda E_{ii}\delta_{ij} + 2\mu E_{ij})}{\partial E_{kl}}$$

$$= \lambda\delta_{ik}\delta_{il}\delta_{ij} + \mu(\delta_{ik}\delta_{jl} + \delta_{il}\delta_{jk})$$

$$= \lambda\mathbf{G}^{ij}\otimes\mathbf{G}^{kl} + \mu(\mathbf{G}^{ik}\otimes\mathbf{G}^{jl} + \mathbf{G}^{il}\otimes\mathbf{G}^{jk}) \tag{A.8}$$

For Cartesian coordinates: $\mathbf{G} = \mathbf{1}$

9.) Derivation of $\partial_{\mathbf{E}}\phi$:

$$\partial_{\mathbf{E}}\phi = \partial_{\mathbf{T}^P}\phi\cdot\partial_{\mathbf{E}}\mathbf{T}^P + \partial_{\mathbf{T}^P}\phi\cdot\partial_{\mathbf{E}^P}\mathbf{T}^P\cdot\partial_{\mathbf{E}}\mathbf{E}^P + \partial_\zeta\phi\cdot\partial_{\mathbf{E}}\zeta + \partial_\zeta\phi\cdot\partial_\xi\zeta\cdot\partial_{\mathbf{E}}\xi$$

$$= -\partial^2_{\mathbf{E}\mathbf{E}^P}\psi(\mathbb{I} + \gamma^P\partial^2_{\mathbf{T}^P\mathbf{T}^P}\phi\cdot\partial^2_{\mathbf{E}^P\mathbf{E}^P}\psi)\partial_{\mathbf{T}^P}\phi - \partial^2_{\mathbf{E}\xi}\psi(\mathbb{I} + \gamma^P\partial^2_{\zeta\zeta}\phi\cdot\partial^2_{\xi\xi}\psi)\partial_\zeta\phi$$

$$= -\partial^2_{\mathbf{E}\mathbf{E}^P}\psi(\mathbb{I} + \gamma^P\partial^2_{\mathbf{T}^P\mathbf{T}^P}\phi\cdot\partial^2_{\mathbf{E}^P\mathbf{E}^P}\psi)\partial_{\mathbf{T}^P}\phi = -\partial^2_{\mathbf{E}\mathbf{E}^P}\psi\mathbb{B}^{-1}\partial_{\mathbf{T}^P}\phi \tag{A.9}$$

With $\partial^2_{\mathbf{E}\xi}U = 0$ and $\mathbb{B} = \mathbb{I} + \gamma^P\partial^2_{\mathbf{T}^P\mathbf{T}^P}\phi\partial^2_{\mathbf{E}^P\mathbf{E}^P}U$

Appendix B.

Fourth-order tensor: Inversion

This appendix shows the inversion of a fourth-order tensor. Take a fourth-order tensor $\mathbb{B}$ into account like it is defined in chapter 4 with its components B_{ijkl} for $i, j, k, l = \{1, 2, 3\}$. Additionally consider two second-order tensor $\mathbf{a}$ and $\mathbf{c}$ with the components a_{ij} and c_{kl}. Defining a linear algebra system $a_{ij} = B_{ijkl} c_{kl}$ with the matrix formulation $\mathbf{a}_9 = \mathbb{B}_{9\text{x}9} * \mathbf{c}_9$ yields:

$$\begin{pmatrix} a_{(11)} \\ a_{(22)} \\ a_{(33)} \\ a_{(23)} \\ a_{(13)} \\ a_{(12)} \\ a_{(32)} \\ a_{(31)} \\ a_{(21)} \end{pmatrix} = \begin{pmatrix} B_{(11)11} & B_{(11)22} & B_{(11)33} & B_{(11)23} & B_{(11)13} & B_{(11)12} & B_{(11)32} & B_{(11)31} & B_{(11)21} \\ B_{(22)11} & B_{(22)22} & B_{(22)33} & B_{(22)23} & B_{(22)13} & B_{(22)12} & B_{(22)32} & B_{(22)31} & B_{(22)21} \\ B_{(33)11} & B_{(33)22} & B_{(33)33} & B_{(33)23} & B_{(33)13} & B_{(33)12} & B_{(33)32} & B_{(33)31} & B_{(33)21} \\ B_{(23)11} & B_{(23)22} & B_{(23)33} & B_{(23)23} & B_{(23)13} & B_{(23)12} & B_{(23)32} & B_{(23)31} & B_{(23)21} \\ B_{(13)11} & B_{(13)22} & B_{(13)33} & B_{(13)23} & B_{(13)13} & B_{(13)12} & B_{(13)32} & B_{(13)31} & B_{(13)21} \\ B_{(12)11} & B_{(12)22} & B_{(12)33} & B_{(12)23} & B_{(12)13} & B_{(12)12} & B_{(12)32} & B_{(12)31} & B_{(12)21} \\ B_{(32)11} & B_{(32)22} & B_{(32)33} & B_{(32)23} & B_{(32)13} & B_{(32)12} & B_{(32)32} & B_{(32)31} & B_{(32)21} \\ B_{(31)11} & B_{(31)22} & B_{(31)33} & B_{(31)23} & B_{(31)13} & B_{(31)12} & B_{(31)32} & B_{(31)31} & B_{(31)21} \\ B_{(21)11} & B_{(21)22} & B_{(21)33} & B_{(21)23} & B_{(21)13} & B_{(21)12} & B_{(21)32} & B_{(21)31} & B_{(21)21} \end{pmatrix} * \begin{pmatrix} c_{11} \\ c_{22} \\ c_{33} \\ c_{23} \\ c_{13} \\ c_{12} \\ c_{32} \\ c_{31} \\ c_{21} \end{pmatrix} \tag{B.1}$$

For a symmetric tensor $\mathbb{B}$ the rows and columns show linear dependencies and the tensor can be rewritten as an equivalent tensor in the form of $\mathbf{B}_{6\text{x}6}$. The linear algebra system from Equation B.1 can be expressed as $\mathbf{a}_6 = \mathbb{B}_{6\text{x}6} * \mathbf{c}_6$

$$\begin{pmatrix} a_{11} \\ a_{22} \\ a_{33} \\ a_{23} \\ a_{13} \\ a_{12} \end{pmatrix} = \begin{pmatrix} B_{(11)11} & B_{(11)22} & B_{(11)33} & 2B_{(11)23} & 2B_{(11)13} & 2B_{(11)12} \\ B_{(22)11} & B_{(22)22} & B_{(22)33} & 2B_{(22)23} & 2B_{(22)13} & 2B_{(22)12} \\ B_{(33)11} & B_{(33)22} & B_{(33)33} & 2B_{(33)23} & 2B_{(33)13} & 2B_{(33)12} \\ 2B_{(23)11} & 2B_{(23)22} & 2B_{(23)33} & 4B_{(23)23} & 4B_{(23)13} & 4B_{(23)12} \\ 2B_{(13)11} & 2B_{(13)22} & 2B_{(13)33} & 4B_{(13)23} & 4B_{(13)13} & 4B_{(13)12} \\ 2B_{(12)11} & 2B_{(12)22} & 2B_{(12)33} & 4B_{(12)23} & 4B_{(12)13} & 4B_{(12)12} \end{pmatrix} * \begin{pmatrix} c_{11} \\ c_{22} \\ c_{33} \\ c_{23} \\ c_{13} \\ c_{12} \end{pmatrix} \tag{B.2}$$

$\mathbf{B}^{-1}$ can be calculated and the coefficients can be shifted back to a fourth-order tensor to get $\mathbb{B}^{-1}$. This yields:

$$A_{ijkl} A^{-1}_{klmn} = I_{ijmn} = \frac{1}{2}(\delta_{im}\delta_{jn} + \delta_{in}\delta_{jm}). \tag{B.3}$$

Appendix C.

Second-order tensor: Spectral decomposition

In this appendix the spectral decomposition of a second-order tensor is demonstrated. Performing a spectral decomposition to a second-order tensor $\mathbf{A}$ yields:

$$\mathbf{A} = \sum_a \lambda_a \mathbf{N}_a \otimes \mathbf{N}_a. \tag{C.1}$$

$\mathbf{N}_a$ and λ_a denote the eigenvectors with the corresponding eigenvalues of the tensor $\mathbf{A}$. The derivation of the eigenvalues λ_a with respect to the tensor $\mathbf{A}$ follows as:

$$\frac{\partial \lambda_a}{\partial \mathbf{A}} = \frac{\partial \lambda_a}{\partial \Big(\sum\limits_b \lambda_b \mathbf{N}_b \otimes \mathbf{N}_b \Big)} = \sum_b \frac{\partial \lambda_a}{\partial \lambda_b} \mathbf{N}_b \otimes \mathbf{N}_b = \mathbf{N}_a \otimes \mathbf{N}_a. \tag{C.2}$$

Next the derivation of the eigenvectors $\mathbf{N}_a$ with respect to its tensor $\mathbf{A}$ is observed. For this take $\mathbf{A}\mathbf{N}_a = \lambda_a \mathbf{A}$ and $\mathbf{I} = \sum_b \mathbf{N}_b \otimes \mathbf{N}_b$ into account. Furthermore it is $\delta_{ab} = \mathbf{N}_a^T \mathbf{N}_b$ which yields:

$$(\mathbf{I} - \mathbf{N}_a \otimes \mathbf{N}_a) \otimes \mathbf{N}_a = \sum_{b \neq a} (\lambda_a - \lambda_b) \mathbf{N}_b \otimes \mathbf{N}_b \frac{\partial \mathbf{N}_a}{\partial \mathbf{A}} \tag{C.3}$$

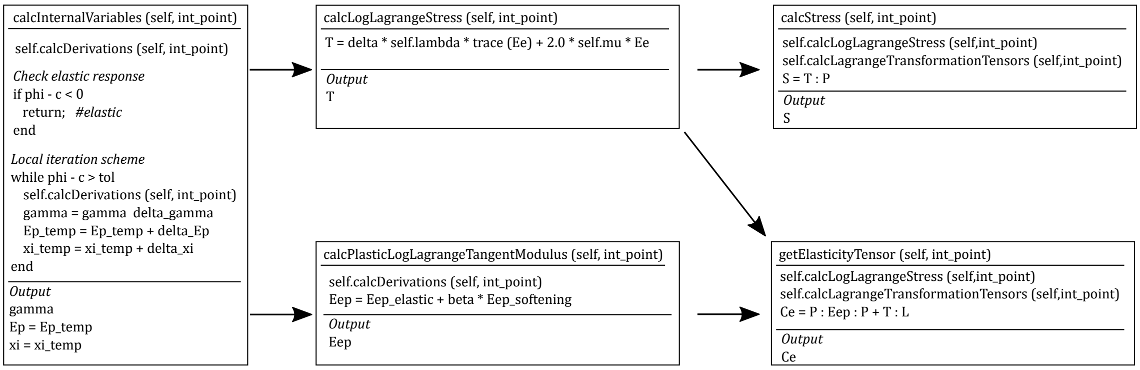
With the equivalence of $(\mathbf{I} - \mathbf{N}_a \otimes \mathbf{N}_a) = \sum_{b \neq a} \mathbf{N}_b \otimes \mathbf{N}_b$ a symmetric tensor $\mathbf{A}$ can be found:

$$2 \frac{\partial (\mathbf{N}_a)_i}{\partial \mathbf{A}_{kl}} = \frac{\partial (\mathbf{N}_a)_i}{\partial \mathbf{A}_{kl}} + \frac{\partial (\mathbf{N}_a)_i}{\partial \mathbf{A}_{lk}} = \sum_{b \neq a} \frac{1}{\lambda_a - \lambda_b} (\mathbf{N}_b)_i [(\mathbf{N}_a)_k (\mathbf{N}_b)_l + (\mathbf{N}_a)_l (\mathbf{N}_b)_k] \tag{C.4}$$

Appendix D.

Matlab Code

In this appendix the programmed Matlab code can be found. The code illustrates the constitutive equations which were added to a object oriented finite element software called ”soofeaM” (software for object-oriented finite element analysis in Matlab) provided by the Institute of Strength of Materials at Graz University of Technology. The program can handle two and three dimensional linear and non-linear problems. Quad, triangle, hexaeder and tetra elements are implemented. The meshes of the simple benchmark examples from chapter 5 were generated with the help of the finite element program Abaqus. The mesh was exported and prepared for the above-mentioned solver. The boundary conditions are modeled directly within the Matlab program.
Firstly the deformation gradient **F** is calculated. This is used to calculate the right Cauchy-Green deformation tensor **C**. Within the function ”calcInternalVariables” the necessary derivations and the internal variables $\mathbf{E}^P$ and ξ are calculated locally for every integration point. With the transformation tensors defined in the function ”calcLagrangeTransformationTensors” the Second Piola Kirchhoff stress **S** and the elastic-plastic logarithmic Lagrangian tangent modulus $\mathbb{E}^{ep}$ are calculated. This routine is performed within the global iteration scheme till a converged state is achieved. A schematic illustration and pseudo-code of used functions can be taken from Figure D.1.



```
calcInternalVariables (self, int_point)
 self.calcDerivations (self, int_point)
 Check elastic response
 if phi - c < 0
   return;   #elastic
 end
 Local iteration scheme
 while phi - c > tol
    self.calcDerivations (self, int_point)
    gamma = gamma  delta_gamma
    Ep_temp = Ep_temp + delta_Ep
    xi_temp = xi_temp + delta_xi
 end
Output
gamma
Ep = Ep_temp
xi = xi_temp
```

```
calcLogLagrangeStress (self, int_point)
 T = delta * self.lambda * trace (Ee) + 2.0 * self.mu * Ee
 Output
 T
```

```
calcStress (self, int_point)
 self.calcLogLagrangeStress (self,int_point)
 self.calcLagrangeTransformationTensors (self,int_point)
 S = T : P
 Output
 S
```

```
calcPlasticLogLagrangeTangentModulus (self, int_point)
 self.calcDerivations (self, int_point)
 Eep = Eep_elastic + beta * Eep_softening
 Output
 Eep
```

```
getElasticityTensor (self, int_point)
 self.calcLogLagrangeStress (self,int_point)
 self.calcLagrangeTransformationTensors (self,int_point)
 Ce = P : Eep : P + T : L
 Output
 Ce
```

Figure D.1. Matlab code. Schematic illustration and pseudo-code of used functions.

```
classdef PlasticLogLagrangeanStrain < nsModel.nsMaterial.Material & nsModel.nsMaterial.AnisotropicPlasticMaterial
    %Plastic material with additive logarithmic strain

    methods (Static)

        function self = PlasticLogLagrangeanStrain(number, E_mod, nu, a1, a2, y0, y11, y22, y33, y12, y23, y13, h, c)

            self@nsModel.nsMaterial.Material(number);
            self@nsModel.nsMaterial.AnisotropicPlasticMaterial(number, E_mod, nu, a1, a2, y0, y11, y22, y33, y12, y23, y13, h, c);

        end


        function FirstInvEe = calcFirstInvariantEe (int_point)
            % Calculate first invariant of elastic logarithmic strain Ee
            % ----------------------------------------------------------

            Ee = int_point.Ee;
            FirstInvEe = trace(Ee);

        end

        function SecondInvEe = calcSecondInvariantEe (int_point)
            % Calculate second invariant of elastic logarithmic strain Ee
            % -----------------------------------------------------------

            Ee = int_point.Ee;
            SecondInvEe = trace(transpose(Ee)*Ee);

        end

        function potential = calcPotential (self,int_point)
            % Calculate strain-energy
            % -----------------------

            FirstInvEe = self.calcFirstInvariantEe (int_point);
            SecondInvEe = self.calcSecondInvariantEe (int_point);

            potential = 0.5 * self.lambda * (FirstInvEe)^2 + self.mu * SecondInvEe;

        end

        function T = calcLogLagrangeStress(self, int_point)
            % Calculate logarithmic stress T
            % ------------------------------

            dimension = length(int_point.material_coordinates);

            delta = eye(dimension);
            Ee = int_point.Ee;

            T = delta * self.lambda * trace(Ee) + 2.0 * self.mu * Ee;

        end

        function Eee = calcElasticLogLagrangeTangentModulus (self, int_point)
            % Calculation of the elastic logarithmic tangent modulus Eee
            % ----------------------------------------------------------

            dimension = length(int_point.material_coordinates);
            delta = eye(dimension);
            d_ijkl = zeros(dimension, dimension, dimension, dimension);

            for i = 1:dimension
                for j = 1:dimension
                    for k = 1:dimension
                        for l = 1:dimension
                            d_ijkl(i,j,k,l) = delta(i,j)*delta(k,l);
                        end
                    end
                end
            end

            d_ikjl = permute(d_ijkl,[1 3 2 4]);
            d_iljk = permute(d_ijkl,[1 3 4 2]);

            Eee = self.lambda * d_ijkl + self.mu * (d_ikjl + d_iljk);

        end

        function [P,L] = calcLagrangeTransformationTensors(self, int_point,case_sel)
            % Calculation of the transformation matrices P and L
            % --------------------------------------------------

            % case_sel = 1 for calculation of Second Piola-Kirchhoff stress S
            % case_sel = 0 for calculation of elastic-plastic Lagrangean tangent modulus C
            % -> less calculation ressources

            dimension = length(int_point.material_coordinates);
            C = int_point.C;

            % Eigenvalues of C and Ee
            EigenValC = eig(C);
            EigenValEe = 0.5*log(EigenValC);

            % Introducing tensor with eigenvalues
            [EigenVec,temp] = eig(C);

```

```
% Calculate parameters d and f
d = EigenValC.^(-1);
f = -2*EigenValC.^(-2);

% Define eigenvalue bases M
M = zeros(dimension, dimension, dimension);

for a = 1:dimension
    for i = 1:dimension
        for j = 1:dimension
            M(i,j,a) = EigenVec(i,a) * EigenVec(j,a);
        end
    end
end

% Initiate parameters
theta = zeros(dimension, dimension);
xi = zeros(dimension, dimension);
eta = 0;

% Check eigenvalues
 if dimension == 3

    tol = 1e-10;        % computional tolerance
    equals = [0 0 0];
    equals(1) = abs(EigenValC(1) - EigenValC(3)) <= tol;
    equals(2) = abs(EigenValC(2) - EigenValC(1)) <= tol;
    equals(3) = abs(EigenValC(3) - EigenValC(2)) <= tol;
 else
    tol = 1e-10;        % computional tolerance
    equals = [0 0];
    equals(1) = abs(EigenValC(1) - EigenValC(2)) <= tol;
 end

% Calculate parameters (3 cases)
% --------------------------------
switch sum(equals)
    case 0
        for a= 1:dimension
            for b= 1:dimension
                theta(a,b) = (EigenValEe(a)-EigenValEe(b))/(EigenValC(a)-EigenValC(b));
                xi(a,b) = (theta(a,b) - 0.5 * d(b))/(EigenValC(a)-EigenValC(b));
                for c= 1:dimension
                    if b~=a
                        if c~=a && c~=b
                             eta = eta + EigenValEe(a)/(2*(EigenValC(a)-EigenValC(b))*...
                                (EigenValC(a)-EigenValC(c)));
                        end
                    end
                end
            end
        end

    case 1 % a = b
        for a= 1:dimension
            for b= 1:dimension
                theta(a,b) = 0.5*d(a);
                xi(a,b) = 1/8*f(a);
            end
        end

        if dimension ==3
            idx = find(equals == 0);
            eta = xi(idx(1),idx(2));
        else
            eta = 1/8*f(1);
        end

    case 3 % a = b = c
        for a= 1:dimension
            for b= 1:dimension
                theta(a,b) = 0.5*d(a);
                xi(a,b) = 1/8*f(a);
            end
        end
        eta = 1/8*f(1);
end
%--------------------------------

% Calculation of help matrices G(a,b) and H(a,b,c)
G = zeros(dimension, dimension, dimension, dimension, dimension, dimension);
H = zeros(dimension, dimension, dimension, dimension, dimension, dimension, dimension, dimension, dimension);

for a = 1:dimension
    for b = 1:dimension
        for c = 1:dimension
            for i = 1:dimension
                for j = 1:dimension
                    for k = 1:dimension
                        for l = 1:dimension
                            for m = 1:dimension
                                for n = 1:dimension

                                    if case_sel == 1
                                        G(i,j,k,l,a,b) = M(i,k,a)*M(j,l,b) + M(i,l,a)*M(j,k,b);
                                    else
                                        G(i,j,k,l,a,b) = M(i,k,a)*M(j,l,b) + M(i,l,a)*M(j,k,b);
                                        H(i,j,k,l,m,n,a,b,c) = M(i,k,a)*M(j,m,b)*M(l,n,c) + M(i,k,a)*M(j,n,b)*M(l,m,c)...
```

```
                                                    + M(i,l,a)*M(j,m,b)*M(k,n,c) + M(i,l,a)*M(j,n,b)*M(k,m,c)...
                                                    + M(j,l,a)*M(i,m,b)*M(k,n,c) + M(j,l,a)*M(i,n,b)*M(k,m,c)...
                                                    + M(j,k,a)*M(i,m,b)*M(l,n,c) + M(j,k,a)*M(i,n,b)*M(l,m,c);
                                            end

                                        end
                                    end
                                end
                            end
                        end
                    end
                end
             end
        end


        % Calculation of dyadic(M) and dyadic(dyadic(M,M),M)
        dyadM = zeros(dimension, dimension, dimension, dimension, dimension); % dyadic(M,M)
        ddyadM = zeros(dimension, dimension, dimension, dimension, dimension,...
            dimension, dimension, dimension, dimension); %dyadic(dyadic(M,M),M)

        for a = 1:dimension
            for i = 1:dimension
                for j = 1:dimension
                    for k = 1:dimension
                        for l = 1:dimension
                            dyadM(i,j,k,l,a) = M(i,j,a) * M(k,l,a);
                        end
                    end
                end
            end
        end

        if case_sel ~=1
            for a = 1:dimension
                for i = 1:dimension
                    for j = 1:dimension
                        for k = 1:dimension
                            for l = 1:dimension
                                for m = 1:dimension
                                    for n = 1:dimension
                                        ddyadM(i,j,k,l,m,n,a) = M(i,j,a)*M(k,l,a)*M(m,n,a);
                                    end
                                end
                            end
                        end
                    end
                end
            end
        end

        % Calculation of fourth-order Lagrangean transformation tensor P
        % Calculation of sixth-order Lagrangean transformation tensor L

        P = zeros(dimension, dimension, dimension, dimension);
        L = zeros(dimension, dimension, dimension, dimension, dimension, dimension);

        for a = 1:dimension
            P = P + d(a) * dyadM(:,:,:,:,a);
            L = L + f(a) * ddyadM(:,:,:,:,:,:,a);

            for b = 1:dimension
                if a ~= b
                    P = P + theta(a,b) * G(:,:,:,:,a,b);
                    L = L + + xi(a,b)*(H(:,:,:,:,:,:,b,a,b)+H(:,:,:,:,:,:,b,b,a)+H(:,:,:,:,:,:,a,b,b));
                    for c= 1:dimension
                        if c ~= a && c ~= b
                            L = L + eta*H(:,:,:,:,:,:,a,b,c);
                        end
                    end
                end
            end

        end

    end

    function tensor4dinv = inverse4dsym( tensor4d, dim )
        % Calculate the inverse of a symmetric fourth-order tensor
        % -------------------------------------------------------
        % works for inversion of 4d tensors with either major & minor symmetries or
        % with just minor symmetries
        % this does not work for 4d tensors with just major symmetries

        if (dim ~= 3)
            stopstop;
        end

        tensor2d = zeros([dim*2 dim*2]);
        tensor4dinv = zeros([dim dim dim dim]);

        tensor2d( 1, 1  ) = tensor4d( 1,1,1,1 );
        tensor2d( 1, 2  ) = tensor4d( 1,1,2,2 );
        tensor2d( 1, 3  ) = tensor4d( 1,1,3,3 );
        tensor2d( 2, 1  ) = tensor4d( 2,2,1,1 );
        tensor2d( 2, 2  ) = tensor4d( 2,2,2,2 );
        tensor2d( 2, 3  ) = tensor4d( 2,2,3,3 );
        tensor2d( 3, 1  ) = tensor4d( 3,3,1,1 );
```

```
            tensor2d( 3, 2  ) = tensor4d( 3,3,2,2 );
            tensor2d( 3, 3  ) = tensor4d( 3,3,3,3 );
            tensor2d( 1, 4+0) = tensor4d( 1,1,2,3 ) * 2.0;
            tensor2d( 1, 4+1) = tensor4d( 1,1,1,3 ) * 2.0;
            tensor2d( 1, 4+2) = tensor4d( 1,1,1,2 ) * 2.0;
            tensor2d( 2, 4+0) = tensor4d( 2,2,2,3 ) * 2.0;
            tensor2d( 2, 4+1) = tensor4d( 2,2,1,3 ) * 2.0;
            tensor2d( 2, 4+2) = tensor4d( 2,2,1,2 ) * 2.0;
            tensor2d( 3, 4+0) = tensor4d( 3,3,2,3 ) * 2.0;
            tensor2d( 3, 4+1) = tensor4d( 3,3,1,3 ) * 2.0;
            tensor2d( 3, 4+2) = tensor4d( 3,3,1,2 ) * 2.0;
            tensor2d( 4+0, 1  ) = tensor4d( 2,3,1,1 ) * 2.0;
            tensor2d( 4+0, 2  ) = tensor4d( 2,3,2,2 ) * 2.0;
            tensor2d( 4+0, 3  ) = tensor4d( 2,3,3,3 ) * 2.0;
            tensor2d( 4+1, 1  ) = tensor4d( 1,3,1,1 ) * 2.0;
            tensor2d( 4+1, 2  ) = tensor4d( 1,3,2,2 ) * 2.0;
            tensor2d( 4+1, 3  ) = tensor4d( 1,3,3,3 ) * 2.0;
            tensor2d( 4+2, 1  ) = tensor4d( 1,2,1,1 ) * 2.0;
            tensor2d( 4+2, 2  ) = tensor4d( 1,2,2,2 ) * 2.0;
            tensor2d( 4+2, 3  ) = tensor4d( 1,2,3,3 ) * 2.0;
            tensor2d( 4+0, 4+0) = tensor4d( 2,3,2,3 ) * 4.0;
            tensor2d( 4+0, 4+1) = tensor4d( 2,3,1,3 ) * 4.0;
            tensor2d( 4+0, 4+2) = tensor4d( 2,3,1,2 ) * 4.0;
            tensor2d( 4+1, 4+0) = tensor4d( 1,3,2,3 ) * 4.0;
            tensor2d( 4+1, 4+1) = tensor4d( 1,3,1,3 ) * 4.0;
            tensor2d( 4+1, 4+2) = tensor4d( 1,3,1,2 ) * 4.0;
            tensor2d( 4+2, 4+0) = tensor4d( 1,2,2,3 ) * 4.0;
            tensor2d( 4+2, 4+1) = tensor4d( 1,2,1,3 ) * 4.0;
            tensor2d( 4+2, 4+2) = tensor4d( 1,2,1,2 ) * 4.0;

            tensor2dinv = inv(tensor2d);

            tensor4dinv(1,1,1,1) = tensor2dinv(1,1);
            tensor4dinv(1,1,2,2) = tensor2dinv(1,2);
            tensor4dinv(1,1,3,3) = tensor2dinv(1,3);
            tensor4dinv(2,2,1,1) = tensor2dinv(2,1);
            tensor4dinv(2,2,2,2) = tensor2dinv(2,2);
            tensor4dinv(2,2,3,3) = tensor2dinv(2,3);
            tensor4dinv(3,3,1,1) = tensor2dinv(3,1);
            tensor4dinv(3,3,2,2) = tensor2dinv(3,2);
            tensor4dinv(3,3,3,3) = tensor2dinv(3,3);
            tensor4dinv(1,1,2,3) = tensor2dinv(1,4+0);
            tensor4dinv(1,1,1,3) = tensor2dinv(1,4+1);
            tensor4dinv(1,1,1,2) = tensor2dinv(1,4+2);
            tensor4dinv(1,1,3,2) = tensor2dinv(1,4+0);
            tensor4dinv(1,1,3,1) = tensor2dinv(1,4+1);
            tensor4dinv(1,1,2,1) = tensor2dinv(1,4+2);
            tensor4dinv(2,2,2,3) = tensor2dinv(2,4+0);
            tensor4dinv(2,2,1,3) = tensor2dinv(2,4+1);
            tensor4dinv(2,2,1,2) = tensor2dinv(2,4+2);
            tensor4dinv(2,2,3,2) = tensor2dinv(2,4+0);
            tensor4dinv(2,2,3,1) = tensor2dinv(2,4+1);
            tensor4dinv(2,2,2,1) = tensor2dinv(2,4+2);
            tensor4dinv(3,3,2,3) = tensor2dinv(3,4+0);
            tensor4dinv(3,3,1,3) = tensor2dinv(3,4+1);
            tensor4dinv(3,3,1,2) = tensor2dinv(3,4+2);
            tensor4dinv(3,3,3,2) = tensor2dinv(3,4+0);
            tensor4dinv(3,3,3,1) = tensor2dinv(3,4+1);
            tensor4dinv(3,3,2,1) = tensor2dinv(3,4+2);
            tensor4dinv(2,3,1,1) = tensor2dinv(4+0,1);
            tensor4dinv(2,3,2,2) = tensor2dinv(4+0,2);
            tensor4dinv(2,3,3,3) = tensor2dinv(4+0,3);
            tensor4dinv(3,2,1,1) = tensor2dinv(4+0,1);
            tensor4dinv(3,2,2,2) = tensor2dinv(4+0,2);
            tensor4dinv(3,2,3,3) = tensor2dinv(4+0,3);
            tensor4dinv(1,3,1,1) = tensor2dinv(4+1,1);
            tensor4dinv(1,3,2,2) = tensor2dinv(4+1,2);
            tensor4dinv(1,3,3,3) = tensor2dinv(4+1,3);
            tensor4dinv(3,1,1,1) = tensor2dinv(4+1,1);
            tensor4dinv(3,1,2,2) = tensor2dinv(4+1,2);
            tensor4dinv(3,1,3,3) = tensor2dinv(4+1,3);
            tensor4dinv(1,2,1,1) = tensor2dinv(4+2,1);
            tensor4dinv(1,2,2,2) = tensor2dinv(4+2,2);
            tensor4dinv(1,2,3,3) = tensor2dinv(4+2,3);
            tensor4dinv(2,1,1,1) = tensor2dinv(4+2,1);
            tensor4dinv(2,1,2,2) = tensor2dinv(4+2,2);
            tensor4dinv(2,1,3,3) = tensor2dinv(4+2,3);
            tensor4dinv(2,3,2,3) = tensor2dinv(4+0,4+0);
            tensor4dinv(2,3,1,3) = tensor2dinv(4+0,4+1);
            tensor4dinv(2,3,1,2) = tensor2dinv(4+0,4+2);
            tensor4dinv(3,2,2,3) = tensor2dinv(4+0,4+0);
            tensor4dinv(3,2,1,3) = tensor2dinv(4+0,4+1);
            tensor4dinv(3,2,1,2) = tensor2dinv(4+0,4+2);
            tensor4dinv(2,3,3,2) = tensor2dinv(4+0,4+0);
            tensor4dinv(2,3,3,1) = tensor2dinv(4+0,4+1);
            tensor4dinv(2,3,2,1) = tensor2dinv(4+0,4+2);
            tensor4dinv(3,2,3,2) = tensor2dinv(4+0,4+0);
            tensor4dinv(3,2,3,1) = tensor2dinv(4+0,4+1);
            tensor4dinv(3,2,2,1) = tensor2dinv(4+0,4+2);
            tensor4dinv(1,3,2,3) = tensor2dinv(4+1,4+0);
            tensor4dinv(1,3,1,3) = tensor2dinv(4+1,4+1);
            tensor4dinv(1,3,1,2) = tensor2dinv(4+1,4+2);
            tensor4dinv(3,1,2,3) = tensor2dinv(4+1,4+0);
            tensor4dinv(3,1,1,3) = tensor2dinv(4+1,4+1);
            tensor4dinv(3,1,1,2) = tensor2dinv(4+1,4+2);
            tensor4dinv(1,3,3,2) = tensor2dinv(4+1,4+0);
            tensor4dinv(1,3,3,1) = tensor2dinv(4+1,4+1);
            tensor4dinv(1,3,2,1) = tensor2dinv(4+1,4+2);
            tensor4dinv(3,1,3,2) = tensor2dinv(4+1,4+0);
            tensor4dinv(3,1,3,1) = tensor2dinv(4+1,4+1);
            tensor4dinv(3,1,2,1) = tensor2dinv(4+1,4+2);
            tensor4dinv(1,2,2,3) = tensor2dinv(4+2,4+0);
            tensor4dinv(1,2,1,3) = tensor2dinv(4+2,4+1);
            tensor4dinv(1,2,1,2) = tensor2dinv(4+2,4+2);
            tensor4dinv(2,1,2,3) = tensor2dinv(4+2,4+0);
            tensor4dinv(2,1,1,3) = tensor2dinv(4+2,4+1);
            tensor4dinv(2,1,1,2) = tensor2dinv(4+2,4+2);
            tensor4dinv(1,2,3,2) = tensor2dinv(4+2,4+0);
            tensor4dinv(1,2,3,1) = tensor2dinv(4+2,4+1);
            tensor4dinv(1,2,2,1) = tensor2dinv(4+2,4+2);
            tensor4dinv(2,1,3,2) = tensor2dinv(4+2,4+0);
            tensor4dinv(2,1,3,1) = tensor2dinv(4+2,4+1);
            tensor4dinv(2,1,2,1) = tensor2dinv(4+2,4+2);
        end

        function [H_norm,dtp_phi,dtptp_phi,dzeta_phi,dzetazeta_phi,depep_U,dxixi_U] = calcDerivations(self,int_point)
            % Calculation of necessary derivations
            % -------------------------------------

            dimension = length(int_point.material_coordinates);
            h = self.h;

            % Calculate logarithmic stress
            T = self.calcLogLagrangeStress(self,int_point);
            Tp = T;

            % Fourth-order Hill tensor
            H = self.calcStructuralTensor(self,int_point);


            % Calculate norm of logarithmic stress tensor with respect
            % to constant fourth-order Hill tensor
            norm = 0;

            for i = 1:dimension
                for j = 1:dimension
                    for k = 1:dimension
                        for l = 1:dimension
                           norm = norm + Tp(i,j)*H(i,j,k,l)*Tp(k,l);
                        end
                    end
                end
            end

            H_norm = sqrt(norm);

            % Calculate H : T^(p)
            H_Tp = zeros(dimension,dimension);
            H_Tp_H_Tp = zeros(dimension,dimension,dimension,dimension);

             for i = 1:dimension
                for j = 1:dimension
                    for k = 1:dimension
                        for l = 1:dimension
                            H_Tp(i,j) = H_Tp(i,j) + H(i,j,k,l)*Tp(k,l);
                        end
                    end
                end
             end

             % Calculate [H : T^(p)] dyad [H : T^(p)]
             for i = 1:dimension
                for j = 1:dimension
                    for k = 1:dimension
                        for l = 1:dimension
                            H_Tp_H_Tp(i,j,k,l) = H_Tp(i,j) * H_Tp(k,l);
                        end
                    end
                end
             end

            % Function Output
            dtp_phi = H_Tp / H_norm;
            dtptp_phi = H / H_norm - H_Tp_H_Tp / H_norm^3;
            dzeta_phi = sqrt(2/3);
            dzetazeta_phi = 0;
            depep_U = self.calcElasticLogLagrangeTangentModulus (self, int_point);    % for cartesian G = 1
            dxixi_U = h;

        end

        function Eep = calcPlasticLogLagrangeTangentModulus (self, int_point)
            % Calculate elastic-plastic logarithmic Lagrangean tangent modulus Eep
            % --------------------------------------------------------------------

            gamma = int_point.gamma;
            dimension = length(int_point.material_coordinates);

            [H_norm,dtp_phi,dtptp_phi,dzeta_phi,dzetazeta_phi,depep_U,dxixi_U] = self.calcDerivations(self,int_point);

            % Check if already on yield surface
            if H_norm < 10E-15
                dtptp_phi = zeros (dimension,dimension,dimension,dimension);
            end

            % Initialize values
            delta = eye(dimension);
            d_mnkl = zeros(dimension, dimension, dimension, dimension);
            B = zeros(dimension, dimension, dimension, dimension);
            Eep_elastic = zeros(dimension, dimension, dimension, dimension);
            Eep_softening = zeros(dimension, dimension, dimension, dimension);
            de_phi = zeros(dimension, dimension);

            % Calculate help variable b and B
            b = 1 + gamma * dzetazeta_phi * dxixi_U;

            for m = 1:dimension
                for n = 1:dimension
                    for k = 1:dimension
                        for l = 1:dimension
                            d_mnkl(m,n,k,l) = 1/2*( delta(m,k)*delta(n,l) + delta(m,l)*delta(n,k));
                            for i = 1:dimension
                                for j = 1:dimension
                                  B(m,n,k,l) = B(m,n,k,l) + gamma * dtptp_phi(m,n,i,j) * depep_U(i,j,k,l);
                                end
                            end
                        end
                    end
                end
            end

            B = B + d_mnkl;

            % Calculate inverse of B
            B_inv = self.inverse4dsym(B,3);

            % Calculate elastic part of Eep
            for i = 1:dimension
                for j = 1:dimension
                    for s = 1:dimension
                        for t = 1:dimension
                            for m = 1:dimension
                                for n = 1:dimension
                                    for k = 1:dimension
                                        for l = 1:dimension
                                            for q = 1:dimension
                                                for r = 1:dimension

                                                    Eep_elastic(i,j,s,t) = Eep_elastic(i,j,s,t) - ...
                                                        gamma * depep_U(i,j,m,n)*B_inv(m,n,k,l)*dtptp_phi(k,l,q,r)*depep_U(q,r,s,t);

                                                end
                                            end
                                        end
                                    end
                                end
                            end
                        end
                    end
                end
            end

            Eep_elastic = Eep_elastic + depep_U;


            % Calculate Eep
            if gamma > 0

                beta = 1;           % Calculation switch
                skalar = 0;

                % Calculate softening part of Eep
                for i = 1:dimension
                    for j = 1:dimension
                        for m = 1:dimension
                            for n = 1:dimension
                                for k = 1:dimension
                                    for l = 1:dimension
                                        de_phi(i,j) = de_phi(i,j) - depep_U(i,j,m,n)*B_inv(m,n,k,l)*dtp_phi(k,l);
                                        skalar = skalar - dtp_phi(i,j)*depep_U(i,j,m,n)*B_inv(m,n,k,l)*dtp_phi(k,l);
                                    end
                                end
                            end
                        end
                    end
                end

                skalar = skalar - dzeta_phi*dxixi_U*1/b*dzeta_phi;

                for i = 1:dimension
                    for j = 1:dimension
                        for s = 1:dimension
                            for t = 1:dimension
                                Eep_softening(i,j,s,t) = de_phi(i,j) * de_phi(s,t);
                            end
                        end
                    end
                end

                Eep_softening = Eep_softening/skalar;

                Eep = Eep_elastic + beta * Eep_softening;

            else
```

```

                Eep = Eep_elastic;

            end

        end

        function H = calcStructuralTensor (self,int_point)
            % Calculate constant fourth-order Hill tensor
            %--------------------------------------------

            dimension = length(int_point.material_coordinates);

            % Initialize input
            a1 = self.a1;       % base vector 1
            a2 = self.a2;       % base vector 2
            a3 = cross(a1,a2);  % base vector 3
            A = [a1 a2 a3];     % matrix of base vectors

            y0 = self.y0;       % initial yield stress
            y11 = self.y11;
            y22 = self.y22;
            y33 = self.y33;
            y12 = self.y12;
            y23 = self.y23;
            y13 = self.y13;

            % Calculate structural tensors M
            M = zeros(dimension,dimension,dimension,dimension);

            for i = 1:dimension
                for j = 1:dimension
                    for k = 1:dimension
                        for l = 1:dimension
                             M(i,j,k,l) = 1/2 * (A(k,i)*A(l,j) + A(k,j)*A(l,i));
                        end
                    end
                end
            end

            % Calculate independet material parameters alpha
            alpha1 = 2/3 * (y0^2/y11^2);
            alpha2 = 2/3 * (y0^2/y22^2);
            alpha3 = 2/3 * (y0^2/y33^2);
            alpha4 = 1/2 * (alpha3 - alpha1 - alpha2);
            alpha5 = 1/2 * (alpha1 - alpha2 - alpha3);
            alpha6 = 1/2 * (alpha2 - alpha3 - alpha1);
            alpha7 = 1/3 * (y0^2/y12^2);
            alpha8 = 1/3 * (y0^2/y23^2);
            alpha9 = 1/3 * (y0^2/y13^2);

            % Calculate fourth-order Hill tensor
            MM = zeros(dimension, dimension, dimension, dimension);

            for i = 1:dimension
                for j = 1:dimension
                    for k = 1:dimension
                        for l = 1:dimension

                            MM(i,j,k,l) = alpha1 * (M(1,1,i,j) * M(1,1,k,l)) + ...
                                 alpha2 * (M(2,2,i,j) * M(2,2,k,l)) + ...
                                 alpha3 * (M(3,3,i,j) * M(3,3,k,l)) + ...
                                 2 * alpha4 * (1/2*(M(1,1,i,j)*M(2,2,k,l) + M(2,2,i,j)*M(1,1,k,l))) + ...
                                 2 * alpha5 * (1/2*(M(2,2,i,j)*M(3,3,k,l) + M(3,3,i,j)*M(2,2,k,l))) + ...
                                 2 * alpha6 * (1/2*(M(1,1,i,j)*M(3,3,k,l) + M(3,3,i,j)*M(1,1,k,l))) + ...
                                 2 * alpha7 * (M(1,2,i,j) * M(2,1,k,l)) + ...
                                 2 * alpha8 * (M(2,3,i,j) * M(3,2,k,l)) + ...
                                 2 * alpha9 * (M(1,3,i,j) * M(3,1,k,l));
                        end
                    end
                end
            end

            H = MM;

        end

        function [Ep_temp,xi_temp,gamma,Ee] = calcInternalVariables(self,int_point)
            % Calculate internal variables Ep and xi
            % --------------------------------------

            % Initialize values
            h = self.h;             % isotropic hardening
            c = self.c;
            dimension = length(int_point.material_coordinates);

            % Calculate elastic logarithmic Lagrangean strain
            int_point.Ee = nsAnalyzer.KinematicTensors.calcElasticLogLagrangeStrain(int_point.Elog,int_point.Ep_temp);
            Ee = int_point.Ee;

            % Load internal variables from previous/converged step
            xi = int_point.xi;
            xi_temp = int_point.xi_temp;
            Ep = int_point.Ep;
            Ep_temp = int_point.Ep_temp;

            gamma = 0;              % in first step always 0
            deltagamma = 1e99;      % for convergence criteria
            % Calculate necessary derivations
            [H_norm,dtp_phi,dtptp_phi,dzeta_phi,dzetazeta_phi,depep_U,dxixi_U] = self.calcDerivations(self,int_point);

            % Calculate zeta
            zeta = -h*xi_temp;

            % Calculate level set function phi
            phi = H_norm + sqrt(2/3) * zeta;

            % Check if elastic or plastic response
            if phi - c < 0  % elastic
                return;
            end

            % Initialize residuals
            R_ep = zeros(dimension,dimension);
            R_xi = 0;

            % Local iteration scheme
            tol = 10E-23;
            iter = 0;

            while abs( deltagamma*(phi - c) ) > tol

                % Calculate necessary derivations
                [H_norm,dtp_phi,dtptp_phi,dzeta_phi,dzetazeta_phi,depep_U,dxixi_U] = self.calcDerivations(self,int_point);

                % Calculate zeta
                zeta = -h*xi_temp;

                % Calculate level set function phi
                phi = H_norm + sqrt(2/3) * zeta;

                % Calculate residuals
                for m = 1:dimension
                    for n = 1:dimension
                        R_ep(m,n) = -Ep_temp(m,n) + Ep(m,n) + gamma * dtp_phi(m,n);
                    end
                end

                R_xi = -xi_temp + xi + gamma * dzeta_phi;

                % Calculate help quantities b and B
                b = 1 + gamma * dzetazeta_phi * dxixi_U;

                d_mnkl = zeros(dimension, dimension, dimension, dimension);
                delta = eye(dimension);
                B = zeros(dimension, dimension, dimension, dimension);

                for m = 1:dimension
                    for n = 1:dimension
                        for k = 1:dimension
                            for l = 1:dimension

                                d_mnkl(m,n,k,l) = 1/2*( delta(m,k)*delta(n,l) + delta(m,l)*delta(n,k));

                                for i = 1:dimension
                                    for j = 1:dimension

                                      B(m,n,k,l) = B(m,n,k,l) + gamma * dtptp_phi(m,n,i,j) * depep_U(i,j,k,l);

                                    end
                                end
                            end
                        end
                    end
                end

                B = B + d_mnkl;

                % Invert B
                B_inv = self.inverse4dsym(B,3);

                % Calculate delta gamma
                f = 0;
                g = 0;

                for i = 1:dimension
                    for j = 1:dimension
                        for k = 1:dimension
                            for l = 1:dimension
                                for m = 1:dimension
                                    for n = 1:dimension

                                        f = f + dtp_phi(i,j)*depep_U(i,j,k,l)*R_ep(m,n)*B_inv(m,n,k,l);
                                        g = g + dtp_phi(i,j)*depep_U(i,j,k,l)*dtp_phi(m,n)*B_inv(m,n,k,l);

                                    end
                                end
                            end
                        end
                    end
                end


                deltagamma = (phi - c - f - dzeta_phi*dxixi_U * R_xi/b)/(g + dzeta_phi*dxixi_U*dzeta_phi/b);

```

```
                gamma = gamma + deltagamma;


                % Calculate delta Ep_temp
                A = zeros(dimension,dimension);

                for m = 1:dimension
                    for n = 1:dimension

                        A(m,n) = R_ep(m,n) + deltagamma * dtp_phi(m,n);

                    end
                end

                deltaEp = zeros(dimension,dimension);

                for m = 1:dimension
                    for n = 1:dimension
                        for k = 1:dimension
                            for l = 1:dimension

                                deltaEp(m,n) = deltaEp(m,n) + B_inv(k,l,m,n)*A(k,l);

                            end
                        end
                    end
                end

                % Calculate Ep_temp
                Ep_temp = Ep_temp + deltaEp;
                Ep_save = int_point.Ep_temp;

                % Calculate xi_temp
                a = R_xi + deltagamma * dzeta_phi;
                deltaxi = a/b;
                xi_temp = xi_temp + deltaxi;

                % Update internal variables
                int_point.Ep_temp = Ep_temp;
                int_point.xi_temp = xi_temp;
                int_point.gamma = gamma;

                % Replace Ee at integration point with new values
                int_point.Ee = nsAnalyzer.KinematicTensors.calcElasticLogLagrangeStrain(int_point.Elog,int_point.Ep_temp);
                Ee = int_point.Ee;

                % Iteration counter
                iter = iter + 1;

                if (iter > 1000) % to avoid endless loop
                  disp('Local iteration needed maximum iteration steps');
                  disp([iter, deltagamma, phi - c, gamma, abs(deltagamma*(phi - c))]);
                  break;
                end

            end

        end


        function S = calcStress(self, int_point)
            % Calculate Second Piola Kirchhoff stress
            %----------------------------------------

            % Calculate logarithmic stress
            T = self.calcLogLagrangeStress(self,int_point);

            % Calculate transformation tensor P (case_sel = 1)
            [P,L] = self.calcLagrangeTransformationTensors(self,int_point,1);

            %Second Piola Kirchhoff stress S
            % S = T : P
            dimension = length(int_point.material_coordinates);
            S = zeros(dimension, dimension);

            for i = 1:dimension
                for j = 1:dimension
                    for k = 1:dimension
                        for l = 1:dimension

                            S(i,j) = S(i,j) + T(k,l) * P (k,l,i,j);

                        end
                    end
                end
            end

        end

        function Ce = getElasticityTensor(self, int_point)
            % Calculate elastic-plastic Lagrangean tangent modulus
            % ----------------------------------------------------

            % Calculate logarithmic stress
            T = self.calcLogLagrangeStress(self,int_point);

            % Calculate transformation tensors P and L (case_sel = 0)
            [P,L] = self.calcLagrangeTransformationTensors(self,int_point,0);
            % Calculate elasruc-plastic logarithmic Lagrangean tangent modulus
            Eep = self.calcPlasticLogLagrangeTangentModulus(self,int_point);

            % Elastic-plastic Lagrangean Tangent Modulus Ce
            % Ce = P : Eep : P + T : L
            dimension = length(int_point.material_coordinates);
            Ce = zeros(dimension, dimension,dimension,dimension);

            for i = 1 : dimension
                for j = 1 : dimension
                    for k = 1 : dimension
                        for l = 1 : dimension
                            for q = 1 : dimension
                                for r = 1 : dimension

                                    Ce(i,j,k,l) = Ce(i,j,k,l) + T(q,r)*L(q,r,i,j,k,l);

                                    for s = 1 : dimension
                                        for t = 1 : dimension

                                            Ce(i,j,k,l) = Ce(i,j,k,l) + P(i,j,q,r)*Eep(q,r,s,t)*P(s,t,k,l) ;

                                        end
                                    end
                                end
                            end
                        end
                    end
                end
            end


        end


    end
end
```

Bibliography

[1] K.-J. Bathe. *Finite element procedures.* Prentice-Hall, Englewood Cliffs, N.J, 2nd ed. edition, 2014.

[2] J.P. Boehler. A simple derivation of representations for non-polynomial constitutive equations in some cases of anisotropy. *ZAMM - Zeitschrift für Angewandte Mathematik und Mechanik*, 59(4):157–167, 1979.

[3] J. Bonet and R. D. Wood. *Nonlinear continuum mechanics for finite element analysis.* Cambridge Univ. Press, Cambridge, 2. ed., reprint edition, 2009.

[4] O. T. Bruhns, H. Xiao, and A. Meyers. Constitutive inequalities for an isotropic elastic strain-energy function based on hencky's logarithmic strain tensor. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 457(2013):2207–2226, 2001.

[5] J. Dietrich. *Praxis der Umformtechnik: Umform- und Zerteilverfahren, Werkzeuge, Maschinen.* Springer Vieweg, Wiesbaden, 12., überarbeitete und erweiterte Auflage, 2018.

[6] B. Eidel and F. Gruttmann. Elastoplastic orthotropy at finite strains: multiplicative formulation and numerical implementation. *Computational Materials Science*, 28(3-4):732–742, 2003.

[7] A. E. Green and P. M. Naghdi. A general theory of an elastic-plastic continuum. *Archive for Rational Mechanics and Analysis*, 18(4):251–281, 1965.

[8] K. Hashiguchi and Y. Yamakawa. *Introduction to finite strain theory for continuum elasto-plasticity.* Wiley series in computational mechanics. John Wiley & Sons, Chichester, 2013.

[9] H. Hencky. Über die Form des Elastizitätsgesetzes bei ideal elastischen Stoffen. *Zeitschrift für technische Physik*, 9:215–220, 1928.

[10] R. Hill. On constitutive inequalities for simple materials - I. *J. Mech. Phys. Solids*, (16):229–242, 1968.

[11] M. Latorre and F. J. Montáns. On the interpretation of the logarithmic strain tensor in an arbitrary system of representation. *International Journal of Solids and Structures*, 51(7-8):1507–1515, 2014.

[12] J. Löblein, J. Schröder, and F. Gruttmann. Application of generalized measures to an orthotropic finite elasto-plasticity model. *Computational Materials Science*, 28(3-4):696–703, 2003.

[13] J. Lubliner. *Plasticity Theory*. Dover Books on Engineering. Dover Publications, Newburyport, 2013.

[14] J. Mandel. Thermodynamics and plasticity. In J. J. Delgado Domingos, M. N. R. Nina, and J. H. Whitelaw, editors, *Foundations of Continuum Thermodynamics*, pages 283–304. Macmillan Education UK, London, 1973.

[15] R. J. Martin, I. Münch, B. Eidel, and P. Neff. A brief history of logarithmic strain measures in nonlinear elasticity. *Proc. Appl. Math. Mech.*, 18(1):e201800366, 2018.

[16] A. Menzel and P. Steinmann. On the spatial formulation of anisotropic multiplicative elasto-plasticity. *Computer Methods in Applied Mechanics and Engineering*, 192(31-32):3431–3470, 2003.

[17] C. Miehe. A constitutive frame of elastoplasticity at large strains based on the notion of a plastic metric. *International Journal of Solids and Structures*, 35(30):3859–3897, 1998.

[18] C. Miehe and N. Apel. Anisotropic elastic-plastic analysis of shells at large strains. a comparison of multiplicative and additive approaches to enhanced finite element design and constitutive modelling. *International Journal for Numerical Methods in Engineering*, 61(12):2067–2113, 2004.

[19] C. Miehe, N. Apel, and M. Lambrecht. Anisotropic additive plasticity in the logarithmic strain space: modular kinematic formulation and implementation based on incremental minimization principles for standard materials. *Computer Methods in Applied Mechanics and Engineering*, 191(47-48):5383–5425, 2002.

[20] C. Miehe and M. Lambrecht. Algorithms for computation of stresses and elasticity moduli in terms of Seth-Hill's family of generalized strain tensors. *Communications in Numerical Methods in Engineering*, 17:337–353, 2001.

[21] R. von Mises. Mechanik der festen Körper im plastisch-deformablen Zustand. *Nachrichten von der Gesellschaft der Wissenschaften zu Göttingen, Mathematisch-Physikalische Klasse*, 1913:582–592, 1913.

[22] P. Papadopoulos and J. Lu. On the formulation and numerical solution of problems in anisotropic finite plasticity. *Computer Methods in Applied Mechanics and Engineering*, 190(37-38):4889–4910, 2001.

[23] J. R. Rice, R.M. McMeeking, D.M. Parks, and E.P. Sorensen. Recent finite element studies in plasticity and fracture mechanics. *Computer Methods in Applied Mechanics and Engineering*, (17-18, Part 2):411–442, 1979.

[24] J. Rösler, H. Harders, and M. Bäker. *Mechanisches Verhalten der Werkstoffe*. Springer Vieweg, Wiesbaden, 5. edition, 2016.

[25] C. Sansour, I. Karsaj, and J. Soric. A formulation of anisotropic continuum elastoplasticity at finite strains. Part I: Modelling. *International Journal of Plasticity*, 22(12):2346–2365, 2006.

[26] J. Schröder, F. Gruttmann, and J. Löblein. A simple orthotropic finite elasto-plasticity model based on generalized stress-strain measures. *Computational Mechanics*, 30(1):48–64, 2002.

[27] B. R. Seth. Generalized strain measure with applications to pysical problems. *MRC Technical Summary Report*, (248):1–14, 1961.

[28] A. Singh, S. Basak, Lin Prakash P.S., G. G. Roy, M. N. Jha, M. Mascarenhas, and S. K. Panda. Prediction of earing defect and deep drawing behavior of commercially pure titanium sheets using CPB06 anisotropy yield theory. *Journal of Manufacturing Processes*, 33:256–267, 2018.

[29] A.J.M. Spencer. Theory of invariants, in: Eringen, A.C. (ed.). *Continuum Physics*, Vol. 1 (Academic Press, New York):239–353, 1971.

[30] I. Tikhovskiy, D. Raabe, and F. Roters. Simulation of earing during deep drawing of an Al–3% Mg alloy (AA 5754) using a texture component crystal plasticity FEM. *Journal of Materials Processing Technology*, 183(2-3):169–175, 2007.

[31] C. Truesdell, W. Noll, and S. S. Antman. *The Non-Linear Field Theories of Mechanics*. Springer-Verlag Berlin Heidelberg, third edition, 2004.

[32] M. H. Ulz. A Green–Naghdi approach to finite anisotropic rate-independent and rate-dependent thermo-plasticity in logarithmic Lagrangean strain–entropy space. *Computer Methods in Applied Mechanics and Engineering*, 198(41-44):3262–3277, 2009.

[33] M. H. Ulz. *A Green-Naghdi approach to finite anisotropic rate-independent and rate-dependent thermo-plasticity in logarithmic Lagrangean strain-entropy space*. Dissertation, Technische Universität Graz, Graz, 2009.

[34] Q.-S. Zheng. Theory of representations for tensor functions—a unified invariant approach to constitutive equations. *Applied Mechanics Reviews*, 47(11):545, 1994.