Department of Construction Sciences
Solid Mechanics

ISRN LUTFD2/TFHF-15/5199-SE(1-70)

Effects of Martensitic Phase Transformation on Advancing Cracks in Austenitic Steel

Master's Dissertation by

Alexander Lundberg and Sara Eliasson

Supervisor:
Håkan Hallberg, Division of Solid Mechanics

Examiner:
Mathias Wallin, Division of Solid Mechanics

Printed by Media-Tryck AB, Lund, Sweden
For information, address:
Division of Solid Mechanics, Lund University, Box 118, SE-221 00 Lund, Sweden
Webpage: www.solid.lth.se

1 Abstract

This paper is based on a constitutive model for an austenitic stainless steel. Previous research in this particular field include investigations of the transformation zones at a crack tip of a stationary crack [27]. Moving on, it would be interesting to consider the transformation toughening related to the martensitic formation. This is the topic of this paper, where propagating cracks are analyzed by exploiting a cohesive zone model.

The phase transformation model [25] is implemented in FORTRAN code. It is utilized in ABAQUS as a user-material subroutine. The cohesive zone model, previously discussed in [33], is also implemented in FORTRAN code as a user-element subroutine. The goal is to capture the effects of the phase transformation, and how it alters the crack tip behavior. A special cohesive zone model which takes the phase transformation in adjacent continuum elements into account is also developed.

The work carried out in this paper is done to gain a better understanding of the implications martensitic transformation in steel has on propagating cracks. Results are presented for different temperatures when the phase transformation is switched on and off. Some simulations are also carried out with the special cohesive zone model and compared to the regular model.

The first step is looking into theory of the cohesive zones and implementing a user-element subroutine for ABAQUS. Implementing a cohesive zone model and using different traction-separation laws proposed by different authors, then adding the user-material subroutine, one is able to compare the propagation of a crack with and without phase transformation.

The model used is a disc-shaped model with a mode I displacement field applied. Looking at the results, comparing a bilinear traction-separation law and a trapezoidal traction-separation law, there are clear differences with martensite present and with a crack propagating without phase transformation. The biggest differences occur at low temperature when significant amounts of phase transformation take place. Also altering the cohesive zone model can give additional effects.

This research is purely based on previous theory by several different authors. It would be interesting to carry out experiments as a reference for the simulations. This could also help gain better a better understanding of how the special cohesive zone model affects the fracture mechanisms.

2 Preface

This report is a master's thesis written during the spring of 2015 at the Division of Solid Mechanics at Lund University, Faculty of Engineering. It is a continuation of the bachelor thesis written during the autumn of 2014 [33], where cohesive zones were studied. With the knowledge from the bachelor thesis on cohesive zone models, a user-element subroutine is implemented in FORTRAN code. The subroutine is used in ABAQUS together with a constitutive model where the martensitic phase transformation is an integral part. The constitutive model is implemented as a user-material subroutine. The theories are combined to examine a mode I crack in a disc shaped specimen. The aim of this project is to examine the transformation toughening related to the martensitic formation.

We want to thank Håkan Hallberg, our supervisor, for the great support he has given and for the commitment he has shown. We also want to thank Mathias Wallin, our examiner, and the people at Solid Mechanics for helping us out when we encountered different problems and for showing great interest in our work.

2.1 Individual Contributions

During this thesis work both Alexander Lundberg and Sara Eliasson have tried to distribute the work evenly among themselves. They have collaborated, trying to discuss problems and trying to make sure they both have equal knowledge about the problem investigated. Both find it important that they can answer questions that may arise.

To simplify things, Sara has had the main responsibility for the report and Alexander the main responsibility that the correct results are produced.

2.2 Ethical considerations

Ethical aspects are not relevant to the subject at hand.

Contents

3 Introduction

The objective of this master's thesis is to build on the work on cohesive zones, previously presented in [33], and combine it with a specific material model developed by Hallberg [25] to gain a better understanding of how the martensitic phase transformation in austenitic steel influences crack propagation.

A bachelor thesis on the theories and implementation of cohesive zones was carried out during the autumn of 2014, and serves as a starting point for the work in this master's thesis project. The aim of the bachelor thesis was to get a sound theoretical foundation, and to enable the investigation of more complex problems.

In the article published by Hallberg in 2007 [25], a constitutive model for martensite transformation in austenitic stainless steel is derived. In a subsequent article, also by Hallberg, from 2011 [27], results are presented for a stationary crack where the martensite transformation at the crack tip is included. Using this constitutive model with a cohesive zone model on an advancing crack would make it possible to investigate how the martensite transformation influences the crack propagation.

As a first step the simulations are run with a cohesive zone model with a constant traction-separation law. The behavior of the crack tip will be compared for cases when the phase transformation is active and then when no phase transformation is present.

Using a cohesive zone model for this problem the first simulation might show a difference in behavior already without considering the influence of phase transformation on fracture properties explicitly. But taking into consideration that the fracture toughness is different for the martensitic and austenitic phases, the material response in the vicinity of the crack tip will change and the traction separation law *might* have to account for these changes. Examples of possible alterations that could be made to the cohesive zone model to account for the phase transformation are proposed.

The phase transformation model [25] is written in FORTRAN code. It can be implemented to ABAQUS as a user-material subroutine. The cohesive zone model is also implemented in FORTRAN code as a user-element subroutine to ABAQUS. The goal with the simulations is to capture the effects of the phase transformation, and how it alters the behavior of the crack tip.

4 Resources and Time

The master's thesis is written at the Division of Solid Mechanics at Lund University, Faculty of Engineering. The division provides us with a workspace where we can carry out the work with the thesis. In the workspace two computers are available to our disposal. The computers are installed with ABAQUS, Matlab and other programs we might need during the thesis work.

Our supervisor, Håkan Hallberg, and our examiner, Mathias Wallin, are both located at the University. It is important to us that Håkan has a good overview of how our work is progressing. Since he is close at hand, it is convenient to drop in or book a meeting with him if any questions or difficulties occur.

The master's thesis covers 30 hp and should be distributed over a 20 week period as a recommendation. A preliminary time schedule is proposed in table 4.1.

Week	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
Writing the project plan																				
Approval of project plan																				
ABAQUS:																				
User-defined subroutine																				
Advancing crack model																				
Learning to use constitutive model																				
Results																				
Report:																				
Literature study on phase transformation																				
Literature study on material parameters																				
Literature study on the constitutive model																				
Writing the report																				
Opposition																				
Submiting the report																				
Presentation																				

Table 4.1: Preliminary time schedule for the master thesis

The actual time schedule is shown in table 4.2.

Week	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
Writing the project plan																				
Approval of project plan																				
ABAQUS:																				
User-defined subroutine																				
Advancing crack model																				
Using the constitutive model																				
Results																				
Report:																				
Literature study on phase transformation																				
Literature study on material parameters																				
Literature study on the constitutive model																				
Writing the report																				
Opposition																				
Submiting the report																				
Presentation																				

Table 4.2: Actual time schedule for the master thesis

The outcome of the time plan is very satisfying, see table 4.2. The literature study consumed about as much time as calculated, but was carried out at different stages of the project. The biggest difference between the two time plans is when working with simulations and looking at results. Since the simulations turned out to be very time consuming this was initiated earlier and continued for a longer time period. The subroutines were continuously worked on and altered throughout the thesis work. There is not a definition when this part is done. Overall the time schedule is good and there was time to complete all tasks.

5 Material Properties

The implemented model is based on calibrations for an austenitic stainless steel. At first there will be an introduction to the material and its properties and the phase transformation involved.

5.1 Applications

Austenitic stainless steels are commonly used in engineering applications due to their versatility and exceptional mechanical properties. An important property of an austenitic steel is its excellent corrosion resistance against different hostile environments. The mechanical properties of stainless steel have a determining role for the suitability of the material for particular uses.

Areas of application involve use at cryogenic temperatures, where the low temperature ductility is a crucial factor. Cryogenic applications involve missiles, space vehicles and liquid natural gas storage tanks. In these applications the low temperature toughness is an essential property. The low temperature toughness is closely connected to brittle fracture, and this can be one reason to investigate how martensitic transformation can affect the crack propagation.

5.2 The Austenitic Stainless Steel

The constitutive model is calibrated against experimental data for a Ni-Cr steel. The steel is referred to as AISI 304 (SUS 304), austenitic stainless steel [26]. The composition is 18% chromium and 8% nickel. It has a carbon content of maximum 0.08 wt% [2].

5.3 The Crystal Structure of Metals

Today metals are used for a wide range of applications. They have a variety of properties which are useful in different areas. Important mechanical properties of metals include strength, ductility, toughness, high melting point and thermal and electric conductivity. The high strength of metals and the fact that they can be formed into geometries such as wires without breaking indicate a strong bond between the atoms in the metal.

Metals have an organized crystalline structure in the solid phase. The crystal lattice of a metal consists of ions, and the ions share the electrons in the outer atom shell and the outermost electrons are not said to be associated with just one ion. As a result the electrons are very mobile which leads to a strong and uniform metallic bond. Since the electrons find it easy to move around this allows for the good heat and electrical conductivity seen in the metals [12]. The crystal structure of the metal is packed

as closely as possible and can be arranged in different ways. The most commonly occurring crystal structures include the BCC-, FCC- and HCP-structure.

BCC stands for body-centered cubic. The BCC unit cell has a net total of two atoms; one in the center and eight eighths in the corners of the cube from pieces of atoms, see figure 1. The FCC-structure is a face-centered cubic arrangement. The FCC unit cell has a net total of four atoms; eight eighths in the corners of the cube, as the BCC-structure, and six halves of atoms on the faces of the cube, see figure 2. The atoms in a FCC layer pack themselves in an empty space in the layer of atoms in the adjacent layer. This results in a higher packing factor for a FCC compared to a BCC. The packing factor is a number for the packing efficiency or packing fraction of the volume in the structure that is occupied by particles [1]. In this report austenitic steel, which has a FCC-structure crystal structure, is considered.

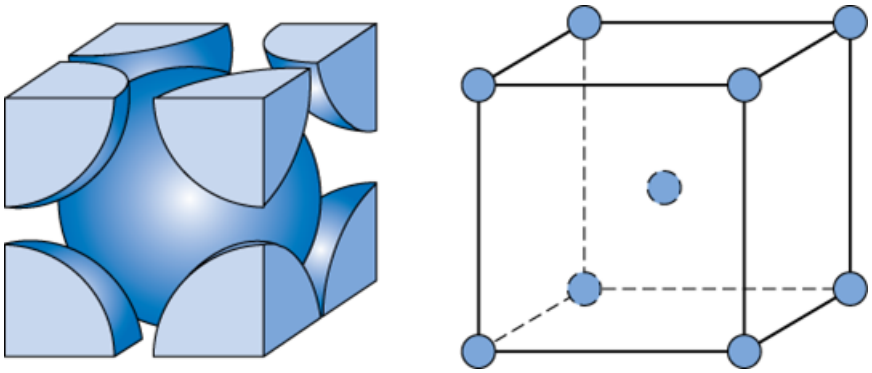


Figure 1: An illustration of the atomic placing of the BCC-structure in a unit cell [1].

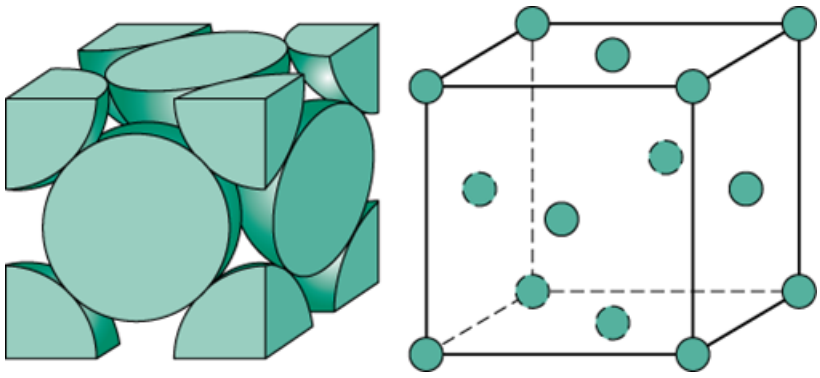


Figure 2: An illustration of the atomic placing of the FCC-structure in a unit cell [2].

The HCP-structure is a hexagonal close-packed crystal structure. Not all structures are cells with cubic symmetry such as FCC and BCC. The HCP-structure's top and bottom face of a unit cell consists of six atoms forming a regular hexagon surrounding one single atom in the center. Another plane in-between the top and bottom faces consists of three additional atoms situated in-between the atoms of the other two planes, see figure 3. A total of six atoms are contained in one unit cell, one sixth of each the top and bottom faced atoms surrounding the center atom which contributes with one half, and a total three atoms in the middle plane. The packing factor for

a HCP is the same as for a FCC [12]. The investigated martensite transformation will transform from austenite with an FCC-structure to an HCP-structure and finally reach a BCC-structure.

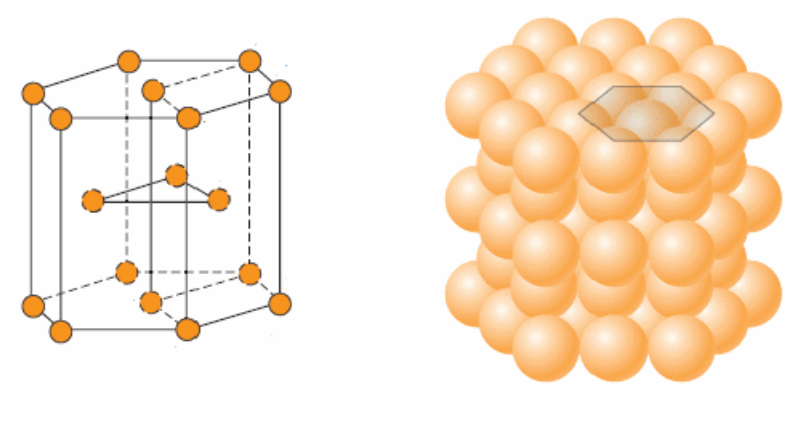


Figure 3: An illustration of the atomic placing of an HCP-structure in a unit cell [3].

5.4 Phase Transformation

Metallic materials possess a wide range of mechanical properties. The mechanical properties are reliant on the characteristics of the microstructure. The development of different kinds of microstructures almost always involve some kind of phase transformation. Phase transformation is a change in the number of phases and/or the characteristics of the phases. There are a variety of phase transformations which are important in the processing of materials. Usually all kind of phase transformations involve some alteration of the microstructure.

There are three main classifications of phase transformation. The first group involve simple *diffusion-dependent transformations*. This type of process involve no change in either the number or composition of the phases present. This group include solidification of a pure metal, allotropic transformations, recrystallisation and grain growth. The second classification is also a diffusion-dependent transformation but here there is some alteration in phase composition and also often in the number of phases present. In the final microstructure there are often two phases present. An example of a reaction included in this classification is the eutectoid reaction. Now the final and third classification is a *diffusion-less transformation*. A transition occurring without diffusion means that the chemical composition remains constant but the atoms are organized in a new crystallographic lattice [10]. In this transformation a metastable phase is produced. High temperature applications generally promote diffusion-dependant phase transformation, while lower temperatures promote the diffusion-less martensitic transformation. The martensitic transformation is a diffusion-less transformation and will be investigated further in this report.

There are two distinct stages in the phase transformation process; nucleation and

growth. The nucleation involves the formation of numerous small particles of the new phase. They then start to increase in size and grow until equilibrium is reached. Most often these two stages are present in the phase transformation and do not occur instantaneously. This is not the case for martensitic transformation. Since it is a diffusion-less process it occurs almost instantaneously. The martensite grains nucleate and grow at a very rapid speed within the austenite. This entails that the martensitic transformation, for all practical purposes, is time independent [12].

5.5 The Phases of Steel

Steel is an alloy that consist of iron and carbon. When you heat a piece of steel to a specific temperature, the steel goes through a phase change. Even if it is not always an obvious change, like the transition from solid to liquid, the crystal structure changes. The main factor controlling the transformation is temperature, but stress, cooling rate and alloy or chemical composition can influence the temperature at which changes take place.

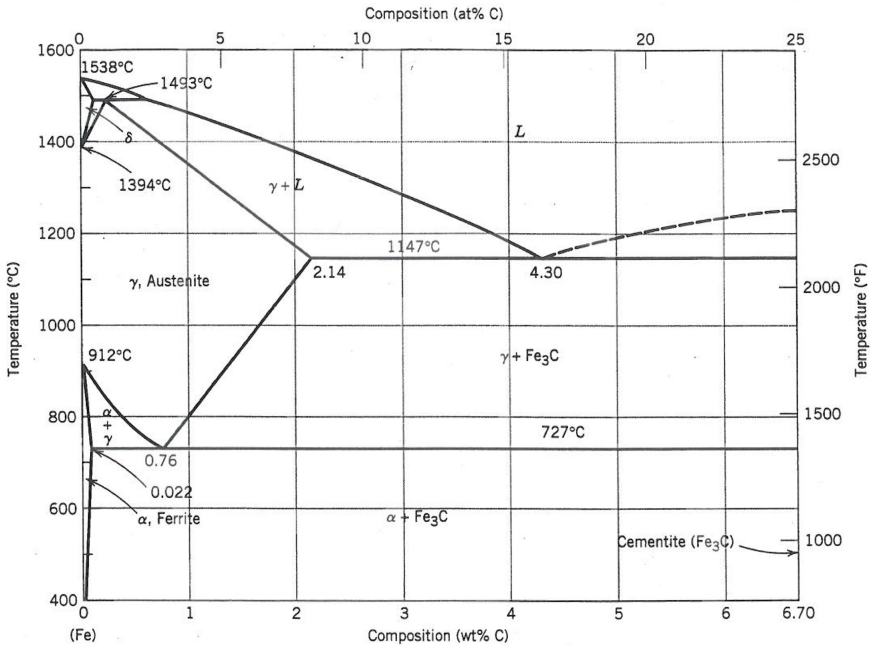


Figure 4: An illustration of the phase diagram for Fe-C [4].

In figure 4 the different phases of a Fe-C alloy are shown. In this phase diagram the influence of alloy composition and temperature changes on phase transitions and solidification is represented graphically. Fe-C has a number of different microstructures appearing at various temperatures and carbon contents. These different microstructures are the crystalline structures that the steel consist of at different temperatures. There is ferrite which is a solid solution and is stable at room temperature. There is also cementite which has a much higher carbon content.

Austenite is also known as gamma iron and, as previously mentioned, has an FCC-structure. Austenite cannot be stable in carbon steel when the temperature is below 727°C. Pearlite has alternating thin layers of cementite and ferrite combined. You receive pearlite if you slowly cool austenite. Pearlite always contain 0.77 % carbon. Bainite is fine carbon needles in a ferrite matrix. When austenite is cooled at a rate slower than the rate needed to form martensite, bainite will form instead [12].

The last phase is martensite, which is obtained when austenite is rapidly cooled. Martensite is not present in the phase diagram because it is not an equilibrium phase. The phase transformation of austenite to martensite along with its implications on the crystal structure is of great interest.

Martensite, unlike austenite which consist of a single phase, the γ-phase, has two phases, the ϵ-phase and the α-phase. The ϵ-phase has an HCP-structure and the α-phase has a BCC-structure. When austenite with FCC-structure is to transform into martensite with BCC-structure the HCP-structure is present during the transformation. It turns out as follows; $\gamma \rightarrow \epsilon \rightarrow \alpha$. This was investigated by Mangonon and Thomas in 1970 and the results also showed that the α-phase was only observed after plastic deformation [34]. The theory of austenite transforming from $FCC \rightarrow HCP \rightarrow BCC$ is also supported in an article by Sherby, Wadsworth, Lesuer and Syn [56].

When stainless steel is deformed, the previously mentioned ϵ-phase is a true intermediate phase in the nucleation of α martensite from the austenite matrix, c.f. [65]. The α-phase martensite is stable at low temperatures, while the austenitic phase is stable at high temperatures. Following the transformation from γ-phase austenite to martensite the amount of transformation is dependent on the temperature and the deformation applied [16].

6 Martensitic Transformation

Little is known about the martensitic transformation. It is, however, known that a large number of atoms experience cooperative movements. This involves a slight change in distances and placement of atoms relative to its neighbours. The movement occurs in such a way that it is possible for the FCC austentite to experience a polymorphic transformation into BCC martensite. All the carbon atoms remain as interstitial impurities in the supersaturated martensite which together with the structure change results in a increase of the volume. Since the martensite transformation is a diffusion-less process it involves only a change in the crystal structure.

The martensitic phase evolves differently depending on the amount of carbon in the steel. At a high carbon content, greater than 1.0 wt%, the martensitic structure changes to a plate-like structure adjoining regions of untransformed austenite. For a carbon content up to 0.6 wt% the structure is described as massive, cubic, lath-like, lenticular, subgrain-containing bundles [56]. If the content is between 0.6-1.0 wt% a mixture of the two structures will occur simultaneously or sequentially during the quenching process. The crystal structure is also important and influenced by the carbon content. Martensite will have a BCC-structure if the content of carbon is less than 0.6 wt%. The different martensitic structures that can be obtained from austenitic steel is due to abrupt changes in the crystal structure and related to the carbon content and changes in properties [56].

As mentioned there is a volume change within the material when it undergoes a phase transformation into martensite. It was shown by Greenwood-Johnson [23] that the volume change during phase transformation can permit plastic deformation of the material even if the loading stress is kept constant beneath a certain level where plasticity is induced. The plastic deformation is induced in the weaker of the two phases present. It has also been shown by Magee (1969) that external loading can trigger martensite formation in preferred directions. These preferred directions cause an orientation effect that influences the macroscopic shape of the loaded body [25]. Looking at the meaning of the Greenwood-Johnson mechanism and the Magee effect conclusions can be drawn that the material undergoing a martensitic phase transformation will show both a volumetric and a deviatoric response along with directional dependency.

Martensite is the hardest and strongest microstructure but also the most brittle. In fact, martensite has negligible ductility. The hardness is dependent on the carbon content, which can range up to 6 wt%. The hardness and strength of martensite is not thought to be related to the microstructure as for pearlitic steels. These properties are said to come from the effectiveness of the interstitial carbon atoms that are set in places that prevent dislocation motion [12]. The fact that martensite is more brittle than the FCC austenite is based on the fact that when the BCC martensite structure is deformed there is a tendency for micro cracks to be formed in the crystal and the intense stress concentration at the tip of such a crack causes it to extend, thereby it would produce a more rapid fracture [44].

The transition from austenite to martensite can evolve in two ways, either by inducing changes in temperature or by variations in the stress- or strain-state of the material. As mentioned austenite is said to be a metastable phase. The term "metastable" is phenomenologically defined by the fact that the parent austenitic matrix will undergo a diffusion-less shear type of martensite tranformation when the material is deformed or cooled to a low temperatue [44]. The growth of the martensitic phase is strongly dependent on the temperature and a too high temperature restrains the martensite growth while a lower temperature promotes it [25]. In cryogenic applications, metastable austenitic stainless steels are liable to martensitic transformation in real operational conditions.

As illustrated in figure 5, the temperature needed to obtain martensite from quenching is very low relative to the one where austenite occurs, and the time needed to reach this temperature is very short.

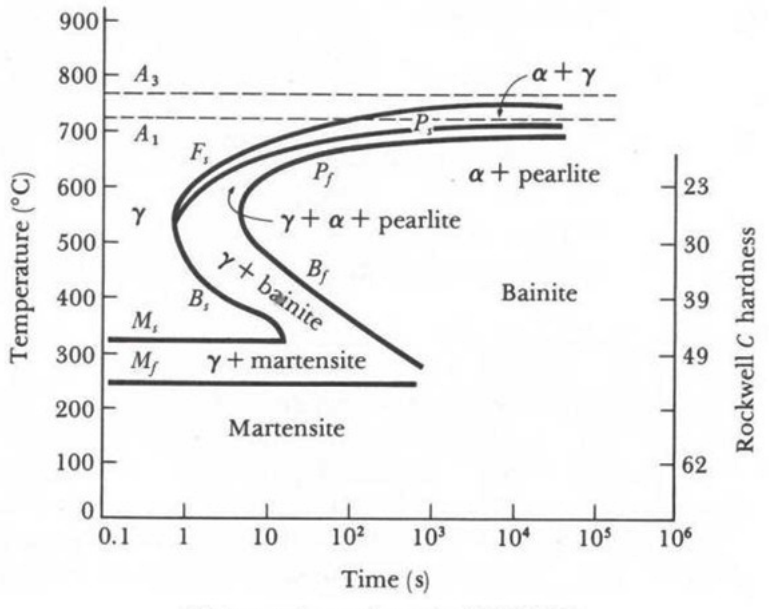


Figure 5: Time-Temperature-Transformation diagram for a Carbon-Steel, taken from [5].

Forming martensite from austenitized iron-carbon alloys by heat treatment, the martensitic phase evolves when the alloy is rapidly cooled to a relatively low temperature. This process is called quenching, which was mentioned earlier. Martensite may be thought of as a competitive transformation product to pearlite and bainite. The rapidness of the cooling process has to be sufficient enough, effectively preventing any carbon diffusion. Any diffusion whatsoever will result in formation of cementite and ferrite phases.

Since the martensitic phase also evolves from variations in the stress- or strain state it is interesting to know the effects of deformation. The formation of martensite involves homogeneous distortion of the parent structure, and it is expected that the externally applied stresses are of importance. Plastic deformation is useful when

forming martensite above the temperature where at martensite is formed, M_s. This is valid if the temperature does not exceed a critical temperature, M_d. Applying elastic stresses when the temperature is over M_s and maintaining them during cooling, can affect the transformation as well. Uniaxial compression or tensile stresses raises the M_s temperature and hydrostatic stresses lower the M_s temperature [60].

There is always a preferential crystallographic structure that corresponds to the lowest energy level. Compared to the original austenitic structure, which is stable at high temperatures, the martensitic phase has a thermodynamically preferred crystal structure at relatively low temperatures. This is why when quenching austenite it tends to transform into martensite.

For a martensite transformation to occur an energy barrier must be overcome. This energy barrier arises from the fact that during the phase transformation, crystal atoms will for a brief period of time assume configurations not corresponding to the lowest energy level. This creates an energy barrier that needs to be overcome. If the driving force for the martensitic transformation is not sufficient, the energy barrier can not be overcome and the austenite is not transformed. It is said to be metastable. When lowering the temperature the difference in energy level of the two structures increases. This difference is the driving force for the transformation, and so when austenite is cooled rapidly below a temperature, M_s, a spontaneous transformation to martensite occurs. Altering the driving force can also be done by supplying the system with mechanical work in form of an applied stress. Transformations above the temperature M_s is said to be mechanically induced. It is also known as stress-assisted since an applied stress is required to provide the additional driving force for the transformation [62].

6.1 Stress- and Strain-Induced Martensite

The fact that austenitic steels are said to be metastable at low temperatures makes it possible for the austenite to undergo a spontaneous transformation into martensite if it is exposed to a sufficient amount of stress or strain [36]. The volume expansion connected to the transformation makes the transformation more extensive under a tensile load than compared to a compressive load [20].

In certain steels, termed TRIP-steels, the carbide precipitation decreases the stability of the austenite. This reduction in stability is caused due to the precipitated carbides wiping out alloying elements in the austenite matrix and thus favoring the martensitic transformation. Often the austenite strength in steels is increased by increasing the dislocation density and the number of carbide precipitates, but not in all cases. The thermo-mechanical treatment in these cases is used to achieve a balance such that the transformation to martensite is obtained on straining, but not at a too low level of stress.

The martensitic phase transformation is a mode of plastic deformation and it may be either stress- or strain-induced. A strain induced transformation only occurs after

plastic deformation has taken place in the austenite. If the transformation is stress-induced the martensite is transformed before any plastic deformation has occurred.

The fracture toughness is dependent on the exact characteristics of the transformation. Strain-induced martensite formation is expected to increase the fracture toughness, as discussed in [8]. This is believed to stem from the highly dissipative phase transformation which, together with the plasticity in the austenite, reduces the energy available for crack propagation and consequently increasing the toughness.

If the martensitic transformation is induced by small elastic stresses in the austenite without plastic deformation, very little of the austenite would be left and its contribution to the overall toughness would be negligible. The stress level at which martensite is formed plays a big role in the toughening of the material. The strain energy reduction seen during the martensitic formation is dependent on the stress at which martensite is formed and also its related so-called invariant shear strain. Thus, lowering the stress level at which the transformation occurs will reduce the effect of toughening in the material [8].

6.2 Transformation Toughening

The process of phase transformation under straining can be seen as a mode of plastic deformation. Hence, the process is capable of absorbing part of the elastic strain energy in the body otherwise available for crack extension. The phase transformation can also greatly influence the mechanical properties of the material. Therefore it is theoretically possible that the phase transformation could lower the fracture toughness, despite the dissipation. The martensitic phase, for example, is hard and brittle and the fracture toughness could therefore be reduced as a result of the phase transformation. If the energy absorbed by the transformation process is insufficient to offset the general brittleness of the martensitic phase the net toughness could therefore decrease. It has, however, been indicated that the toughness increment associated with the martensitic phase transformation in austenitic steels is generally positive [7].

In a study by Yi and Gao [69], transformation zones for stationary and advanced cracks have been analysed in shape memory alloys. In this study it is shown that the martensite transformation will increase the toughness of the shape memory alloy and reduce the stress intensity at the crack tip.

6.3 Fracture Toughness for Austenite and Martensite

There is a substantial difference in strength between the phases austenite and martensite [62]. Investigators have suggested that stress- or strain-induced phase transformation enhance the energy absorption and thereby increase the fracture toughness [48]. The term "toughness" is used to describe the ability of a material to absorb energy before and during fracture. Brittleness and ductility are qualities further used to de-

scribe materials which exhibit low and high toughness, respectively, on a macroscopic scale.

At the macroscopic scale the most brittle fracture mechanism is known as cleavage. Fracture occurs by separation along crystallographic planes across which atomic bonds have been broken. The fracture surface is flat and shiny. Comparing this to the needle shaped martensite produced by strain-induced transformation, the plate-like martensite will behave in a more brittle manner. The ductile fracture process involves nucleation, growth and coalescence of voids, which form around particles or inclusion present in the material matrix, forming elliptical cracks [62].

When the crack propagates the fracture toughness can vary due to the phase transformation. The behavior of the fracture process can also vary from a ductile to a more brittle manner. Experiments carried out by Stavehaug revealed that the fracture mode can be very different when transformation is present, with micrographs showing large amounts of crack tip blunting as well as crack branching [62].

There are many different mechanisms contributing to the toughness increment active at the same time and it is important to understand their individual contribution and how they can be modelled. The contribution from the dissipative transformation process as well as the relaxation of stresses due to the volume expansion has to be taken into account by the constitutive relation for the continuum.

A method for modelling the alteration of the fracture process, which is not captured by the continuum model, is also proposed in this paper. The model can account for the decreased ductility by exploiting a special traction-separation law which is dependant on the martensite fraction, z, cf. section 11.

6.4 Effects of Martensitic Transformation

Many studies have been made on the effects of martensitic transformation. As previously mentioned, when studying martensitic phase transformation and its implications on fracture there are a few aspects to look at. The summary of the relevant work covering this field of research suggests that the martensitic transformation can decrease the crack growth rate [36].

Hornbogen [28] investigated austenitic steels capable of strain-induced martensitic transformation. The primary topic of his paper is fatigue crack growth, and an analysis of the volume change occurring when the austenite transform into martensite is carried out. Results pointing to volume increases of up to 5% ahead of the crack tip have been reported [35]. The volume change is dependent on the composition of the metal and it is according to [36] and [20], typically about $2-3\%$ for the 304-type stainless steels. If the material that increases in volume is surrounded by a sufficient amount of untransformed material compression stresses will occur. These compression stresses, if high enough, can minimize the stresses around the crack tip and contribute to slowing down the crack propagation. If the compression stresses are very high they can even

cause a closure of the crack.

In another paper by Mei et al. [36], on the topic of fatigue crack growth in 304-type stainless steel the author concludes two important things from the experiments which were carried out. At first, the deformation-induced martensitic transformation increases the fatigue resistance. Mei then investigates the different mechanisms contributing to influence on martensite transformation on crack growth. There are many different mechanisms, for example incremental strain added to the crack tip from the martensite transformation, dual phase microstructure on the crack path, but the most important one is the transformation strain appearing. In the crack tip an interference in the crack tip stress field is caused by the strain connected to the martensite transformation. Looking back at the different types of phases in the steel, the $\gamma \rightarrow \alpha$ transformation involves a volume expansion and a shear strain of about 10%. The constraint of the surrounding material not experiencing a volume expansion places the region under compression.

A lot of authors have been looking into fatigue crack growth in relation to phase transformation. The cohesive element implemented in this project has a damage formulation that defines the unloading and reloading behavior of the interface, which makes it suitable for cyclic loading. Since the continuum material model used in this paper is implemented with isotropic hardening, effects on hardening by cyclic loading are not captured. If cyclic loading and fatigue crack growth was to be analyzed, kinematic hardening should be implemented.

6.5 The constitutive model for phase transformation

The constitutive model used is previously implemented by Hallberg [25]. It is a model for diffusion-less phase transitions calibrated for an austenitic stainless steel undergoing large deformations. The model is derived within a thermodynamic framework, where the second law of thermodynamics plays an important role.

The model will not be discussed further in this paper. Instead the reader is referred to the article published by Hallberg, cf. [25].

7 Linear Elastic Fracture Mechanics

There are three different types of pure crack modes, see figure 6. Mode I is when the loading is applied normal to the crack plane and tends to open the crack. Mode II is in-plane shear loading and tends to slide the crack faces over each other. Mode III is out-of-plane shear, as seen in figure 6 the crack edges are moved across each other in opposite directions [6]. Also mixed modes may occur as combinations of any of the three basic modes.

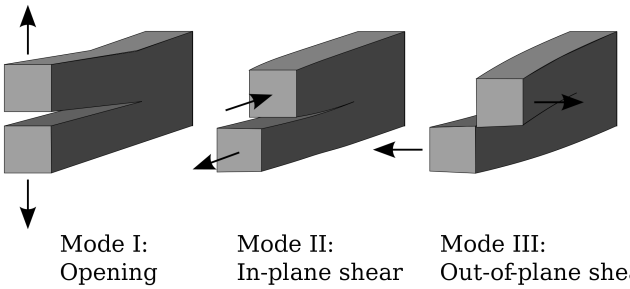


Figure 6: Illustration of the three different crack modes, taken from [6].

Early work in the field of fracture mechanics regarding stress concentrations around elliptical holes was developed by C.E. Inglis in 1913 [29]. His theory predicted that the stresses at a perfectly sharp crack tip approach infinity. In other words; that the material would have zero strength [51]. Instead of analysing the stress state in the vicinity of the crack tip, A.A. Griffith developed a theory based on energy-balance [24].

The energy criterion is based on the principle that crack extension can only occur when the energy available for crack growth is sufficient to overcome the resistance of the material. This resistance includes all types of energy dissipation related to crack extension, for example surface energy and plastic work [6].

The early work by Griffith was later developed further by G.R. Irwin [30]. Irwin defined the energy release rate, G, as the rate of change in potential energy with crack area for a linear elastic material. Crack extension occurs when the energy release rate reaches a critical value, $G = G_c$, which is a measure of fracture toughness.

On the atomic scale, fracture occurs when the applied stress and work are sufficient to sever the atomic bonds. The bond between atoms is caused by the attractive electrostatic force between opposite charges. The cohesive strength can be derived in terms of the cohesive stress. This is shown in figure 7 and provides the theoretical strength as

$$\sigma_c \approx \frac{E}{\pi} \tag{7.1}$$

Experiments have, however, shown that the actual fracture strength is several orders

of magnitude lower than the theoretical value. This discrepancy is related to stress concentrations around flaws in the material which magnify the stresses locally.

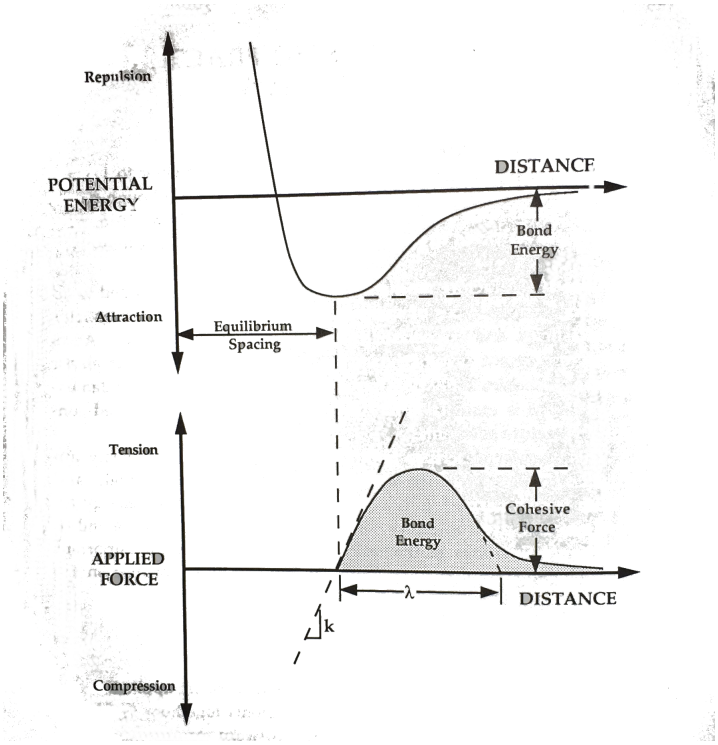


Figure 7: Potential energy and force as a function of atomic separation, taken from [7].

Griffith's energy balance is based on the first law of thermodynamics which states that a system that goes from a state of non-equilibrium to a state of equilibrium will have a net decrease in energy. The interpretation of the application on cracks is that a crack can only grow if the fracture process results in constant or decreased total energy [24]. Hence, the critical condition for crack extension can be defined as the point where crack growth occurs under equilibrium conditions, with zero net change in total energy [6]. The balance equation under equilibrium conditions for an incremental increase in crack area, dA, can therefore be expressed as

$$\frac{dE}{dA} = \frac{d\Pi}{dA} + \frac{dW_s}{dA} = 0 \tag{7.2}$$

which is equivalent to

$$-\frac{d\Pi}{dA} = \frac{dW_s}{dA} \tag{7.3}$$

where E, is the total energy, W_s, is the work needed to create two new surfaces and Π is the potential energy in the form of strain energy and work done by external forces. For an edge crack, two new surfaces are created when a crack is formed. Thus the expression for W_s takes the form

$$\frac{dW_s}{dA} = 2\gamma_s \tag{7.4}$$

where γ_s is the material specific surface energy.

Griffith first developed his theory while investigating cracks in glass. It has since been shown that his equation is valid for ideally brittle solids. The theory does, however, greatly underestimate the fracture strength of ductile materials such as metals. A modification to Griffith's approach was later developed by Irwin and Orowan [51]. The modified version of the energy-balance equation accounts for plastic flow. In brittle materials cracks form simply by breaking the atomic bonds, and the surface energy represents the total energy of the broken bonds in a unit area. Crack propagation in ductile metals, on the other hand, include a plastic zone in the vicinity of the crack tip. In this plastic region dislocation motion occurs. The process of plastic flow around the crack tip contributes to additional energy dissipation. The modified expression takes the form

$$W_f = \gamma_s + \gamma_p \tag{7.5}$$

where γ_p is the plastic work per unit area of surface created. It has been shown that fracture in ductile metals under highly plastic conditions involve dissipation mostly due to plastic work, rather than separation. In other words γ_s is small compared to γ_p as discussed by Siegmund and Brocks, cf. [59].

Irwin later developed an energy criterion for fracture that is, in essence, equivalent to Griffiths criterion. The concept is based on the energy release rate, G, defined as a measure of energy available for an increment of crack extension [6]

$$G = -\frac{d\Pi}{dA} \tag{7.6}$$

Crack extension occurs when the energy release rate reaches a critical value, i.e when

$$G = G_c = \frac{dW_s}{dA} = 2W_f \tag{7.7}$$

Important to note is that these theories are derived under the assumption that the material response is strictly linear elastic. This means that inaccurate results may be produced when applying the model on non-linear problems. The criterion for this theory to be valid is that the global behavior of the structure must be linear elastic, while plasticity must be confined to small regions around the crack tip [6].

8 The Cohesive Zone Model

Failure and fracture is a big part of many fields in engineering. It is therefore important to gain understanding of failure processes and fractures. To analyze these phenomena efficiently in arbitrary geometries, a general numerical method that can describe the initiation and evolution of a crack is needed. The method should include and be able to simulate the initial loading, the damage initiation with initial debonding, and the damage evolution until complete separation and failure has occurred. A method used for these kind of problems is modeling with cohesive zones. Cohesive zone models have been proven useful in many different varieties of fracture issues in homogeneous solids as well as in analyzing interface crack problems.

In this project a cohesive zone model will be used to analyze crack propagation. For the analysis a numerical FE model will be implemented. The cohesive zone is defined as an interface inbetween the structure faces where the crack advance will take place. The interface consists of cohesive elements which are set to behave in the same way as a crack would when it propagates. So for a cohesive zone model no elements but the cohesive elements are damaged. The crack will only propagate where cohesive elements are modeled. This requires a pre-defined crack path.

Cohesive zone models are based on theory from Barenblatt and Dugdale [9, 18]. Their method for using cohesive zones to represent a crack propagation path is very similar to Griffith's theory based on a surface energy that measures the resistance against crack advance.

Barenblatt and Dugdale's theory on cohesive zones is based on a method trying to explain how a crack advances. The cohesive zone is formulated to imitate the behavior ahead of the crack tip. Both Barenblatt and Dugdale divide the crack surfaces into two separate regions, one part that is stress free and another part which is loaded by cohesive stresses. Dugdale assumed that there is a plastic zone near the crack tip. Within this plastic zone there is a stress acting across the crack that is equal to the yield strength, σ_γ. Dugdale examined the yielding of steel sheets and investigated yielding in the sheets at the end of slits. He obtained a relation between plastic yielding and external applied load and found that the influence of yielding was approximately represented by a long crack extending into the region that had a stress equal to the yield stress. Dugdale's theory holds for plane stress but the crack opening stresses can be greater than the equivalent stress in a multiaxial stress state.

Barenblatt investigated brittle materials and instead of a constant stress in the cohesive zone he let the stress vary with deformation in the zone ahead of the crack tip. The stress of the cohesive zones works as a restraining stress that keeps the separating surfaces together. It corresponds to atomic or molecular attractions. The restraining stress, the stress in the cohesive zone, can be seen as a function of the separation distance, $\sigma = \sigma(\delta)$, see figure 8.

In figure 9 it is seen in each end of a crack the difference between Dugdale's and Barenblatt's theories. Looking at the right end we see Barenblatt's crack model where

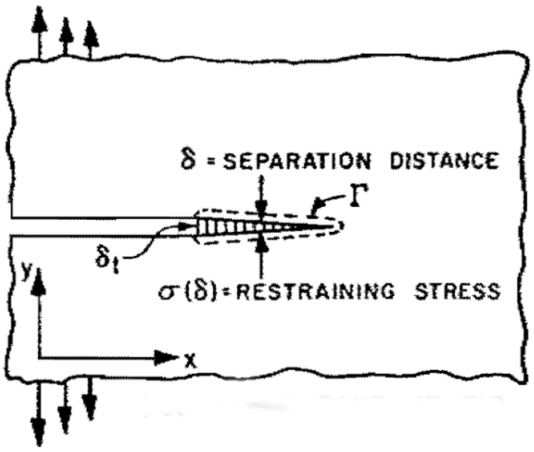


Figure 8: The cohesive zone ahead of a crack tip, taken from [8].

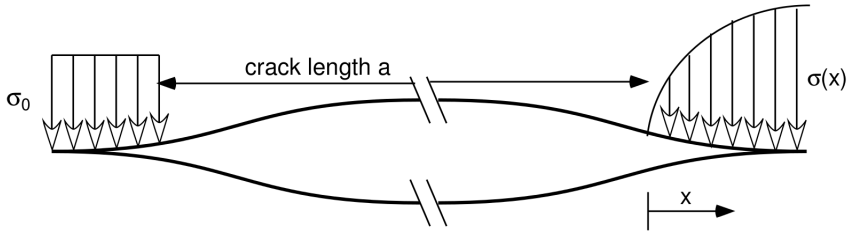


Figure 9: To the left is Dugdale's crack model and to the right is Barenblatt's crack model showing the separation of a crack connected by a cohesive interface, taken from [9].

the cohesive stress varies with the crack tip displacement. To the left Dugdale's crack model where the cohesive zone has a constant stress equal to the yield stress can be seen.

When using cohesive elements a constitutive relation describes the behavior of the failing cohesive material elements in front of the crack tip. The constitutive relation for a cohesive interface is such that the traction across the interface varies depending on the separation of the crack. With increasing separation, the traction reaches a maximum, then starts to decrease and is eventually reduced to zero. When the traction is zero a complete decohesion is allowed. A typical cohesive stress-displacement diagram is shown in figure 10. The constitutive behavior of the cohesive elements is described by a traction-separation law. The parameters that set the properties of the cohesive zone is the cohesive strength (a peak stress required for separation) and a cohesive energy (separation work per unit area) [57, 64].

Looking at figure 8, Γ represents the contour tracing an arbitrary path surrounding the crack tip, and δ_t is the separation distance at the crack tip. To connect cohesive theory with Griffith's work we evaluate the J-integral for the cohesive zone as

$$J = \int_{\Gamma} (W \mathrm{d}y - \mathbf{T}\frac{\partial \mathbf{u}}{\partial x}\mathrm{d}s) \tag{8.1}$$

Shrinking the contour, Γ, down to the lower and upper surface of the cohesive zone and making use of the path independence of the line integral results in $dy = 0$ on Γ, the J-integral becomes

$$J = -\int_{CZ} \sigma(\delta)\frac{\mathrm{d}\delta}{\mathrm{d}x}\mathrm{d}x = -\int_{CZ} \frac{\mathrm{d}}{\mathrm{d}x}\left\{\int_0^{\delta} \sigma(\delta)\mathrm{d}\delta\right\}\mathrm{d}x = \int_0^{\delta_t} \sigma(\delta)\mathrm{d}\delta \tag{8.2}$$

When a cracked structure is exposed to some external loads the crack surfaces are subjected to forces which restrains them from separating. These forces can be seen as cohesive forces. The cohesive stress is a function of the relative displacement between the crack surfaces, $\sigma = \sigma(\delta)$. The external loads will increase δ until it reaches δ^*, see figure 10. When δ^* is reached the bond between the crack faces breaks and new free surfaces are created.

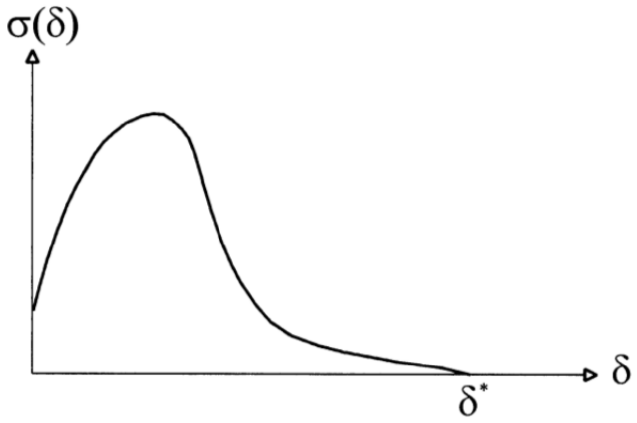


Figure 10: A typical stress-displacement diagram for a cohesive element, taken from [10].

When two new free surfaces are created, the atoms can be considered to be pulled apart. They are slowly moving out of range from their neighbours. For the process where new, traction-free surfaces are created, the cohesive stresses perform an amount of work according to

$$W = \int_0^{\delta^*} \sigma(\delta)\mathrm{d}\delta \tag{8.3}$$

According to equation (8.2) this relation is equal to the J-integral. To propagate a crack through the distance Δa, the surface energy needed corresponds to

$$\Delta U_s = \int_0^{\Delta a} \int_0^{\delta^*} \sigma(\delta) \mathrm{d}\delta \mathrm{d}x = \Delta a \int_0^{\delta^*} \sigma(\delta) \mathrm{d}\delta \tag{8.4}$$

The area under the traction-separation curve is by definition twice the surface energy. Recalling equation (7.4), γ_s is the surface energy for one new free surface created and for a crack we have two new free surfaces. This gives us

$$\int_0^{\delta^*} \sigma(\delta) \mathrm{d}\delta = 2\gamma_s + \gamma_p \tag{8.5}$$

If the cohesive zone is negligible in size compared to the characteristic lengths of the structure around the crack it can be concluded that Griffiths theory and the theory of atomic cohesive forces are identical, cf. equation (7.4).

The information above on cohesive zone models is collected from several authors, see references [4, 18, 37, 39, 50, 53, 68]

8.1 Traction-Separation Law

The cohesive elements are initially made of a damage-free bulk material which will eventually describe the damage and failure of the structure. The damage behavior is typically described by a traction-separation law, as discussed in the previous section, also cf. [54].

Depending on the material, one has to choose a suitable traction-separation law which correctly describes the fracture behavior of the particular material. The cohesive elements model the initial loading, the damage initiation and the damage evolution. These three parts can be identified in a curve for the traction-separation law [3].

When modeling fracture in ductile materials, such as steel, the deformation process involves the processes of elasticity, plasticity and damage. This means that the shape of the traction-separation law cannot easily be determined experimentally, and would have to be assumed [54]. The cohesive model is a phenomenological model and it does not model the real physical fracture process. There is no evidence for which the correct traction-separation law is, and several different approaches can be found in the literature.

The traction-separation laws used in this report are implemented in the user-defined element subroutine, UEL, for ABAQUS.

9 Abaqus Modeling

To be able to assess the implications of the phase transformation in the vicinity of a crack tip in austenitic steel and simulating the propagation of a crack using cohesive elements, ABAQUS is used. ABAQUS is a commercial FE software. The constitutive model is implemented as a user-defined subroutine, UMAT (User MATerial). This makes it possible to create a model and then run it with the desired constitutive model. The cohesive element is implemented in a user-subroutine, UEL (User ELement).

The user subroutines are called through the input-file to ABAQUS. The input-file is a file that generates the geometry and the mesh. It also contains information about the loads and boundary conditions. The input-file calls the user subroutine for the calculations of the cohesive elements and then again for the constitutive law connected to the material of the continuum elements. An example of an input-file and how it is written is attached in appendix A.

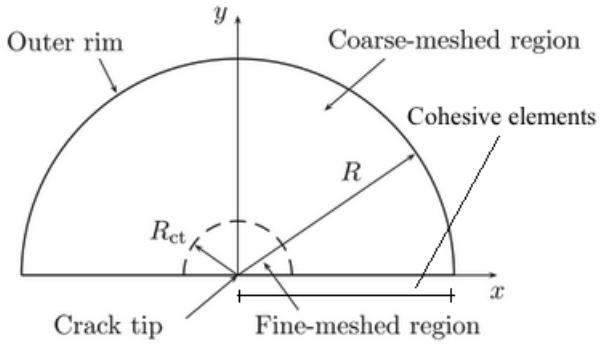


Figure 11: Illustration of the disc modeled in ABAQUS, taken from [11]

For the simulation a disc model is used. The model is implemented as a disc and half of the geometry is seen in figure 11. The bottom left boundary of the half-disc represents the traction-free surface of the crack, while the bottom right side represents the ligament. Along the ligament the cohesive elements are placed. The fine-meshed region in the model is divided into more parts of different fine-meshed regions. The disc has a radius of $200mm$.

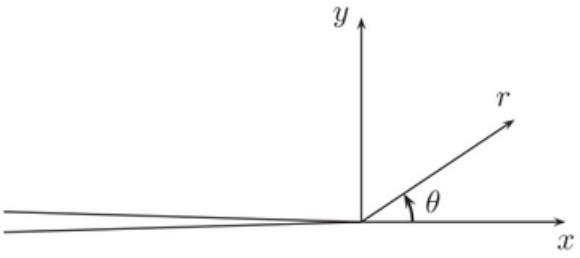


Figure 12: Illustration of the vicinity of a crack tip with its polar coordinates defined, taken from [12]

When modeling in ABAQUS, a cylindrical coordinate system is used. In figure 12 the definition of the polar coordinates in the vicinity of the crack tip can be seen. This coordinate system is used for the displacement field in x- and y-direction for the outer rim of the disc.

A picture of the final ABAQUS model from the CAE of ABAQUS can be seen in figure 13. In this figure the boundary conditions applied are also shown.

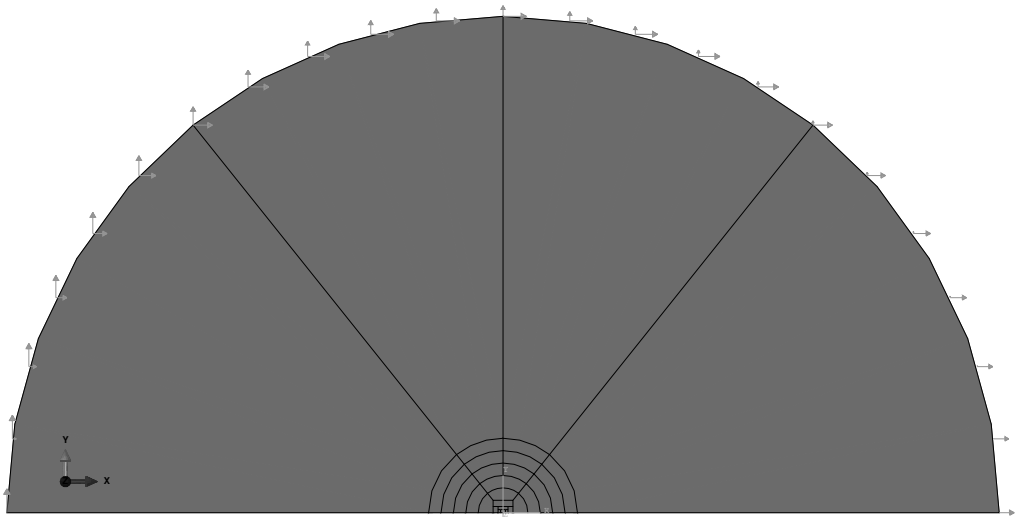


Figure 13: The model implemented in ABAQUS with boundary conditions applied.

Crack tip conditions are studied using a mode I crack, and the applied displacement field on the outer rim of the disc-shaped geometry is defined as

$$u_x = \frac{K_I}{2\mu}\sqrt{\frac{r}{2\pi}}\cos(\frac{\theta}{2})[\kappa - 1 + 2\sin^2(\frac{\theta}{2})] \tag{9.1}$$

$$u_y = \frac{K_I}{2\mu}\sqrt{\frac{r}{2\pi}}\sin(\frac{\theta}{2})[\kappa + 1 - 2\cos^2(\frac{\theta}{2})] \tag{9.2}$$

where

$$\kappa = \begin{cases} \frac{3-\nu}{1+\nu} & \text{(Plane stress)} \\ 3 - 4\nu & \text{(Plane strain)} \end{cases} \tag{9.3}$$

This displacement field assures a pure mode I loading at the crack tip. There will be one displacement along the outerrim of the disc in x-direction and one in y-direction depending on the local cylindrical coordinate system, (r, θ), at the center of the disc. Where r will represent the radius and θ the angle of placement. The coordinate system can be seen in figure 12. The stress intensity factor, K_I, will control the load amplitude.

Looking at the displacement it should be a pure mode I crack loading. This is not true when the crack starts to grow. However, the crack will not grow to a size where this has to be taken into consideration. This is why it is neglect that the displacement field will not be a pure mode I crack, since the crack length is small relative the radius of the disc.

The geometry, mesh and boundary conditions are easiest generated through ABAQUS, via the user interface. Then it is possible to write an input-file and to add wanted user subroutine commands. The mesh on the model generated can be seen in figure 14. The figures 15 and 16 give a better view of the fine meshed region in the crack tip area. A total of 269752 continuum elements were used. A total of 400000, user-defined, cohesive elements were employed.

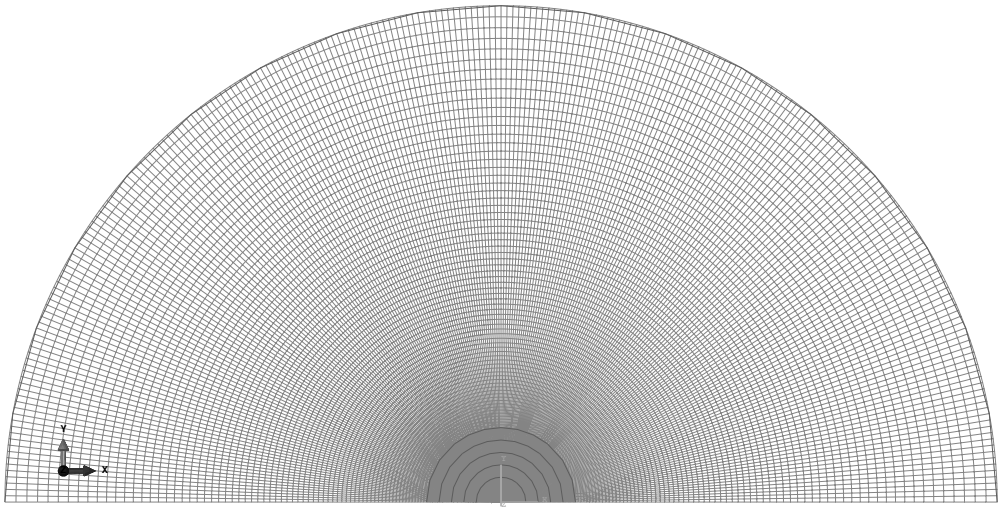


Figure 14: The mesh generated for the ABAQUS model.

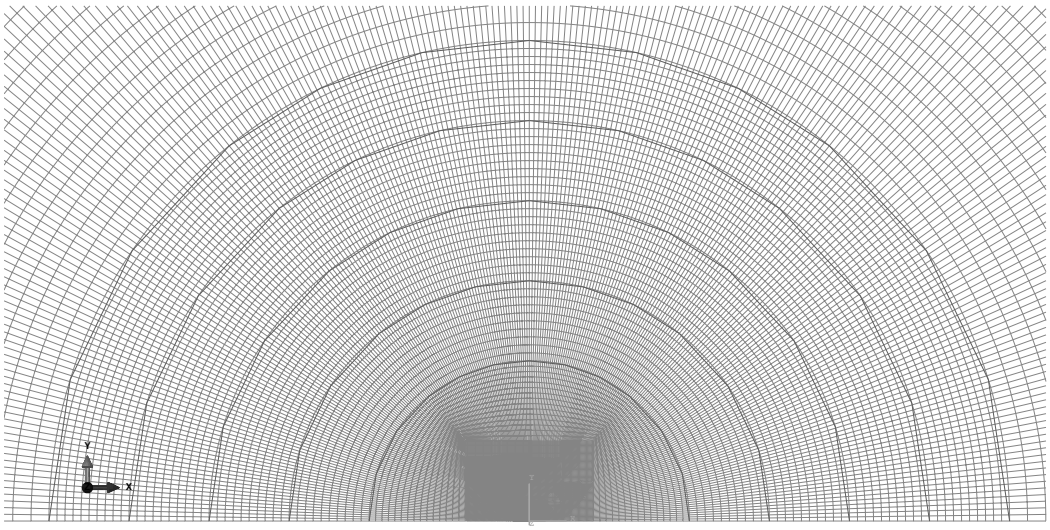


Figure 15: A closer view of the fine meshed region of the ABAQUS model.

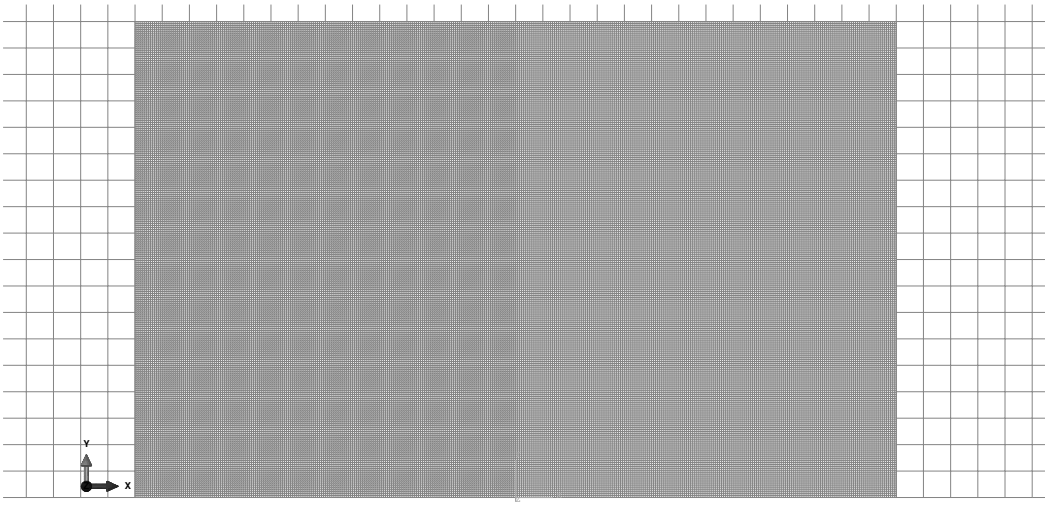


Figure 16: The very fine meshed region chosen according to characteristic length of the ABAQUS model.

For the continuum elements the ABAQUS element *CPE8RT* is used. It is an eight node quadratic element with reduced integration and with bilinear integration of the temperature field. The node numbering and integration points can be seen in figure 17.

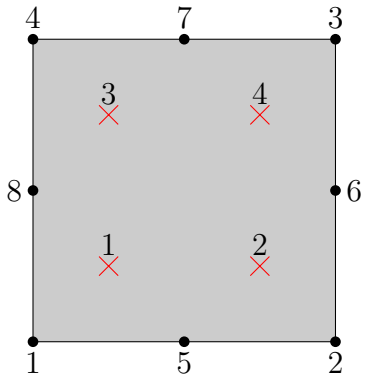


Figure 17: Node and integration point numbering of the 8-node continuum element with reduced integration

9.1 Numerical Considerations

The numerical integration of the linear interface element is chosen according to the Newton-Cotes scheme. While Gauss' integration scheme is perhaps the most commonly used scheme for the numerical integration of continuum elements, it is not the preferred method for interface elements. Schellekens and De Borst compared the Newton-Cotes scheme to the Gauss scheme and the results showed that the former yielded superior performance when used in combination with linear interface elements.

The use of Gauss quadrature led to oscillatory traction profiles along the interface, which are believed to occur due to spurious zero-energy modes [55].

The strain energy of an elastic body can be written as

$$U = \frac{1}{2}\boldsymbol{u}^{\boldsymbol{T}}\boldsymbol{K}\boldsymbol{u} \tag{9.4}$$

where $\boldsymbol{u}$ is the nodal displacement vector and $\boldsymbol{K}$ is the stiffness matrix and $(\bullet)^T$ denotes the transposed quantity. Since the stiffness matrix is positive semi-definite it follows that

$$U \geq 0 \tag{9.5}$$

When full integration is used the equality sign holds only for the rigid-body displacement mode, which is intuitive since no deformation occurs when the body rotates or translates rigidly. When the stiffness matrix is derived using reduced integration, i.e. not exact integration, new displacement modes that create zero strain energy can occur. Such displacement modes are called spurious zero-energy modes [45].

According to [55] the oscillatory traction profile occurs due to high traction gradients over the interface element and coupling between degrees of freedom in nodes of opposing sides of the element when using linear elements with Gauss' integration scheme [22]. A possible solution to this problem could be to use a higher number of integration points, which could eliminate the zero-energy modes. However, numerical results presented in [4] implies that the use of additional integration points could deacrease the overall robustness of the solution procedure, leading to a reduction in stable increment size.

9.2 Martensite Fraction in the Cohesive Element

When writing the UEL code, a method which links the traction-separation law to the martensite fraction is needed. To enable this the martensite fraction, z, in the cohesive elements' integration points must be obtained. A problem with the implementation is that in ABAQUS there is no convenient way of obtaining the state variables of the continuum elements in the user element subroutine. In order to attain this these variables in the user element, *common blocks* had to be exploited. Common blocks makes a variable available in any subroutine.

From the user material subroutine, the martensitic fraction is calculated for the integration points in the continuum element. From these integration points in the continuum element the z-value has to be extrapolated to the integration points in the cohesive element. In figure 18 the steps can be followed.

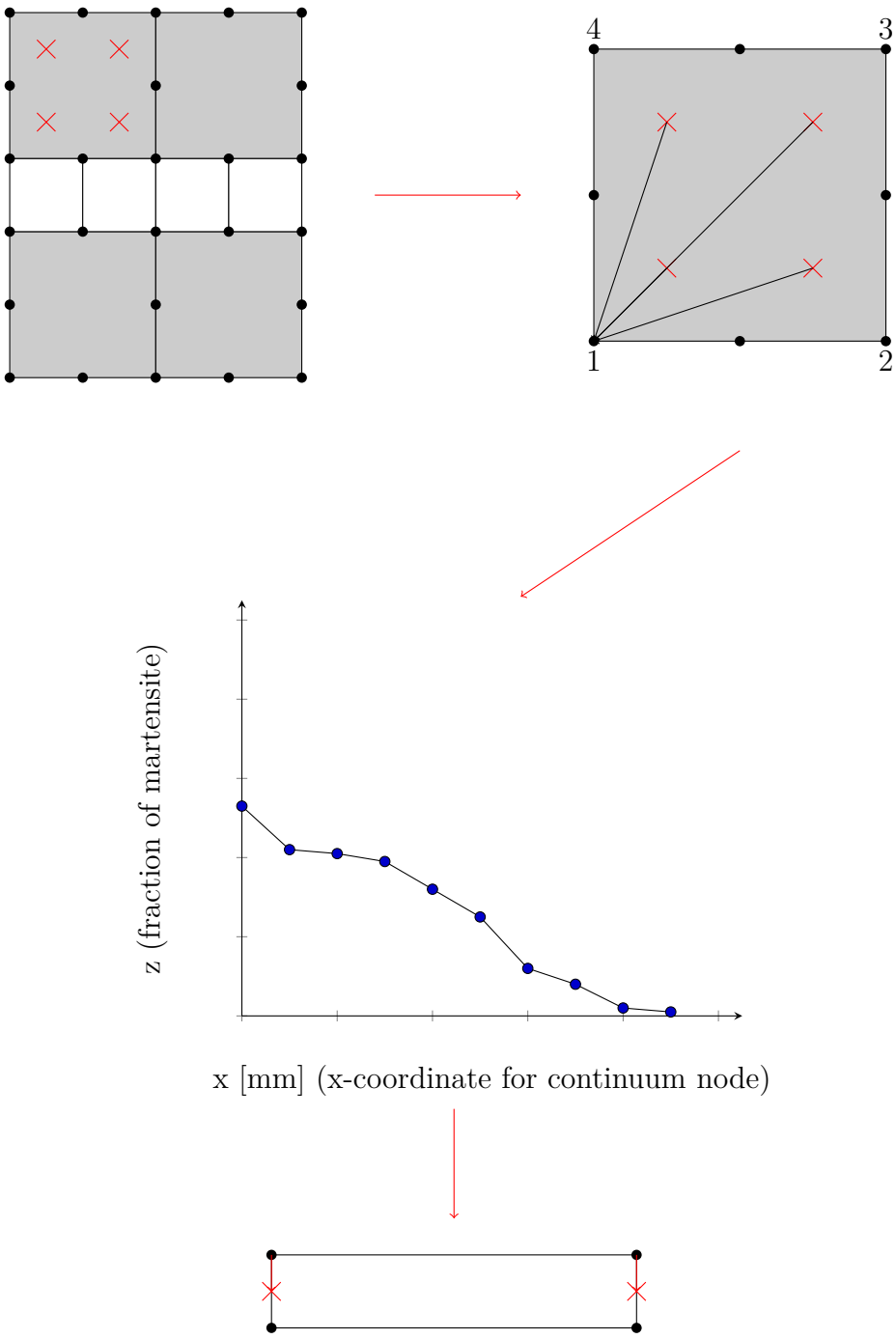


Figure 18: Flowchart showing the interpolation scheme to find the z-value in the cohesive elements integration points.

The aim of the calculation is to find the z-fraction in the integration points for the cohesive elements, so that it can be used to influence the traction-separation law. Looking at the geometry of our problem, the cohesive nodes are not connected to the continuum nodes. The elements are instead tied together with a *tie constraint* in ABAQUS. There is more than one cohesive element connected to one continuum element. In order to find the z-value for the cohesive element integration points, several steps are needed.

The basic idea is to find curve that describes the martensitic fraction as a function of the coordinate along the crack plane. The first step is to extrapolate the z-value from the integration points in the continuum element to the continuum nodes. Since reduced integration is used, there are only four integration points in a continuum element. If the shape functions for the 8-node element are used, the result will be an under-determined system of equations. Eight shape functions but only four integrations points are available to solve the following system of equations

$$z_{IP,i} = \sum_{j=1}^{8} z_{node} * N_j^{IPi}(\zeta, \theta) \tag{9.6}$$

Written in matrix form the equation system to solve looks like the following

$$\begin{bmatrix} z_{IP1} \\ z_{IP2} \\ z_{IP3} \\ z_{IP4} \end{bmatrix} = \begin{bmatrix} N_1^{IP1} N_2^{IP1} N_3^{IP1} - - - N_8^{IP1} \\ N_1^{IP2} N_2^{IP2} N_3^{IP2} - - - N_8^{IP2} \\ N_1^{IP3} N_2^{IP3} N_3^{IP3} - - - N_8^{IP3} \\ N_1^{IP4} N_2^{IP4} N_3^{IP4} - - - N_8^{IP4} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \\ z_5 \\ z_6 \\ z_7 \\ z_8 \end{bmatrix} \tag{9.7}$$

There are eight unknowns, and the system cannot be solved by inverting the shape function matrix since it is non-square and therefore non-invertible. A pseudoinverse could be exploited to solve this issue, but since the solution is ambiguous other methods are preferred.

Since the goal is to aquire z as a function of the position along the crack plane, it is deemed sufficient to only extrapolate the z-value to the corner nodes of the element along the crack. This can be done by using the linear shape functions of a 4-node element

$$\begin{aligned} N_1 &= \tfrac{1}{4}(1-\zeta)(1-\theta) \\ N_2 &= \tfrac{1}{4}(1+\zeta)(1-\theta) \\ N_3 &= \tfrac{1}{4}(1+\zeta)(1+\theta) \\ N_4 &= \tfrac{1}{4}(1-\zeta)(1+\theta) \end{aligned} \tag{9.8}$$

With this system of equations it is now possible to solve the equation (9.6) unambiguously. The system can now be written as

$$\begin{bmatrix} z_{IP1} \\ z_{IP2} \\ z_{IP3} \\ z_{IP4} \end{bmatrix} = \begin{bmatrix} N_1^{IP1} N_2^{IP1} N_3^{IP1} N_4^{IP1} \\ N_1^{IP2} N_2^{IP2} N_3^{IP2} N_4^{IP2} \\ N_1^{IP3} N_2^{IP3} N_3^{IP3} N_4^{IP3} \\ N_1^{IP4} N_2^{IP4} N_3^{IP4} N_4^{IP4} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \end{bmatrix} \tag{9.9}$$

It is simply solved by inverting the square shape function matrix.

Since linear interpolation is exploited, it is possible that the martensitic fraction becomes negative. This is an unphysical artifact created by the interpolation. The z-value should be between the 0 and 1. If the value of z turns out to be negative it is instead set to zero. This value is updated every iteration.

Solving the equation below will provide the z-value for four continuum nodes, the numbering and which nodes can be seen in the second step in figure 18.

$$\boldsymbol{z}_{node} = \boldsymbol{N}^{-1} \boldsymbol{z}_{IP} \tag{9.10}$$

The next step is to set up a function $z(x)$, where x is the x-coordinate for the continuum nodes along the ligament. To make the calculations easier linear interpolation is used between node values. It is also important to know that the node which is connected to two continuum elements might be declared two different values depending on the z-value in the continuum element. To solve this problem a mean value is calculated between the contribution from the adjacent elements.

Now that the z-value is known as a function of the x-coordinate for the system, it is possible to find the value for the cohesive nodes. The x-coordinate used is the coordinate from the undeformed configuration, which simplifies making the $z(x)$-function and then using this function for the cohesive elements. In this case the deformation or displacement in x-direction does not need to be considered.

Since there is more than one cohesive element along one continuum element, the continuum nodes inbetween which the cohesive element is located must be found. A linear interpolation is then done, taking into consideration the placement of the cohesive element. From the following equation, z is calculated.

$$\frac{z - z_0}{x - x_0} = \frac{z_1 - z_0}{x_1 - x_0} \tag{9.11}$$

The last step, after knowing the values in the cohesive nodes, is the interpolation to the integration points of the cohesive element. This is done using the shape functions. In this case Newton-Cotes integration is used together with the following shape functions

$$\begin{aligned} N_1 &= \tfrac{1}{2}(1 - \zeta) \\ N_2 &= \tfrac{1}{2}(1 + \zeta) \end{aligned} \tag{9.12}$$

The following system of equations is solved and the z-value for the integration points in the cohesive element are obtained.

$$\begin{bmatrix} z_{IP1} \\ z_{IP2} \end{bmatrix} = \begin{bmatrix} N_1^{IP1} N_2^{IP1} \\ N_1^{IP2} N_2^{IP2} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} \tag{9.13}$$

9.3 Implementation issues

When implementing the UEL in FORTRAN, a few *rules* need to be followed due to the restrictions of ABAQUS. Since *common blocks* had to be used to be able to find the correct values and to be able to perform the changes in the traction-separation law due to the phase transformation, it was also necessary to use hard coding to find the elements along the ligament.

The numbering of the integration points in the elements along the ligament can be seen in figure 17. However, using this numbering it also has to be taken in to consideration which side of the element lies along the ligament. It is also noted that the elements are rotated 90 degrees counterclockwise after the mesh structure operation,*sweep* is used. The very fine-meshed region has square elements, nut when leaving the fine-meshed region the element geometry changes to fit the disc. The disc is circular and to get the desired shape of the elements a sweeping structure is used which results in rotated elements, compared to the very fine-meshed region. This is also implemented using hard coding. When reaching the coordinate for where the sweeping structure starts, other integration point values has to be picked. The element are placed along the ligament as shown in figure 19, where the ligament is along the lower edge. It is also shown how the elements are rotated when using the sweep structure.

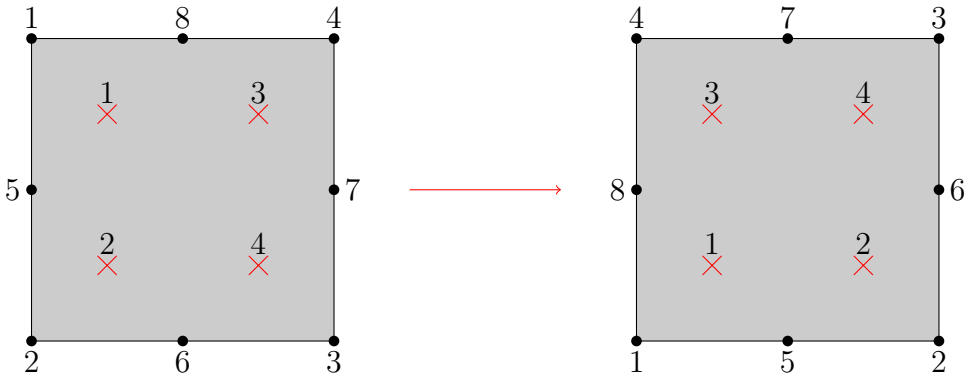


Figure 19: Illustration of the rotation of the element when going from structured square elements to a sweep structure in ABAQUS.

The constitutive relation of the cohesive element is characterized in terms of the separation of the nodes of the cohesive element. It is natural that the separation of the nodes can be negative for certain displacement modes. This leads to penetration of the continuum elements, which is undesirable. In order to prevent this behavior

a contact criterion is implemented into the cohesive element model. The contact is modeled with a penalty formulation meaning that for displacement modes that causes negative separation in the cohesive element, a very high tangential stiffness, or penalty stiffness, is exploited. This method allows for small amounts of penetration, which can often be neglected. When a sufficiently fine mesh is used, the contact between the continuum elements decreases drastically and this issue disappears. This topic is discussed further in section 10.4.

9.4 Cohesive Zone Modeling in Abaqus

The cohesive element is implemented as a user-defined subroutine for ABAQUS. A few pre-defined traction-separation laws are available in the user interface of ABAQUS, but in order to create a fully customizable cohesive element a user-defined subroutine is needed.

The cohesive elements are modeled as an own separate part of the model. The cohesive zone is a one element thick layer in between the regions where the crack is presumed to propagate. The cohesive element has a linear displacement formulation and the stress in the third direction does not affect the element behavior, thus there is no difference between plane stress and plane strain condition. The separations and stresses in the cohesive elements are calculated in each increment at the integration points according to the traction-separation law. When the critical separation energy is reached, the element has failed. The integration point which contributes to the stiffness obtains the status *failed*. Once an integration point has lost its stiffness, it is *failed* and it can never obtain another status [15].

10 Cohesive Zone Modeling in UEL

In the UEL subroutine a user code which defines the tangential stiffness matrix and the internal force vector of the element has to be implemented. The tangential stiffness matrix and the internal force vector is the input to ABAQUS. To understand the implementation it is interesting to know how the element is defined and how the stiffness matrix and the force vector of the cohesive element is formulated.

10.1 Cohesive Zone Elements

The problems analyzed in this paper are all two-dimensional, which means that two-dimensional elements are used. The element have four nodes, and the sides connecting to a continuum element behaves as a truss element. This means that the cohesive elements have a linear behavior. The cohesive element has four nodes with two degrees of freedom per node. Different from the continuum elements the cohesive elements does not have a temperature dependency. In figure 20, an element is defined with the nodes 1 to 4. Note that the numbering of the element starts in the lower left corner and works its way counter clockwise through the nodes around the elements. This is usually done to prevent negative areas to occur when using determinants to calculate element area.

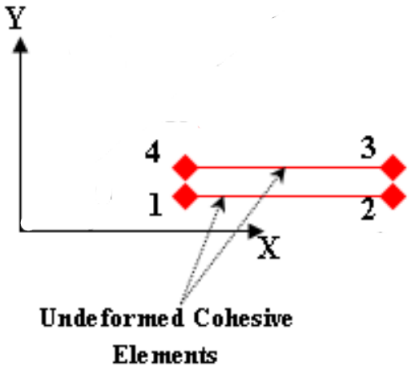


Figure 20: An undeformed cohesive element defined in a global coordinate system (X,Y), taken from [13]

10.2 Local coordinate system

Since the cohesive element will deform and move in space a local coordinate system that traces the motion of the element is needed. Calculating the separations in the local coordinate system ensures that rotations do not affect the results. The force vector and the stiffness matrix will be defined in the local coordinate system and then transformed back to the global coordinate system which is input for ABAQUS. The local coordinate system is chosen such that it is moving with the element using

a midsection face. The midsection face is a bisector between the upper and lower surface of the element [53], see figure 21.

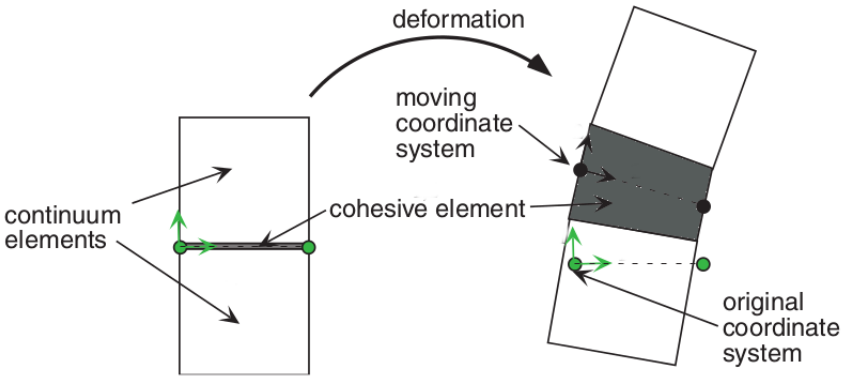


Figure 21: Illustration of the local coordinate system of the cohesive element moving with the deformation of the element, taken from [14].

A coordinate transformation matrix Λ that is defined as

$$\mathbf{\Lambda} = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix} \tag{10.1}$$

where θ is the angle the element is rotated compared to origin reference element. Using (10.1) it is possible to transform the global coordinates, X, into local coordinates, x, as

$$\mathbf{x} = \mathbf{\Lambda X} \tag{10.2}$$

Using the definition from (10.1) a rotational matrix, R, is defined as

$$\mathbf{R} = \begin{bmatrix} \mathbf{\Lambda} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{\Lambda} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{\Lambda} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{\Lambda} \end{bmatrix} \tag{10.3}$$

The rotational matrix in equation (10.3) transforms the global nodal displacements into local nodal displacements as follows [47]

$$\tilde{\mathbf{u}} = \mathbf{R}\bar{\mathbf{u}} \tag{10.4}$$

Where the local nodal displacement vector is defined as

$$\tilde{\mathbf{u}} = \begin{bmatrix} \tilde{u_1} \\ \tilde{u_2} \\ \tilde{u_3} \\ \tilde{u_4} \\ \tilde{u_5} \\ \tilde{u_6} \\ \tilde{u_7} \\ \tilde{u_8} \end{bmatrix} \tag{10.5}$$

since each node has two degrees of freedom.

10.3 Finite element formulation

The FE-formulation of the cohesive element can be derived from the principle of virtual work. This approach is explained by both Park and Scheider, c.f. [47, 54]. Making use of the fact that the virtual work performed by the external traction, $\mathbf{T}_{\text{ext}}$, on the boundary, Γ, is equal to the sum of the virtual strain energy in the domain, Ω, and the cohesive fracture energy evaluated on the fracture surface, Γ_c, the governing equation can be written in a weak formulation as

$$\int_{\Omega} \delta\boldsymbol{\epsilon} : \boldsymbol{\sigma} \; dV + \int_{\Gamma_c} \delta\Delta \cdot \mathbf{T}_c \; dS = \int \delta\mathbf{u} \cdot \mathbf{T}_{\text{ext}} \; dS \tag{10.6}$$

where $\delta\boldsymbol{\epsilon}$, $\delta\mathbf{u}$ and $\delta\Delta$ are the virtual strain, virtual displacement and virtual separation respectively. The stress tensor in the deformed configuration is denoted $\boldsymbol{\sigma}$ and $\mathbf{T}_c$ is the cohesive traction along the fracture surface. Looking at a two-dimensional problem, two directions of stress; normal and shear, exist. For the mode I crack problem at hand, traction only occur in the normal direction.

$$\mathbf{T}_c = \begin{bmatrix} \mathbf{T}_s \\ \mathbf{T}_n \end{bmatrix} \tag{10.7}$$

In equation (10.6) the first term on the left hand side is associated with the internal force of volumetric elements and the second term is related to the internal cohesive force of cohesive surface elements. The right hand term corresponds to the external force.

The domain Ω is discretized into finite elements, and the displacement field $\mathbf{u}$ is approximated by the product of the nodal displacement $\bar{\mathbf{u}}$ and the matrix of shape functions.

$$\mathbf{u}(\mathbf{X}) = \mathbf{N}\bar{\mathbf{u}} \tag{10.8}$$

In equation (10.8), $\mathbf{N}$ is the matrix with shape functions defined as

$$\mathbf{N} = \begin{bmatrix} N_1 & 0 & N_2 & 0 \\ 0 & N_1 & 0 & N_2 \end{bmatrix} \tag{10.9}$$

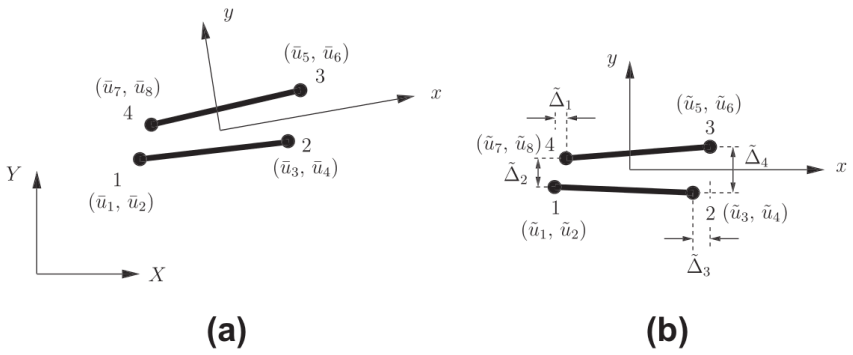


Figure 22: A two-dimensional linear cohesive element with the nodal displacement in global coordinates in a) and local coordinates in b), taken from [15].

The local separation, $\tilde{\Delta}$, is approximated using the nodal displacement $\tilde{\mathbf{u}}$, and is defined in terms of normal and tangential nodal separation in each end of the cohesive element, see figure 22. An illustration of how the global quantities are transformed into local quantities with the rotational matrix from equation (10.3) is also shown in the previously mentioned figure. The local separations are defined as

$$\tilde{\boldsymbol{\Delta}} = \begin{bmatrix} \tilde{\Delta}_1 \\ \tilde{\Delta}_2 \\ \tilde{\Delta}_3 \\ \tilde{\Delta}_4 \end{bmatrix} = \begin{bmatrix} \tilde{u}_7 - \tilde{u}_1 \\ \tilde{u}_8 - \tilde{u}_2 \\ \tilde{u}_5 - \tilde{u}_3 \\ \tilde{u}_6 - \tilde{u}_4 \end{bmatrix} \tag{10.10}$$

Since the local separations are based on the global nodal displacement, the global coordinates are first transformed to local coordinates with equation (10.2). Similarly the global nodal displacement is transformed to the local nodal displacement with equation (10.4). From the local nodal displacement the local nodal separation, $\tilde{\Delta}$, along the surface normal and tangential directions can be obtained. This is expressed as

$$\tilde{\Delta} = \mathbf{L}\tilde{\mathbf{u}} \tag{10.11}$$

where $\mathbf{L}$ is a local displacement-separation relation matrix. With the information about the local nodal displacement and the local separations, equation (10.10) and (10.5) respectively, $\mathbf{L}$ is defined as

$$\mathbf{L} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & -1 & 0 & 0 \end{bmatrix} \tag{10.12}$$

From here the separation along a cohesive surface element is now given by interpolating the nodal separation using the shape functions for the element.

$$\boldsymbol{\Delta}(\mathbf{x}) = \mathbf{N}\tilde{\boldsymbol{\Delta}} \tag{10.13}$$

The interpolation is done using two integration points along the natural coordinate ξ. Since a two-dimensional, linear element is used only two integration points along the mid-section of the element are needed. The shape functions expressed in the natural coordinate ξ are defined as

$$N_1 = \frac{1-\xi}{2}, \quad N_2 = \frac{1+\xi}{2} \tag{10.14}$$

Recalling equations (10.4) and (10.11) from earlier, it is possible to write (10.13) as

$$\boldsymbol{\Delta}(\mathbf{x}) = \mathbf{N}\tilde{\boldsymbol{\Delta}} = \mathbf{NL}\tilde{\mathbf{u}} = \mathbf{NLR}\bar{\mathbf{u}} \tag{10.15}$$

Where **NLR** is written as a global displacement-separation relation matrix, $\mathbf{B}_c$.

$$\boldsymbol{\Delta}(\mathbf{x}) = \mathbf{B}_c\bar{\mathbf{u}} \tag{10.16}$$

Expressed in matrices it takes the following form

$$\mathbf{B}_c\bar{\mathbf{u}} = \begin{bmatrix} N_1 & 0 & N_2 & 0 & -N_2 & 0 & -N_1 & 0 \\ 0 & N_1 & 0 & N_2 & 0 & -N_2 & 0 & -N_1 \end{bmatrix} \mathbf{R}\bar{\mathbf{u}} \tag{10.17}$$

Based on the approximated displacement field, the internal force vector for the cohesive element is given as

$$\mathbf{f}_{\text{coh}} = \int_{\Gamma_c} \mathbf{B}_c^T \mathbf{T}_c \, dS \tag{10.18}$$

The stresses are evaluated at the surfaces of the cohesive element, i.e. the surfaces adjacent the continuum surfaces. These can be seen as the top and bottom surfaces at which the cohesive stresses are transferred to the neighbouring continuum elements.

The cohesive stresses $\mathbf{T}_c$ in equation (10.18) are non-linear functions of the displacement. This makes it impossible to extract the nodal displacement from the integral, and a linear system of equations can not be set up. This means that the tangential

stiffness matrix needs to be derived. The tangential stiffness matrix is the change of the internal forces corresponding to infinitesimal changes in displacements [31].

To find the tangential stiffness matrix, equation (10.18) is differentiated. When differentiating the internal forces the total differential of the traction is used as

$$d\mathbf{T}_c = \frac{\partial \mathbf{T}_c}{\partial \mathbf{\Delta}} \cdot d\mathbf{\Delta} \tag{10.19}$$

and

$$d\mathbf{\Delta} = \mathbf{B}_c d\bar{\mathbf{u}} \tag{10.20}$$

For $d\mathbf{\Delta}$ the difference $\Delta\mathbf{\Delta}$ is used. Looking at the total displacement at the time $t+\Delta t$ we get $\mathbf{\Delta}^{t+\Delta t} = \mathbf{\Delta}^t + \Delta\mathbf{\Delta}$. The next step is to write the differential formulation of the equation (10.18) so that the tangential stiffness matrix $\mathbf{K}_{\text{coh}}$ can be found. The tangential stiffness can finally be expressed as

$$\mathbf{K}_{\text{coh}} = \frac{\partial \mathbf{f}_{\text{coh}}}{\partial \bar{\mathbf{u}}} = \int_{\Gamma_c} \mathbf{B}_c^T \frac{\partial \mathbf{T}_c}{\partial \mathbf{\Delta}} \mathbf{B}_c \, dS \tag{10.21}$$

The term $\frac{\partial \mathbf{T}_c}{\partial \mathbf{\Delta}}$ on the right hand side of equation (10.21), is called the tangent modulus matrix and is denoted as $\mathbf{C}$ [41], providing

$$\mathbf{C} = \frac{\partial \mathbf{T}_c}{\partial \mathbf{\Delta}} = \begin{bmatrix} \frac{\partial \mathbf{T}_s}{\partial \Delta_s} & \frac{\partial \mathbf{T}_s}{\partial \Delta_n} \\ \frac{\partial \mathbf{T}_n}{\partial \Delta_s} & \frac{\partial \mathbf{T}_n}{\partial \Delta_n} \end{bmatrix} \tag{10.22}$$

With this definition the tangential stiffness matrix for the cohesive element can be expressed as

$$\mathbf{K}_{\text{coh}} = \int_{\Gamma_c} \mathbf{B}_c^T \mathbf{C} \mathbf{B}_c \, dS \tag{10.23}$$

To be able to solve this, a law that defines the traction-separation relation must be chosen.

As earlier mentioned the cohesive element is implemented as a user-defined element, UEL, in FORTRAN code suited for ABAQUS as a user-subroutine.The UEL-subroutine is attached in appendix B.

10.4 Length scales

Very large stress and strain gradients occur inherently in regions around the crack tip. The gradients put limitations on the maximum allowable element size. If the elements in the cohesive zone are insufficiently small the cohesive zone is unable to accurately

resolve the stress field at the crack tip. There are different length scales that needs to be considered in the finite element model. The first one is the macroscopic length scale $\boldsymbol{L}$ which is related to the geometry. It can often be identified with, for example, the ligament length. The second one is the mesh size, l_m, which is a non-physical length scale. It is crucial that the mesh size is much smaller than the macroscopic length for the mesh to be able to resolve the stress distribution near the tip of the crack. The third length scale is the cohesive zone length, l_z, which is determined by the material parameters [63]. The cohesive zone length is a measure of the length over which the cohesive constitutive relation plays a role [21].

According to [63] the cohesive zone length for a mode I crack can be calculated with the following equation

$$l_z = \frac{\pi}{8} E' \frac{\Gamma_c}{T_{max}^2} \tag{10.24}$$

Where E is

$$E = \begin{cases} E & \text{(Plane stress)} \\ \frac{E}{1-\nu^2} & \text{(Plane strain)} \end{cases} \tag{10.25}$$

Young's modulus will be used for a plane stress condition. Since there will be some rounding of the numbers already, the choice of plane stress instead of plain strain will not make that much of a difference.

The choice of parameters is based on a parameter study and the choice of parameters done for the TS-law, cf. section 11.1-11.3. So for the calculations the following choices are made, $T_{max} = 1000$ MPa and $\Gamma_c = 65 \frac{N}{mm}$, important to know is that the unit of the equation is $\frac{J}{m^2}$. Too small elements is undesirable, since it will result in extremely slow simulations.

In order to fully resolve the stress field, 2-5 elements over the cohesive zone length is recommended [21]. This makes it necessary to also divide the length with number of elements wanted over a cohesive zone length. With $E' = 200277 MPa$, $\Gamma_c = 65 * 10^{-3} \frac{N}{mm}$ and $T_{max} = 1000 MPa$, the cohesive length according to eq. (10.24) will become

$$l_z = 0.00511 [mm] \tag{10.26}$$

Even if the choice of Γ_c later is a smaller number which results in a smaller cohesive zone length, it is not taken into consideration, since the amount of elements will be too high.

11 Traction-Separation Laws

A great number of different traction-separation laws have been proposed in the literature. Since there are many different assumptions of the traction-separation law a few will be described with reference to the authors. They all build on the same fundamental ideas, but include a few differences.

There are three different TS-laws that have been implemented in the present work in the UEL FORTRAN code used in ABAQUS. A bilinear, a trapezoidal and an exponential model. All three models needs to be able to simulate the different stages of crack propagation. The TS-law must involve the initial loading, the damage initiation and the damage evolution. The three models are based on a failure displacement δ_f, this is where the cohesive element is said to fail. Once a cohesive element has failed it will never again retain traction. There is also a first critical displacement δ_{cr1} and a second critical displacement δ_{cr2}. These displacements mark the damage initiation. The critical displacement can be expressed using a fraction with respect to δ_f as

$$\lambda_{cr1} = \frac{\delta_{cr1}}{\delta_f} \tag{11.1}$$

$$\lambda_{cr2} = \frac{\delta_{cr2}}{\delta_f} \tag{11.2}$$

This is illustrated as an example for one TS-laws in figure 23.

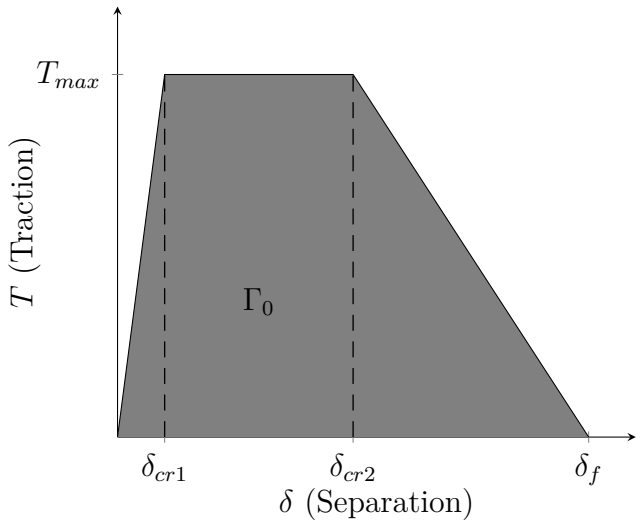


Figure 23: An example of a TS-law.

Then there is the cohesive energy. The cohesive energy is defined as the area underneath the traction-separation curve.

$$\Gamma_0 = \int_0^{\delta_0} T(\delta) d\delta \tag{11.3}$$

The energy is connected to the material as the fracture toughness. The last parameter defining the TS-law, is the maximum traction, T_{max}. This is the maximum traction that the cohesive element can obtain. With these parameters the TS-laws can be uniquely defined. The input to the FORTRAN UEL code are the critical displacements, the maximum traction and the fracture toughness.

In the simulations carried out in ABAQUS only a mode I crack is examined. This results in just looking at the traction normal to the crack surface, i.e the shear mode is neglected. In the 2D problem the effective displacement and an effective traction turns out as follows [41]

$$\lambda_e = \sqrt{(\frac{\delta_n}{\delta_f})^2} \tag{11.4}$$

$$t_e = \sqrt{(t_n)^2} \tag{11.5}$$

Here δ_n is the normal displacement, and t_n is the normal traction.

Since phase transformation is occurring in front of the crack tip, the traction-separation laws will change with the martensite fraction, z. The TS-laws are changed by applying theory of how the martensite transformation would affect the properties of the material that affects the crack propagation.

Changing the TS-law might result in taking effects from the phase transformation into account more than once since also the continuum elements will depend on the material behavior. For example, the volume change in the material might give some change even without changing the TS-law. These factors have to be taken into consideration later on when looking at the results.

11.1 Study of Cohesive Parameters

The choice of values for the parameters in the traction-separation law is very important. Since the cohesive zone model is a phenomenological one, it is a difficult task choosing these parameters without actual data to support the choice. In this study different articles are studied, which propose cohesive zone models for different types of steel.

It might not be the best choice to look into different types of steel, since different steels have very different properties. Some are more ductile and some are more brittle. It is proposed that a bilinear TS-law is more suited for brittle fracture and a trapezoidal TS-law is more suited for ductile fracture [15]. However the parameters will vary depending on which steel is examined. Just looking at a few different articles, the values for the TS-law are very different from each other. This is why it would have been a good idea to support the choice of parameters with an actual analysis from an example in reality.

Analysing the parameter related to the initial stiffness, δ_{cr1}, the choice should be as small as numerically possible in order to ensure realistic pre-crack behavior [11]. In reality a crack will occur or propagate as soon as a separation of the two surfaces occur. It is therefore desirable that the separation is very small before the cohesive element reaches the softening state. The parameter δ_{cr2} however, is chosen according to the fracture process and is material dependent. An example of choice of parameters for a trapezoidal law, which can also be applied on the bilinear law for δ_{cr1}, is $\delta_{cr1} = 0.05$ and $\delta_{cr2} = 0.5$. This choice proved successful for steels according to [11].

Looking at the choice of maximum traction, T_{max}, and fracture toughness, Γ_c, the suggested parameters range from approximately between $500 - 2000$ MPa and $6 - 110 \quad \frac{N}{mm}$ respectively. Varying authors have been looking into different steels. A common choice is pipeline steels [19, 49, 70]. Also parameters for aluminum [42], a C300 steel [46], and a 304 stainless steel [32] is found. The different choices can be seen in table 11.1.

	$T_{max} \quad [MPa]$	$\Gamma_c \quad [\frac{N}{mm}]$
Pipeline Steels	1329 and $T_{max} = 3 * \sigma_0$, where $\sigma_0 = \frac{E}{450}$	100 and $6 - 8$
Aluminium	800	13.8
C300 Steel	2000	15
304 Stainless Steel	1100	110

Table 11.1: Table showing values from the parameter study for cohesive parameters.

An overview of the different parameters used in the literature reveals a huge disparity, even among similar materials. This makes the exact choice of parameters very difficult since it depends on a lot of different factors.

Looking at simulations carried out while varying the parameters, the simulations show very high values of martensite fraction in the elements near the crack-tip if T_{max} is too high. Also, the process zone, meaning the length of the zone where the cohesive elements are active, becomes very long if Γ_c is too high. Combining the experience from previous simulations and different parameters from literature the choice is based on an average of the parameters found in the literature study, $T_{max} = 1000$ MPa and $\Gamma_c = 35 \quad \frac{N}{mm}$. The choices made for the two different TS-laws, bilinear and trapezoidal, is shown below.

11.2 Bilinear model

One of the simplest possible models is the bilinear traction-separation law, see figure [24]. This model has the following three stages [41, 5, 61]

$$t_n = \begin{cases} T_{max} \frac{1}{\lambda_{cr1}} \left(\frac{\delta_n}{\delta_f} \right) & \lambda_e < \lambda_{cr1} \\ T_{max} \frac{1-\lambda_e}{1-\lambda_{cr1}} \frac{1}{\lambda_e} \left(\frac{\delta_n}{\delta_f} \right) & \lambda_e > \lambda_{cr1} \\ 0 & \lambda_e > 1 \end{cases} \tag{11.6}$$

And the energy, that is the area underneath the curve, can be calculated as

$$G_c = \frac{T_{max} \delta_f}{2} \tag{11.7}$$

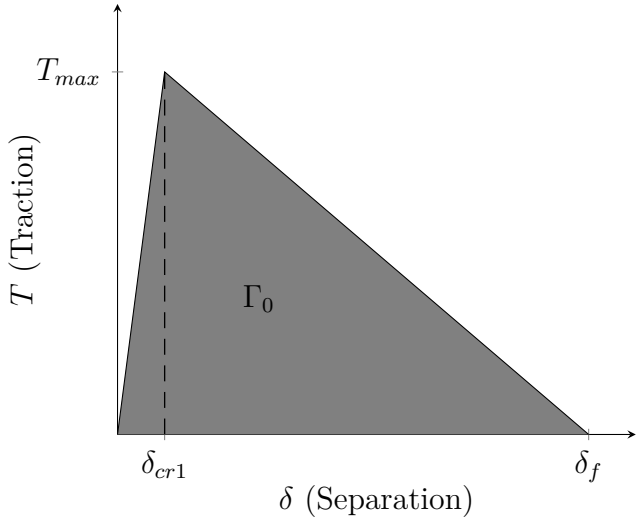


Figure 24: Bilinear traction-separation law.

The bilinear model is suited for brittle fracture [15]. When analyzing the crack length for the bilinear model, the cohesive element is defined as *open* when the separation has reached δ_{cr1}. The crack tip is placed an element has last reached the softening part of the TS-law.

The parameters used for the trapezoidal TS-law are: $\Gamma_c = 17$, $T_{max} = 1000MPa$ and $\lambda_{cr1} = 0.005$.

11.3 Trapezoidal model

The trapezoidal model is suited for ductile fracture [15]. The implementation is based on theory from Tvergaard and Hutchinson [64]. It is very similar to the bilinear model, but for this model there is also a part with constant traction, see figure 25. The four stages are described by

$$t_n = \begin{cases} T_{max}\frac{1}{\lambda_{cr1}}\left(\frac{\delta_n}{\delta_f}\right) & \lambda_e < \lambda_{cr1} \\ T_{max} & \lambda_{cr1} < \lambda_e < \lambda_{cr2} \\ T_{max}\frac{1-\lambda_e}{1-\lambda_{cr2}}\frac{1}{\lambda_e}\left(\frac{\delta_n}{\delta_f}\right) & \lambda_e > \lambda_{cr2} \\ 0 & \lambda_e > 1 \end{cases} \tag{11.8}$$

The energy, which is represented by the area under the curve can be calculated as

$$G_c = \frac{1}{2}T_{max}(\delta_f - \delta_{cr1} + \delta_{cr2}) \tag{11.9}$$

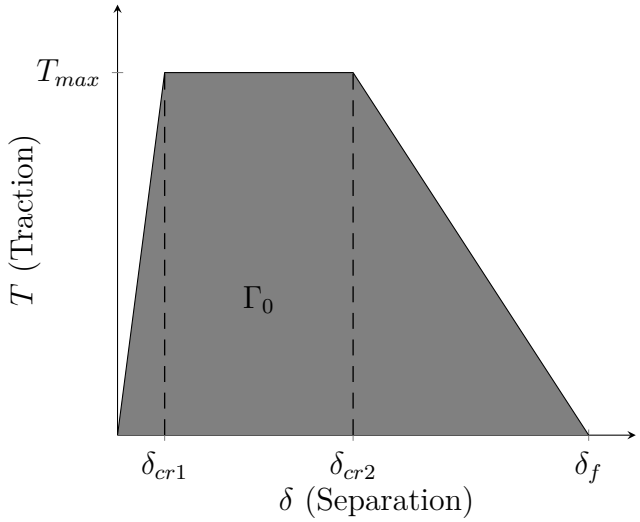


Figure 25: Trapezoidal traction-separation law.

When analyzing the crack length for the trapezoidal model, the cohesive element is defined as *open* when the separation has reached δ_{cr2}. The crack tip is placed where an element has last reached the softening part of the TS-law.

The parameters used for the trapezoidal TS-law are: $\Gamma_c = 25$, $T_{max} = 1000MPa$, $\lambda_{cr1} = 0.05$ and $\lambda_{cr2} = 0.5$.

11.4 Exponential model

The exponential model is based on a potential. By deriving a cohesive potential energy, Φ, the cohesive traction is found. This is based on theory from both Needleman and Xu [67, 38]. The mechanical response of the interface is described through a constitutive relation that gives a dependence of the traction. The traction response is seen as an exponential curve which is used as a base to implement the exponential model [5], see figure 26. The model is based on the following equation

$$T = \frac{\partial \phi}{\partial \delta} = \frac{\delta_n}{\delta_{cr1}} T_{max} * \exp(1 - \frac{\delta_n}{\delta_{cr1}}) \tag{11.10}$$

The energy, the area under the curve, is given as

$$G_c = T_{max} \delta_f * \exp(1) \tag{11.11}$$

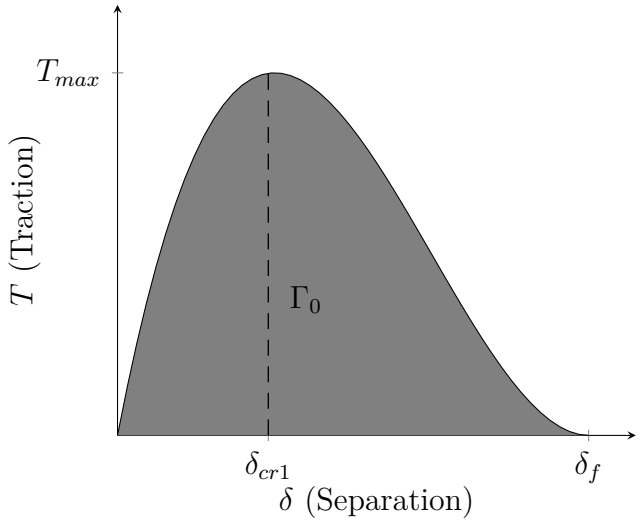


Figure 26: Exponential traction-separation law.

The exponential model described is just a suggestion of an exponential TS-law. Exponential curves can have very different shapes and one has to find one suited for specific geometry, material and problem to be implemented.

The exponential model is not used for further simulation in the present work. It was implemented but not used for the simulations since the literature study reveals more use of the bilinear and the trapezoidal model to simulate fracture in steel.

11.5 A Special Alteration of the TS-law

The continuum model might not be sufficient to capture the effect of the phase transformation on the propagating crack. That is why a method for modeling the alteration of the fracture process, which is not captured by the continuum model, is proposed. The model can account for the decreased ductility by exploiting a special traction-separation law which is dependant on the martensitic fraction, z.

The bilinear and the trapezoidal TS-law is altered, and the alteration is based on how the mechanical properties of the austenitic steel are affected by the martensitic transformation. A similar approach was proposed in an article by Olden *et al* [43]. In this paper a model which combines hydrogen embrittlement with cohesive zones is

proposed. When hydrogen diffuses into the steel structure it becomes brittle and the surface energy decreases. This effect is accounted for by introducing a bilinear TS-law in which the peak traction is lowered with increasing hydrogen content, effectively decreasing the fracture energy of the material. The results presented in this paper show good conformation with the experiments carried out.

A similar reasoning can be used to argument for the need to develop a similar method for the more brittle martensitic phase. The model proposed in this paper has no experimental results to compare to, but it is interesting to see the effects of such changes. In order to fully validate this method experiments would need to be conducted in the future.

There are a multitude of different ways that the phase transformation could plausibly alter the TS-law. The proposed model is such that the peak traction is taken as a linear combination of the traction suitable for the austenitic phase and the traction suitable for the martensitic phase, i.e.

$$T_{max} = T_{max,a} * (1 - z) + z * T_{max,m} \tag{11.12}$$

In order to retain the basic appearance of the bilinear model it is modeled such that the initial stiffness is kept constant, and the point of peak traction moves along the same tangent throughout the deformation, see figure 27. Moreover, the maximum traction is kept constant once the critical displacement corresponding to the current maximum traction is reached. This means that additional phase transformation does not affect the TS-law once damage is initiated. The fracture energy is kept constant which means that the failure displacement reduces with increasing martensitic fracture.

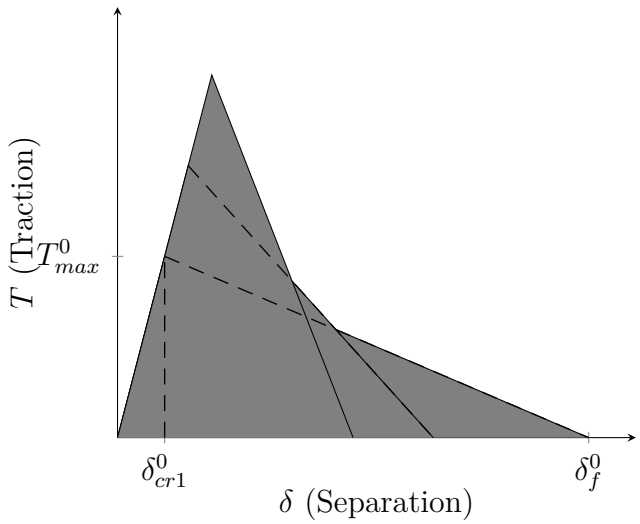


Figure 27: The altered bilinear traction-separation law.

The proposed model is very simple, and arguments can be made to claim that other implementations could be more accurate. The main reason why this model is chosen is that the tangential stiffness is kept constant, which greatly increases the stability

of the solution. If the phase transformation is allowed to change the characteristics of the TS-law after damage is initiated, steep stiffness increases could occur during the softening state.

The trapezoidal law is changed based on the fact that the austenite is more ductile than the martensite. From previous sections it is mentioned that the bilinear TS-law and the trapezoidal law is more suited for brittle respectively ductile fracture simulations. Using this information and scaling the δ_{cr2} with z-fraction the TS-law will make the behavior of the crack propagation more brittle with a higher z-fraction. The scaling is done with a linear combination, see equation below, and z=1 gives $\delta_{cr2} = \delta_{cr1}$.

$$\delta_{cr2} = \delta_{cr2} * (1 - z) + z * \delta_{cr1} \tag{11.13}$$

The change is illustrated in figure 28.

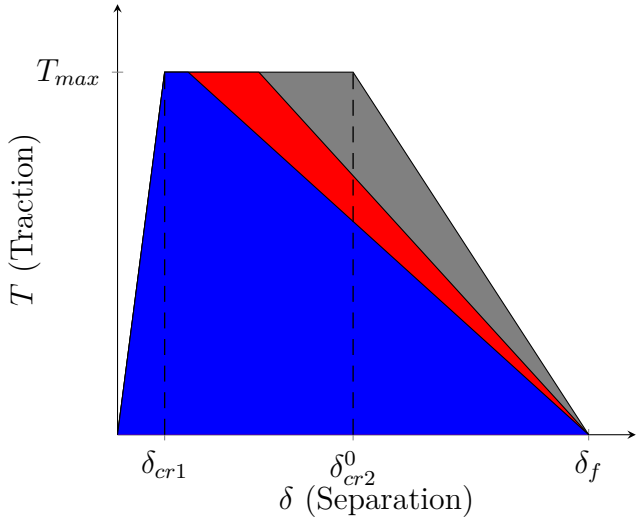


Figure 28: The altered trapezoidal traction-separation law.

In this alteration of the TS-law it might be more correct to also change the maximum traction used. This because martensite is harder than austenite. However, in this case it is just interesting to see the changes given by altering the law to a more brittle behavior and see if it gives any effect.

12 Results

12.1 Crack Propagation Simulations

The simulations are carried out at four different temperatures, 213K, 233K, 293K and 313K. Each temperature has special input parameters in the input-file for ABAQUS. The simulations have been very time consuming and with the computers provided by the department a simulation could take up to about 48 hours.

The simulations are run with phase transformation switched on and off. Also simulations where the TS-law is altered are run at the lower temperatures.

Looking at the results it is interesting to see how the crack length increases with increasing stress intensity factor. The crack length and the crack tip opening displacement (CTOD) are plotted against the stress intensity factor, K_I. It is also interesting to look at the z-fraction at the crack tip at lowest temperature. It is also interesting to see how the proposed TS-laws change with the phase transformation, and its implications on the results at low temperatures.

When plotting the results, the definition of where the crack tip is located should be well-defined and straight forward. It would as a first guess be located when the traction in the cohesive element has reached 0. This is however not the case. According to Xu and Needleman [67] once crack growth initiates there is a continuous dependence of the cohesive surface traction. This means that the crack tip is located where the last element entered the softening part of the TS-law. For the bilinear law it will be when $\delta_n > \delta_{cr1}$ and for the trapezoidal law it will be when $\delta_n > \delta_{cr2}$.

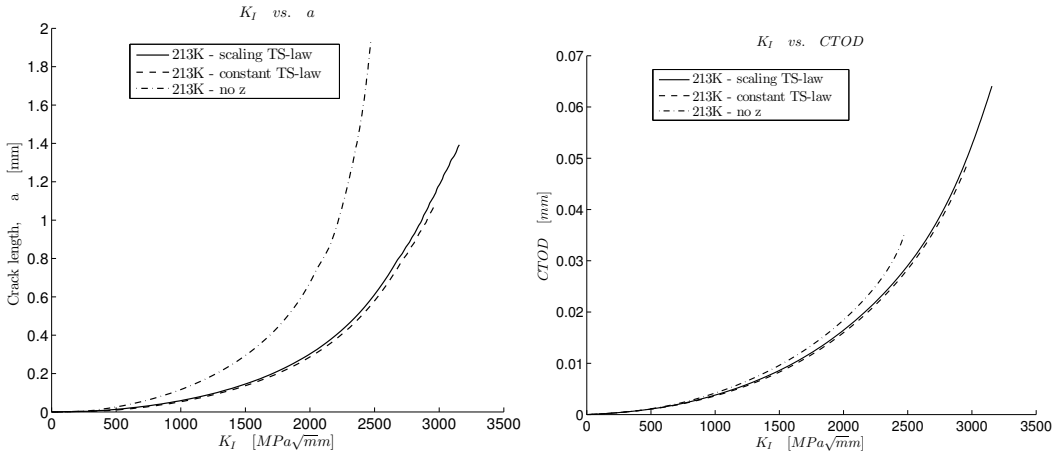
The CTOD is defined as the crack tip opening displacement at the original crack tip. This is chosen because it is the most convenient definition when extracting the data from ABAQUS.

Even if there is a possibility to run simulations with four different temperatures it was chosen not to look at 313K since no phase transformation was present at this temperature. The results will already coincide at 293K where the phase transformation is negligible.

12.2 Bilinear TS-law

The results for the simulations run with and without phase transformation with the bilinear TS-law are presented below. Three different temperatures are used for the simulations.

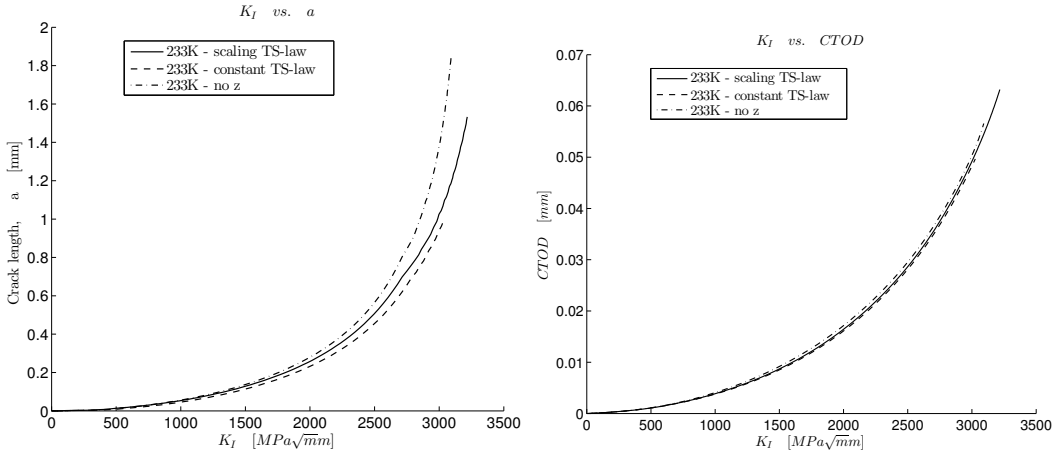
12.2.1 213 K



(a) **(b)**

Figure 29: Figure (a) shows the crack length, a, as a function of the stress intensity factor and figure (b) shows the $CTOD$ as a function of the stress intensity factor, both at 213 K.

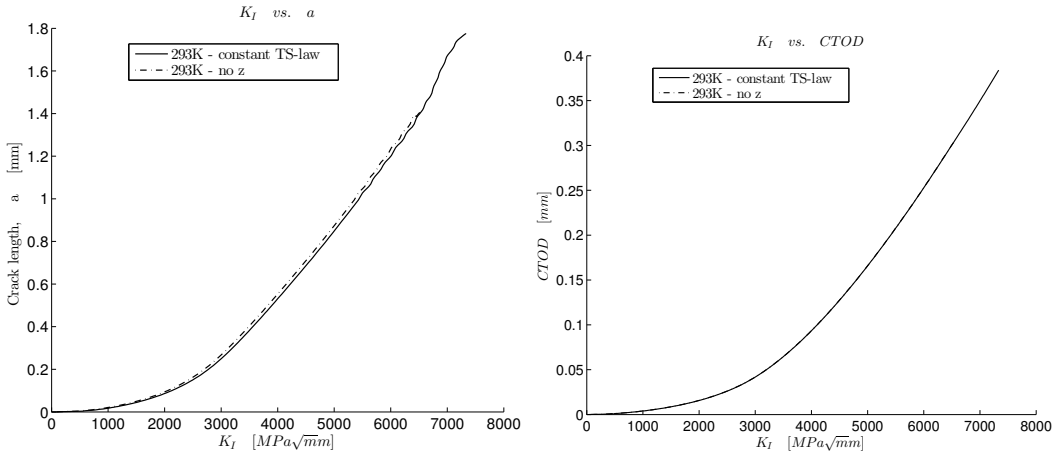
12.2.2 233 K



(a) **(b)**

Figure 30: Figure (a) shows the crack length, a, as a function of the stress intensity factor and figure (b) shows the $CTOD$ as a function of the stress intensity factor, both at 233 K.

12.2.3 293 K



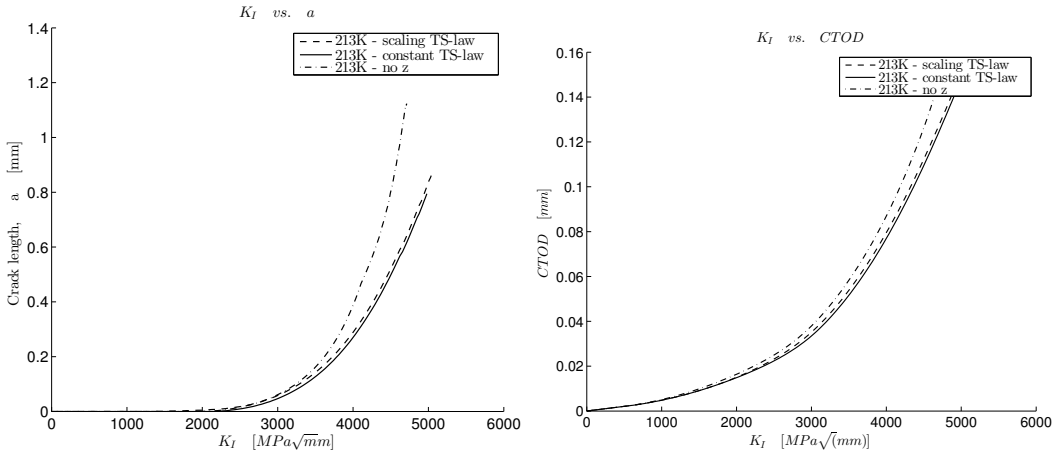
(a) **(b)**

Figure 31: Figure (a) shows the crack length, a, as a function of the stress intensity factor and figure (b) shows the $CTOD$ as a function of the stress intensity factor, both at 293 K.

12.3 Trapezoidal TS-law

The results for the simulations run with and without phase transformation with the trapezoidal TS-law are presented below. Three different temperatures are used for the simulations.

12.3.1 213 K



(a) **(b)**

Figure 32: Figure (a) shows the crack length, a, as a function of the stress intensity factor and figure (b) shows the $CTOD$ as a function of the stress intensity factor, both at 213 K.

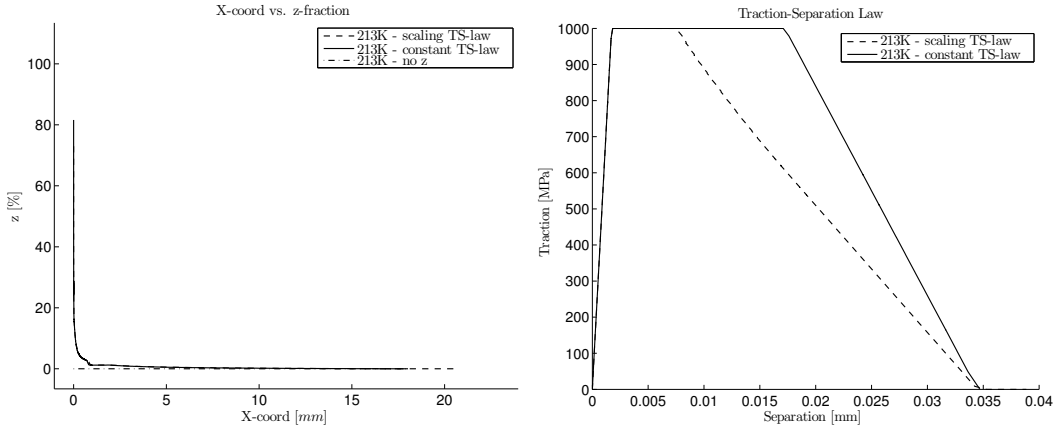


Figure 33: Figure (a) shows the z-fraction, z, as a function of the x-coordinate at the crack tip and figure (b) shows the TS-law when altered and when constant, both at 213 K.

12.3.2 233 K

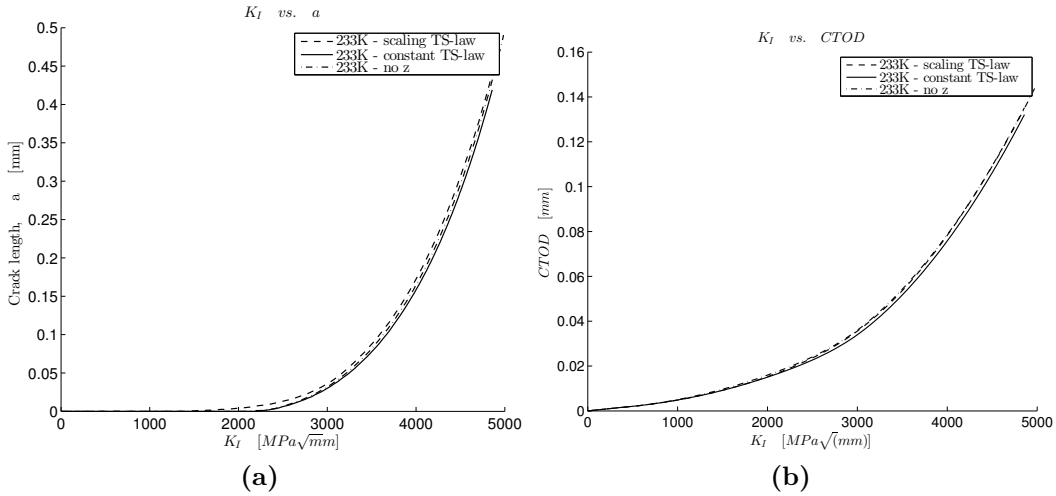
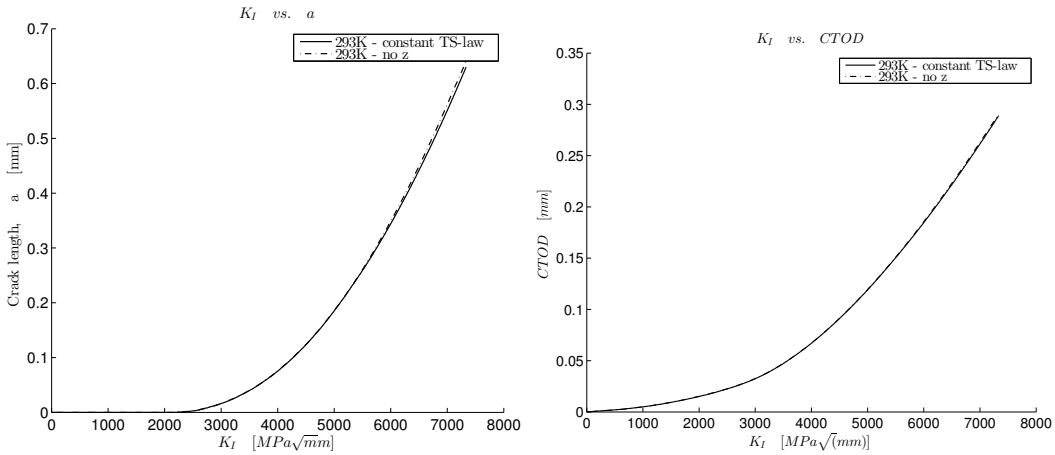


Figure 34: Figure (a) shows the crack length, a, as a function of the stress intensity factor and figure (b) shows the $CTOD$ as a function of the stress intensity factor, both at 233 K.

12.3.3 293 K



(a) **(b)**

Figure 35: Figure (a) shows the crack length, a, as a function of the stress intensity factor and figure (b) shows the $CTOD$ as a function of the stress intensity factor, both at 293 K.

13 Discussion and Conclusions

It is clear that temperature has a major effect on when phase transformation occurs. Running the same model at different temperatures clearly shows that there is most martensite at the temperature 213K. At the temperature 293K the simulations show a negligible fraction of martensite, which barely influences the crack propagation. This is why the choice was made not to look at the fourth temperature available for simulation, 313K. Since the crack propagation is examined when phase transformation is present, the highest temperature yields the same results for when the phase transformation is turned on and off.

Comparing the results from 213K looking at both a bilinear and a trapezoidal TS-law. There are visible differences in the behavior of both the crack length and the CTOD. The crack length and the CTOD are plotted against the stress intensity factor, K_I, which scales the applied displacement field. The plots shows that the crack with no phase transformation will open up more, and propagate further than when phase transformation is present for the same load amplitude.

Looking at the higher temperature, 233K, there are still visible differences, but when 293K is reached the difference in behavior is negligible. There is zero or almost zero martensite present and the curves showing both the crack length and the CTOD plotted against K_I almost coincide.

Looking at the alteration of the trapezoidal TS-law, it yields results that show that the crack propagates further. Since this implementation makes the fracture more brittle, thus lowering the fracture energy, the results turn out as expected. This is seen especially in figure 32.

Results of simulations when using the altered bilinear TS-law shows that the crack propagates slower, see figure 29. This is explained by the fact that the maximum traction is increased, which contributes to higher stresses in the vicinity of the crack tip. The higher stresses promote more martensitic phase transformation, which slows down the crack.

We can also see differences between the different TS-laws. This seen if one compares figures 29 and 32, looking at the load applied and the length of the crack. It is also seen in the amount of martensite transformed at the crack tip. So this makes the choice of TS-law very important.

The results show that the phase transformation affects the crack length slightly more than the CTOD throughout the simulations. This is most likely because the CTOD is affected by the shape of the crack tip. Crack tip blunting will have major implications on this quantity, which means that the crack length is a more suitable measurement.

It is interesting to think about what might cause these changes in behavior of the crack propagation. There are many different mechanisms that could give an effect. The different mechanisms are discussed earlier in this paper. The martensitic transformation yields a local increase in volume, which creates stresses that counteract the

stresses causing the crack to open.

The phase transformation can be seen as a plastic deformation and needs energy to take place. The process is a dissipative process which means that it absorbs energy which could otherwise be used for crack propagation. This could also be a mechanism affecting the crack propagation behavior.

Another mechanism that affects the crack propagation is the change in material properties as martensite is formed. Austenite is softer and more ductile than martensite. The fracture energy of the martensitic phase is also likely lower, since the phase is more brittle. These factors might also have an influence on the propagating crack.

It is very hard to determine which the main mechanisms contributing to the increase in toughness are. When measurements of fracture toughness are carried out, all mechanisms are active at the same time, and one has to be careful not to account for the effects multiple times when looking at the different mechanisms.

It was previously mentioned that the temperature affects the phase transformation. It also affects the material properties. The input for the user-material subroutine is calibrated for four different temperatures and the input is different for every temperature. This is important to acknowledge when looking at the stress intensity factor scaling the load applied. Comparing the different temperatures it shows much higher values for 293K than the lower temperatures, compare for example figure 29 and 31. This is because the crack tip is also affected by the material parameters, which change with the temperature. For example, the yield criterion in the constitutive model is temperature dependent. This makes comparisons between different temperatures hard.

It should be noted that the choice of TS-law and its parameters is very important for the accuracy of the simulations. The model is very sensitive to these changes, since they not only affect the results by themselves, but also the phase transformation in the surrounding continuum. For example, small increases in cohesive energy and maximum traction will slow down the crack by itself, but it will also increase the stresses around the crack tip, which promotes more phase transformation.

As can be seen in figure 33, the martensitic fraction is very high at the initial crack tip. From basic fracture mechanics it is known that, as we are closing in on the sharp crack tip, a stress singularity exists. This means that there are very high stress gradients. When very small elements are used to resolve these gradients, a very high fraction of martensite occurs locally in one element at the crack tip. Appart from this single element, between 5-10% martensite is transformed around the crack tip, which is reasonable.

14 Future Work

Another topic that is frequently discussed in literature related to the martensitic transformation and fracture mechanics is fatigue crack growth. As mentioned previously, the cohesive element implemented in this project includes a damage formulation which makes it compatible with cyclic loading. The phase transformation model which is used throughout this paper is however based on isotropic J_2-plasticity [26]. This type of plasticity model states that yielding occurs when the tensile stress reaches the a critical value, the yield stress, for uniaxial loading. If the loading is then reversed, the model predicts elastic unloading until the yield stress in compression is reached. Experimental results show that this prediction is inaccurate for metals and steel [52]. Uniaxial tests show that after being loaded plastically in tension or compression, the specimen yields at much lower stresses when the loading is reversed. This effect is called the *Bauschinger effect*. In order to capture this type of behavior during cyclic loading other hardening models are needed. When isotropic hardening is used the yield surface will most likely expand during the first cycles, and then stay fixed since the same load cycle is repeated. A more suitable model for this type of analysis is kinematic hardening, which means that the yield surface retains its shape and size, while the position in the deviatoric plane changes with plastic deformation. This means that the Bauschinger effect can be captured, since yielding occurs earlier when loading is reversed. Since the derivation and implementation of such a model is fairly complex and time consuming, it is not included in this project. It would, however, be an interesting topic for future projects.

The parameters chosen for the simulations carried out in this paper are chosen based on results from different articles. No experiments or measurements to predict these parameters are currently available. Since the choice of parameters is close to the mean values of the parameters found in the literature, there is at least *some* support. Since the cohesive zone model is a phenomenological model, experiments could aid in choosing parameters and understanding the behavior of martensite and austenite during fracture. It would also be interesting to find more specific data on the difference in fracture toughness between the two phases. The martensitic phase is harder and more brittle, which means that fracture toughness most likely decreases, while the maximum traction could increase.

The idea of altering the TS-law is based on the fact that the cohesive zone model is a separate part of the whole model. It is connected to the continuum element but is not affected by the phase transformation in any way. From the results it is possible to see that the crack propagation is clearly affected without altering the TS-law. The results also reveal that altering the TS-law can have macroscopic effects on the crack propagation. This change is hard to connect to the theory since it is very hard to say which mechanisms are affecting the crack propagation. By altering the TS-law, several mechanism affecting the behavior might be taken in to consideration more than once. This means that the altering of the TS-law might be unnecessary, but it could be a way to account for the effects that may not captured by the continuum

model. This topic needs to be investigated further in order to draw conclusions.

Another important aspect that needs further investigation is the different mechanisms which affects the toughness increment. There are several mechanisms active in the constitutive model at the same time. It is known that there is an increase in volume when the phase transformation takes place. This volume expansion can lead to compressive stresses which relaxes the stresses around the crack tip. Other mechanisms which affects the the behavior of the crack tip includes hardening and energy dissipated by the phase transformation. Different geometries and boundary conditions could be used to promote or limit the effects of the volume expansion. If insufficient material surrounds the transformed martensite, the volume expansion will not give rise to compressive stresses. Further investigations could reveal which mechanisms are important, and if the proposed TS-law which is altered by the phase transformation is necessary to capture these effects.

Earlier mentioned is that the simulations are very time consuming. There are more than 600 000 elements in the model used for the simulations. This is definitely something that could be improved. For these simulation, a good choice would have been to use symmetry. But for some reason, still not clear why. The implementation of symmetry along the crack ligament made the cohesive elements behave i a strange way. There was not put too much time into trying to solve this problem which is why this should be done if continuing working with this model. It would halve the amount of elements an speed up the simulation significantly.

15 References

[1] https://www.nde-ed.org/EducationResources/CommunityCollege/Materials/Structure/metallic_structures.htm
NDT Resource center, collected 150415

[2] http://www.tecni-cable.co.uk/s.nl/ctype.KB/it.I/id.136/KB.36576/.f
Tecnicable, collected 180515

[3] *Abaqus Analysis, User's Manual: Volume IV*. Dassault Systèmes, 2011

[4] G. Alfano, M.A. Crisfield, *Finite Element Interface Models for Delamination Analysis of Laminated Composites: Mechanical and Computational Issues*, 2001

[5] M. Alfano, F. Furgiuele, A. Leonardi, C. Maletta, G.H. Paulino, *Mode I Fracture of Adhesive Joints Using Tailored Cohesive Zone Models*. Springer Science, 2008

[6] T.L. Anderson, *Fracture Mechanics - Fundamentals and Applications*. Department of Mechanical Engineering, Texas A&M University. CRC Press, 1991

[7] S. Antolovich, B. Singh, *On the Toughness Increment Associated with the Austenite to Martensite Phase Transformation in TRIP Steels*. Metallurgy Transactions 1971;2:2135-41

[8] S.D. Antolovich, D. Fahr, *An Experimental Investigation of the Fracture Characteristics of Trip Alloys*. Engineering Fracture Mechanics, Vol. 4, pages 133-144, 1972

[9] G.I. Barenblatt, *The Mathematical Theory of Equilibrium Cracks in Brittle Fracture*. Advances in Applied Mechanics vol. 7, pp 55-129, 1962

[10] M. Breveiller, F.D. Fischer, *Mechanics of Solids with Phase Changes*. CISM International Centre for Mechanical Sciences, Springer, 1997

[11] W. Brocks, D. Arafah, M. Madia, *Exploiting Symmetries of FE Models and Application to Cohesive Elements*. Milano/Kiel, 2013

[12] W.D. Callister, Jr & D.G Rethwisch, *Fundamentals of Materials Science and Engineering - An integrated approach*. John Wiley & Sons 2008, Third Edition

[13] J. Chen M. Crisfield, *Predicting Progressive Delamination of Composite Material Specimens via Interface Elements*. Mechanics of Composite Materials and Structures 6:301-317, 1999

[14] M.A. Crisfield, G.A.O. Davies, H.B. Hellweg, Y. Mi, *Progressive Delamination Using Interface Elements*. Journal of Composite Materials, 1998

[15] A. Cornec, I. Scheider, K-H. Schwalbe, *On the Practical Application of the Cohesive Model*. Institute for Materials Research, GKSS Research Centre Geesthacht, 2003

[16] J. Dash, H.M. Otte, *The Martensite Transformation in Stainless Steel.* Acta Metallurgica, Vol. 11, issue 10, pages 1169-1178, 1963

[17] T. Diehl, *Modeling Surface-Bonded Structures with ABAQUS Cohesive Elements: Beam-Type Solutions.*

[18] D.S. Dugdale, *Yielding of Steel Sheets Containing Slits*, Engineering Department, Univeristy College ov Swansea, 1959

[19] A.J. Dunbar, B. Eng, *Simulation of Ductile Crack Propagation in Pipeline Steels Using Cohesive Zone Modeling.* Department of Mechanical and Aerospace Engineering, Carleton University, Ottawa, Ontario, 2011

[20] A.J. Evily, J.L. Gonzalez Velazquez, *Fatigue Crack Tip Deformation Processes as Influenced by the Environment.* Metallurgical Transactions, Vol. 23A, 1992

[21] M.L. Falk, A. Needleman, J.R. Rice, *A Critical Evaluation of Cohesive Zone Models of Dynamic Fracture.* Jounal de Physique, 11, pp. Pr5-43-50, 2001

[22] Y. Freed, L. Banks-Sills, *Crack growth resistance of shape memory alloys by means of a cohesive zone model.* Journal of the Mechanics and Physics of Solids 55, pages 2157-2180, 2007

[23] G.W. Greenwood, R.H. Johnson, *The deformation of metals under small stresses during phase transformation.* 1964

[24] A.A. Griffith, *The Phenomena of Rupture and Flow in Solids.* 1920

[25] H. Hallberg, P. Håkansson, M. Ristinmaa, *A Constitutive Model for the Formation of Martensite in Austenitic Steels under Large Strain Plasticity.* Division of Solid Mechanics, Lund University, 2007

[26] H. Hallberg, P. Håkansson, M. Ristinmaa, *Thermo-Mechanically Coupled Model of Diffusionless Phase Transformation in Austenitic Steel.* Division of Solid Mechanics, Lund University, 2010

[27] H. Hallberg, L. Banks-Sills, M. Ristinmaa, *Crack Tip Transformation Zones in Austenitic Stainless Steel.* Division of Solid Mechanics, Lund University, 2011

[28] E. Hornbogen, *Martensitic Transformation at a Propagating Crack.* Acta Metallurgica, Vol. 26, pages 147-152, 1978

[29] C.E Inglis, *Stresses In a Plate Due to the Presence of Cracks and Sharp Corners.* Cambridge, 1913

[30] G.R Irwin, *Analysis of Stresses and Strains Near the End of a Crack Traversing a Plate.* Journal of Applied Mechanics 24, 361–364, 1957

[31] S. Krenk, *Non-Linear Modeling and Analysis of Solids and Structures.* Cambridge University Press, 2009

[32] H. Li, H. Yuan, *Simulation of Fracture and Fatigue Damage in Stainless Steel 304 Using a Cohesive Zone Model.* Department of Mechanical Engineering,

University of Wuppertal, Germany, 2013

[33] A. Lundberg, S. Eliasson, *Investigation and Comparison of Cohesive Zone Models for Simulation of Crack Propagation*, Division of Solid Mechanics, Lund University, 2015

[34] P.L. Mangonan, G. Thomas, *The Martensite Phases in 304 Stainless Steel.* Metallurgical transactions, Vol. 1, 1970

[35] H.R. Mayer, S.E. Stanzl-Tschegg, Y. Sawaki, M. Hühner, E. Hornbogen, *Influence of Transformation-Induced Crack Closure on Slow Fatigue Crack Growth Under Variable Amplitude Loading.* Fatigue Fracture Engineering Material Structure, Vol. 18, No. 9, pages 935-948, 1995

[36] Z. Mei, J.W. Morris Jr., *Influence of Deformation-Induced Martensite on Fatigue Crack Propagation in 304-Type Steels.* Metallurgical Transactions, Vol. 21A, 1990

[37] N. Moës, T. Belytschko, *Extended Finite Element Method for Cohesive Crack Growth.* Department of Mechanical Engineering, Northwestern University, USA, 2001

[38] A. Needleman, *A Continuum Model for Void Nucleation by Inclusion Debonding.* Journal of Applied Mechanics, Vol. 54, 1987

[39] A. Needleman, *An Analysis of Decohesion Along an Imperfect Interface* - 1988. International Journal of Fracture vol.42 pp.21-40, 1990

[40] F. Nilsson, *Fracture Mechanics - From theory to applications.* Royal Institute of Technology, 2001

[41] K. Ng, *Integration of Computational Models and Experimental Characterization to study internal frost damage in cementitious materials.* Michigan Technological University, 2012

[42] O. Nguyen, E.A. Repetto, M. Ortiz, R.A. Radovitzky, *A Cohesive Model of Fatigue Crack Growth.* International Journal of Fracture, Vol. 110, pages 351-369, 2001

[43] V. Olden, C. Thaulow, R. Johnsen, E. Ostby, *Cohesive zone modeling of hydrogen-induced stress cracking in 25% Cr duplex stainless steel.* Norwegian University of Science and Technology, 2007

[44] Y. Oshida, A. Deguchi, *Martensite Formation and the Related Toughness in 304 Stainless Steel During Low Temperatue Fatigue.* Fatigue Fracture Engineering Material Structure, Vol. 10, No. 5, pages 363-372, 1987

[45] N. Ottosen, H. Petersson, *Introduction to the Finite Element Method.* Prentice Hall International, 1991

[46] A. Pandolfi, P.R. Guduru, M. Ortiz, A.J. Rosakis, *Three Dimensional Cohesive-Element Analysis and Experiments of Dynamic Fracture in C300 Steel.* Inter-

national Journal of Solids and Structures, Vol. 37, pages 3733-3760, 2000

[47] K. Park, G.H. Paulino, *Computational Implementation of the PPR Potential-Based Cohesive Model in ABAQUS: Educational Perspective.* School of Civil & Environmental Engineering - Yonsei University and Department of Civil & Environmental Engineering - University of Illinois, 2012

[48] E.R. Parker, V.F. Zackay, *Enhancement of Fracture Toughness in High Strength Steel ny Microstructural Control.* Engineering Fracture Mechanics, Vol. 5, pages 147-165, 1973

[49] X.B. Ren, Z.L. Zhang, B.Nyhus, *Effect of Residual Stress on Cleavage Fracture Toughness by Using Cohesive Zone Model.* Fatigue and Fracture of Engineering Materials & Structures, Vol. 34, pages 592-603, 2011

[50] J.R. Rice, *A Path Independent Integral and the Approximate Analysis of Strain Concentration by Notches and Cracks.* Journal of Applied Mechanics Vol. 35, pp. 379-386, 1968

[51] D. Roylance, *Introduction to Fracture Mechanics.* 2001

[52] N. Ottosen, M. Ristinmaa, *The Mechanics of Constitutive Modeling.* Elsevier Ltd, First Edition 2005

[53] I. Scheider, *Cohesive Model for Crack Propagation Analyses of Structures with Elastic-Plastic Material Behavior.* GKSS Research Center Geesthacht, 2001

[54] I. Scheider, W. Brocks, *The Effect of the Traction Separation Law on the Results of Cohesive Zone Crack Propagation Analyses*, 2003

[55] J.C.J. Schellekens, R. De Borst, *On the Numerical Integration of Interface Elements.* International Journal for Numerical Methods in Engineering, vol. 36, pages 43-66, 1993

[56] O.D. Sherby, J. Wadsworth, D.R. Lesuer, C.K. Syn, *Revisiting the Structure of Martensite in Iron-Carbon Steels.* 2008

[57] T. Siegmund, W. Brocks, *The Role of Cohesive Strength and Separation Energy for Modeling of Ductile Fracture.* Fatigue and Fracture Mechanics: 30th Volume, pages 139-151, 2000

[58] C.S. Shin, S.H. Hsu, *On the Mechanisms and Behaviour of Overload Retardation in AISI 304 Stainless Steel.* Int J Fatigue, Vol. 15, pages 181-192, 1993

[59] T. Siegmund, W. Brocks *A Numerical Study on the Correlation Between the Work of Separation and the Dissipation Rate in Ductile Fracture*, 2000

[60] R.E. Smallman, *Modern Physical Metallurgy.* Butterworths, 1962

[61] S.H. Song, G.H. Paulino, W.G. Buttlar, *A Bilinear Cohesive Zone Model Tailored for Fracture of Asphalt Concrete Considering Viscoelastic Bulk Material.* Elsevier, 2006

[62] R.G. Stringfellow, *Mechanics of Strain-induced Transformation Toughening in Metastable Austenitic Steels.* Massachusetts Institute of Technology, 1990

[63] V. Tomar, J. Zhai, M.Zhou, *Bounds for Element Size in a Variable Stiffness Cohesive Finite Element Model.* Int. J. Numer. Meth. Engng, 61, pages 1894-1920, 2004

[64] V. Tvergaard, J.W. Hutchinson, *The Relation Between Crack Growth Resistance and Fracture Process Parameters in Elastic-Plastic Solids.* 1992

[65] J.A. Venables, *The Martensite Transformation in Stainless Steel.* Philosophical Magazine, Vol. 7, issue 73, pages 35-44, 1962

[66] d. Xie, A.M. Waas, *Discrete Cohesive Zone Model for Mixed-Mode Fracture Using Finite Element Analysis.* Department of Aerospace Engineering, The University of Michigan, 2006

[67] X.P. Xu, A. Needleman, *Numerical Simulations of Fast Crack Growth In Brittle Solids.* J. Mech. Phys. Solids, Vol. 42, No. 9, pages 1397-1434, 1994

[68] A. Yavari, *Generalization of Barenblatt's Cohesive Fracture Theory for Fractal Cracks.* California Institute of Technology, 2001

[69] S. Yi, S. Gao, *Fracture Toughening Mechanism of Shape Memory Alloys Due to Martensite Transformation*, 1999

[70] Y. Zeng, *Feasibility Study of Cohesive Zone Model on Crack Propagation in Pipeline Steel Under Monotonic and Fatigue Loading.* Faculty of Civil Engineering and Geosciences, Delft University of Technology, Netherlands, 2015

16 Refereces - Images

[1] http://classconnection.s3.amazonaws.com/38/flashcards/512038/png/bcc1316360320702.png

[2] http://classconnection.s3.amazonaws.com/38/flashcards/512038/png/fcc1316360426857.png

[3] http://4.bp.blogspot.com/-Nc2sB5h4vxc/U8ksI053IpI/AAAAAAAAADE/1MhCapQuapM/s1600/preview_html_m1468904e.gif

[4] http://oregonstate.edu/instruct/engr322/Exams/AllExams/S14/MTTwoImages/MT2-5.1.jpg

[5] http://practicalmaintenance.net/wp-content/uploads/TTT-curve-for-carbon-steel-AISI-1050.jpg

[6] http://upload.wikimedia.org/wikipedia/commons/thumb/e/e7/Fracture_modes_v2.svg/2000px-Fracture_modes_v2.svg.png

[7] Image from figure 2.1 in
T.L. Andersson, *Fracture Mechanics - Fundamentals and Applications*. Department of Mechanical Engineering, Texas A&M University. CRC Press, 1991.

[8] Image from figure 7a in
J.R. Rice, *A Path Independent Integral and the Approximate Analysis of Strain Concentration by Notches and Cracks*. Journal of Applied Mechanics Vol. 35, pp. 379-386, 1968

[9] Image from figure 1.1 in
I. Scheider, *Cohesive Model for Crack Propagation Analyses of Structures with Elastic-Plastic Material Behavior*. GKSS Research Center Geesthacht, 2001

[10] Image from figure 3b in
A. Yavari, *Generalization of Barenblatt's Cohesive Fracture Theory for Fractal Cracks*. California Institute of Technology, 2001

[11] Image from figure 2 in
H. Hallberg, L. Banks-Sills, M. Ristinmaa, *Crack Tip Transformation Zones in Austenitic Stainless Steel*. Division of Solid Mechanics, Lund University, 2011

[12] H. Hallberg, L. Banks-Sills, M. Ristinmaa, *Crack Tip Transformation Zones in Austenitic Stainless Steel*. Division of Solid Mechanics, Lund University, 2011

[13] Image from figure 2.13 in
K. Ng, *Integration of Computational Models and Experimental Characterization to study internal frost damage in cementitious materials*. Michigan Technological University, 2012

[14] Image from figure 3.4 in
I. Scheider, *Cohesive Model for Crack Propagation Analyses of Structures with*

Elastic-Plastic Material Behavior. GKSS Research Center Geesthacht, 2001

[15] Image from figure 6 in
K. Park, G.H. Paulino, *Computational Implementation of the PPR Potential-Based Cohesive Model in ABAQUS: Educational Perspective*. School of Civil & Environmental Engineering - Yonsei University and Department of Civil & Environmental Engineering - University of Illinois, 2012

A Appendix

This is an example of an input-file used in ABAQUS. It shows the essential parts needed when using the two user subroutines, UMAT and UEL.

∗ ∗ An input-file for a test model
∗ ∗
∗ ∗Job name: Example_model
∗ ∗
∗ Preprint, echo=NO, model=NO, history=NO, contact=NO
∗ ∗
∗ ∗PARTS (Here we define the different parts needed in our model)
∗ ∗
∗ Part, name=Bottom
∗ Node
1, 5., 0.
2, 0., 0.
3, −5., 0.
4, 5., −5.
5, 0., −5.
6, −5., −5.
∗ Element, type=CPE4RT
1, 1, 2, 5, 4
2, 2, 3, 6, 5
∗ ∗
∗ Nset, nset=Set-1, generate
1, 6, 1
∗ Elset, elset=Set-1
1, 2
∗ ∗
∗ ∗Section: Section-1
∗ Solid Section, elset=Set-1, material=Material-1
,
∗ End Part
∗ ∗
∗ ∗Creating a part of our cohesive element.
∗ ∗
∗ Part, name=Cohesive
∗ Node
1, −5, 0
2, −4, 0
3, −3, 0
4, −2, 0
5, −1, 0
6, 0, 0

7, 1, 0
8, 2, 0
9, 3, 0
10, 4, 0
11, 5, 0
12, −5, 0
13, −4, 0
14, −3, 0
15, −2, 0
16, −1, 0
17, 0, 0
18, 1, 0
19, 2, 0
20, 3, 0
21, 4, 0
22, 5, 0
* *
* *Connecting the cohesive elements to the UEL.
* *
*USER ELEMENT, TYPE=U1, NODE=4, COORDINATES=2, PROPERTIES=6,
– VARIABLES=11
1, 2
* ELEMENT, TYPE=U1, ELSET=ONE
1, 1, 2, 13, 12
2, 2, 3, 14, 13
3, 3, 4, 15, 14
4, 4, 5, 16, 15
5, 5, 6, 17, 16
6, 6, 7, 18, 17
7, 7, 8, 19, 18
8, 8, 9, 20, 19
9, 9, 10, 21, 20
10, 10, 11, 22, 21
* *
* Nset, nset=TieNodeslavetop, generate
11, 22, 1
* Nset, nset=TieNodeslavebot, generate
1, 11, 1
* *
* *Setting the properties for the UEL.
* *
* *! PROPS(1) - Fracture energy
* *! PROPS(2) - Maximum traction
* *! PROPS(3) - alpha –> 1 for bilinear
* *! –> 2 for exponential

```
** *! -> 3 for trapeziodal
** *! PROPS(4) - lambdacritical1 = deltaTmax1/deltacritical
** *! PROPS(5) - lambdacritical2 = deltaTmax2/deltacritical
** *! — OBS! Only needed when alpha=3
** *! PROPS(6) - Thickness
** *
** UEL PROPERTY, ELSET=ONE
25, 1000, 3, 0.05, 0.5, 1
1
** End Part
** *
** Part, name=Top
** Node
1, −5., 0.
2, 0., 0.
3, 5., 0.
4, −5., 5.
5, 0., 5.
6, 5., 5.
** Element, type=CPE4RT
1, 1, 2, 5, 4
2, 2, 3, 6, 5
** *
** Nset, nset=Set-1, generate
1, 6, 1
** Elset, elset=Set-1
1, 2
** *
** *Section: Section-1
** Solid Section, elset=Set-1, material=Material-1
,
** End Part
** *
** *ASSEMBLY
** *
** Assembly, name=Assembly
** *
** Instance, name=Bottom-1, part=Bottom
** End Instance
** *
** Instance, name=Cohesive-1, part=Cohesive
** End Instance
** *
** Instance, name=Top-1, part=Top
** End Instance
```

* *
* *Making different sets of the geometry
* *
* *All nodes
* *
* Nset, nset=allbot, instance=Bottom-1, generate
1, 6, 1
* Elset, elset=allbot, instance=Bottom-1
1, 2
* Nset, nset=alltop, instance=Top-1, generate
1, 6, 1
* Elset, elset=alltop, instance=Top-1
1, 2
* *
* *Nodes for BC
* *
* Nset, nset=BCbot, instance=Bottom-1, generate
4, 6, 1
* Elset, elset=BCbot, instance=Bottom-1
1, 2
* Nset, nset=BCtop, instance=Top-1, generate
4, 6, 1
* Elset, elset=BCtop, instance=Top-1
1, 2
* *
* *Outer rim for bc and displacement
* *
* Nset, nset=Outerrim, instance=Bottom-1
1, 3, 4, 5, 6
* Nset, nset=Outerrim, instance=Top-1
1, 3, 4, 5, 6
* Elset, elset=Outerrim, instance=Bottom-1
1, 2
* Elset, elset=Outerrim, instance=Top-1
1, 2
* *
* *Surfaces...
* *
* Elset, elset=surfcrackbotS1, internal, instance=Bottom-1
1, 2
* Surface, type=ELEMENT, name=surfcrackbot
surfcrackbotS1, S1
* Elset, elset=surfcracktopS1, internal, instance=Top-1
1, 2
* Surface, type=ELEMENT, name=surfcracktop

surfcracktopS1, S1
* *
* *Surfaces for tie constraint
* *
* Elset, elset=SurfTiebotS1, internal, instance=Bottom-1
1, 2
* Surface, type=ELEMENT, name=SurfTiebot
SurfTiebotS1, S1
* Elset, elset=SurfTietopS1, internal, instance=Top-1
1, 2
* Surface, type=ELEMENT, name=SurfTietop
SurfTietopS1, S1
* *
* Nset, nset=slavetop, instance=Cohesive-1, generate
11, 22, 1
* SURFACE, type=node, name=SLAVETOPSURF
slavetop, 1
* Nset, nset=slavebot, instance=Cohesive-1, generate
1, 11, 1
* SURFACE, NAME=SLAVEBOTSURF, TYPE=NODE
slavebot, 1
* *
* *Creating tie constraints
* *
* *Constraint: tiebot
* Tie, name=tiebot, type=NODE TO SURFACE, adjust=yes
SLAVEBOTSURF, SurfTiebot
* *
* *Constraint: tietop
* Tie, name=tietop, type=NODE TO SURFACE, adjust=yes
SLAVETOPSURF, SurfTietop
* *
* End Assembly
* *
* *PREDEFINED FIELDS
* *
* *Name: specimeninittemp Type: Temperature
* Initial Conditions, type=TEMPERATURE
alltop, 233.
allbot, 233.
* *
* *MATERIALS
* *
* Material, name=Material-1
* Conductivity

45.,
* Density
$7.8e-09$,
* Specific Heat, dependencies=1
$4.5e+08$, , 0.
$4.15e+08$, , 1.
* User Material, constants=27, unsymm
$77000., 167000., 609., 13.3211, 1175., 1175., 625., 625.$
$0.29, 198.083, 214.119, 1245.5, 1., 29.5307, 2.6491, 1.64$
$0.81, 0.185, 0.35, 1e-05, 233., 233., -1.25, 0.83$
$0.28, 0, 1$
* *
* *Create z-field using USDFLD
* USER DEFINED FIELD
* Depvar
14,
* *
* * − − − − − − − − − − − −
* *
* *STEP: Step-1
* *
* Step, name=Step-1, nlgeom=YES, inc=100000
* Coupled Temperature-displacement, creep=none, deltmx=10.
$5e-02, 1., 1e-16, 0.1$
* *
* *BOUNDARY CONDITIONS
* *
* *Name: bcedgetemp Type: Temperature
* Boundary
$outerrim, 11, 11, 233.$
* *
* *The displacement BC
* *
* Restart, write, frequency=10
* Print, solve=NO
* *
* *FIELD OUTPUT: F-Output-1
* *
* Output, field
* Node Output
CF, NT, RF, RFL, U
* Element Output, directions=YES
HFL, LE, PE, PEEQ, PEMAG, S, SDV
* Contact Output
CDISP, CSTRESS

* *
* *HISTORY OUTPUT: H-Output-1
* *
* Output, history
* Energy Output
ALLAE, ALLCD, ALLDMD, ALLEE, ALLFD, ALLIE, ALLJD, ALLKE, ALLKL,
– ALLPD, ALLQB, ALLSD, ALLSE, ALLVD, ALLWK, ETOTAL
* Contact Output
ENRRT11, ENRRT12, ENRRT13
* End Step

B Appendix

This is the FORTRAN code for the UEL-subroutine implemented. This code is used to get the correct behavior of the cohesive element.

```
! ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
! UEL - User-element subroutine for ABAQUS in FORTRAN code
! ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
! Cohesive-model in 2D
! OBS! It is important to remember that the number of
! cohesive elements are hardcoded as 400000.
! ----------------------------------------------------------------
! For the input-file:
!          PROPS(1) - Fracture energy [N/mm] or [J/mm^2]
!          PROPS(2) - Maximum traction [Mpa]
!          PROPS(3) - alpha --> 1 for bilinear,
!                           --> 2 for exponential
!                           --> 3 for trapeziodal
!          PROPS(4) - lambda_cr1 = delta_Tmax1/delta_critical
!          PROPS(5) - lambda_cr2 = delta_Tmax2/delta_critical
!          --- OBS! Only needed when alpha=3
!          PROPS(6) - Thickness
! ----------------------------------------------------------------
! SVARS(1) = T_n (In GP 1) at delta_n_max
! SVARS(2) = T_n (In GP 2) at delta_n_max
! SVARS(3) = delta_n_max (for the element in GP1)
! SVARS(4) = delta_n_max (for the element in GP2)
! SVARS(5) = T_n (In GP 1) at current step
! SVARS(6) = T_n (In GP 2) at current step
! SVARS(7) = 0 or 1, failed status GP1
! SVARS(8) = 0 or 1, failed status GP2
! SVARS(9) = 0 or 1, softening status in GP2
! SVARS(10) = 0 or 1, failed status GP1
! SVARS(11) = 0 or 1, failed status GP2
! Remember to put VARIABLES = NSVARS in the input-file
! ----------------------------------------------------------------
 SUBROUTINE UEL (RHS, AMATRX, SVARS, ENERGY, NDOFEL, NRHS, &
  NSVARS, PROPS, NPROPS, COORDS, MCRD, NNODE, U, DU, V, A, &
  JTYPE, TIME, DTIME, KSTEP, KINC, JELEM, PARAMS, NDLOAD, &
  JDLTYP, ADLMAG, PREDEF, NPREDF, LFLAGS, MLVARX, DDLMAG, &
  MDLOAD, PNEWDT, JPROPS, NJPRO, PERIOD)
 INCLUDE 'ABA_PARAM.INC'
  DIMENSION RHS(MLVARX,*), AMATRX(NDOFEL,NDOFEL), PROPS(*), &
  SVARS(*), ENERGY(8), COORDS(MCRD, NNODE), U(NDOFEL), &
  DU(MLVARX,*), V(NDOFEL), A(NDOFEL), TIME(2), PARAMS(*), &
  JDLTYP(MDLOAD,*), ADLMAG(MDLOAD,*), DDLMAG(MDLOAD,*), &
  PREDEF(2, NPREDF, NNODE), LFLAGS(*), JPROPS(*)
! ----------------------------------------------------------------
```

```
! From the subroutine k3d we will get the stiffness matrix
! and the force vector
! ------------------------------------------------------------------
 call k3d(RHS,AMATRX,PROPS,COORDS,U,NDOFEL, &
                  MCRD,NNODE,NRHS, SVARS, TIME, JELEM)

 RETURN
 END
! ------------------------------------------------------------------
! ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 SUBROUTINE k3d(RHS, AMATRX, PROPS, COORDS, U, &
     NDOFEL, MCRD, NNODE, NRHS, SVARS, TIME, JELEM)
! ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 INCLUDE 'ABA_PARAM.INC'
  DIMENSION cur_coor(MCRD,NNODE), Rot_M(2,2), &
  COORDS(MCRD,NNODE), ShapeN(4), residual_global(NDOFEL,NRHS), &
   stiff_global(NDOFEL,NDOFEL), dummy_mat(MCRD,NDOFEL), &
   cur_coor_mid(MCRD,2),Trac_Jacob(MCRD,MCRD), Trac(MCRD,NRHS), &
   Transformation_M(NDOFEL,NDOFEL),stiff_local(NDOFEL,NDOFEL), &
   trans_dummy(NDOFEL,NDOFEL),residual_local(NDOFEL,NRHS), &
   cur_coor_local(MCRD,NNODE),delta_n(2), Bmat(MCRD, NDOFEL), &
   Bmat_T(NDOFEL,MCRD), Transformation_M_T(NDOFEL,NDOFEL), &
   GP_coord(2), U(ndofel), AMATRX(NDOFEL,NDOFEL), PROPS(*), &
   delta_u_l_gp(2), RHS(NDOFEL,NRHS), SVARS(11), TIME(2)
 DOUBLE PRECISION a_Jacob, dx, dy, el_length, shape_n1, SVARS, &
                  shape_n2, thickness, T_n, T_n_ny, delta_n_max, &
                  del_c, TIME, cracklength
! ------------------------------------------------------------------
 integer             :: ii, fail_stat
 integer, parameter  :: out_unit_trac=9996
 integer, parameter  :: out_unit_sep=9997
 integer, parameter  :: out_unit_crack=9998
! ------------------------------------------------------------------
! Defining the common blocks needed, these are also
! connected to the UMAT
! ------------------------------------------------------------------
  dimension k_z_x_kont_nodes(363*2,2), &
      k_z_x_func(363+1,2), k_global_coords_coh(400000,2)
  dimension z_coh_nodes(2,1)
 double precision k_z_x_kont_nodes, k_z_x_func, &
          k_global_coords_coh
 double precision z_coh_value, z_coh_nodes
 integer  JELEM, k_coh_count, found, k, l, i,  k_crack_elem
 common /z_x_function_values/ k_z_x_kont_nodes
 common /cohesive_coordinates/ k_global_coords_coh
 common /k_coh_counter/k_coh_count
 common /k_cracklength_counter/ k_crack_elem
! ------------------------------------------------------------------
```

```
! Sort this vector so we get the x-z function
! ------------------------------------------------------------------
  k_z_x_func = k_z_x_func*0.0d0
  call kbubblesort(k_z_x_kont_nodes,k_z_x_func,2)
! ------------------------------------------------------------------
! Saving the coordinates for the cohesive elements
! ------------------------------------------------------------------
 if (k_coh_count.eq.0) then
    k_global_coords_coh(JELEM,1) = COORDS(1,4)
    k_global_coords_coh(JELEM,2) = COORDS(1,3)
 endif
 if (JELEM.eq.400000.and.k_coh_count.eq.0) then
    k_coh_count = 1
 endif
! ------------------------------------------------------------------
! Resets all the values in the matrices and vectors to zero
! ------------------------------------------------------------------
 call k_mat_zero(AMATRX, NDOFEL, NDOFEL)
 call k_mat_zero(RHS, NDOFEL, NRHS)
 call k_mat_zero(Trac, MCRD, NRHS)
 call k_mat_zero(Trac_Jacob, MCRD, MCRD)
 call k_mat_zero(Bmat, MCRD, NDOFEL)
 call k_mat_zero(Bmat_T, NDOFEL, MCRD)
 call k_mat_zero(Transformation_M, NDOFEL, NDOFEL)
 call k_mat_zero(Transformation_M_T, NDOFEL, NDOFEL)
 call k_mat_zero(residual_local, NDOFEL, NRHS)
 call k_mat_zero(residual_global, NDOFEL, NRHS)
 call k_mat_zero(delta_u_l_gp,2,NRHS)
!------------------------------------------------------------------
! Updates the coordinates with respect to the displacements
! ------------------------------------------------------------------
       do i=1,MCRD
         do j=1,NNODE
            cur_coor(i,j) = COORDS(i,j) + U(2*(j-1)+i)
         end do
       end do
! ------------------------------------------------------------------
! Calculates the mid-plane
! ------------------------------------------------------------------
 do i = 1,2
   cur_coor_mid(i,1) = 0.5d0*(cur_coor(i,1) + cur_coor(i,4))
   cur_coor_mid(i,2) = 0.5d0*(cur_coor(i,2) + cur_coor(i,3))
 enddo
! ------------------------------------------------------------------
! Calculates the element length and dx, dy so that an
! angle can be found. In this way we are able to find
! our rotation matrix
! ------------------------------------------------------------------
```

```
        dx = cur_coor_mid(1,2)- cur_coor_mid(1,1)
        dy = cur_coor_mid(2,2)- cur_coor_mid(2,1)
        el_length = sqrt(dx**2+dy**2)
        Rot_M(1,1)= dx/el_length
        Rot_M(1,2)= dy/el_length
        Rot_M(2,1)= -dy/el_length
        Rot_M(2,2)= dx/el_length
! ------------------------------------------------------------------
! Must define the Jacobian between the referens and
! original coordinates. It is seen as a scale factor for
! the length, just need to check along a line integral.
! We do the integrations in the constant a_Jacob.
! ------------------------------------------------------------------
 thickness = PROPS(6)
 a_Jacob = el_length/2*thickness !Weight is one for two GP
! ------------------------------------------------------------------
! Calculate the transformation matrix
! ------------------------------------------------------------------
 call k_transformation_matrix_2D(Rot_M, Transformation_M, &
                                 NDOFEL)
 call k_mat_transpose(Transformation_M, Transformation_M_T, &
                      NDOFEL, NDOFEL)
! ------------------------------------------------------------------
! Calculates the local coordinates
! ------------------------------------------------------------------
! cur_coord_local=Rot_M*cur_coord (matrix multiplication)
! ------------------------------------------------------------------
   do i= 1,NNODE
        cur_coor_local(1,i) = Rot_M(1,1)*cur_coor(1,i)&
                             +Rot_M(1,2)*cur_coor(2,i)
        cur_coor_local(2,i) = Rot_M(2,1)*cur_coor(1,i)&
                             +Rot_M(2,2)*cur_coor(2,i)
   enddo
! ------------------------------------------------------------------
! Calculates the local opening/separation of the element
! y-coordinates, n in local coordinates
! ------------------------------------------------------------------
    delta_n(1) = cur_coor_local(2,4) - cur_coor_local(2,1)
    delta_n(2) = cur_coor_local(2,3) - cur_coor_local(2,2)
! ------------------------------------------------------------------
! Finding the z-value in the nodes, using the
! k_global_coords_coh(JELEM,*) and k_z_x_func to save in
! the vector z_coh_nodes.
! ------------------------------------------------------------------
 if (k_z_x_func(1,2).ne.0) then
    z_coh_nodes(1,1) = 0.0d0
    z_coh_nodes(2,1) = 0.0d0
```

```
      do l = 1,2
        found = 0
        if (JELEM.eq.1) then
           z_coh_nodes(1,1) = k_z_x_func(1,1)
           z_coh_nodes(2,1) = (k_global_coords_coh(JELEM,2)- &
                               k_z_x_func(1,2))*(k_z_x_func(2,1)&
                              -k_z_x_func(1,1))/(k_z_x_func(2,2)&
                              -k_z_x_func(1,2))+k_z_x_func(1,1)
           found = 1
        endif

        do k = 1,size(k_z_x_func,DIM=1)
        if (k_global_coords_coh(JELEM,l).le.k_z_x_func(k,2).and.&
                                  found.eq.0.and.JELEM.ne.1) then
            z_coh_nodes(l,1) = (k_global_coords_coh(JELEM,l)- &
                               k_z_x_func(k-1,2))*(k_z_x_func(k,1)&
                              -k_z_x_func(k-1,1))/(k_z_x_func(k,2)&
                              -k_z_x_func(k-1,2))+k_z_x_func(k-1,1)
            found = 1
        endif
        enddo
      enddo
    endif
!-----------------------------------------------------------------
! Using two Gauss points. The separation and z-value in
! the GP is wanted. GP lies along the coordinate axis s,
! which makes it very easy in 2D.
!-----------------------------------------------------------------
!       GP_coord(1)=-0.577350269189626
!       GP_coord(2)=0.577350269189626
!-----------------------------------------------------------------
! If we are using Newton-Cotes Integration!!
!-----------------------------------------------------------------
       GP_coord(1)=-1.0d0
       GP_coord(2)=1.0d0
!-----------------------------------------------------------------
! Calculates for both GP
! Both the z-value for the Gauss points and the local
! separation
!-----------------------------------------------------------------
    alpha = PROPS(3)
    del_c = PROPS(1)/PROPS(2)*(PROPS(3)+1.0d0)
    do i=1,2
       shape_n1 = 0.5*(1-GP_coord(i))
       shape_n2 = 0.5*(1+GP_coord(i))

       ! Local separation in GP
       !-----------------------
```

```
      delta_u_l_gp(i) = delta_n(1)*shape_n1+delta_n(2)*shape_n2
      delta_n_max = SVARS(2+i)
      T_n = SVARS(i)

      ! Z-values in GP
      !-----------------
       z_coh_value = z_coh_value*0.0d0
       z_coh_value = z_coh_nodes(1,1)*shape_n1+&
                     z_coh_nodes(2,1)*shape_n2
!-----------------------------------------------------------------
! Failed status ------> UPDATES WITH SVARS(10)
!-----------------------------------------------------------------
!-----------------------------------------------------------------
! Update for Unloading/Reloading, SVARS
!-----------------------------------------------------------------
! If del_u_gp > del_n_max we want to update SVARS(2+i),
! since it is the largest separation.
          if (delta_u_l_gp(i).GT.delta_n_max) THEN
             SVARS(2+i) = delta_u_l_gp(i)
          endif
!-----------------------------------------------------------------
! Here a function is called that calculates the C-matrix
! and the traction vector. It will call different
! subroutines depending on which traction-separation
! law is used (the alpha variable).
!
! alpha = 1 - bilinear
! alpha = 2 - exponential
! alpha = 2 - trapezoidal
!-----------------------------------------------------------------
! Resets values for the elements
     SVARS(9) = 0.0d0
!-----------------------------------------------------------------
 if (SVARS(9+i).NE.1) THEN
    if (alpha.eq.1) THEN
       call k_compute_traction_jacobian_bilinear (PROPS, &
              delta_u_l_gp, i, Trac, Trac_Jacob, NRHS, &
              JELEM, MCRD, delta_n, delta_n_max, T_n, &
              z_coh_value, SVARS)
    else if (alpha.eq.2) THEN
       call k_compute_traction_jacobian_exponential(PROPS, &
              delta_u_l_gp, i, Trac, Trac_Jacob, NRHS, &
              JELEM, MCRD, delta_n, z_coh_value, SVARS)
    else if (alpha.eq.3) THEN
       call k_compute_traction_jacobian_trapeziodal(PROPS, &
              delta_u_l_gp, i, Trac, Trac_Jacob, NRHS, &
              JELEM, MCRD, delta_n, z_coh_value, SVARS)
    endif
```

```
      T_n_ny = Trac(2,1)
      SVARS(4+i) = T_n_ny
!-----------------------------------------------------------------
! Update for cracklength with SVARS(9), and writing it in
! textfile
!-----------------------------------------------------------------
      if (JELEM.eq.1) THEN
         k_crack_elem = 0
      endif
      if (i.eq.2.and.SVARS(9).eq.1) then
         k_crack_elem = JELEM
      endif
      if (i.eq.2.and.JELEM.eq.400000.and.k_crack_elem.gt.0) THEN
         cracklength = k_crack_elem*5*10**(-4)
         open (unit=out_unit_crack, &
               file="/home/Guest/cracktip_pos.txt", &
               status="old",action="write", position="append")
         write (out_unit_crack,*) TIME(2), k_crack_elem, cracklength
         close (unit=out_unit_crack)
      endif
!-----------------------------------------------------------------
! Update for Reloading/Unloading, SVARS
!-----------------------------------------------------------------
      if ((delta_u_l_gp(i).GT.delta_n_max)) THEN
         SVARS(i) = T_n_ny
      endif
   else
      Trac(2,1) = 0.0d0
      Trac(1,1) = 0.0d0
      Trac_Jacob(1,1) = 0.0d0
      Trac_Jacob(2,2) = 0.0d0
      Trac_Jacob(1,2) = 0.0d0
      Trac_Jacob(2,1) = 0.0d0
      SVARS(i) = 0
   endif
!-----------------------------------------------------------------
! Calculates the B-matrix
!-----------------------------------------------------------------
            Bmat(1,1)=shape_n1
            Bmat(1,2)=0.0d0
            Bmat(1,3)=shape_n2
            Bmat(1,4)=0.0d0
            Bmat(1,5)=-shape_n2
            Bmat(1,6)=0.0d0
            Bmat(1,7)=-shape_n1
            Bmat(1,8)=0.0d0
            Bmat(2,1)=0.0d0
            Bmat(2,2)=shape_n1
```

```
            Bmat(2,3)=0.0d0
            Bmat(2,4)=shape_n2
            Bmat(2,5)=0.0d0
            Bmat(2,6)=-shape_n2
            Bmat(2,7)=0.0d0
            Bmat(2,8)=-shape_n1
!-----------------------------------------------------------------
! Does the matrix multiplication to get AMATRX AND RHS
!-----------------------------------------------------------------
 call k_mat_transpose (Bmat, Bmat_T, NDOFEL, MCRD)
 call k_matrix_multiplication(Trac_Jacob, Bmat, dummy_mat, &
      MCRD, NDOFEL, MCRD)
 call k_matrix_multiplication(Bmat_T, dummy_mat, stiff_local, &
      NDOFEL, NDOFEL, MCRD)
 call k_matrix_multiplication(Transformation_M_T, stiff_local, &
      trans_dummy, NDOFEL, NDOFEL, NDOFEL)
 call k_matrix_multiplication(trans_dummy, Transformation_M, &
      stiff_global, NDOFEL, NDOFEL, NDOFEL)
 call k_matrix_multiplied_scalar(stiff_global, NDOFEL, NDOFEL, &
      a_Jacob)
 call k_matrix_add(stiff_global, AMATRX, NDOFEL, NDOFEL)
 call k_matrix_multiplication (Bmat_T, Trac, residual_local, &
      NDOFEL, NRHS, MCRD)
 call k_matrix_multiplication(Transformation_M_T, &
      residual_local, residual_global, NDOFEL, NRHS, NDOFEL)
 call k_matrix_multiplied_scalar (residual_global, NDOFEL, NRHS, &
      a_Jacob)
 call k_matrix_add(residual_global, RHS, NDOFEL, NRHS)
 enddo
!-----------------------------------------------------------------
 RETURN
 END
!-----------------------------------------------------------------
!+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 SUBROUTINE k_transformation_matrix_2D(Rot_M, Transformation_M, &
                                       NDOFEL)
!+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
       include 'ABA_PARAM.INC'
       dimension Transformation_M(NDOFEL,NDOFEL), Rot_M(2,2)
       DOUBLE PRECISION dum
       do i = 1,4
          dum = 2*(i-1)
          Transformation_M(dum+1, dum+1) = Rot_M(1,1)
          Transformation_M(dum+1, dum+2) = Rot_M(1,2)
          Transformation_M(dum+2, dum+1) = Rot_M(2,1)
          Transformation_M(dum+2, dum+2) = Rot_M(2,2)
       enddo
       RETURN
```

```
      END
! ------------------------------------------------------------------
! ------------------------------------------------------------------
! Here are the different variants of traction-separation laws
! alpha = 1 - bilinear
! alpha = 2 - exponential
! aplha = 3 - trapeziodal
! ------------------------------------------------------------------
! ------------------------------------------------------------------
! BILINEAR
! ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 SUBROUTINE k_compute_traction_jacobian_bilinear (PROPS, &
            delta_u_l_gp, i, Trac, Trac_Jacob, NRHS, &
            JELEM, MCRD, delta_n, delta_n_max, T_n, &
            z_coh_value, SVARS)
! ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 include 'ABA_PARAM.INC'
 dimension Trac_Jacob(MCRD,MCRD), PROPS(*), Trac(MCRD,NRHS), &
           SVARS(11), delta_n(2), delta_u_l_gp(2), &
           delta_n_gp(2), kdel_cc_max(400000,2)
 DOUBLE PRECISION lambda_e, del_c, Trac, Trac_Jacob, &
           z_coh_value, T_max, Fracture_e, alpha, ambda_cr, &
           del_cc, delta_n, del_eff, a1, T_n, delta_n_max, &
           delta_n_gp, delta_u_l_gp, kdel_cc_max, &
           SVARS, ini_stiff
 integer i, JELEM
 common/del_cc_max/ kdel_cc_max
! ------------------------------------------------------------------
! Define all variables
! ------------------------------------------------------------------
 Fracture_e = PROPS(1)
 delta_n_gp(1)=delta_u_l_gp(1)
 delta_n_gp(2)=delta_u_l_gp(2)
 del_eff = sqrt(delta_n_gp(i)**2) ![mm]
 ini_stiff = PROPS(2)/((PROPS(1)/PROPS(2)*&
             (PROPS(3)+1.0d0))*PROPS(4))

 if (del_eff.le.kdel_cc_max(JELEM,i)) then
 ! If altering the TS-law with z-fraction remove comment
    !T_max = PROPS(2)*(1-z_coh_value)+PROPS(2)*z_coh_value
    T_max = PROPS(2)
    del_cc_max(JELEM,i) = T_max/ini_stiff
 else
    T_max = ini_stiff*del_cc_max(JELEM,i)
 end if
    !T_max = PROPS(2)
    del_cc = T_max/ini_stiff
    del_c = 2*Fracture_e/T_max
```

```
      alpha = PROPS(3)
      ambda_cr = del_cc/del_c
! Not altering the TS-law
! ------------------------------------------------------------------
      !del_c = PROPS(1)/PROPS(2)*(PROPS(3)+1.0d0)
      !del_cc = del_c*ambda_cr   ![mm]
! ------------------------------------------------------------------
! Variables special for this TS_law
! ------------------------------------------------------------------
      lambda_e = sqrt((delta_n_gp(i)/del_c)**2) ![mm]
      anew_del_c=del_c
! ------------------------------------------------------------------
! Unloading/Reloading
! ------------------------------------------------------------------
 if ((delta_n_gp(i).LT.delta_n_max).AND.&
          (delta_n_max.GT.del_cc)) THEN
      del_cc = delta_n_max
      T_max = T_n
      ambda_cr = del_cc/del_c
 endif
! ------------------------------------------------------------------
! Loading
! ------------------------------------------------------------------
! Solving overlapping issue
 if ((delta_n_gp(i).LT.0).AND.(delta_n_gp(2).LT.0)) THEN
!           write(*,100) 'Contact'
!           100 format (A,F)
            Trac(2,1) = 10*T_max*(delta_n_gp(i))/del_cc
            Trac(1,1) = 0.0d0
            Trac_Jacob(1,1) = 0.0d0
            Trac_Jacob(1,2) = 0.0d0
            Trac_Jacob(2,1) = 0.0d0
            Trac_Jacob(2,2) = 10*T_max/(ambda_cr*del_c)
        else
        if (del_eff.le.del_cc) then
!           write(*,101) 'loading'
!           101 format (A,F)
            Trac(2,1) = T_max*delta_n_gp(i)/del_cc
            Trac(1,1) = 0.0d0
            Trac_Jacob(1,1) = 0.0d0
            Trac_Jacob(1,2) = 0.0d0
            Trac_Jacob(2,1) = 0.0d0
            Trac_Jacob(2,2) = T_max/(ambda_cr*del_c)
! ------------------------------------------------------------------
! Softening
! ------------------------------------------------------------------
 elseif ((del_eff.gt.del_cc).AND.(del_eff.lt.del_c))then
!           write(*,102) 'softening'
```

```
!           102 format (A,F)
      Trac(2,1) = T_max*(1-lambda_e)/(1-ambda_cr)*1/&
                  lambda_e*(delta_n_gp(i)/del_c)
      Trac(1,1) = 0.0d0
      a1 = del_c*T_max/(1-ambda_cr)*(delta_n_gp(i)/&
           (lambda_e*del_c**2))**2+(1-lambda_e)*&
           (del_c*T_max)/(1-ambda_cr)*(1/&
           (lambda_e*del_c**2)-1/lambda_e**3*&
           delta_n_gp(i)**2/del_c**4)
      Trac_Jacob(1,1) = 0.0d0
      Trac_Jacob(1,2) = 0.0d0
      Trac_Jacob(2,1) = 0.0d0
      Trac_Jacob(2,2) = a1
! ----------------------------------------------------------------
! Status = Failed
! ----------------------------------------------------------------
         elseif (del_eff.gt.del_c) then
            !write(*,103) 'Failed'
            !103 format (A,F)
            if (i.eq.1) then
               SVARS(10) = 1.0d0
            endif
            if (i.eq.2) then
!              SVARS(9) = 1
               SVARS(11) = 1.0d0
            endif

            Trac(2,1) = 0.0d0
            Trac(1,1) = 0.0d0
            Trac_Jacob(1,1) = 0.0d0
            Trac_Jacob(2,2) = 0.0d0
            Trac_Jacob(1,2) = 0.0d0
            Trac_Jacob(2,1) = 0.0d0
         endif
         endif
         RETURN
         END
! ----------------------------------------------------------------
! EXPONENTIAL
! ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 SUBROUTINE k_compute_traction_jacobian_exponential (PROPS, &
              delta_u_l_gp, i, Trac, Trac_Jacob, NRHS, &
              JELEM, MCRD, delta_n, z_coh_value, SVARS)
! ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 include 'ABA_PARAM.INC'
 dimension Trac_Jacob(MCRD,MCRD), PROPS(*), Trac(MCRD,NRHS), &
       delta_n(2), delta_u_l_gp(2), delta_n_gp(2), SVARS(11)
 DOUBLE PRECISION lambda_e, del_c, SVARS, &
```

```
                T_max, Fracture_e, alpha, ambda_cr, del_cc, &
                del_eff, Trac, z_coh_value
 integer i, JELEM
! ------------------------------------------------------------------
! Define all variables
! ------------------------------------------------------------------
      Fracture_e = PROPS(1)
      T_max = PROPS(2)
      alpha = PROPS(3)
      ambda_cr = PROPS(4)
      delta_n_gp(1)=delta_u_l_gp(1)
      delta_n_gp(2)=delta_u_l_gp(2)
! ------------------------------------------------------------------
! Variables special for this TS_law
! ------------------------------------------------------------------
      del_c = Fracture_e/(T_max*ambda_cr*2.718281828)
      del_cc = del_c*ambda_cr

     ! If altering the TS-law with z-fraction
     !-------------------------------------------------------------
      !T_max = PROPS(2)*(1-z_coh_value)+5*PROPS(2)*z_coh_value
      !del_c = Fracture_e/(T_max*ambda_cr*2.718281828)

      del_eff = sqrt(delta_n_gp(i)**2)
      lambda_e = sqrt((delta_n_gp(i)/del_c)**2)
! ------------------------------------------------------------------
! Loading/Softening
! ------------------------------------------------------------------
! Solving overlapping issue
 if ((delta_n_gp(i).LT.0).AND.(delta_n_gp(2).LT.0)) THEN
!          write(*,200) 'Contact'
!          000 format (A,F)
    Trac(2,1) = 1*(delta_n_gp(i)/del_cc)*T_max*EXP&
                (1-delta_n_gp(i)/del_cc)
    Trac(1,1) = 0.0d0
    Trac_Jacob(1,1) = 0.0d0
    Trac_Jacob(1,2) = 0.0d0
    Trac_Jacob(2,1) = 0.0d0
    Trac_Jacob(2,2) = 1*(T_max*EXP(1-delta_n_gp(i)/del_cc)* &
                      (del_cc-delta_n_gp(i)))/(del_cc**2)
 else
    if (del_eff.le.del_c) then
!          write(*,201) 'Loading/softening'
!          201 format (A,F)
       Trac(2,1) = (delta_n_gp(i)/del_cc)*T_max*EXP
                   (1-delta_n_gp(i)/del_cc)
       Trac(1,1) = 0.0d0
       Trac_Jacob(1,1) = 0.0d0
```

```
        Trac_Jacob(1,2) = 0.0d0
        Trac_Jacob(2,1) = 0.0d0
        Trac_Jacob(2,2) = (T_max*EXP(1-delta_n_gp(i)/del_cc)* &
                          (del_cc-delta_n_gp(i)))/(del_cc**2)
!-----------------------------------------------------------------
! Status = Failed
!-----------------------------------------------------------------
      elseif (del_eff.gt.del_c) then
!         write(*,202) 'Failed'
!         202 format (A,F)
          if (i.eq.1) then
             SVARS(10) = 1.0d0
          endif
          if (i.eq.2) then
!             SVARS(9) = 1
             SVARS(11) = 1.0d0
          endif

          Trac(2,1) = 0.0d0
          Trac(1,1) = 0.0d0
          Trac_Jacob(1,1) = 0.0d0
          Trac_Jacob(2,2) = 0.0d0
          Trac_Jacob(1,2) = 0.0d0
          Trac_Jacob(2,1) = 0.0d0
      endif
      endif
      RETURN
      END
!-----------------------------------------------------------------
!-----------------------------------------------------------------
! TRAPEZOIDAL
!+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 SUBROUTINE k_compute_traction_jacobian_trapeziodal (PROPS, &
            delta_u_l_gp, i, Trac, Trac_Jacob, NRHS, &
            JELEM, MCRD, delta_n, z_coh_value, SVARS)
!+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 include 'ABA_PARAM.INC'
 dimension Trac_Jacob(MCRD,MCRD), PROPS(*), Trac(MCRD,NRHS), &
      delta_n(2), delta_u_l_gp(2), delta_n_gp(2), SVARS(11)
 DOUBLE PRECISION lambda_e, del_c, Trac, SVARS, &
   T_max, Fracture_e, alpha, lambda_cr1, lambda_cr2, del_cc, &
               del_eff, a1, del_ccc, z_coh_value
 integer JELEM, i
!-----------------------------------------------------------------
! Define all variables
!-----------------------------------------------------------------
     Fracture_e = PROPS(1)
     T_max = PROPS(2)
```

```
      alpha = PROPS(3)
      lambda_cr1 = PROPS(4)
      lambda_cr2 = PROPS(5)
      delta_n_gp(1)=delta_u_l_gp(1)
      delta_n_gp(2)=delta_u_l_gp(2)
!-----------------------------------------------------------------
! Variables special for this TS_law
!-----------------------------------------------------------------
      del_c = 2*PROPS(1)/(PROPS(2)*(1-lambda_cr1+lambda_cr2))
      del_cc = del_c*lambda_cr1
      del_ccc = del_c*lambda_cr2
      dist_x = del_c-del_ccc
      del_eff = sqrt(delta_n_gp(i)**2)
      lambda_e = sqrt((delta_n_gp(i)/del_c)**2)
!-----------------------------------------------------------------
! Changing the TS-law with respect to the z value
!-----------------------------------------------------------------
      del_ccc = del_ccc*(1-z_coh_value)+z_coh_value*del_cc
      lambda_cr2 = del_ccc/del_c
!-----------------------------------------------------------------
! Loading
!-----------------------------------------------------------------
! Solving overlapping issue
 if ((delta_n_gp(i).LT.0).AND.(delta_n_gp(2).LT.0)) THEN
!           write(*,300) 'Contact'
!           300 format (A,F)
     Trac(2,1) = 10*T_max*(delta_n_gp(i))/del_cc
     Trac(1,1) = 0.0d0
     Trac_Jacob(1,1) = 0.0d0
     Trac_Jacob(1,2) = 0.0d0
     Trac_Jacob(2,1) = 0.0d0
     Trac_Jacob(2,2) = 10*T_max/(lambda_cr1*del_c)
 else
     if (del_eff.le.del_cc) then
!           write(*,301) 'loading'
!           301 format (A,F)
        Trac(2,1) = T_max*delta_n_gp(i)/del_cc
        Trac(1,1) = 0.0d0
        Trac_Jacob(1,1) = 0.0d0
        Trac_Jacob(1,2) = 0.0d0
        Trac_Jacob(2,1) = 0.0d0
        Trac_Jacob(2,2) = T_max/(lambda_cr1*del_c)
!-----------------------------------------------------------------
! Constant Traction
!-----------------------------------------------------------------
     elseif (del_eff.le.del_ccc) then
!           write(*,302) 'Const. Traction'
!           302 format (A,F)
```

```
!           if (i.eq.2) then
!              SVARS(9) = 1
!           endif
            Trac(2,1) = T_max;
            Trac(1,1) = 0.0d0
            Trac_Jacob(1,1) = 0.0d0
            Trac_Jacob(1,2) = 0.0d0
            Trac_Jacob(2,1) = 0.0d0
            Trac_Jacob(2,2) = 0.0d0
! ------------------------------------------------------------------
! Softening
! ------------------------------------------------------------------
      elseif ((del_eff.gt.del_ccc).AND.(del_eff.lt.del_c))then
         if (i.eq.2) then
            SVARS(9) = 1.0d0
         endif
!           write(*,303) 'softening'
!           303 format (A,F)
         Trac(2,1) = T_max*(1-lambda_e)/(1-lambda_cr2)*1/&
                           lambda_e*(delta_n_gp(i)/del_c)
         Trac(1,1) = 0.0d0
         a1 = del_c*T_max/(1-lambda_cr2)*(delta_n_gp(i)&
                         /(lambda_e*del_c**2))**2+&
                          (1-lambda_e)*(del_c*T_max)&
                         /(1-lambda_cr2)*(1/(lambda_e*&
                          del_c**2)-1/lambda_e**3*delta_n_gp(i)&
                          **2/del_c**4)
         Trac_Jacob(1,1) = 0.0d0
         Trac_Jacob(1,2) = 0.0d0
         Trac_Jacob(2,1) = 0.0d0
         Trac_Jacob(2,2) = a1
! ------------------------------------------------------------------
! Status = Failed
! ------------------------------------------------------------------
         elseif (del_eff.gt.del_c) then
            if (i.eq.1) then
               SVARS(10) = 1.0d0
            endif
            if (i.eq.2) then
               SVARS(9) = 1.0d0
               SVARS(11) = 1.0d0
            endif
!           write(*,304) 'Failed'
!           304 format (A,F)
            Trac(2,1) = 0.0d0
            Trac(1,1) = 0.0d0
            Trac_Jacob(1,1) = 0.0d0
            Trac_Jacob(2,2) = 0.0d0
```

```
            Trac_Jacob(1,2) = 0.0d0
            Trac_Jacob(2,1) = 0.0d0
         endif
         endif
         RETURN
         END
!+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
         SUBROUTINE k_mat_transpose(A,B,n1,n2)
!+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
         include 'ABA_PARAM.INC'
         dimension A(n2,n1),B(n1,n2)
         do i=1,n1
            do j=1,n2
               B(i,j) = A(j,i)
            enddo
         enddo
         RETURN
         END
!-----------------------------------------------------------------------
!+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
subroutine k_matrix_add(A,B,n1,n2)
!+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
         include 'ABA_PARAM.INC'
         dimension A(n1,n2),B(n1,n2)
         do i=1,n1
            do j=1, n2
               B(i,j)=B(i,j)+A(i,j)
            enddo
         enddo
         return
         end
!-----------------------------------------------------------------------
!+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
         SUBROUTINE k_vector_zero(A,n1)
!+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
         include 'ABA_PARAM.INC'
         dimension A(n1)
         do i=1,n1
            A(i)=0.0d0
         enddo
         RETURN
         END
!-----------------------------------------------------------------------
!+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
         SUBROUTINE k_matrix_multiplication(A,B,C,n1,n2,n3)
!+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
         include 'ABA_PARAM.INC'
         dimension A(n1,n3),B(n3,n2),C(n1,n2)
```

```
      call k_mat_zero(C,n1,n2)
      do i=1,n1
         do j=1,n2
            do k=1, n3
               C(i,j) = C(i,j)+A(i,k)*B(k,j)
            enddo
         enddo
      enddo
      RETURN
      END
!-----------------------------------------------------------------
!+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE k_matrix_multiplied_scalar(A,n1,n2,fac)
!+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      include 'ABA_PARAM.INC'
      dimension A(n1,n2)
      do i=1,n1
         do j=1, n2
            A(i,j) = A(i,j)*fac
         enddo
      enddo
      RETURN
      END
!+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE k_mat_zero(A,n1,n2)
!+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      include 'ABA_PARAM.INC'
      dimension A(n1,n2)
      do i=1,n1
         do j=1, n2
            A (i,j) = 0.0d0
         enddo
      enddo
      RETURN
      END
!-----------------------------------------------------------------
!+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  subroutine kbubblesort(V,S,n)
!+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
! V a mx2 matrix - n=2 or 1, which coloumn you want to
! sort after (At the moment it works only for a 726x2 matrix)
! --> number of continuum elements along ligiment
! Sorts the components of the input matrix V,
! according to column using the Bubblesort algorithm
! The sorted matrix is returned in V with the elements in
! ascending order according to column n. After sorting the
! components it fixes the problem with double values at
! nodes and then returns S.
```

```
! ------------------------------------------------------------------
! V = Unsorted vector on input and sorted vector on output [1:n]
! n = coloumn to sort after

   double precision :: V(363*2,2), tmp1(2,1), tmp2(2,1), &
                       S(363+1,2)
   integer          :: i, j, nvals, p, n, s_counter

   nvals = size(V,DIM=1)

   do i=1,nvals-1
      do j=1,nvals-i
         if (V(i+j,n)<V(i,n)) then !Sorts after coloumn n
            tmp1(1,1) = V(i,1)
            tmp2(1,1) = V(i+j,1)
            tmp1(2,1) = V(i,2)
            tmp2(2,1) = V(i+j,2)
            V(i,1) = tmp2(1,1)
            V(i+j,1) = tmp1(1,1)
            V(i,2) = tmp2(2,1)
            V(i+j,2) = tmp1(2,1)
         end if
      end do
   end do
! ------------------------------------------------------------------
! Now we want to sort out the nodes that are connected,
! and take out the middle value of them, but not the first
!and last node.
! ------------------------------------------------------------------
   s_counter = 1
   S(1,1) = V(1,1)
   S(1,2) = V(1,2)
   S(363+1,1) = V(363*2,1)
   S(363+1,2) = V(363*2,2)

   do p = 2,size(V,DIM=1)-2,2
      s_counter = s_counter+1
      S(s_counter,1) = 0.5*(V(p,1)+V(p+1,1))
      S(s_counter,2) = 0.5*(V(p,2)+V(p+1,2))
   enddo

   return
   end subroutine kbubblesort
! ------------------------------------------------------------------
! ==================================================================
```