UNIVERSITY OF THESSALY
SCHOOL OF ENGINEERING
DEPARTMENT OF MECHANICAL ENGINEERING

Doctor of Philosophy Dissertation

Numerical Simulation of High-Cyclic Fatigue of Steel Structural Members

by

Gregory C. Sarvanis

Diploma of Civil Engineering, University of Thessaly, 2013
Master of Science (M.Sc.), University of Thessaly, 2015

Supervisor: Professor Spyros A. Karamanos

Submitted in partial fulfilment
of the requirements for the degree
of Doctor of Philosophy

2024

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ
ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ
ΤΜΗΜΑ ΜΗΧΑΝΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ

Διδακτορική Διατριβή

Αριθμητική Προσομοίωση Πολυκυκλικής Κόπωσης Μεταλλικών Δομικών Στοιχείων

από

Γρηγόριο Χ. Σαρβάνη

Πολιτικός Μηχανικός, Πανεπιστημίου Θεσσαλίας, 2013
Μεταπτυχιακό Δίπλωμα Ειδήκευσης (Μ.Δ.Ε.), Πανεπιστήμιο Θεσσαλιας, 2015

Επιβλέπων: Καθηγητής, Σπύρος Α. Καραμανος

Υπεβλήθη για την εκπλήρωση μέρους
των απαιτήσεων για την απόκτηση
του Διδακτορικού Διπλώματος

2024

Examination Committee

Dr. Spyros A. Karamanos (*Supervisor*)

Professor Department of Mechanical Engineering, University of Thessaly

Dr. Antonios E. Giannakopoulos (*Member of Advisory Committee*)

Professor School of Applied Mathematical and Physical Sciences, National Technical University of Athens

Dr. Vissarion Papadopoulos (*Member of Advisory Committee*)

Professor School of Civil Engineering, National Technical University of Athens

Dr. Costas Papadimitriou

Professor Department of Mechanical Engineering, University of Thessaly

Dr. Alexis Kermanidis

Associate Professor Department of Mechanical Engineering, University of Thessaly

Dr. Anna Zervaki

Assistant Professor School of Naval Engineering, National Technical University of Athens

Dr. Michalis Agoras

Assistant Professor Department of Mechanical Engineering, University of Thessaly

Κλείνοντας έναν σημαντικό κύκλο της ζωής μου, που λέγετε διδακτορικές σπουδές, θα ήθελα να ευχαριστήσω μεσα απο την καρδιά μου όλους τους ανθρώπους που μου στάθηκαν και με βοήθησαν όλα αυτά τα χρόνια. Θα ήθελα να ευχαριστήσω τον καθηγητή μου Σπύρο Καραμάνο όχι μόνο για την ακαδημαϊκή υποστήριξη που μου παρείχε αλλά κυρίως για τις αξίες και την αγάπη για μάθηση που μου μετέδωσε. Θα ήθελα να ευχαριστήσω την Αννα Ζερβάκη, τον Πολυνίκη Βαζούρα, Χάρη Παπαθεοχάρη, Ηλία Μπέλλα και Ιωάννα Παπαδίωτη για όλες τις συζητήσεις και ανταλλαγές απόψεων που έχουμε κάνει, ακαδημαϊκές ή μη. Θα ήθελα να ευχαριστήσω τους γονείς μου και την αδερφή μου οι οποίοι πάντοτε πίστευαν σε μένα και με ενθάρυναν πάντα να προχωράω μπροστά. Τέλος θα ήθελα να ευχαριστήσω τον άνθρωπό που ήταν πάντα εκεί, που αποτελεί την πηγή έμπνευσης και τον λόγο να γίνομαι καλύτερος, την γυναίκα μου.

Η διατριβή αυτή είναι αφιερώμενη
στην Γιαννούλα, στον Χρήστο και στην Αναστασία

Σαρβάνης Χ. Γρηγόρης

Table of Contents

1 Abstract....7
2 Περίληψη....8
3 Scope of the Present Dissertation and Novelties....9
4 Introduction....11
4.1 Fatigue of steels....11
4.2 Numerical techniques of crack growth simulation....12
4.3 Cohesive Elements....15
5 Developed Numerical Model....18
5.1 Description of the algorithm....18
5.2 Stress Calculation based on displacements....20
5.3 Developed Cohesive Element....24
5.4 Thermodynamic consistency of the model....34
5.5 J-Integral Calculation....37
6 Experimental Results....41
7 Simulation of Experiments....48
7.1 Compact specimen tests....48
7.2 SENB specimen tests....58
8 Experimental results for future research....64
9 Conclusion....73
10 References....74
11 Appendix....79
11.1 Cycle counting....79
11.2 Algorithm for J-integral calculation per element....81
11.3 Algorithm for cohesive element....83
11.4 Algorithm for bulk element calculations....94
11.5 Algorithm for finding q values of J-integral....97
11.6 Algorithm for finding paths for J-ntegral....100

1 Abstract

High cyclic fatigue is considered as one of the most common failure mechanisms of steel components. Over the last decades a lot of methodologies have been proposed for the design of steel components against high cyclic fatigue and lot of commercial softwares have been developed in order to help civil and mechanical engineers to address this problem. The scope of the present research effort is to study the problem of high cyclic fatigue of steel members using numerical methods. In the framework of this dissertation a robust and efficient algorithm has been developed in order to predict the crack tip position, the crack propagation direction and the crack growth rate of steel members. More specific, the crack growth rate is estimated based on the calculation of J-integral around the crack tip, while a novel methodology for an effective loading cycle skipping is proposed. The developed methodology, which employs Finite Element method and cohesive elements, has been verified against experimental results of two different specimens geometries (compact and SENB).

2 Περίληψη

Η πολυ-κυκλική κόπωση θεωρείται μια απο τις πιο συχνές αιτίες αστοχίας μεταλλικών δομικών στοιχείων. Τις τελευταίες δεκαετίες έχουν προταθεί πολλές μεθοδολογίες για των σχεδιασμό μεταλλικών δομικών στοιχείων έναντι πολυ-κυκλικής κόπωσης και πολλά εμπορικά πρόγραμμα έχουν αναπτυχθεί με σκοπό να επιταχύνουν τους υπολογισμούς που πραγματοποιούνται απο τους εκάστοτε μηχανικούς. Ο σκοπός της παρούσας διατριβής είναι να μελετήσει το πρόβλημα της πολυ-κυκλικής κόπωσης με την χρήση της μεθόδου των πεπερασμένων στοιχείων. Πιο συγκεκριμένα αναπτύχθηκε ένας αλγόριθμος ο οποίος μπορεί σε ένα μοντέλο πεπερασμένων στοιχείων να εντοπίσει το κρίσιμο σημείο στο οποίο θα αναπτυχθεί κάποια ρωγμή, να προσδιορίσει την διεύθυνση αυτής και να εκτιμίσει τον ρυθμό ανάπτυξής της. Το ολοκλήρωμα J και τα στοιχεία συνοχής (cohesive) συνδιάστηκαν ώστε να μπορέσει να γίνει η εκτίμηση του ρυθμού ανάπτυξης της ρωγμής. Επισπρόσθετα η μεθοδολογία που αναπτύχθηκε προσφέρει μια αποτελεσματική προσπελαση των κύκλων φόρτισης πράγμα που κάνει το συγκεκριμένο αλγόριθμό ιδιαίτερα αποδοτικό σε υπολογιστικό χρόνο. Τέλος η προτεινώμενη μεθοδολογία πιστωποιήθηκε με πειραματικές δοκιμές οι οποίες πραγματοποιήθηκαν στα πλαίσια της συγκεκριμένης διατριβής.

3 Scope of the Present Dissertation and Novelties

Steel structures and components are quite prone to high cyclic fatigue. There are reported several failures of key steel components due to fatigue loading such as railway structures or aircraft failures [1]. A bottomless literature exists on this subject, which can safely leads to the conclusion that significant experience and knowledge have been developed around high cyclic fatigue. The basic approaches for the calculation of fatigue damage in structures can be divided into two main categories [2]. The first one is the total-life approach which incorporates the concept of S-N curves and various methodologies for taking into account several phenomena like stress concentrations, mean stress and multi-axial loading. While the second one is the Defect-tolerant approach, which is more conservative and involves empirical crack growth laws based on fracture mechanics. The state of the art in the design of steel structures against fatigue is to perform finite element analysis for the calculation of stresses in an un-cracked body and use of these stresses to predict fatigue life, using the total-life approach, as a post-processing calculation. An alternative approach is the calculation of crack propagation, due to high cyclic loading, during a finite element analysis. The benefits of such method are related to the account of changes of the stress field at the critical areas of the structure due to the development of the crack and the accurate calculation of critical crack length. Nevertheless, this approach is not commonly used due to the fact that requires a lot of pre- and post-processing effort such as re-meshing, results mapping, manual calculation of stress intensity factors and consecutive finite element analyses.

The cohesive elements can be used for the simulation of crack propagation in finite element meshes, eliminating the need of re-meshing technics and results mapping. This type of elements have been used quite successfully in de-lamination process of adhesive joints, and a lot of research has been conducted in this area. On the other hand, limited application of cohesive elements in fracture of bulk materials and especially in high cyclic fatigue of bulk material have been reported in the literature. Three are the main open issues in the usage of cohesive elements in the crack propagation of bulk materials [3]. The first one is related to a straight forward methodology for the calibration of cohesive elements in both monotonic and cyclic loading. The second one is related to the fact that steel components usually needs millions of cycles in order to fail. Therefore, it is practically impossible with the current hardware capabilities to simulate all this loading history in one analysis, as a result a methodology for skipping most of the cycles is mandatory. The last open problem in cohesive elements is the artificial compliance that is introduced in the model when cohesive elements is used in a large portion of the structure.

The scope of the present dissertation is to address effectively the open issues that stated in the previous paragraph. Therefore, in the framework of the present research effort a robust and effective algorithm has been developed capable to predict the position of crack tip, the direction of crack propagation and the crack growth rate in steel components. The methodology combines the cohesive elements with the J-integral calculation around the crack tip in order to overcome the problem of cyclic skipping and the material calibration of the Finite Element model. The main novelties of this dissertation can be summarized below:

- A modification of an existing cohesive element of the literature in order to account for cyclic loading has been developed. The proposed damage accumulation is simple and it can be calibrated automatically when the cohesive element is used together with the developed algorithm.
- The material calibration for cyclic loading can be achieved using a straight forward procedure. More specific the calibration process requires simply the experimental data of a crack growth rate test (Δ**K**-da/dN) or the parameters of Paris law.
- A novel cyclic skipping methodology is proposed which uses the J-integral calculation for the estimation of stress intensity factor and the cohesive element length in order to calculate the loading cycles that the crack needs to propagate through all element length.
- The developed algorithm is capable to predict automatically the position of crack tip, the direction of crack propagation and the crack growth rate during analysis, without the need of re-meshing or the need of consecutive analyses. The crack length is evaluated at the end of each step of the analysis.
- The evaluation of stress intensity factor is totally automatic. The algorithm is capable to find the crack tip at the end of each step of the analysis, and to find the appropriate element paths around the crack tip for the calculation of J-integral.

Moreover experimental tests have been conducted in order to be used for the verification of the developed algorithm. More specifically, the proposed methodology has been verified against five crack growth tests, three in CT-specimens, two in SENB-specimens and one monotonic CTOD tests. It is also important to mention the limitations of the current work. The current algorithm has been developed for 2D analysis, and can be applied effectively only in problems that are under plain strain conditions. Moreover, the usage of cohesive elements creates the limitation of crack propagation only in the direction of cohesive elements. Therefore, the mesh affects the direction of the crack propagation, this problem can be alleviated using a relatively dense mesh.

4 Introduction

4.1 Fatigue of steels

Fatigue strength of materials and especially the fatigue strength of steels started to concern engineers the second half of 19th century during industrial revolution. By the beginning of 20th century until now a lot of progress has been done, reported amongst thousands of papers [1], towards the understanding and the predictability of the fatigue failure of structures. A historical survey of fatigue in the last century can be found in [4, 5, 6]. The last analyzed the fatigue of materials and structures from different points of view including material phenomenon, prediction models and fatigue properties of structures, summarized the milestones in this area in the last half of the 20th century, like [7, 8, 9] and many others.

The high cyclic fatigue process generally involves three distinct stages: crack initiation, crack propagation, and final fracture. Crack initiation occurs at stress concentration areas such as notches, inclusions, or surface irregularities. These imperfections serve as the starting points for small cracks that develop as the material is subjected to cyclic loading. Once a crack has been initiated, the process enters the second stage, which is the crack propagation. During this phase, the crack grows incrementally with each load cycle, often along slip planes in the crystalline structure of the steel. The rate of crack growth depends on factors such as the magnitude of the cyclic stresses, the frequency of loading, the presence of corrosive environments, and the material properties of the steel. The final fracture, occurs when the crack has grown sufficiently large that the remaining cross-sectional area of the material can no longer support the applied load.

Several factors influence the fatigue behavior of steel, including material composition, surface finish, temperature, and loading conditions. The microstructure of steel, determined by its composition and heat treatment, plays a significant role in its fatigue resistance. For example, steels with a fine-grained structure typically exhibit better fatigue properties than those with a coarse-grained structure [2]. Smooth surfaces with fewer defects or irregularities tend to exhibit better fatigue resistance, as they reduce the likelihood of crack initiation. Surface treatments, such as shot peening or case hardening, can also enhance fatigue resistance by introducing compressive residual stresses that impede crack initiation. Environmental conditions, such as temperature and exposure to corrosive media, further influence fatigue behavior. Elevated temperatures can accelerate crack growth, while corrosive environments can lead to stress corrosion cracking, a phenomenon where chemical interactions exacerbate mechanical fatigue.

4.2 Numerical techniques of crack growth simulation

Three are the most common numerical techniques for simulation of crack propagation numerically, the cohesive elements, which will be analyzed in detail in the next sections, the virtual crack closure technique and the extend finite element method. Those methodologies have been developed initially for monotonic fracture, nevertheless their application can be extended also in cyclic loading and fatigue. The virtual crack closure technique (VCCT) is based on the principles of linear elastic fracture mechanics (LEFM) and is commonly implemented in finite element analysis to assess the structural integrity and fatigue life of components [10, 11, 12]. VCCT involves incrementally extending a pre-existing crack in a virtual environment, allowing for the calculation of stress intensity factors (SIFs) at the crack tip. The method is based on the assumption that the strain energy released when a crack is extended by a certain amount is the same as the energy required to close the crack by the same amount. This simplification makes VCCT effective in determining the energy release rate using the displacements and forces at the crack tip, which can be obtained from FEA results. The virtual nature of this method allows for the evaluation of complex geometries and loading scenarios that are difficult to replicate experimentally.

For a crack propagating by an infinitesimal length $\Delta\alpha$, the energy release rate for each fracture mode can be calculated using equation 1,

$$G=\frac{1}{2\Delta\alpha}\left(F_I\ \Delta u_I\ +F_{II}\ \Delta u_{II}\ +F_{III}\ \Delta u_{III}\ \right) \qquad 1$$

where F_I , F_{II} and F_{III} are the forces at the crack tip associated with Modes I, II, and III, respectively, and Δu_I , Δu_{II} and Δu_{III} are the corresponding displacements at the crack faces after propagation. More specific, for Mode I, F_I is the opening force, and Δu_I is the relative displacement in the normal direction, for Mode II, F_{II} represents the shear force acting parallel to the crack, while Δu_{II} is the relative sliding displacement and for Mode III, F_{III} is the out-of-plane shear force, and Δu_{III} is the corresponding tearing displacement.

In the Virtual Crack Closure Technique (VCCT), the process begins by creating a finite element model of the structure with a predefined crack tip. Next, the crack is virtually extended by a small increment, and the displacements and reaction forces at the crack tip are obtained from the finite element analysis. Using these displacements and forces, the energy release rates for each fracture mode (Mode I, II, and III) are then calculated. Finally, these energy release rates are compared to the material critical energy release rate, or fracture toughness. If the computed energy release rate exceeds the material fracture toughness, crack propagation is predicted. In general VCCT is easy to implement in existing finite element models and requires only the forces and displacements at the crack

tip, making it computationally efficient. Moreover, it allows for the calculation of energy release rates in mixed-mode fracture scenarios, where more than one type of crack propagation mode is present, while it provides accurate results when applied to small crack increments.

The extended finite element method (XFEM) is also an advanced numerical technique used to model and analyze problems involving discontinuities such as cracks, voids, and material interfaces [13, 14, 15]. The traditional finite element method requires the mesh to conform to the geometry of discontinuities, while XFEM incorporates additional functions that allow the representation of these discontinuities independently of the mesh. This is achieved by enriching the solution space with functions that capture the behavior around the crack tip or discontinuity, such as unit step functions for jumps and asymptotic functions for crack-tip singularities. As a result, XFEM can accurately simulate crack initiation and crack propagation without the need for frequent re-meshing.

The key idea behind XFEM is to modify the classical displacement approximation in FEM to capture discontinuous behavior using enrichment functions. In a standard FEM formulation, the displacement field $u(x)$ is approximated within an element using shape functions $N_i\ (x)$ and nodal displacements u_i as presented in equation 2.

$$u(x)=\sum_{i=1}^{n} N_i\ (x)u_i \tag{2}$$

On the other hand in XFEM, additional enrichment terms are added to account for discontinuities and singularities. The displacement field is modified as shown in equation 3.

$$u(x)=\sum_{i=1}^{n} N_i\ (x)u_i + \sum_{j\in I_{enrichment}} N_j\ (x)H(x)a_j + \sum_{k\in I_{tip}} N_k\ (x)\sum_{l=1}^{4} F_I\ (x)b^l{}_k \tag{3}$$

where u_i is the standard displacement at the node i, $H(x)$ is the Heaviside step function to model discontinuities across a crack surface, a_j represents the nodal degrees of freedom (DOF) for the enriched nodes associated with discontinuities, $F_l(x)$ are the crack tip enrichment functions, which capture the singular stress field near the crack tip, and $b^l{}_k$ are the additional nodal degrees of freedom for capturing the crack tip behavior. This formulation introduces two types of enrichment. The Heaviside Enrichment, which is used to model the displacement jump across a crack surface, and takes the value of +1 on one side of the crack and -1 on the other, introducing a discontinuity in the displacement field and the Crack Tip Enrichment, which is used to capture the singularity in the stress field near the crack tip. This enrichment is based on asymptotic solutions for the stress field near the crack tip.

In XFEM the crack propagation is handled without re-meshing. The crack is allowed to grow along a path determined by the stress intensity factors (SIFs), which can be calculated from the enriched displacement field near the crack tip. The direction of crack

growth is often determined using fracture criteria such as the maximum circumferential stress criterion or the energy release rate criterion. The energy release rate G, which governs crack propagation, is calculated using equation 4,

$$G=\frac{K_I{}^2}{E'}+\frac{K_{II}{}^2}{E''} \quad 4$$

where K_I and K_{II} are the Mode I and Mode II stress intensity factors, and E', E'' are the effective modulus of elasticity for plane stress or plane strain conditions. Crack growth is typically modeled as an extension of the crack along the predicted path, using the enriched displacement field to update the stress and displacement fields as the crack propagates.

Each of the three aforementioned approaches offers advantages and disadvantages for simulating crack growth in materials. VCCT is primarily used in linear elastic fracture mechanics and is best suited for simulating crack growth in brittle materials where crack propagation paths are predetermined. While, XFEM allowing for the representation of discontinuities without conforming the mesh to the crack geometry, making it particularly advantageous for simulating complex crack patterns and dynamic crack propagation without any need of re-meshing. More-over extended finite element method is well-suited for cases where the crack path is not known in advance, therefore is effective in handling cracks that initiate and propagate in unpredictable directions. On the other hand, cohesive elements offer an effective method for modeling the process of crack initiation and crack growth by simulating the material fracture process zone with traction-separation laws, capturing non-linear fracture behaviors like de-lamination and yielding. Cohesive elements are advantageous for modeling ductile fracture and adhesive joints but can be computationally intensive as they require a dense mesh in the regions where the crack is expected to propagate.

4.3 Cohesive Elements

Cohesive elements considered as a special category of finite elements and they are used in computational fracture mechanics for the accurate simulation of crack initiation and crack propagation. They are able to describe the entire de-lamination process, starting from the initial formation of the crack to the propagation stage until the final separation of bodies. An important characteristic of cohesive elements is their capability to handle stress singularities near crack tips due to the fact that they offer a more accurate representation of the displacement field around the crack tip. Usually cohesive elements obviate the need for fine meshing near crack tips, a property that reduces significant the computational cost.

In general the traction – separation relationship of a cohesive element is described by a phenomenological mechanical relation across the cohesive surface which is depicted schematically in Figure 1. The force – displacement response under a monotonic loading can be divided into two main regions. In the first one, as the cohesive surfaces separates, the magnitude of the traction increases until an ultimate value ($T_{strength}$), while in the second one the traction decreases towards zero as the separation of the two cohesive surfaces increases monotonically ($s_{failure}$). In the case of cycling loading the traction should be decreased gradually for every loading cyclic. This macroscopic mechanical relation tries to describe several mechanisms, that take place in the material during monotonic and cyclic loading, like separation of atomic planes, ductile void growth and coalescence.

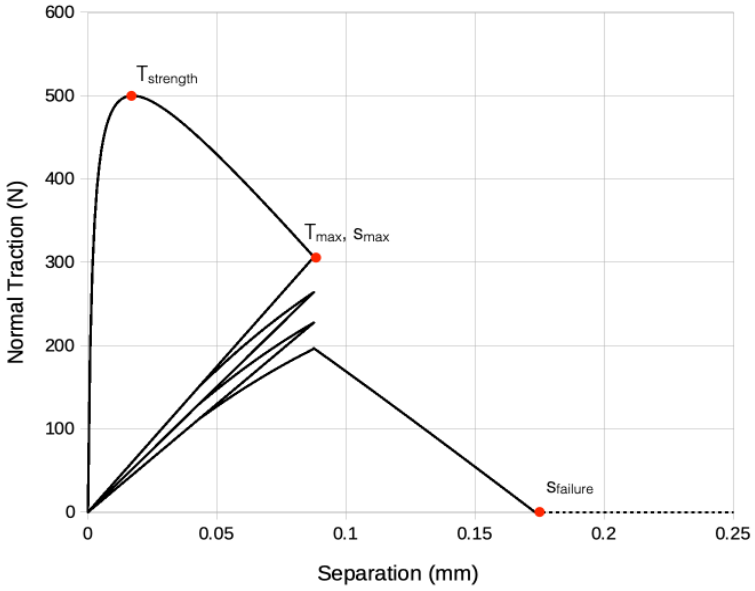

Figure 1. Schematic representation of traction separation response.

In the region near crack tip the stress distribution is deviate from linear elasticity and the distribution of cohesive forces are quite complicated, therefore the phenomenological approach of a high cohesive force that reduces rapidly away from the crack tip was an obvious solution that introduced for the first time in [16] and [17]. The distribution of cohesive traction proposed in [16] and [17] resulted to a finite maximum opening stress in the crack, which actually resolves the problem of the infinite opening stress which is predicted by the linear elastic theory. The work in [16] and [17] was the foundation for the development of cohesive elements which is initially implemented in a computational framework in [18]. More specific, in [18] the problem of concrete fracture was addressed by combining the concept of cohesive traction in a linear elastic concrete body. The cohesive traction concept found also some early applications in dynamic numerical simulations of earthquake raptures [19, 20, 21].

In general cohesive constitutive relations can be divided into two main categories the intrinsic and the extrinsic cohesive formulation. The first category concerns a traction-separation law of the cohesive element that it is initially elastic as shown in Figure 2, while the second category concerns a traction-separation law that it is initially rigid. The intrinsic formulation has been proposed in [24, 25, 26] together with the concept of cohesive surfaces as an initial/boundary value formulation without the need of any initial crack. The extrinsic formulation is introduced in [27] where the cohesive surfaces are added in the body when a critical condition is met. In [28] the possibility of numerical issues due to abrupt introduction of cohesive surfaces, in the case of extrinsic formulation, is discussed while in [29] the delay or suppress of the predicted onset of crack branching was observed for initially rigid cohesive elements.

A large number of scientific papers have been published regarding cohesive constitutive laws for monotonic loading. Some of the first traction – separation laws were proposed in [30, 31, 32] which combine an exponential tensile law with a periodic shear relationship. While in [25] an improved cohesive law was developed, which allows a complete loss of shear stiffness. Despite the effectiveness of cohesive law proposed in [25] and the simplicity of its numerical implementation there was an inconsistency at the work of separation between normal and shear direction which was properly addressed in [33, 34]. Effective and well established cohesive laws that are able to account for dissipation and hysteresis can be found also in [35, 36, 37], while in [38] a class of thermodynamic consistence cohesive laws was introduced, which guarantees that the damage of the model is increasing monotonically.

A lot of research effort exist not only in the development of new cohesive laws but also in the calibration methodologies of the cohesive elements [3]. In the case of monotonic loading different approaches can be found in the literature for the calibration of the cohesive law.

Significant efforts have been done in [39, 40, 41, 42, 43] based on molecular dynamics for the development of a calibration methodology. While different calibration approaches can be found for different materials such as porous plastic [44], ductile metal plates [45] and heterogeneous adhesives [46]. Moreover several researchers tried to develop experimental testing in order to determine parameters for the monotonic cohesive laws such as [47, 48, 49, 50, 51], while in [52] various calibration methods are discussed and compared. There is a gap in the literature regarding robust calibration methods of cohesive laws for the case of high cyclic fatigue loading of bulk materials and especially for steels. One of the main scopes of the present dissertation is to try to fill this gap.

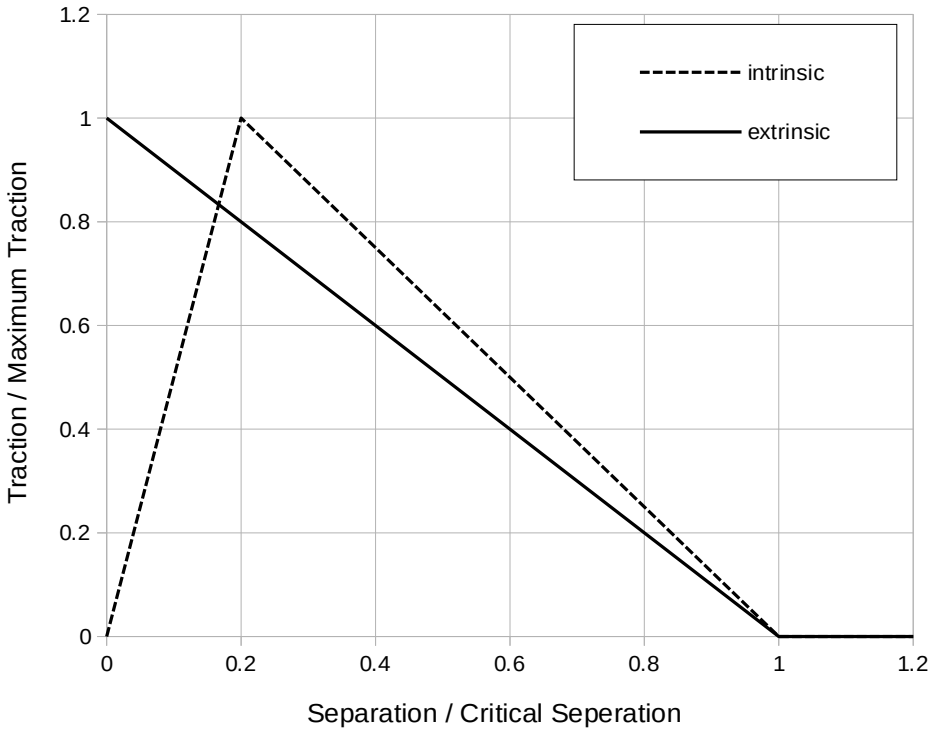

Figure 2. Schematic representation of intrinsic and extrinsic formulation.

5 Developed Numerical Model

5.1 Description of the algorithm

The algorithm developed for this analysis is composed of two main components. The first component involves the implementation of a cohesive element, which has been integrated using the "UEL" user subroutine within the commercial software Abaqus. This cohesive element can also function independently. During each increment of the analysis, the subroutine for the cohesive element is invoked to compute the right-hand-side vector, the Jacobian matrix, and the damage experienced by each cohesive element due to either monotonic or cyclic loading.

The second component of the algorithm is executed at the conclusion of each analysis step, performing a series of critical calculations. Initially, it identifies the critical cohesive element that has sustained the greatest damage. It then determines the node corresponding to the crack tip and identifies the direction in which the crack is propagating. Following this, the algorithm traces the element paths necessary for the calculation of the J-integral. Using the displacement field calculated by Abaqus, it then computes the strains and stresses of the elements required for the J-integral calculation. The J-integral is computed for each increment within the current step, creating a time history.

From the J-integral time history, the algorithm calculates the stress intensity factor range using a simple cycle counting algorithm. It then determines the crack growth rate based on the computed stress intensity factor range, according to a predefined relationship. Next, the algorithm estimates the number of cycles required for the crack to propagate through the length of the critical cohesive element. Once this is determined, it sets the damage parameter equal to 1, which indicates total failure of the critical element. Additionally, it calibrates the parameter of the cohesive elements that governs damage accumulation, based on the estimated number of cycles needed for the critical element's failure.

As a result, the algorithm provides, at the end of each step, the number of cycles required for the complete failure of the critical cohesive element and the subsequent extension of the crack by one element length. This method eliminates the need for re-meshing to account for crack extension, as well as the need for results mapping to incorporate the stress and strain fields from the previous step. By utilizing the J-integral, the algorithm effectively estimates the crack growth rate and calculates the number of cycles necessary for the crack

to propagate through the length of the critical cohesive element, making it a highly efficient and robust cycle-skipping technique.

The key steps of the algorithm are also summarized in the following bullets:

- Finds the critical cohesive element with the largest damage.
- Finds the node which corresponds to crack tip and the crack propagation direction.
- Finds the element paths for the calculation of J-integral.
- Calculates strains and stresses of elements, which will be used for J-integral, using the displacement field calculated by Abaqus.
- Calculates the J-integral for each increment of the current step (time history).
- Calculates the stress intensity factor range ΔK, from the J-integral time history of the step, using a simple cycle counting algorithm (section 11.1).
- Calculates the crack growth rate $(\frac{d\alpha}{dN})$ based on the calculated ΔK, from the given relationship.
- Estimates the number of the cycles needed in order the crack to propagate through the critical cohesive element length, and sets the damage parameter equal to 1 (total failure) for the critical element.
- Calibrates the parameter of the cohesive elements which drives the damage accumulation, based on the estimated number of cycles for failure of the critical element.

5.2 Stress Calculation based on displacements

Solid mechanics deals with the deformation of solid bodies under external loads. The primary unknowns in the system are the displacement fields, which describe how points in the material move relative to their original positions. Once the displacements are known, stresses and strains can be derived. In this section, the basic methodology of stress calculation based on the displacement field using linear finite element formulation for solid elements will be outlined.

The displacement field $u(x)$ represents the movement of each point x in the solid body from its original (undeformed) position to its deformed position and is presented in equation 5.

$$u(x)=\begin{bmatrix} u_x\ (x) \\ u_y\ (x) \\ u_z\ (x) \end{bmatrix} \tag{5}$$

where u_x , u_y , and u_z are the displacements in the x, y, and z directions respectively. The strains in the material describe how much deformation occurs and can be derived from the displacement field based on equation 6. For small deformations, the strain tensor $\boldsymbol{\varepsilon}$ is a linear function of the displacement gradient, given by the symmetric part of the displacement gradient tensor.

$$\varepsilon=\frac{1}{2}(\nabla \boldsymbol{u}+(\nabla \boldsymbol{u})^T\ ) \tag{6}$$

While strains can be represented in a matrix form for 3D problems in equation 7,

$$\begin{bmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{zz} \\ \gamma_{xy} \\ \gamma_{yz} \\ \gamma_{zx} \end{bmatrix} = \begin{bmatrix} \frac{\partial u_x}{\partial x} \\ \frac{\partial u_y}{\partial y} \\ \frac{\partial u_z}{\partial z} \\ \frac{\partial u_x}{\partial y}+\frac{\partial u_y}{\partial x} \\ \frac{\partial u_y}{\partial z}+\frac{\partial u_z}{\partial y} \\ \frac{\partial u_x}{\partial z}+\frac{\partial u_z}{\partial x} \end{bmatrix} \tag{7}$$

where ε_{xx} , ε_{yy} , ε_{zz} are the normal strains, and γ_{xy} , γ_{yz} , γ_{zx} are the shear strains. In linear elastic materials, stresses are related to strains by Hooke's law. For isotropic materials, the constitutive relation is described by equation 8.

$$\sigma = D\varepsilon \tag{8}$$

While stresses can be expressed in a matrix form for 3D problems as shown in equation 9

$$\sigma = \begin{bmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{zz} \\ \tau_{xy} \\ \tau_{yz} \\ \tau_{zx} \end{bmatrix} \tag{9}$$

where σ_{xx} , σ_{yy} , σ_{zz} are the normal stresses, and τ_{xy} , τ_{yz} , τ_{zx} are the shear stresses. The elasticity matrix D for an isotropic material is given by equation 10.

$$D = \frac{E}{(1+\nu)(1-2\nu)} \begin{bmatrix} 1-\nu & \nu & \nu & 0 & 0 & 0 \\ \nu & 1-\nu & \nu & 0 & 0 & 0 \\ \nu & \nu & 1-\nu & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1-2\nu}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1-2\nu}{2} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1-2\nu}{2} \end{bmatrix} \tag{10}$$

In FEM, the solid body is discretized into smaller elements, and the displacement field is approximated within each element. Each element is defined by its nodal points, and the displacement is assumed to vary according to the values of displacements at these nodes. The field of displacement can be defined using linear polynomial functions with respect to x and y coordinates (2 degree of freedom of each node). As a result the polynomial relationships that defines the displacements in every point in the element can be represented by equations 11 and 12, which can also represented in a matrix form as shown in equation 13.

$$u = a_1 + a_2\, x + a_3\, y + a_4\, xy \tag{11}$$

$$v = a_5 + a_6\, x + a_7\, y + a_8\, xy \tag{12}$$

The calculation of the generalized coordinates α_i can be achieved by solving the linear system presented by equation 14, which is the result of the calculation of the displacements u and v of each node of the element.

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} 1 & x & y & xy & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & x & y & xy \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \\ a_8 \end{bmatrix} \quad 13$$

$$\begin{bmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \\ u_3 \\ v_3 \\ u_4 \\ v_4 \end{bmatrix} = \begin{bmatrix} 1 & -a & -b & ab & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -a & -b & ab \\ 1 & a & -b & -ab & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & a & -b & -ab \\ 1 & a & b & ab & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & a & b & ab \\ 1 & -a & b & -ab & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -a & b & -ab \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \\ a_8 \end{bmatrix} \quad 14$$

Equation 14 can be written in compact form of equation 15, while equation 16 and 17 give the solution of the linear system of equations.

$$\boldsymbol{d} = A\,\boldsymbol{\alpha} \quad 15$$

$$\boldsymbol{\alpha} = A^{-1}\,\boldsymbol{d} \quad 16$$

$$\boldsymbol{\alpha} = \frac{1}{4ab} \begin{bmatrix} ab & 0 & ab & 0 & ab & 0 & ab & 0 \\ -b & 0 & b & 0 & b & 0 & -b & 0 \\ -a & 0 & -a & 0 & a & 0 & a & 0 \\ 1 & 0 & -1 & 0 & 1 & 0 & -1 & 0 \\ 0 & ab & 0 & ab & 0 & ab & 0 & ab \\ 0 & -b & 0 & b & 0 & b & 0 & -b \\ 0 & -a & 0 & -a & 0 & a & 0 & a \\ 0 & 1 & 0 & -1 & 0 & 1 & 0 & 1 \end{bmatrix} \boldsymbol{d} \quad 17$$

Therefore the displacement of each point inside the element can be calculated based on the node displacements as shown in equation 18.

$$\boldsymbol{u} = \begin{bmatrix} 1 & x & y & xy & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & x & y & xy \end{bmatrix} [A]^{-1}\,\boldsymbol{d} \quad 18$$

Rewriting equation 18 in the form of equation 19, one can define the shape functions N_i, where are presented by equations 20 to 23.

$$\boldsymbol{u} = \begin{bmatrix} N_1 & 0 & N_2 & 0 & N_3 & 0 & N_4 & 0 \\ 0 & N_1 & 0 & N_2 & 0 & N_3 & 0 & N_4 \end{bmatrix} \boldsymbol{d} \quad 19$$

$$N_1 = \frac{1}{4}\left(1-\frac{x}{a}\right)\left(1-\frac{y}{b}\right) \quad (20)$$

$$N_2 = \frac{1}{4}\left(1+\frac{x}{a}\right)\left(1-\frac{y}{b}\right) \quad (21)$$

$$N_3 = \frac{1}{4}\left(1+\frac{x}{a}\right)\left(1+\frac{y}{b}\right) \quad (22)$$

$$N_4 = \frac{1}{4}\left(1-\frac{x}{a}\right)\left(1+\frac{y}{b}\right) \quad (23)$$

Using the shape functions, equation 7 can be take the form of equation 24, which can be expressed in a compact form with equation 25, where matrix B is given by equation 26.

$$\varepsilon = \begin{bmatrix} u_{,x} \\ u_{,y} \\ u_{,x} + u_{,x} \end{bmatrix} \begin{bmatrix} N_{1,x} & 0 & N_{2,x} & 0 & N_{3,x} & 0 & N_{4,x} & 0 \\ 0 & N_{1,y} & 0 & N_{2,y} & 0 & N_{3,y} & 0 & N_{4,y} \\ N_{1,y} & N_{1,x} & N_{2,y} & N_{2,x} & N_{3,y} & N_{3,x} & N_{4,y} & N_{4,x} \end{bmatrix} \begin{bmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \\ u_3 \\ v_3 \\ u_4 \\ v_4 \end{bmatrix} \quad (24)$$

$$\varepsilon = B\,\boldsymbol{d} \quad (25)$$

$$B = \frac{1}{4ab}\begin{bmatrix} y-b & 0 & -y+b & 0 & y+b & 0 & -y-b & 0 \\ 0 & x-a & 0 & -x-a & 0 & x+a & 0 & -x+a \\ x-a & y-b & -x-a & -y+b & x+a & y+b & -x+a & -y-b \end{bmatrix} \quad (26)$$

Therefore using equation 25 and equation 8 one can calculate the stresses in a plane strain element using the displacements of the element nodes.

5.3 Developed Cohesive Element

In a continuum body every material point can be expressed by a vector $\boldsymbol{X}$ at the reference configuration and by a vector $\boldsymbol{x}$ at the current configuration. Therefore, assuming a constant Cartesian coordinate system, the displacement field $\boldsymbol{u}$ and the deformation gradient F can be expressed by equations 27 and 28, respectively.

$$\boldsymbol{u}=\boldsymbol{X}-\boldsymbol{x} \tag{27}$$

$$F=\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{X}} \tag{28}$$

Equation 29 describes the principle of virtual work for the case of a body with cohesive surfaces [24, 26]. Due to the existence of cohesive surfaces, there is the need of two constitutive models in order to define correctly the balance of energy. The first one relates stresses and strains of the bulk material, while the second one relates tractions and separations of cohesive surfaces.

$$\int_V s:\delta F dV-\int_{S_i} \boldsymbol{T}\cdot\delta\boldsymbol{\Delta}\, dS=\int_{S_{ex}} \boldsymbol{T}\cdot\delta\boldsymbol{u}\, dS-\int_V \rho\frac{\partial^2 \boldsymbol{u}}{\partial t^2}\cdot\delta\boldsymbol{u}\, dV \tag{29}$$

where s is the first Piola–Kirchhoff stress tensor, $\boldsymbol{\Delta}$ is the separation of cohesive surface, V, S_i , and S_{ex} , are the volume of the examined body, the internal surface area and the external cohesive surface area, respectively. The density of the bulk material is ρ, while the traction vector on the cohesive surface in the reference configuration is $\boldsymbol{T}$. The presence of a cohesive surface results in the addition of one more term to the energy balance as shown in equation 29. The first Piola–Kirchhoff stress tensor s produces work on the deformation gradients F over the bulk material, while the tractions $\boldsymbol{T}$ produce work on the separation of the cohesive surfaces $\boldsymbol{\Delta}$. Therefore, for the cohesive element the separation $\boldsymbol{\Delta}$ is a deformation measure, and the tractions $\boldsymbol{T}$ is the corresponding conjugate stress measure, which can be defined according to equation 30, where $\boldsymbol{v}$ is the normal vector at the cohesive surface.

$$\boldsymbol{T}=\boldsymbol{v}\, s \tag{30}$$

The traction–separation relationship used in the present study can be derived by a potential function which is described by equations 31, 32 and 33 and has been proposed in [33]. The main benefit of this model is that it offers a consistent traction separation law due to the fact that the separation traction is derived from a potential.

$$\Psi(\Delta_n\ ,\Delta_t\ )=min(\varphi_n\ ,\varphi_t\ )+H_n\ H_t \tag{31}$$

$$H_n = \Gamma_n \left(1-\frac{\Delta_n}{\delta_n}\right)^{\alpha}\left(\frac{m}{\alpha}-\frac{\Delta_n}{\delta_n}\right)^{m}+\langle \varphi_n - \varphi_t \rangle \quad 32$$

$$H_t = \Gamma_t \left(1-\frac{\Delta_t}{\delta_t}\right)^{\beta}\left(\frac{n}{\beta}-\frac{\Delta_t}{\delta_t}\right)^{n}+\langle \varphi_t - \varphi_n \rangle \quad 33$$

Δ_n and Δ_t of equations 31, 32 and 33 are the normal and the tangential separation of the cohesive surfaces respectively. The model energy constant parameters Γ_n , Γ_t , and the non-dimensional parameters m, n can be calculated based on the equations 34, 35, 36 and 37, respectively. The Macaulay bracket is defined as $\langle x \rangle = (|x|+x)/2$.

$$\Gamma_n = (-\varphi_n)^{\frac{\langle \varphi_n - \varphi_t \rangle}{\varphi_n - \varphi_t}}\left(\frac{\alpha}{m}\right)^{m} \quad 34$$

$$\Gamma_t = (-\varphi_t)^{\frac{\langle \varphi_t - \varphi_n \rangle}{\varphi_t - \varphi_n}}\left(\frac{\beta}{n}\right)^{n} \quad 35$$

$$m = \frac{\alpha(\alpha-1)\lambda_n^{\,2}}{1-\alpha\lambda_n^{\,2}} \quad 36$$

$$n = \frac{\beta(\beta-1)\lambda_t^{\,2}}{1-\beta\lambda_t^{\,2}} \quad 37$$

The critical differential separation of cohesive surfaces at which the element stiffness becomes zero are given by the following equation 38 and 39.

$$\delta_n = \frac{\varphi_n}{\sigma_{max}}\alpha\lambda_n \left(1-\lambda_n\right)^{\alpha-1}\left(\frac{\alpha}{m}+1\right)\left(\frac{\alpha}{m}\lambda_n+1\right)^{m-1} \quad 38$$

$$\delta_t = \frac{\varphi_t}{\tau_{max}}\beta\lambda_t \left(1-\lambda_t\right)^{\beta-1}\left(\frac{\beta}{n}+1\right)\left(\frac{\beta}{n}\lambda_t+1\right)^{n-1} \quad 39$$

The derivatives of the potential presented in equation 31 with respect to normal and tangential separation give the normal and tangential cohesive tractions as shown in equations 40 and 41.

$$\frac{\partial \Psi(\Delta_n, \Delta_t)}{\partial \Delta_n} = \frac{\Gamma_n}{\delta_n}\left[m\left(1-\frac{\Delta_n}{\delta_n}\right)^{\alpha}\left(\frac{m}{\alpha}+\frac{\Delta_n}{\delta_n}\right)^{m-1} - \alpha\left(1-\frac{\Delta_n}{\delta_n}\right)^{\alpha-1}\left(\frac{m}{\alpha}+\frac{\Delta_n}{\delta_n}\right)^{m}\right]H_t \quad 40$$

$$\frac{\partial \Psi(\Delta_n, \Delta_t)}{\partial \Delta_t} = \frac{\Gamma_t}{\delta_t}\left[n\left(1-\frac{\Delta_t}{\delta_t}\right)^{\beta}\left(\frac{m}{\alpha}+\frac{\Delta_n}{\delta_n}\right)^{m-1} - \beta\left(1-\frac{\Delta_t}{\delta_t}\right)^{\beta-1}\left(\frac{n}{\beta}+\frac{\Delta_t}{\delta_t}\right)^{n}\right]H_n \quad 41$$

As shown in Figure 3, the first part of the response, until the reach of the ultimate strength, is described by a non-linear response, while the second part, from the ultimate strength until total failure is controlled by shape factors α and β. Values larger than 2.0 result to a steep reduction of the traction with respect to separation, a behavior that is

closer to a brittle fracture, while values smaller than 2.0 result to a gentle reduction of the traction which is more representative for a plastic fracture.

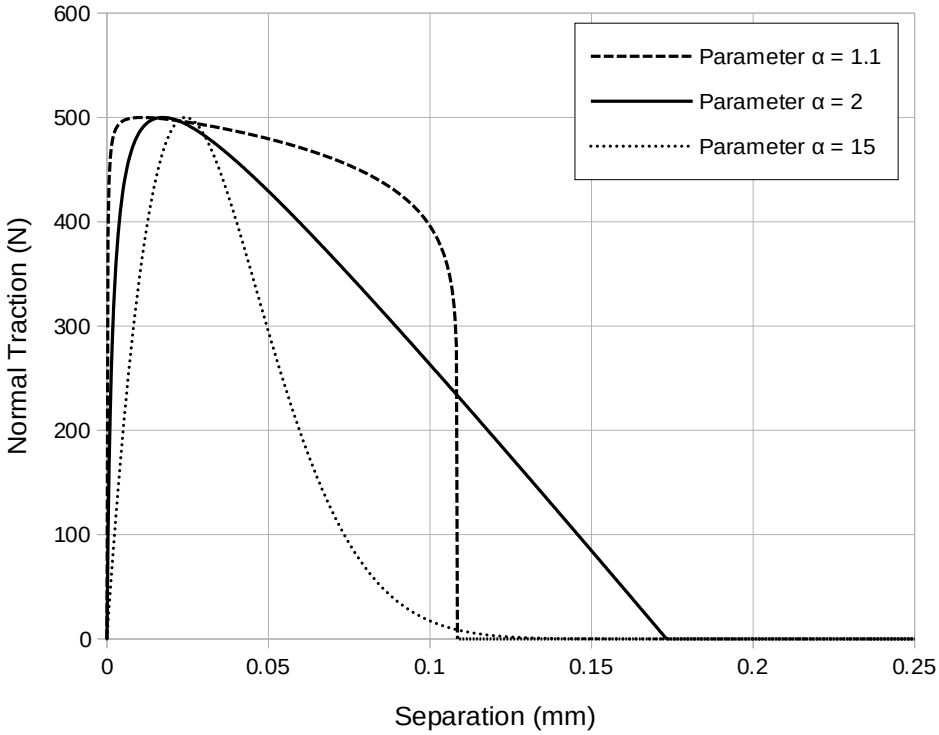

Figure 3. Normal traction with respect to normal sepαration for different values of shape factor a.

The aforementioned traction-separation relationship has been implemented in the user-subroutine "UEL" of commercial software Abaqus, as described in [33] and [53]. The implementation requires the calculation of right-hand-side vector, which is actually the nodal forces due to induced stresses in the element and the Jacobian matrix. The numerical model requires four parameters for each direction (normal and shear), in order to be calibrated for monotonic loading. These parameters are the fracture energies (φ_n , φ_t), which represents the area under traction-separation relationship, the maximum traction strengths (σ_{max} , τ_{max}), the shape factors (α, β), and the initial slope indicators (λ_n , λ_t), which is actually the ratio of the crack opening, which corresponds to maximum traction, with the final crack opening, which corresponds to failure.

The cohesive law described above is implemented as a constitutive relationship governing interface elements in a finite element mesh. The developed cohesive elements have zero thickness, and they introduced in the mesh by duplicating grid points along bulk element edges, as shown in Figure 4. The developed 2D linear cohesive element consists of four nodes, and each node has only two transitional degrees of freedom. Thus, the element global nodal displacement can be described by the vector of equation 42. While equation 43 describes the transformation of global displacement vector into the local displacement vector which is the displacement of each node in the normal and shear direction of the

element as shown in Figure 4. The introduction of the cohesive elements in the finite element mesh is becoming much easier due to the zero thickness of the developed element.

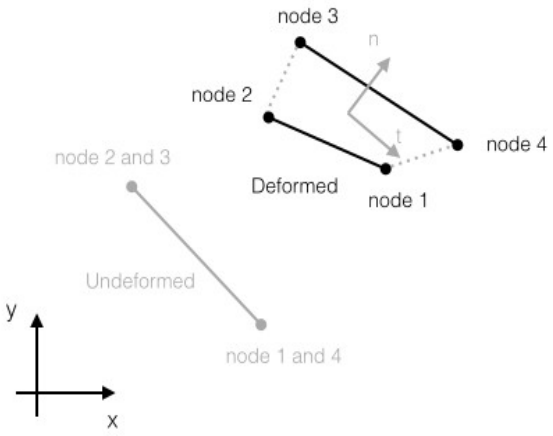

Figure 4. Schematic representation of undeformed and deformed cohesive element.

$$\boldsymbol{u}=[u_1\ ,u_2\ ,u_3\ ,u_4\ ,u_5\ ,u_6\ ,u_7\ ,u_8\ ] \tag{42}$$

$$\boldsymbol{u}_l=R\boldsymbol{u} \tag{43}$$

In equation 43 the rotational matrix R is given by equation 44

$$R=\begin{bmatrix} L & 0 & 0 & 0 \\ 0 & L & 0 & 0 \\ 0 & 0 & L & 0 \\ 0 & 0 & 0 & L \end{bmatrix} \tag{44}$$

where the 2D transformation matrix L is given by equation 45.

$$L=\begin{bmatrix} cos\theta & \sin\theta \\ -sin\theta & \cos\theta \end{bmatrix} \tag{45}$$

Parameter θ denotes the angle between the global and the local coordinate system. A counter clockwise scheme has been selected in the current computational implementation, with the lower left node of the element to be the first node, as shown in Figure 4. The nodal normal separation and tangential separation can be obtained from equation 46 based on the local nodal displacements.

$$\Delta_1=u_7-u_1\ ,\Delta_2=u_8-u_2\ ,\Delta_3=u_5-u_3\ ,\Delta_4=u_6-u_4 \tag{46}$$

The shape functions matrix is given by equation 47, while shape function N_1 and N_2 are given by equation 48.

$$N=\begin{bmatrix} N_1 & 0 & N_2 & 0 \\ 0 & N_1 & 0 & N_2 \end{bmatrix} \tag{47}$$

$$N_1=\frac{1-\xi}{2}, N_2=\frac{1+\xi}{2} \tag{48}$$

The global displacement–separation relation matrix B_c is given by equation 49, where $A=cos\theta$ and $B=\sin\theta$. The calculation of right-hand-side vector, and the Jacobian matrix are calculated numerically using two integration points (Gauss points) per element.

$$B_c = \begin{bmatrix} -AN_1 & -BN_1 & -AN_2 & -BN_2 & AN_2 & BN_2 & AN_1 & BN_1 \\ BN_1 & -AN_1 & BN_2 & -AN_2 & -BN_2 & AN_2 & -BN_1 & AN_1 \end{bmatrix} \quad 49$$

In the case of cyclic loading the tractions T_n and T_t of the element are calculated based on equations 50 and 51, where $\frac{\partial \Psi}{\partial \Delta_n}$ and $\frac{\partial \Psi}{\partial \Delta_t}$ are given by equations 40 and 41, while the scalar parameters d_{cn} and d_{ct} denote the damage due to cyclic loading in the normal and shear direction and are calculated based on equations 52 and 53. This is a common approach for damage mechanics that have been used by many researchers [38, 55, 56]. The damage accumulation is depended on the maximum traction strengths (σ_{max} ,τ_{max}), the surface separations which correspond to total failure (δ_n , δ_t), and the maximum cohesive surface separation s_{max} of each loading cycle, as shown in Figure 1.

$$T_n = \frac{\partial \Psi}{\partial \Delta_n}(1-d_{cn}) \quad 50$$

$$T_t = \frac{\partial \Psi}{\partial \Delta_t}(1-d_{ct}) \quad 51$$

$$\dot{d}_{cn} = f_n \, \dot{\Delta}_n \, \frac{s_{max}}{\delta_n \, \sigma_{max}} \quad 52$$

$$\dot{d}_{ct} = f_t \, \dot{\Delta}_t \, \frac{|s_{max}|}{\delta_t \, \tau_{max}} \quad 53$$

The rate of cyclic damage parameters $\dot{d}_{cn}$ and $\dot{d}_{ct}$ are linearly depended to the rate of cohesive surfaces separations $\dot{\Delta}_n$ and $\dot{\Delta}_t$, and in the numerical implementation they are activated only if the rate of cohesive surfaces separations are positive. Moreover, another criterion has been used which introduces a traction threshold, through the parameter g, for the activation of the cyclic damage as shown in equation 54, a similar threshold has been used in [59]. This criterion gives the capability to the model to accumulate damage only if the traction is larger than a specific value. In Figures 5 and 6 the effect of parameter g is shown, the smaller the parameter g is, the sooner the accumulation of damage starts at the reloading branch. Another important characteristic of equations 52 and 53 is that the damage accumulation rate is strongly related to the maximum cohesive surface separation s_{max} . More-specific, the larger the maximum cohesive surface separation s_{max} is, the larger is the cyclic damage accumulation rate, as shown in Figures 7 and 8, which is an expected behavior. Moreover, in the case of force control analysis, equations 52 and 53 have the ability to increase the rate of damage accumulation as the number of loading cycles increases due to the fact that s_{max} increases. While in the case of displacement control

analysis the rate of damage accumulation remains constant, since s_{max} remains constant, as shown in Figure 9.

$$\dot{d}_c = f\,\dot{\Delta}\frac{s_{max}}{T_{strength}\,\delta} \qquad if \quad \dot{\Delta}>0 \ \text{ and } \ (T - g\cdot T_{max})>0 \tag{54}$$

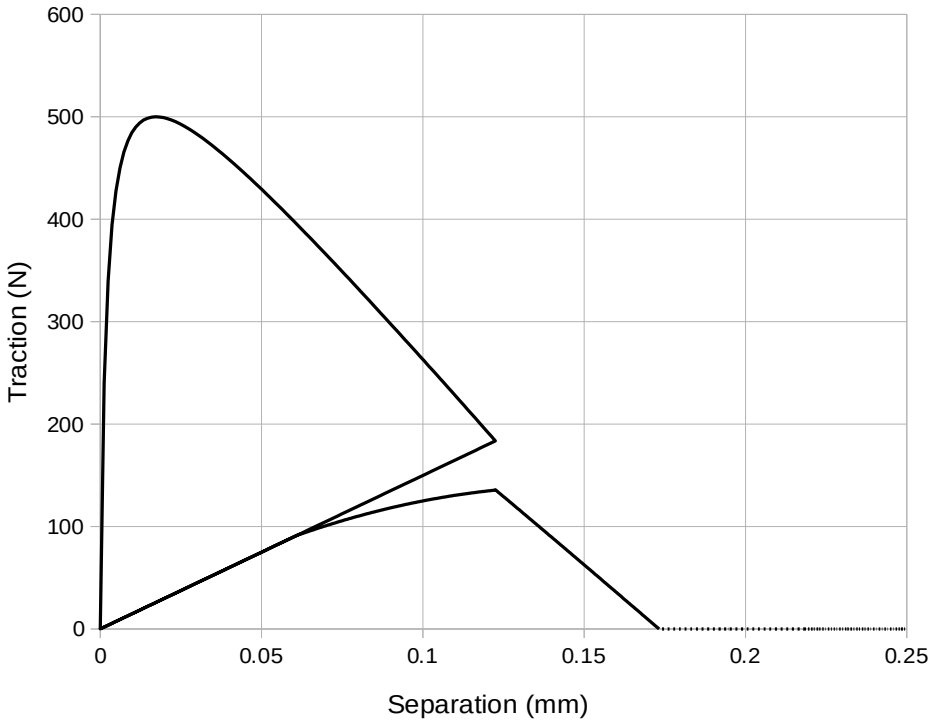

Figure 5. Traction – Separation relationship for one loading cycle and threshold parameter $g = 0.5$, the rest parameters of the model are $\sigma_{max}=1000$, $\varphi_n=100$, $\alpha=2.0$, $\lambda_n=0.1$.

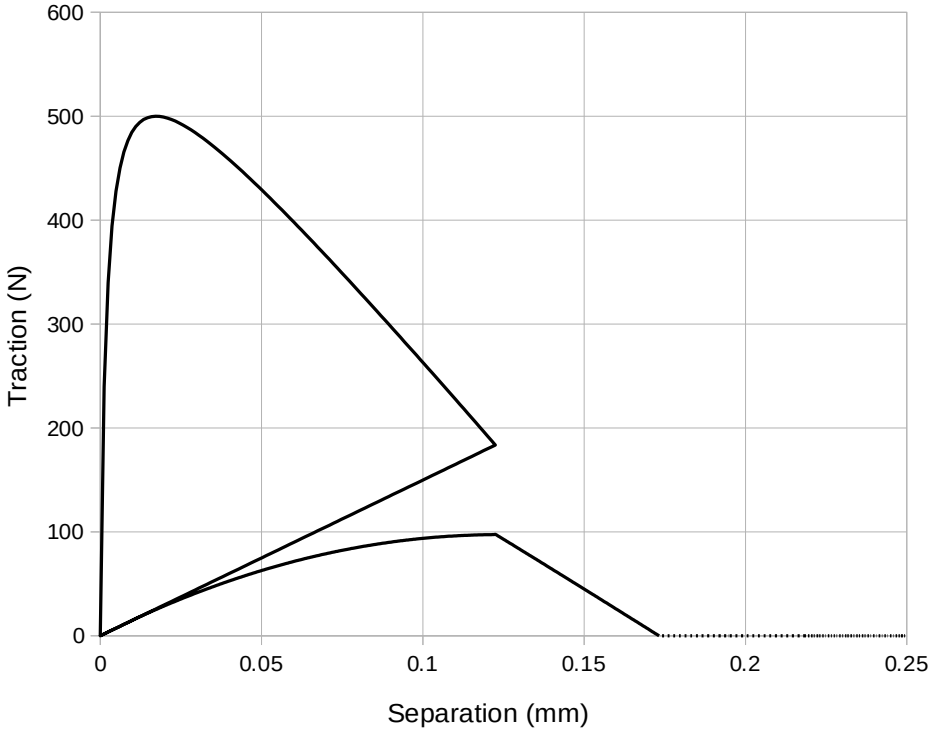

Figure 6. Traction – Separation relationship for one loading cycle and threshold parameter $g = 0.1$, the rest parameters of the model are $\sigma_{max} = 1000$, $\varphi_n = 100$, $\alpha = 2.0$, $\lambda_n = 0.1$.

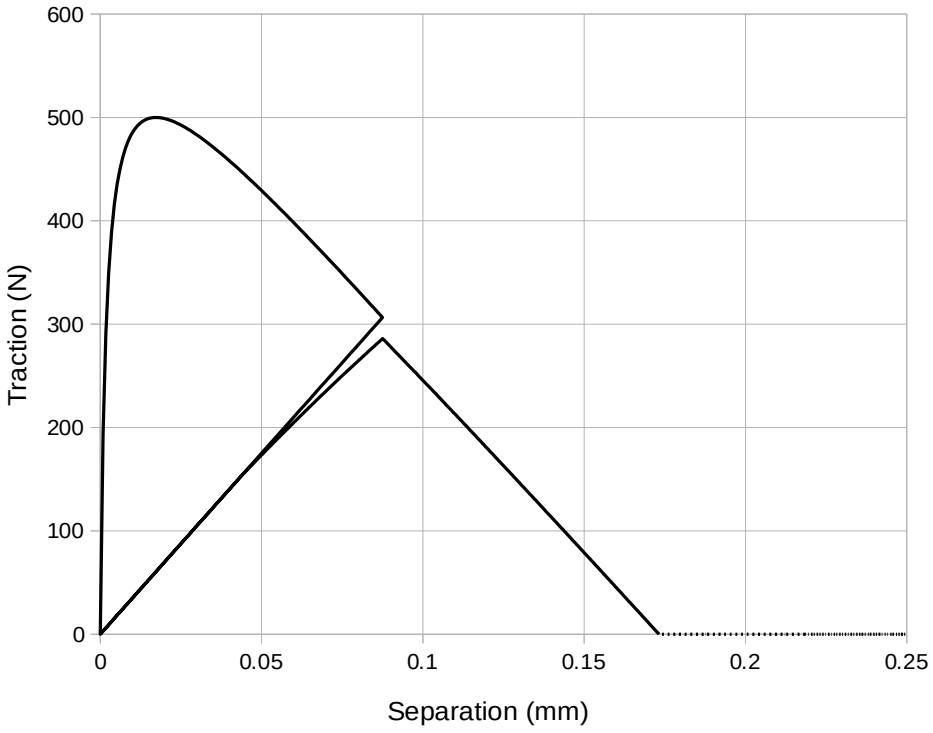

Figure 7. Traction – Separation relationship for one loading cycle after small monotonic deformation, the parameters of the model are $\sigma_{max} = 1000$, $\varphi_n = 100$, $\alpha = 2.0$, $\lambda_n = 0.1$.

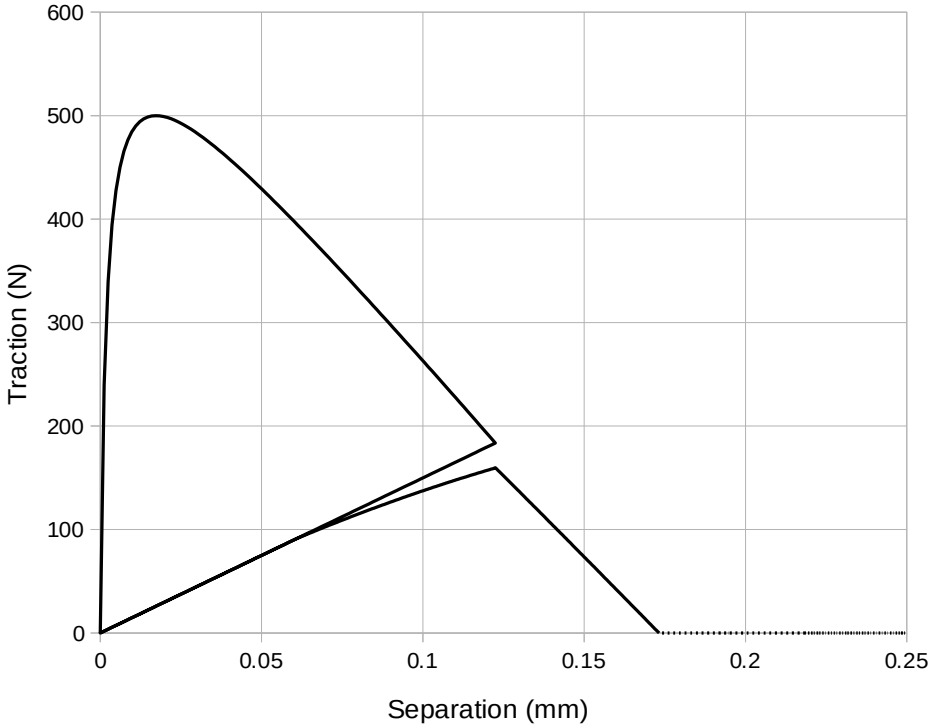

Figure 8. Traction – Separation relationship for one loading cycle after large monotonic deformation, the parameters of the model are $\sigma_{max}=1000$, $\varphi_n=100$, $\alpha=2.0$, $\lambda_n=0.1$.

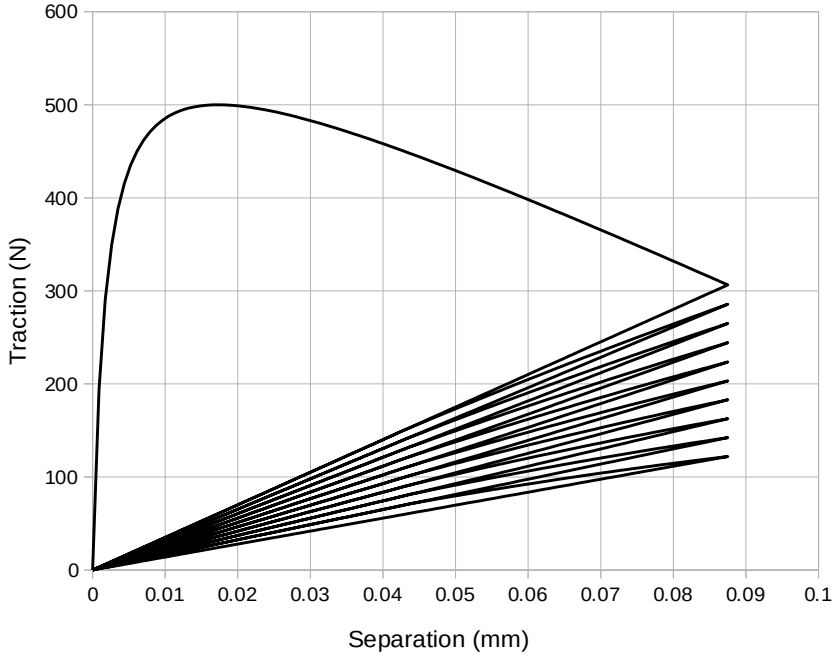

Figure 9. Traction – Separation relationship for many loading cycle under displacement control analysis.

The simple form of the rate of cyclic damage parameters $\dot{d}_{cn}$ and $\dot{d}_{ct}$ offers the capability to auto-calibrate, and auto-correct the parameters f_n and f_t during the analysis when the cohesive element is used together with the proposed algorithm. More-specific, the quantity

h of equation 55, which corresponds to the summation of all parameters that affect damage accumulation rate, is stored and at the end of the step the corrected parameters f_n and f_t are calculated based on equation 56, where N_f is the required number of cycles in order the element under consideration to fail as estimated by the methodology with the J-integral. It has to be mentioned that this methodology is not exact but it produces fair results in the case where only one cohesive element of the model accumulates significant cyclic damage, in that case the maximum cohesive element separation s_{max} remains approximately the same for every cycle.

$$h=\sum_{i=1}^{N} \dot{\Delta} \frac{s_{max}}{\sigma_{max}\ \delta} \tag{55}$$

$$f_{corrected} = \frac{1}{\sum_{j=1}^{N_f} h \cdot f_{initial}} \tag{56}$$

The advantage of the aforementioned methodology is that in every step of the analysis, except of the first one, the critical cohesive element starts to accumulate cyclic damage based on corrected and more representative values (not exact) of parameters f_n and f_t. Those values have been calculated based on the stress intensity factors of the previous step. Therefore parameters f_n and f_t are auto-corrected during the analysis and becoming larger as the crack length becomes larger which is the expected result. In Figure 10 the value $f_n = 2000$ has been used in an one-element test and the algorithm calculates the quantity $h = 2.21 \cdot 10^{-5}$ for one loading cycle. Moreover, the value 10 cycles has been enforced in the model, as the needed cycles in order the examined cohesive element to fail completely, and the algorithm calculates that the correct parameter f_n for the 10 cycles is equal to 4254. In Figure 11 the traction-separation response of the same model but with the corrected value $f_n = 4254$ is shown for 10 loading cycles. As it can be observed the element has been totally failed at the end of the 10^{th} cycle as expected.

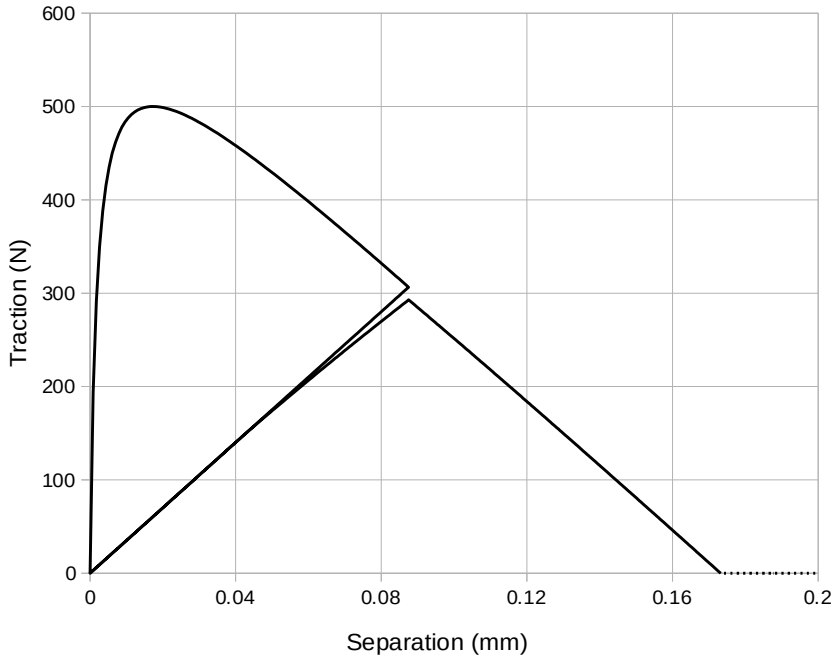

Figure 10. Traction – Separation relationship for one loading cycle and parameter $f = 2000$, the rest parameters of the model are $\sigma_{max} = 1000$, $\varphi_n = 100$, $\alpha = 2.0$, $\lambda_n = 0.1$.

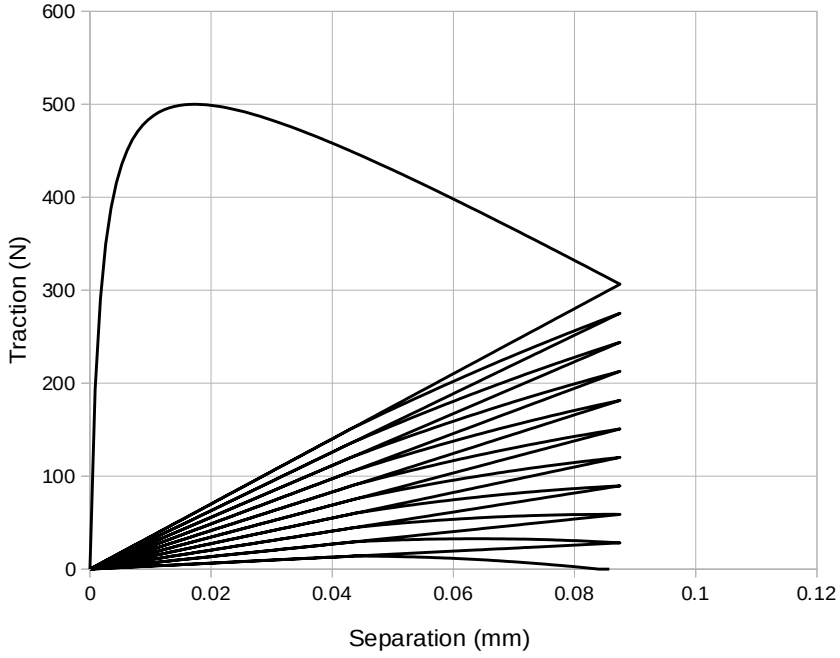

Figure 11. Traction – Separation relationship for 10 loading cycles and parameter $f = 4254$, the rest parameters of the model are $\sigma_{max} = 1000$, $\varphi_n = 100$, $\alpha = 2.0$, $\lambda_n = 0.1$.

5.4 Thermodynamic consistency of the model

The satisfaction or not of 2nd law of thermodynamics is usually referred as thermodynamic consistency of a model. In order a cohesive constitutive law to be thermodynamic consistent the damage of the model should be increasing monotonically, in other words the damage that has been induced in the material can not be undone. In general the cohesive models that have been derived from a potential are thermodynamic consistent, nevertheless there are cases where the unloading/loading relations of the cohesive law are not directly derived from potential functions which may lead to a violation of 2nd law of thermodynamics. The cohesive model used in this dissertation and proposed in [33] belongs to this category. Some corrections in the unloading/loading relations of the model have been proposed in [54] in order to be thermodynamic consistent. In the present section the thermodynamic consistency of the used model is analyzed and the monotonic increasing behavior of the proposed cyclic damage parameter is checked.

In the present study damage is defined as the reduction in the material stiffness using a scalar parameter $d \in [0,1]$, value 0 corresponds to no damage while value 1 corresponds to total failure. This is a common approach in damage mechanics ([38, 55, 56]). Assuming that potential energy can be decomposed into shear and normal components ([57, 58]) the general Helmholtz free energy function takes the form of equation 57,

$$\Psi = \sum_{i=1}^{n} (1-d_i)\Psi_i = (1-d_n)\Psi_n + (1-d_t)\Psi_t \qquad 57$$

where Ψ_n is the potential energy in the normal direction, while Ψ_t is potential energy in the shear direction. Scalar parameters d_n and d_t are the damage in the normal and the shear direction, respectively. Equation 58 is an alternative expression, proposed in [38] for anisotropic Helmholtz free energy which assumes that the damage parameters are multiplicatively coupled.

$$\Psi = \sum_{i=1}^{n} \prod_{j=1}^{n} (1-d_i{}^{j})\Psi_i \qquad 58$$

Equation 59 represents the dissipation inequality and it is defined as the difference of stress power $\mathring{w} = \boldsymbol{T}\,\boldsymbol{\Delta}$ and the time derivate of Helmholtz free energy, where $\boldsymbol{T}$ is the traction vector and $\boldsymbol{\Delta}$ is the vector of the differential displacements of the cohesive element surfaces.

$$D = \mathring{w} - \dot{\Psi} \geq 0 \qquad 59$$

Therefore using equation 58 and the definition of stress power equation 59 takes the following form.

$$D=\mathring{w}-\dot{\Psi}=\boldsymbol{T}\cdot\boldsymbol{\Delta}-(1-d_n)\frac{\partial\Psi_n}{\partial\boldsymbol{\Delta}}\boldsymbol{\Delta}+\Psi_n\ \dot{d}_n\ -(1-d_t)\frac{\partial\Psi_t}{\partial\boldsymbol{\Delta}}\boldsymbol{\Delta}+\Psi_t\ \dot{d}_t\geq 0 \quad (60)$$

Assuming that the normal and the shear traction can be expressed by equations 61 and 62, respectively the dissipation inequality can be expressed by equation 63. The components of the potential energy can be zero or positive therefore in order to equation 63 to be always true the time derivatives of damage parameters should be non-negative ($\dot{d}_n \geq 0$ and $\dot{d}_t \geq 0$), which actually means that any damage that have been induced to the material can not be reversed.

$$T_n=(1-d_n)\frac{\partial\Psi_n}{\partial\boldsymbol{\Delta}} \quad (61)$$

$$T_t=(1-d_t)\frac{\partial\Psi_t}{\partial\boldsymbol{\Delta}} \quad (62)$$

$$D=\Psi_n\ \dot{d}_n+\Psi_t\ \dot{d}_t\geq 0 \quad (63)$$

Based on the work done in [54], the cohesive model presented in [33] can be written in such a way that the potential function matches the form of the Helmholtz free energy. More specific, the components of the potential energy Ψ_n and Ψ_t can be written as follow

$$\Psi_n=\frac{1}{2}K_n\ \Delta_n\ ,\ \ \Psi_t=\frac{1}{2}K_t\ \Delta_t \quad (64)$$

where Δ_n and Δ_t are the differential displacements of the two cohesive surfaces in the normal and shear directions, while K_n and K_t are given by equations 65 and 66.

$$K_n=\frac{-\Gamma_n}{{\delta_n}^2}\left(\frac{m}{\alpha}\right)^{m-1}(m+\alpha)\left[\Gamma_t\left(\frac{n}{\beta}\right)^n\langle\varphi_t-\varphi_n\rangle\right] \quad (65)$$

$$K_t=\frac{-\Gamma_t}{{\delta_t}^2}\left(\frac{n}{\beta}\right)^{n-1}(n+\beta)\left[\Gamma_n\left(\frac{m}{\alpha}\right)^m\langle\varphi_n-\varphi_t\rangle\right] \quad (66)$$

Parameters Γ_n and Γ_t can be considered as energy constants and can be calculated according to equations 34 and 35. The critical differential displacements of cohesive surfaces at which the element stiffness becomes zero are given by the equations 38 and 39, while the non-dimensional exponents, m and n, can be computed by equations 36 and 37, respectively. The damage parameters at the normal and shear directions can be expressed by equations 67 and 68 and they are functions of separation history parameters κ_n and κ_t , parameters $\acute{T}_n$ and $\acute{T}_t$ can be calculated by equations 69 and 70.

$$d_n(\kappa_n,\kappa_t)=1-\frac{T'_n(\kappa_n,\kappa_t)}{K_n\ \kappa_n} \quad (67)$$

$$d_t\left(\kappa_n,\kappa_t\right)=1-\frac{T'_t\left(\kappa_n,\kappa_t\right)}{K_t\,\kappa_t} \tag{68}$$

$$\begin{aligned}\acute{T}_n\left(\kappa_n,\kappa_t\right)=&\frac{\Gamma_n}{\delta_n}\left[m\left(1-\frac{\kappa_n}{\delta_n}\right)^{\alpha}\left(\frac{m}{\alpha}+\frac{\kappa_n}{\delta_n}\right)^{m-1}-\alpha\left(1-\frac{\kappa_n}{\delta_n}\right)^{\alpha-1}\left(\frac{m}{\alpha}+\frac{\kappa_n}{\delta_n}\right)^{m}\right]\\&\times\left[\Gamma_t\left(1-\frac{\kappa_t}{\delta_t}\right)^{\beta}\left(\frac{n}{\beta}+\frac{\kappa_t}{\delta_t}\right)^{n}+\langle\varphi_t-\varphi_n\rangle\right]\end{aligned} \tag{69}$$

$$\begin{aligned}\acute{T}_t\left(\kappa_n,\kappa_t\right)=&\frac{\Gamma_t}{\delta_t}\left[n\left(1-\frac{\kappa_t}{\delta_t}\right)^{\beta}\left(\frac{n}{\beta}+\frac{\kappa_t}{\delta_t}\right)^{n-1}-\beta\left(1-\frac{\kappa_t}{\delta_t}\right)^{\beta-1}\left(\frac{n}{\beta}+\frac{\kappa_t}{\delta_t}\right)^{n}\right]\\&\times\left[\Gamma_n\left(1-\frac{\kappa_n}{\delta_n}\right)^{\alpha}\left(\frac{m}{\alpha}+\frac{\kappa_n}{\delta_n}\right)^{m}+\langle\varphi_n-\varphi_t\rangle\right]\end{aligned} \tag{70}$$

Therefore the normal and shear tractions can be expressed by equations 71 and 72. The numerical scheme from increment i to next increment $i+1$ of equations 73 and 74 can ensure that the separation history parameters κ_n and κ_t will increase monotonically [54]. Based in equations 67, 68, 69, 70 is it obvious that also damage parameters d_n and d_t will increase monotonically due to the fact that they are functions of separation history parameters κ_n and κ_t . Therefore the used cohesive element is thermodynamic consistent.

$$T_n=\acute{T}_n\left(\kappa_n,\kappa_t\right)\frac{\Delta_n}{\kappa_n} \tag{71}$$

$$T_t=\acute{T}_t\left(\kappa_n,\kappa_t\right)\frac{\Delta_t}{\kappa_t} \tag{72}$$

$$\kappa_n{}^{i+1}=max\left(\kappa_n{}^{i},\Delta_n{}^{i+1}\right) \tag{73}$$

$$\kappa_t{}^{i+1}=max\left(\kappa_t{}^{i},|\Delta_t{}^{i+1}|\right) \tag{74}$$

The proposed damage parameter for the cycling loading should increase also monotonically. Therefore the rate of the parameters for cyclic damage accumulation should be positive. Based on equations 52, 53 and the if condition of equation 54 it is obvious that $\dot{d}_{cn}\geq 0$ and $\dot{d}_{ct}\geq 0$ for every possible case, since all the variables of equations 52 and 53 are positive.

5.5 J-Integral Calculation

The 1st Law of Thermodynamics can be expressed in a rate form in equation 75. The terms on the left side of the equation denote the energy input into an object as mechanical work and heat. The terms on the right side represent the various forms of energy within the object that the external energy can be converted into. It is important to be mentioned that kinetic energy, strain energy, and thermal energy are reversible. However, the energy used in crack propagation is not reversible. For quasi-static mechanical loading, equation 75 simplifies to equation 76, which shows that the work energy is divided into internal strain energy, and the breaking of atomic bonds. The energy release associated with bond breaking and crack growth is related to J-Integral.

$$\frac{dW_{external}}{dt}+\frac{dQ_{heat}}{dt}=\frac{dK_{velocity}}{dt}+\frac{dW_{strain}}{dt}+\frac{dU_{thermal}}{dt}+\frac{dD_{crack}}{dt} \quad 75$$

$$\frac{dW_{external}}{dt}=\frac{dW_{strain}}{dt}+\frac{dD_{crack}}{dt} \quad 76$$

Applying the chain rule to equation 76 with respect to crack length a, one can result to equation 77.

$$\frac{dW_{external}}{da}\frac{da}{dt}=\frac{dW_{strain}}{da}\frac{da}{dt}+\frac{dD_{crack}}{da}\frac{da}{dt} \quad 77$$

Therefore, by factoring out the term $\frac{da}{dt}$ equation 77 becomes

$$\frac{dW_{external}}{da}=\frac{dW_{strain}}{da}+\frac{dD_{crack}}{da} \quad 78$$

The equation 78 expresses the conservation of energy per unit of crack growth. Dividing also all terms by the material thickness B and by rearranging the terms of 78 one can obtained the definition of J.

$$J=\frac{dD_{crack}}{Bda}=\frac{dW_{external}}{Bda}-\frac{dW_{strain}}{Bda} \quad 79$$

where Bda is the increment of area created by the crack growth, da.

J-Integral is one of the most fundamental quantities in linear and nonlinear fracture mechanics, and it has been introduced in [60]. The J integral represents the amount of energy released per unit area of crack surface increase and is presented in equation 80.

$$J=-\int_{\Gamma}[\sigma_{i,j}\ \frac{du_i}{dx_1}-W\delta_{1,j}\ ]n_j\ ds \quad 80$$

The numerical calculation of J-Integral has been performed based on the methodology proposed in [61]. According to this methodology the line integral of equation 80 is transformed to an area integral represented by equation 81. This transformation is quite convenience for the calculation of J-Integral in finite element models since it is possible to choose a path of elements that encloses the crack tip, this path is actually an area around the crack. Using the J-integral value the stress intensity factor can be easily calculated from equation 82, in the case of linear elastic fracture mechanics.

$$J=\int_A [\sigma_{i,j} \frac{du_i}{dx_1} - W\delta_{1,j}] \frac{dq_1}{dx_j} dA \quad (81)$$

$$K=\sqrt{\frac{E\cdot J}{1-\nu^2}} \quad (82)$$

The finite element formulation for the calculation of J-integral based on equation 81 is described below. In equations 83 and 84 the physical coordinates and the displacements are given with respect to nodal coordinates X_{iK} and nodal displacements U_{iK} , while in equation 85 the parameter q_i , which helps us to transform the line integral of equation 80 in the area integral of equation 81 is presented.

$$x_i = \sum_{K=1} N_K \; X_{iK} \quad (83)$$

$$u_i = \sum_{K=1} N_K \; U_{iK} \quad (84)$$

$$q_i = \sum_{K=1} N_K \; Q_{iK} \quad (85)$$

N_K is the shape functions of the elements, while Q_{iK} is the value of parameter q_i at nodal K, and takes the value 0 at the nodes of the external boundary of the area A and the value 1 at the nodes of the internal boundary of the area. Finally, the J-integral is given by the double sum of equation 86 where the derivative of q_1 with respect to x_i is given by equation 87 and w_p is the weight factor of integration points.

$$J=\sum_{elelemnts\in path} \sum_{p=1} [(\sigma_{i,j} \frac{du_i}{dx_1} - W\delta_{1,j}) \frac{dq_1}{dx_j} det(\frac{dx_K}{d\eta_K})]_p \; w_p \quad (86)$$

$$\frac{dq_1}{dx_i}=\sum_{i=1}\sum_{J=1} \frac{dN_i}{d\eta_J}\frac{d\eta_J}{dx_j} Q_{1i} \quad (87)$$

The developed algorithm finds the elements paths that defines the area A automatically based on the predicted crack tip, as shown in Figure 12. More specific the algorithm determines the first path as the path that includes only the elements that have the node

which is consider as the crack tip, then it finds the next path which includes all the elements that have common nodes with elements of the previous path. More specific, the algorithm begins by initializing matrices that will store the identified paths. Subsequently the code searches for the element associated with the crack tip node and it loops through all nodes in order to identify the elements connected to the node corresponding to the crack tip. Once found, the connected elements are stored as the first path. After the first path is identified, the subroutine proceeds to find additional paths. For each element in the previous path, it retrieves its nodes and then finds candidate elements connected to those retrieved nodes. Candidate elements are added to paths only if they are not already part of any existing path. This is done by comparing each candidate element with all elements in previously identified paths. The process repeats for all requested paths, with each path being built from elements connected to the previous one. This allows the subroutine to trace the propagation of connections through the structure.

The developed algorithm calculates the values of parameter q_i (equation 85), based on the following procedure. For each element of the examined path, the algorithm identifies the two neighboring elements, which is important for evaluating how the examined element is positioned relative to its neighbors. Subsequently the geometric centers of the examined element and its neighbors are computed using the coordinates of their nodes. For each of the four nodes of the current element, the subroutine computes two values, based on the relative positions of the element center and its neighboring elements. This is done using a form of cross-product between vectors formed by the element center and its neighbors. If both values are positive, the corresponding q_i is set to 1. Otherwise, it is set to 0.

It is known that the J -integral calculation is not representative for elements paths that are very close to the crack tip. Therefore, the calculation of J-integral is performed for several elements paths and the stable value is kept as final result. In Figure 13, a comparison among the J-integral values of the developed algorithm and the corresponding calculation of Abaqus is offered.

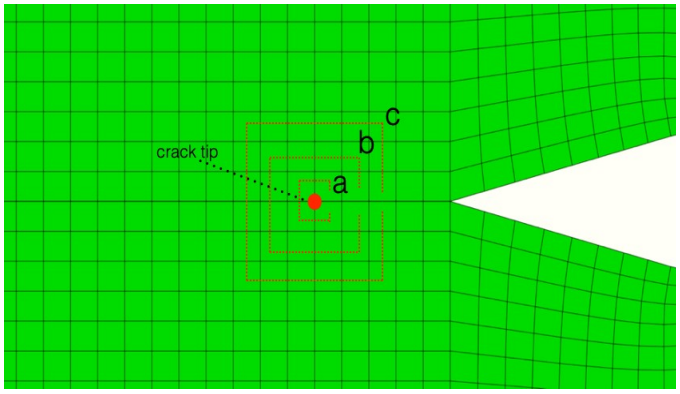

Figure 12. Different element paths (red dot lines) identify automatically by the algorithm around the crack tip.

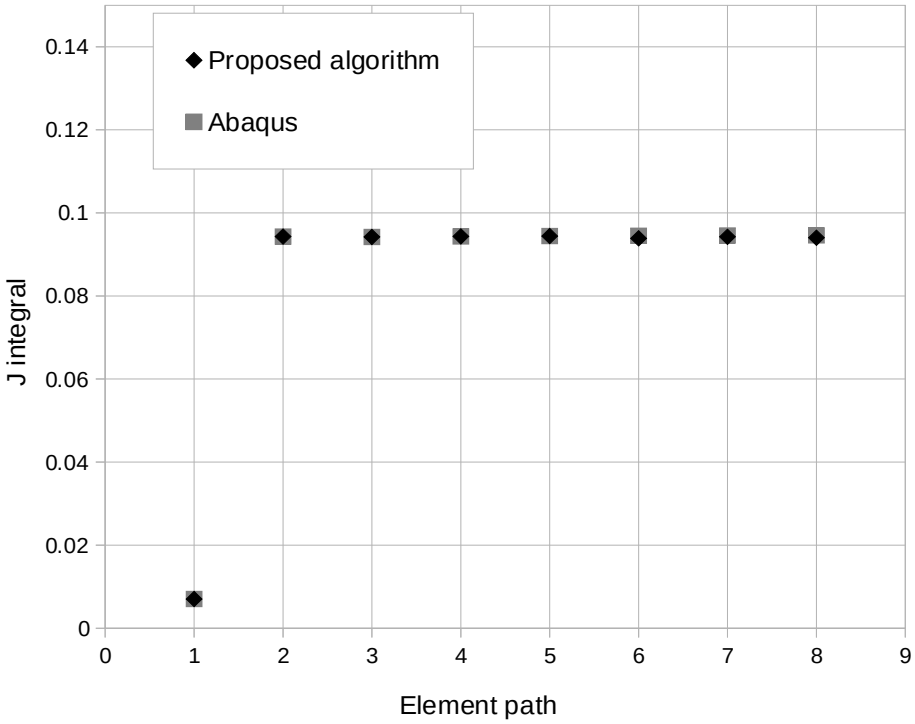

Figure 13: J-Integral calculation for different paths, comparison of proposed algorithm against Abaqus.

6 Experimental Results

Experimental testing has been conducted in the framework of the present research effort in order to be used as verification of the developed numerical model. Two different specimens geometries have been used, the crack growth rate test specimens (compact) and the single edge notch bend specimens (SENB). The experiments took place in the Laboratory of Mechanics and Strength of Materials at the department of Mechanical Engineering of University of Thessaly. A mild steel grade S355 was used for the experiments, and the basic mechanical properties as far as the chemical decomposition can be found in Tables 1 and 2, respectively. A uni-axial tensile test (Figure 14) was performed in order to obtain the stress-strain response of the material under consideration, which is shown Figure 15.

Table 1. Material properties of steel under consideration.

Steel material	Nominal yield strength (MPa)	Rt0.2 (MPa)	UTS (MPa)
Mild (S355)	355	404	486

Table 2 Chemical composition of steel under consideration (%).

Steel Grade	C	Si	Mn	P	S	Cu	Cr	Mo	Al	V	Ti	Nb	Ceq
S355	0.10	0.21	1.22	0.02	0.01	0.22	0.18	0.04	0.03	0.01	0.02	0.03	0.373

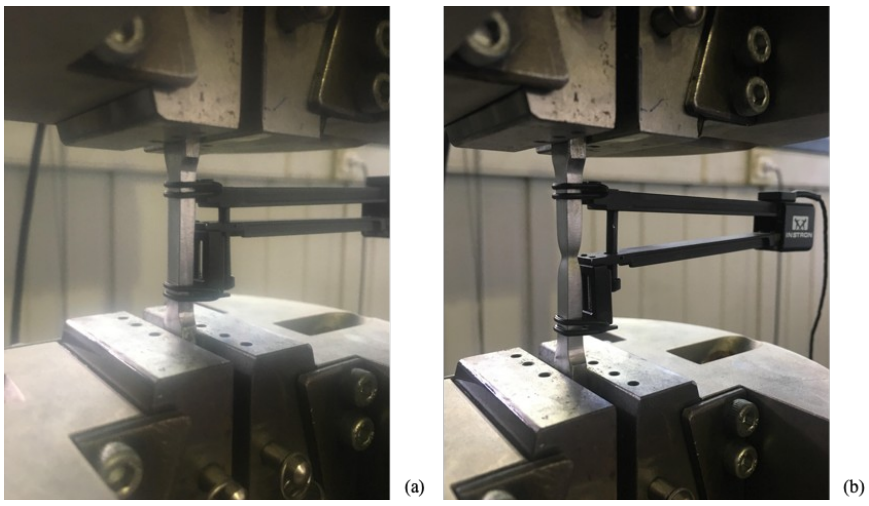

Figure 14. Experimental setup of Stress-Strain curve test of S355 steel grade, (a) undeformed configuration, (b) deformed configuration.

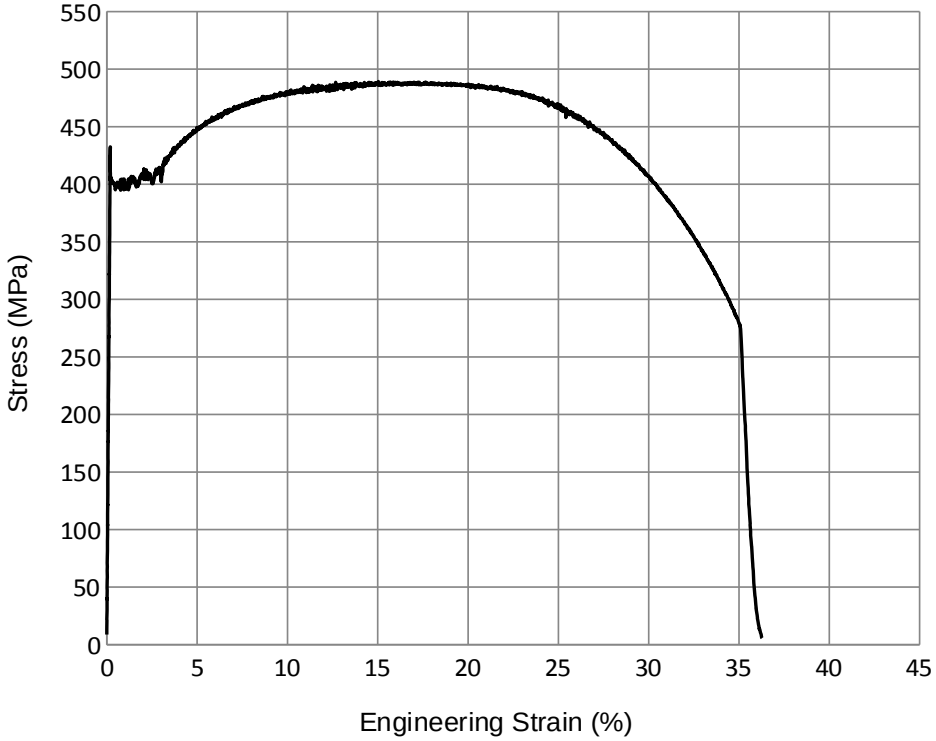

Figure 15. Experimental Stress-Strain curve of S355 steel grade.

The compact specimens shown in Figure 16 were used for the measurement of crack growth rate of the material under investigation. The specimen dimensions were chosen according to [62, 63] (Figure 16) and the experimental setup is presented in Figure 17. The crack mouth opening was measured with an Instron clip gage (Figure 17a), while the crack length was monitored using a digital camera and a macro lens as shown in Figure 17b. Three different stress ratios were examined R=0.1, R=0.5 and R=0.7, while the crack length, the crack growth rate, and the stress intensity factors were calculated in all cases according to [62, 63], based on the measurements of crack mouth opening and applied forces, which can be found at Table 3. A 2 mm crack was induced to all three specimens prior the regular test by a pre-cracking cycling loading in order to eliminate possible effects of the manufacturing process of the specimens.

Table 3. Applied forces at the crack growth test specimens.

Stress Ratio	Maximum Force (kN)	Minimum Force (kN)
0.1	11.0	1.10
0.5	18.0	9.00
0.7	29.1	20.37

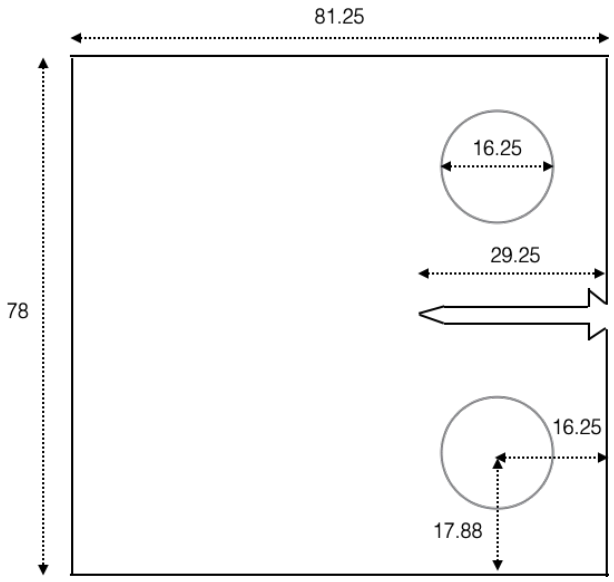

Figure 16. Dimensions in mm of crack growth rate test specimen, according to [62, 63] (thickness was equal to 15 mm).

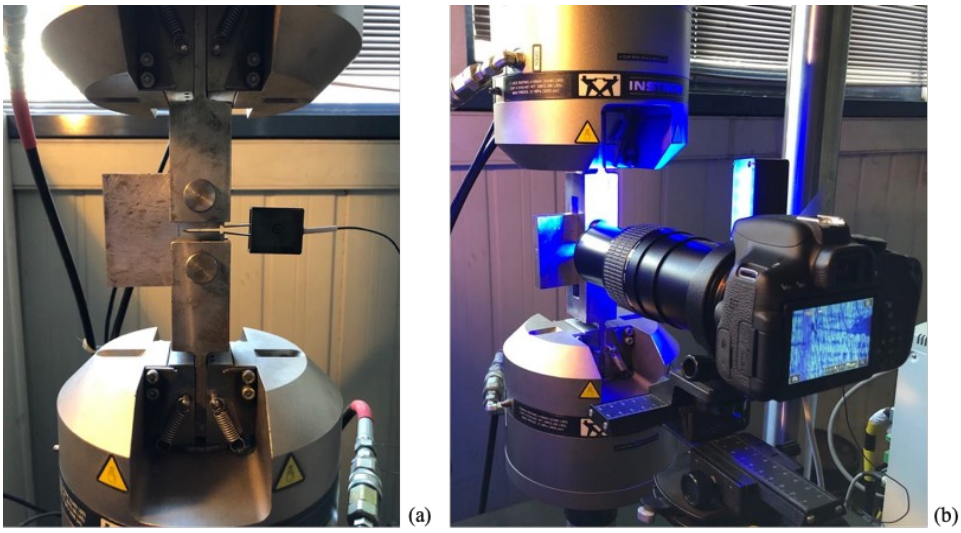

Figure 17. (a) Experimental setup for crack growth rate tests. (b) Monitoring of crack propagation.

All specimens have been extracted by the same plate, but from different directions. More specific, the specimens used for tests with $R = 0.1$ and $R = 0.7$, have been extracted from the same direction of the plate, while the specimen used in test with $R = 0.5$ has been extracted from the perpendicular direction with respect to other two specimens. In Figures 18 and 19 the crack growth rates with respect to stress intensity factor ranges, ΔK, are presented for all test cases. As shown in Figure 18 the crack growth rates of test with $R = 0.7$ are larger than $R = 0.1$ for the same stress intensity factor ranges, ΔK, which is the expected result. The data from Figures 18 and 19 will be used as input in the proposed

numerical model, while the results, which are presented in Figures 20 and 21, will be used in order to verify that the developed algorithm can predict accurately the crack propagation. More specific, in those Figures the crack length, as calculated according to [62, 63] with respect to loading cycles is presented for all examined cases.

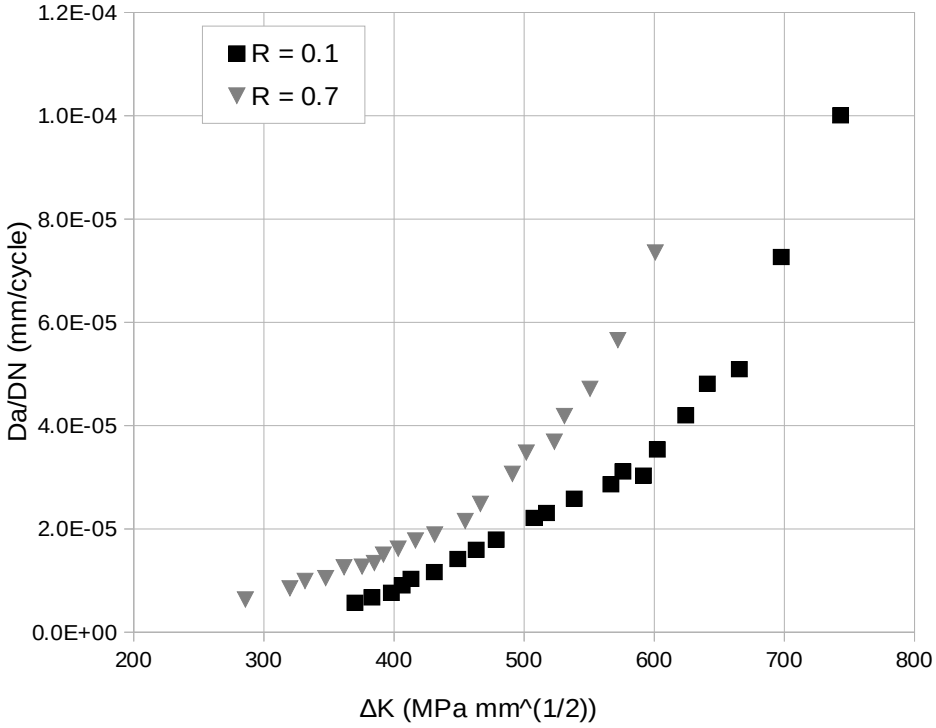

Figure 18. Experimental results of crack growth rate with respect to stress intensity factor amplitude for $R = 0.1$ and $R = 0.7$.

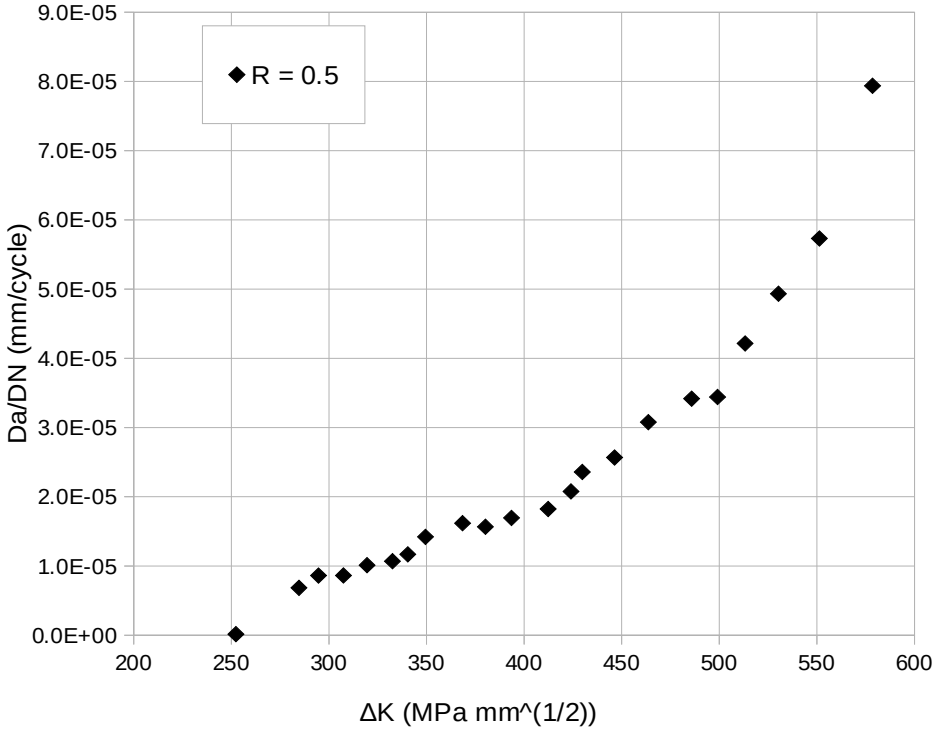

Figure 19. Experimental results of crack growth rate with respect to stress intensity factor amplitude for $R = 0.5$.

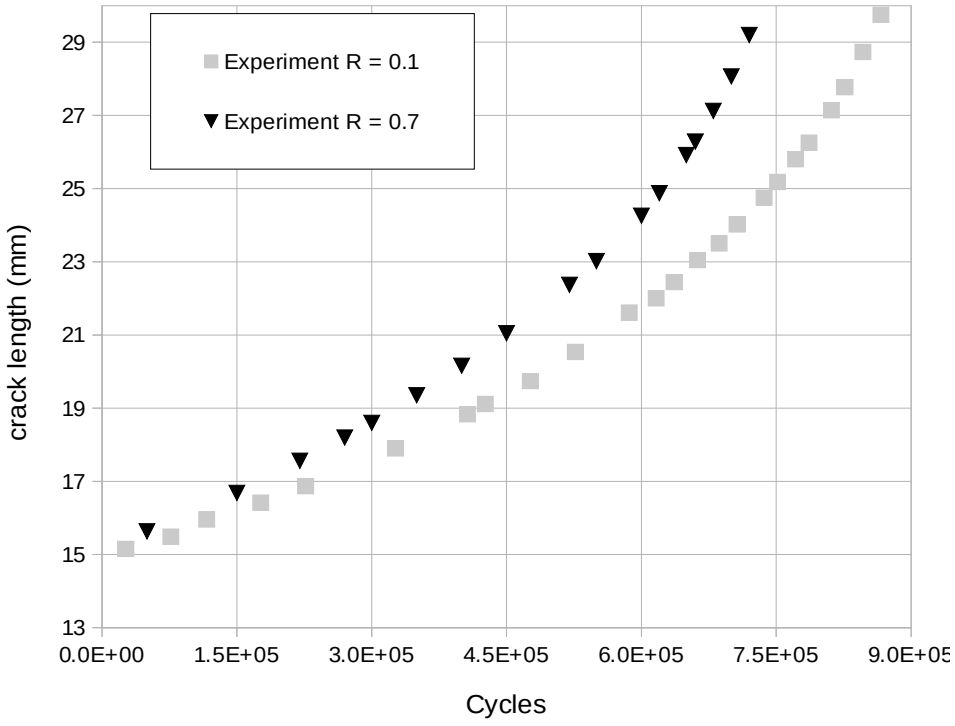

Figure 20: Experimental crack length with respect to loading cycles for $R = 0.1$ and $R = 0.7$.

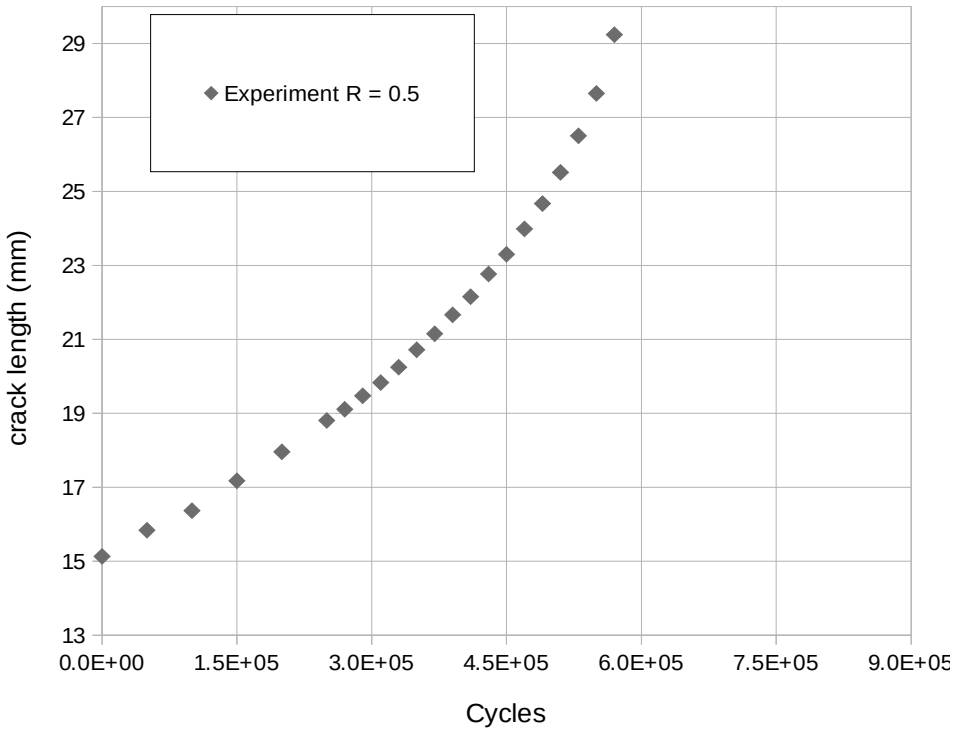

Figure 21: Experimental crack length with respect to loading cycles for $R = 0.5$

The SENB specimens were tested under 3-point bending to determine the CTOD critical displacement. The nominal cross-sectional dimensions of the SENB specimens were 13 mm (width) and 26 mm (height), with a span of 109 mm (Figure 22a). The tests were conducted at room temperature (RT) in an MTS 318.10 loading frame, which employs an MTS FlexTest SE digital controller in accordance to [65]. The mouth opening was measured using video-extensometer, which employs digital image correlation technics, as shown in Figure 22b. The load versus the crack mouth opening is presented in Figure 23, and these results will be also used in order to verify that the selected cohesive model is capable to describe accurately the crack propagation under monotonic loading of the material under consideration.

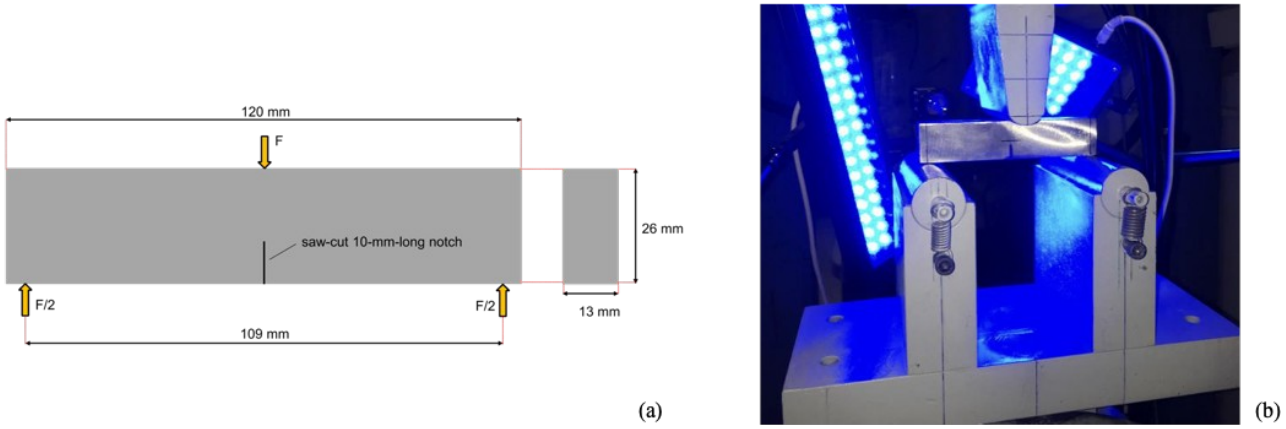

Figure 22. Schematic representation of CTOD specimen (a), and experimental CTOD test setup (b).

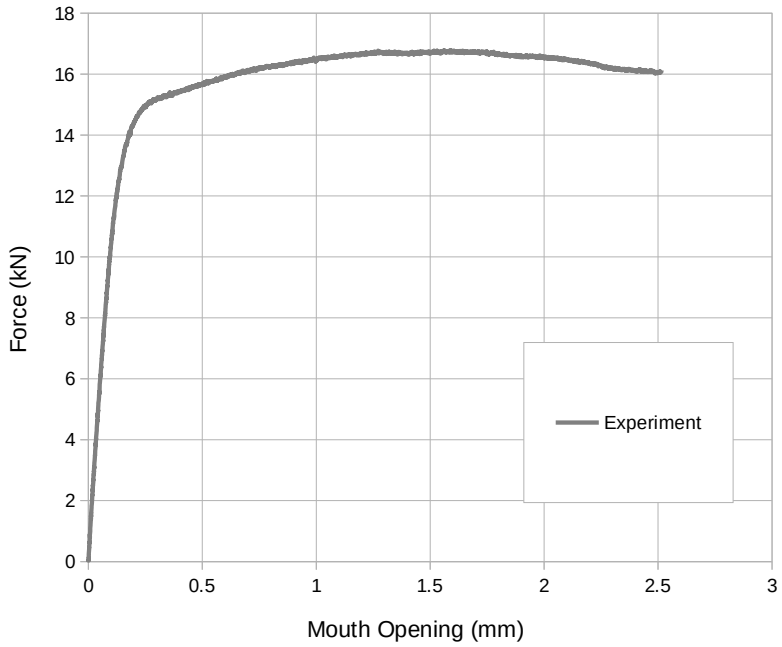

Figure 23: Experimental results of the CTOD test.

7 Simulation of Experiments

7.1 Compact specimen tests

The experimental results, presented in the previous section were used in order to verify the proposed methodology. The developed model for the simulation of crack growth rate tests is shown in Figure 24, and it consists by 4-node plane strain elements with length equal to 0.5 mm, while rigid surfaces and contact algorithm were used for the simulation of pins. The analysis was force control with two loading cycles in every step. A spring element with very small stiffness was used in order to alleviate numerical problems until the rigid surfaces come to contact with the specimen, as shown in Figure 24a. The experimental results of Figures 18 and 19 were used as input in the numerical model.

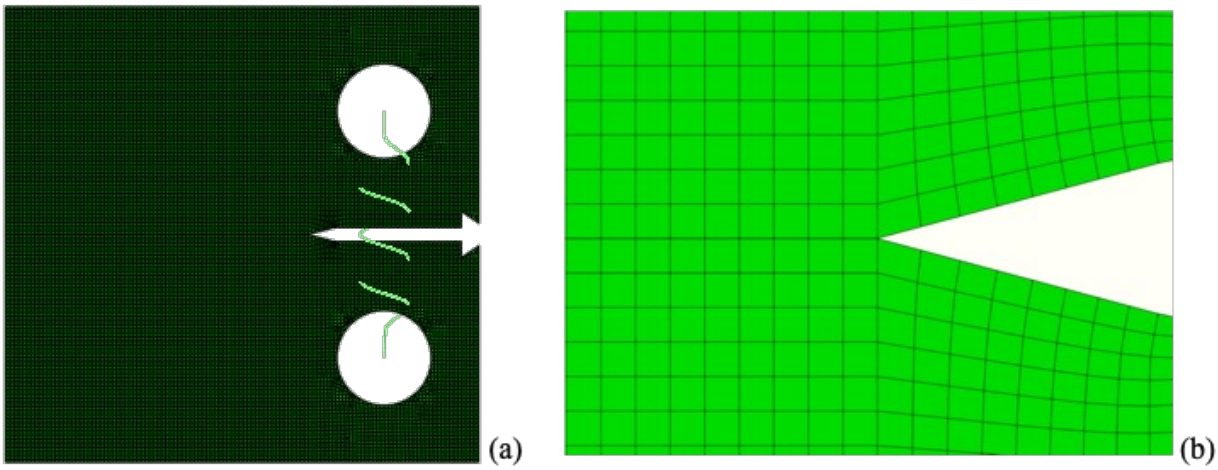

Figure 24. (a) Numerical model for the simulation of crack growth tests. (b) Mesh at the area of crack notch.

In Figures 25, 26, and 27 the crack length with respect the loading cycles is shown for the proposed methodology and the experimental results for loading ratio R equal to 0.1, 0.5 and 0.7, respectively. The comparison among the developed methodology and the experimental results is consider excellent for all examined loading ratios. It has to be mentioned though that the simulation of the experiments concerns the crack propagation after the pre-crack loading cycles, which for the specific specimens corresponds to an initial crack of 2 mm length. In Figures 28, 29, and 30 the crack growth rate da/dN with respect to ΔK is presented for all examined loading ratio R. The fact that the numerical ΔK are very close to the experimental estimations of stress intensity factors is a strong indicator that the methodology used for calculation of J-integral produces correct results. In the Figures 31, 32, and 33 a more macroscopic comparison of experimental and numerical results is presented. More specific, the increase of crack mouth opening is depicted with

respect to loading cycles. The comparison is quite good for all examined loading ratios, which indicates that the proposed approach for numerical simulation of crack propagation is able to capture also the changes in the specimen stiffness due to development of cracks.

In Figures 34, 35 and 36 the crack propagation is depicted for the three examined cases for several stages of fatigue life. Moreover, the stress concentration behind the crack tip is illustrated in those figures. In all cases the maximum stress increases as the crack length increases as expected. It should be noted that a scale factor has been applied to the displacements in order the crack opening to be more visible.

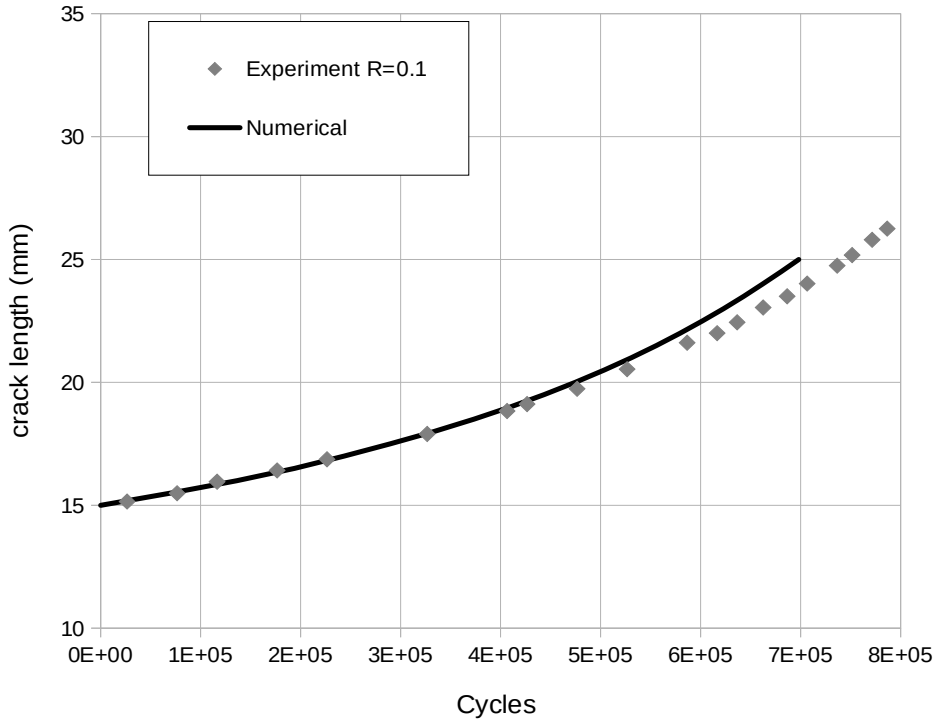

Figure 25: Crack length with respect to loading cycles, comparison of numerical prediction against experimental results of crack growth test with R=0.1.

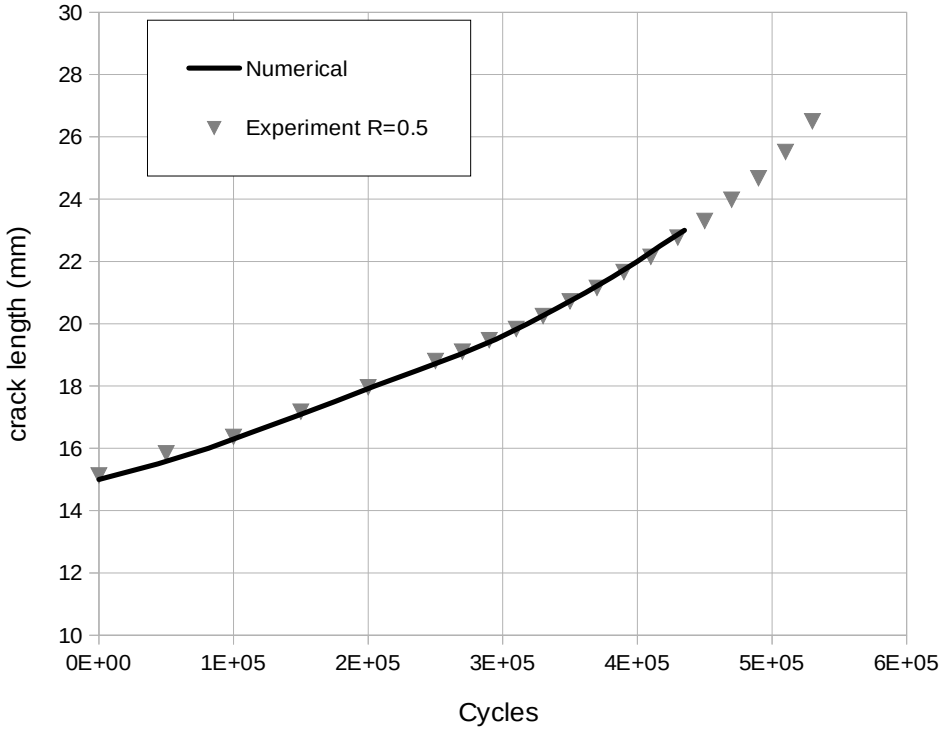

Figure 26: Crack length with respect to loading cycles, comparison of numerical prediction against experimental results of crack growth test with R=0.5.

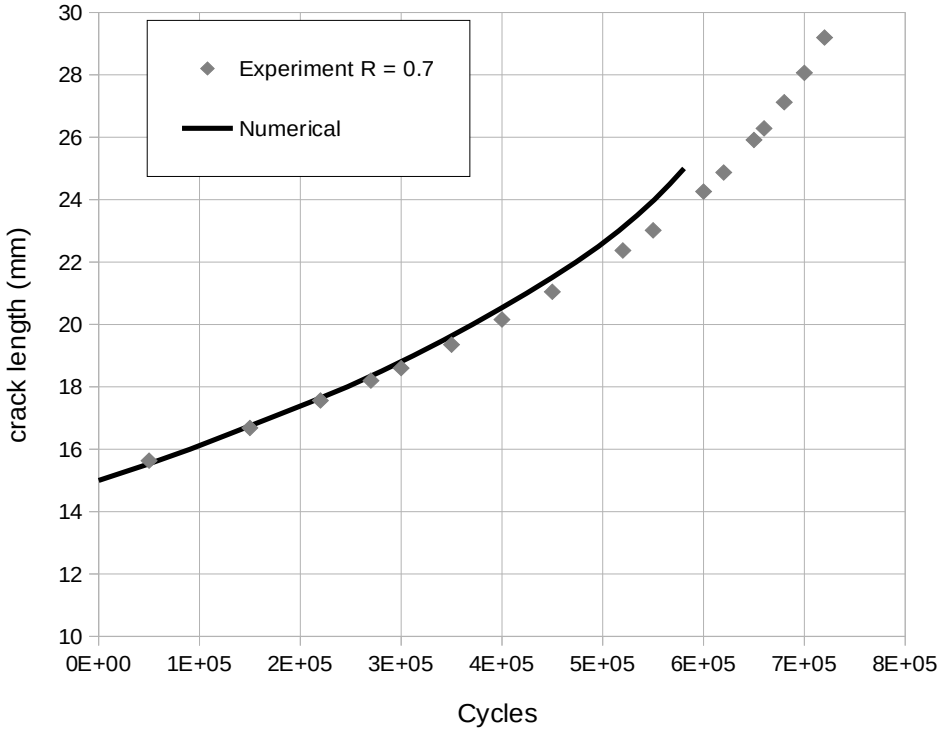

Figure 27: Crack length with respect to loading cycles, comparison of numerical prediction against experimental results of crack growth test with R=0.7.

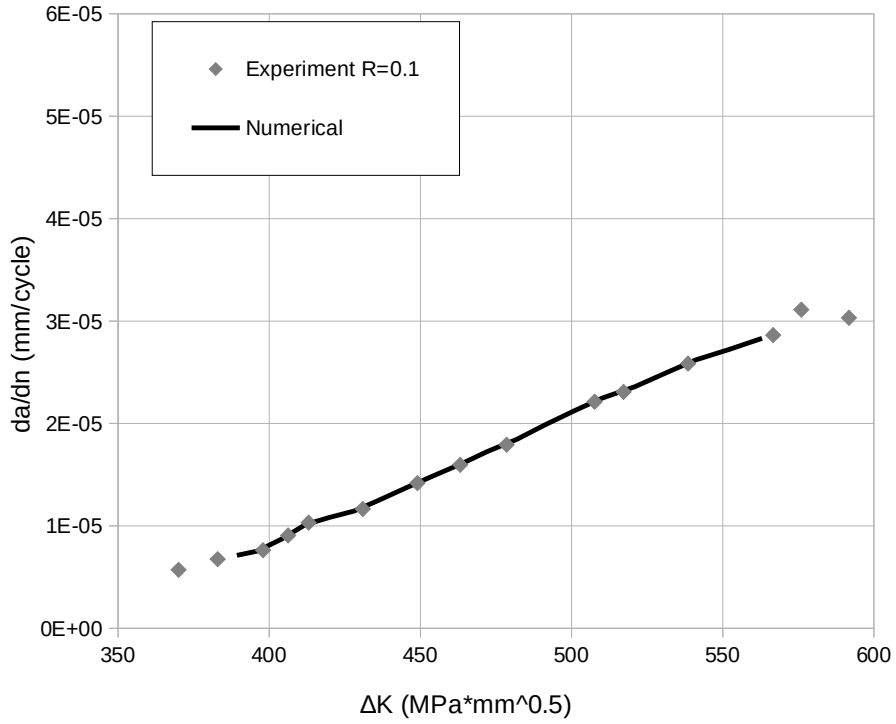

Figure 28: da/dN with respect to ΔK, comparison of numerical prediction against experimental results of crack growth test with R=0.1.

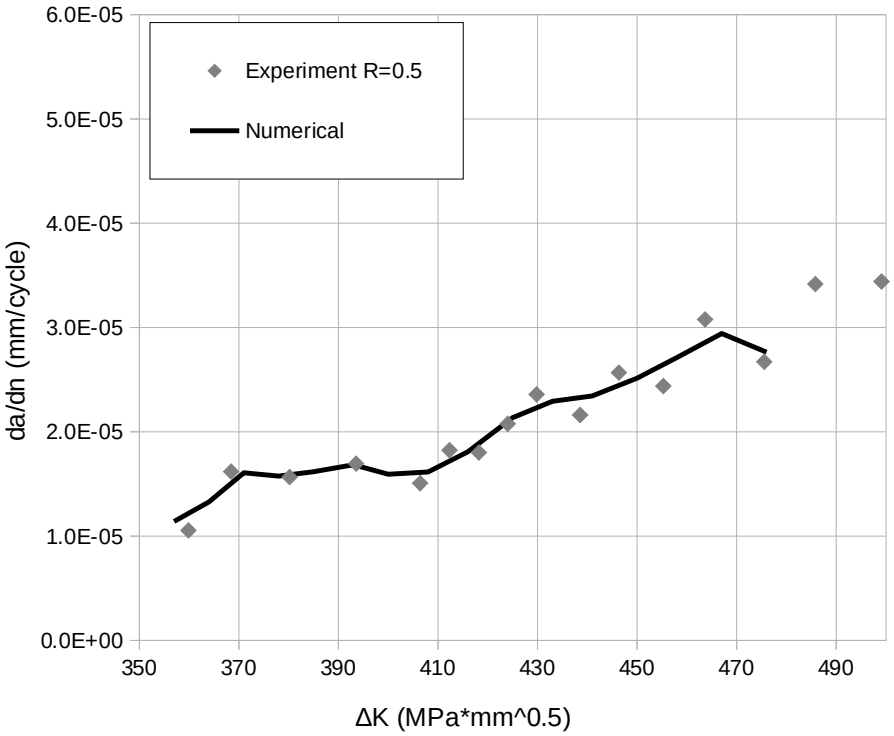

Figure 29: da/dN with respect to ΔK, comparison of numerical prediction against experimental results of crack growth test with R=0.5.

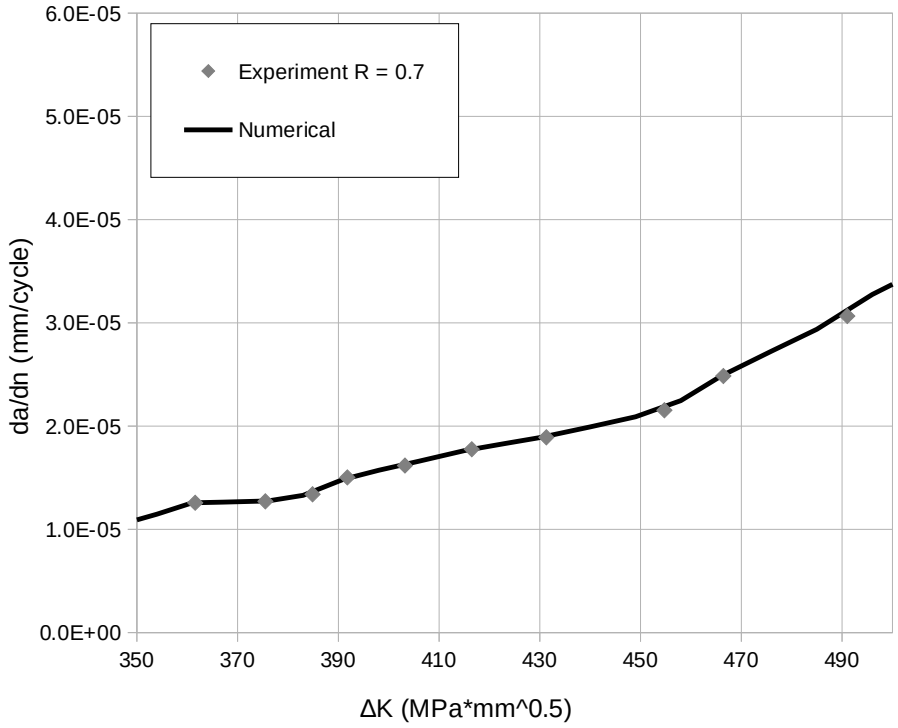

Figure 30: da/dN with respect to ΔK, comparison of numerical prediction against experimental results of crack growth test with R=0.7.

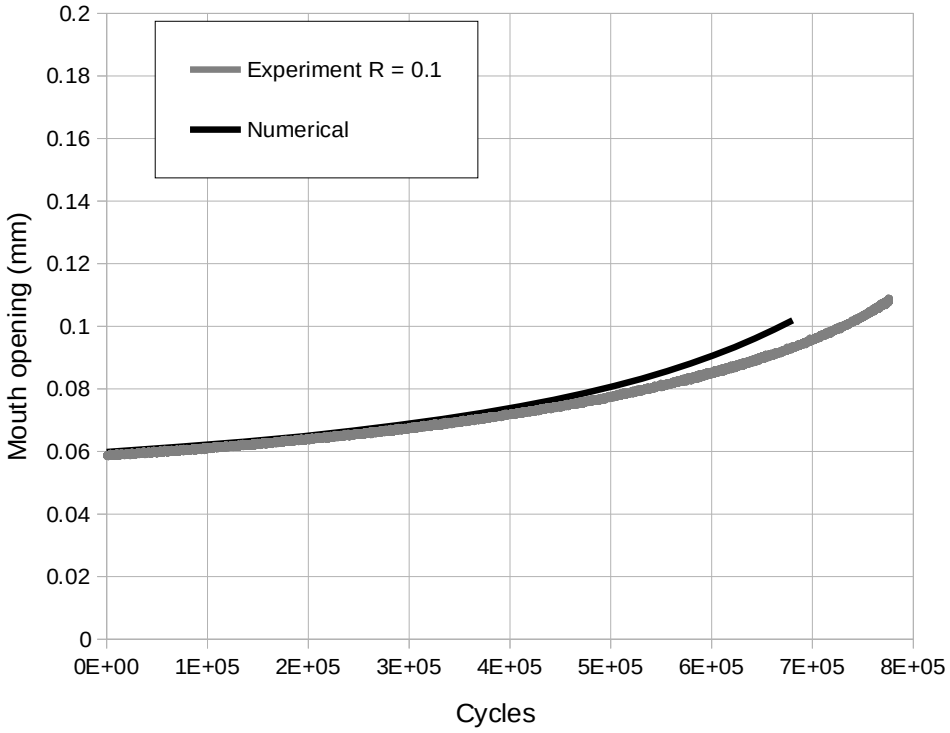

Figure 31: Mouth opening with respect to loading cycles, comparison of numerical prediction against experimental results of crack growth test with R=0.1.

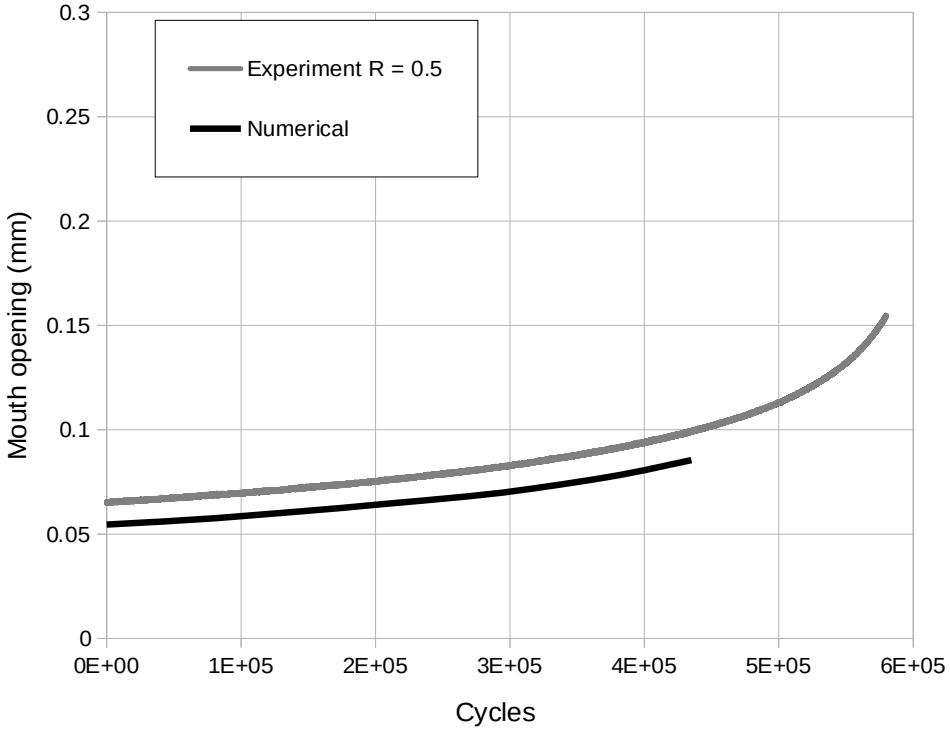

Figure 32: Mouth opening with respect to loading cycles, comparison of numerical prediction against experimental results of crack growth test with R=0.5.

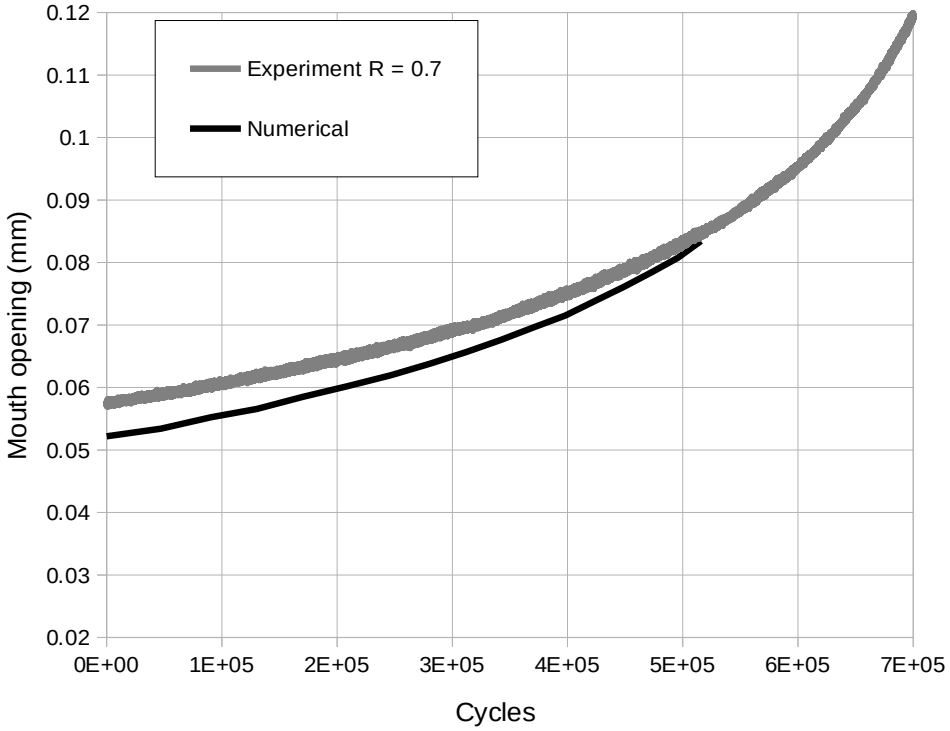

Figure 33: Mouth opening with respect to loading cycles, comparison of numerical prediction against experimental results of crack growth test with R=0.7.

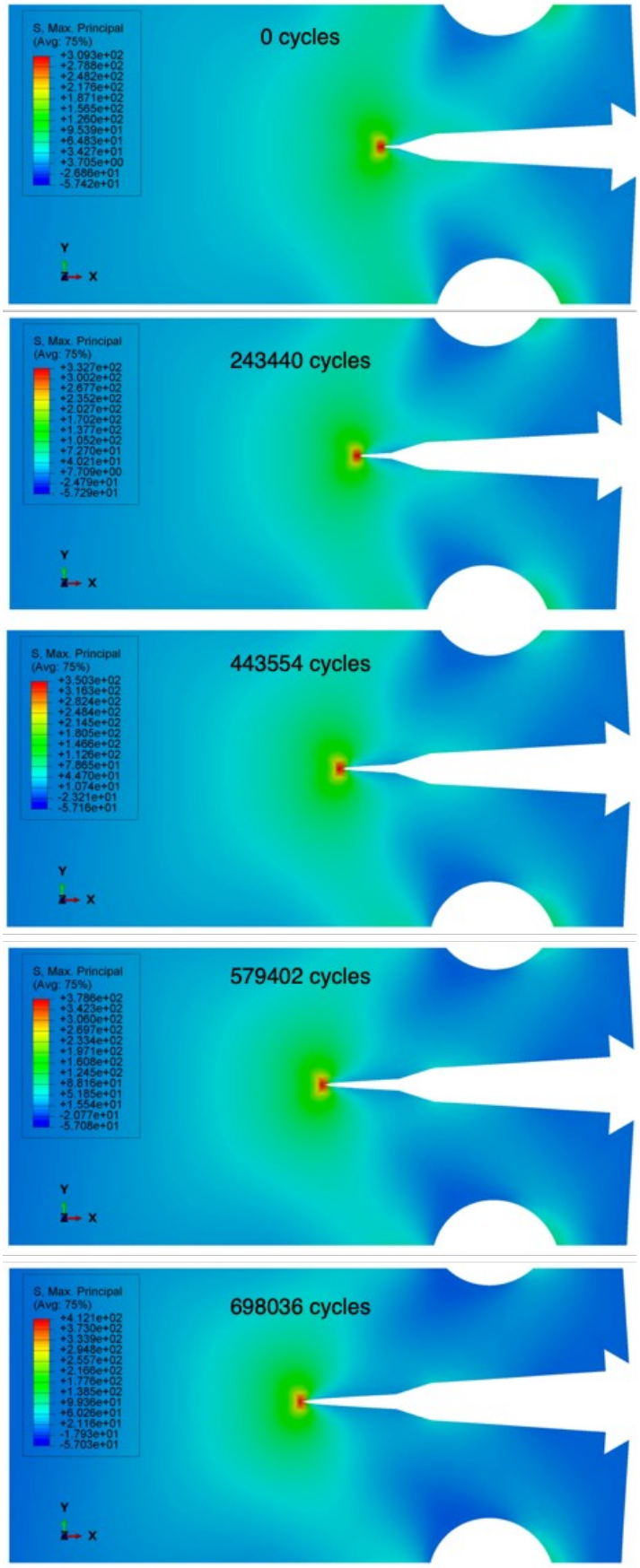

Figure 34: Stress distribution (max principal) and crack propagation for test with R=0.1, (scale factor has been applied to displacements).

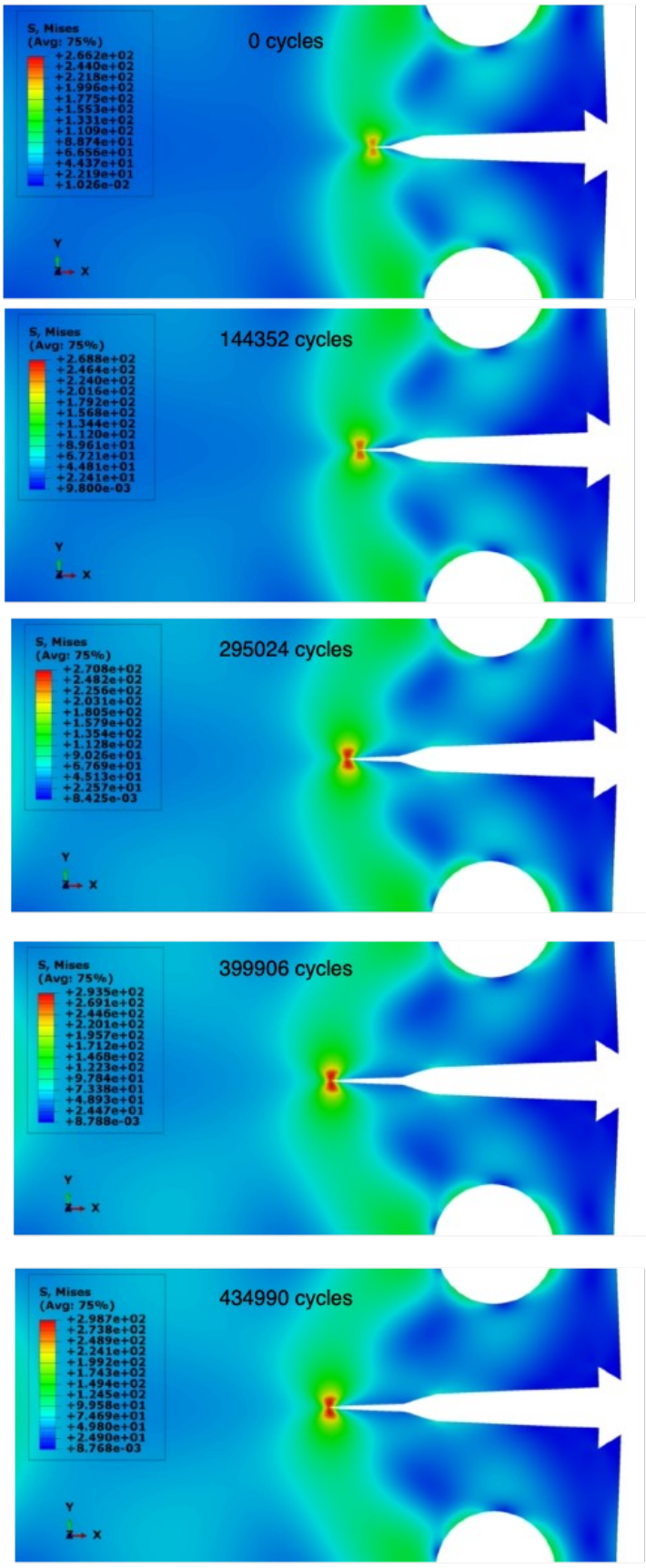

Figure 35: Stress distribution (von Mises) and crack propagation for test with R=05, (scale factor has been applied to displacements).

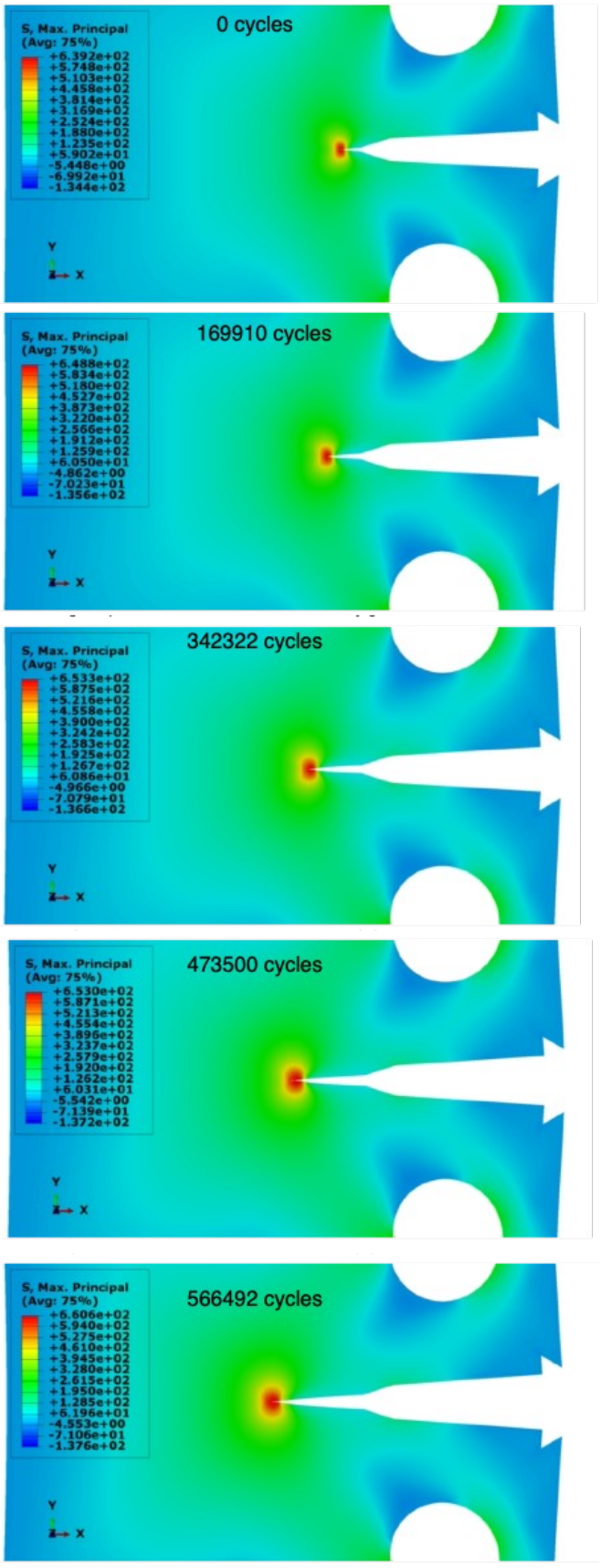

Figure 36: Stress distribution (max principal) and crack propagation for test with R=0.7, (scale factor has been applied to displacements).

An important feature of the proposed methodology is the cyclic skipping, which consist one of the major problems that arrises when cohesive elements are used in high cyclic fatigue problems. As mentioned previously, in the developed methodology cyclic skipping is an automated process. The algorithm, at the end of each step, calculates the cycles needed in order the critical cohesive element to fail and sets the damage parameter of the element equal to 1.0. This actually means that the crack increases one element length at the end of every step and the cycles needed for this increase are returned as an output from the algorithm. In the aforementioned numerical models 20 steps have been defined with 2 loading cycles in every step, therefore 20 cohesive elements are expected to fail, one in each step. In Table 4 the calculated loading cycles for the 3 tests are tabulated together with the actual simulated cycles in order to point out the efficiency in cyclic skipping of the proposed methodology.

Table 4. Comparison between actual simulated cycles and calculated loading cycles.

Stress Ratio	Simulated cycles	Calculated cycles
0.1	40	698,036
0.5	32	434,990
0.7	40	580,690

7.2 SENB specimen tests

The experimental results of CTOD test, presented in the previous section were used in order to verify that the chosen cohesive model is capable to predict the crack propagation under monotonic loading for the material under consideration. The developed model for the simulation of CTOD test is shown in Figure 37, and it consists by 4-node plane strain elements with length equal to 0.4 mm, while rigid surfaces and contact algorithm were used for the simulation of cylinders. The analysis was conducted under a displacement control scheme, while the physical crack of 2 mm, due to pre-cracking loading, was taken into account in the numerical model by deleting five cohesive elements at the tip of the artificial crack. In Figure 38 and Figure 40 the deformed shape and the stress distribution are depicted for several stages of loading, while in Figure 39 the applied load with respect to mouth opening relationship is presented for both experimental and numerical results. The comparison indicates that the chosen cohesive model is capable to predict accurately the crack propagation under monotonic loading.

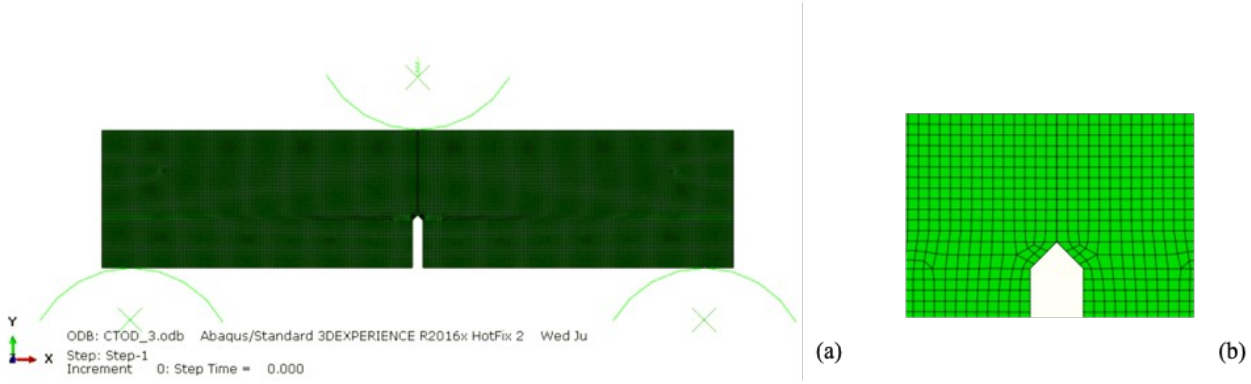

Figure 37. (a) Numerical model for the simulation of CTOD tests. (b) Mesh at the area of crack notch.

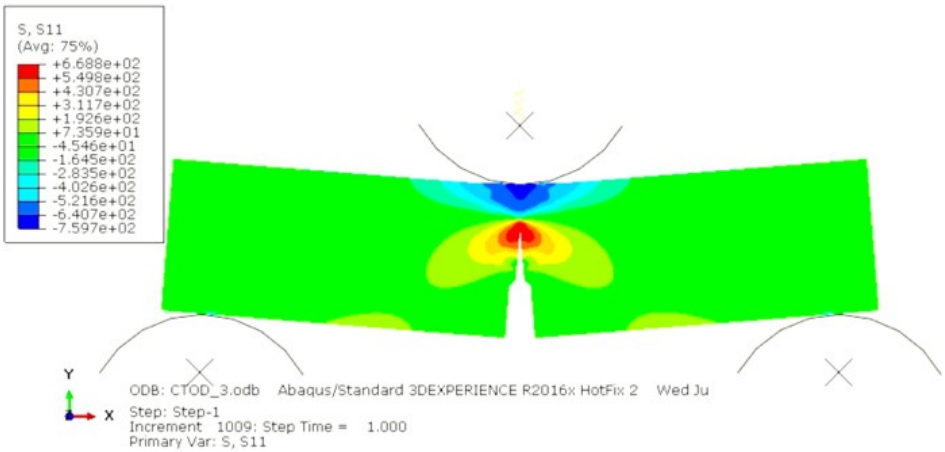

Figure 38. Final deformed shape and distribution of normal stress at x axis.

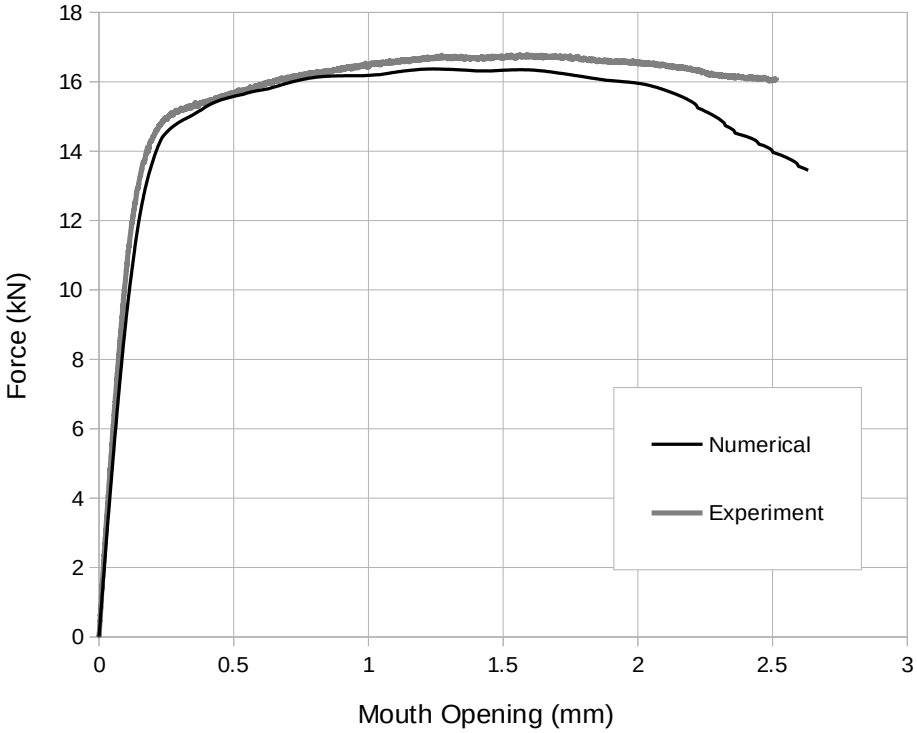

Figure 39. Applied load with respect to mouth opening, the parameters of the model are $\sigma_{max}=8800$, $\varphi_n=5000$, $\alpha=1.4$, $\lambda_n=0.016$.

The aforementioned SENB specimens were used also for the verification of the proposed method in the case of fatigue loading. More-specific, two specimens were subjected to cyclic loading, under a force control scheme, with a stress ratio $R = 0.1$. It is important to mention that the crack growth rate da/dN with respect to ΔK relationship of compact specimens, presented in the previous section, was used as input data for the simulation of those two experiments. More-over it has to be pointed out that in those two cases no physical crack exists at the start of the simulation, only the artificial crack due to the geometry of the specimens. At the first test a maximum force of 6.5 kN was used until the crack reach the 0.8 mm length, while for the rest of the experiment the maximum force was set to 4.5 kN. The second experiment performed at a constant maximum force equal to 3.0 kN for the entire test. In Table 5 the experimental and numerical results in terms of loading cycles are presented for the examined crack lengths, while in Figure 41 the crack length with respect to loading cycles is shown for the first tabulated test of Table 5. The comparison between experimental measurements and numerical predictions, for both tests, is consider excellent, while the cycle skipping efficiency of the proposed method is also presented in the results of Table 5. In Figure 42 the crack propagation and the distribution of max principal stress in depicted for the cyclic test with maximum force of 3.0 kN.

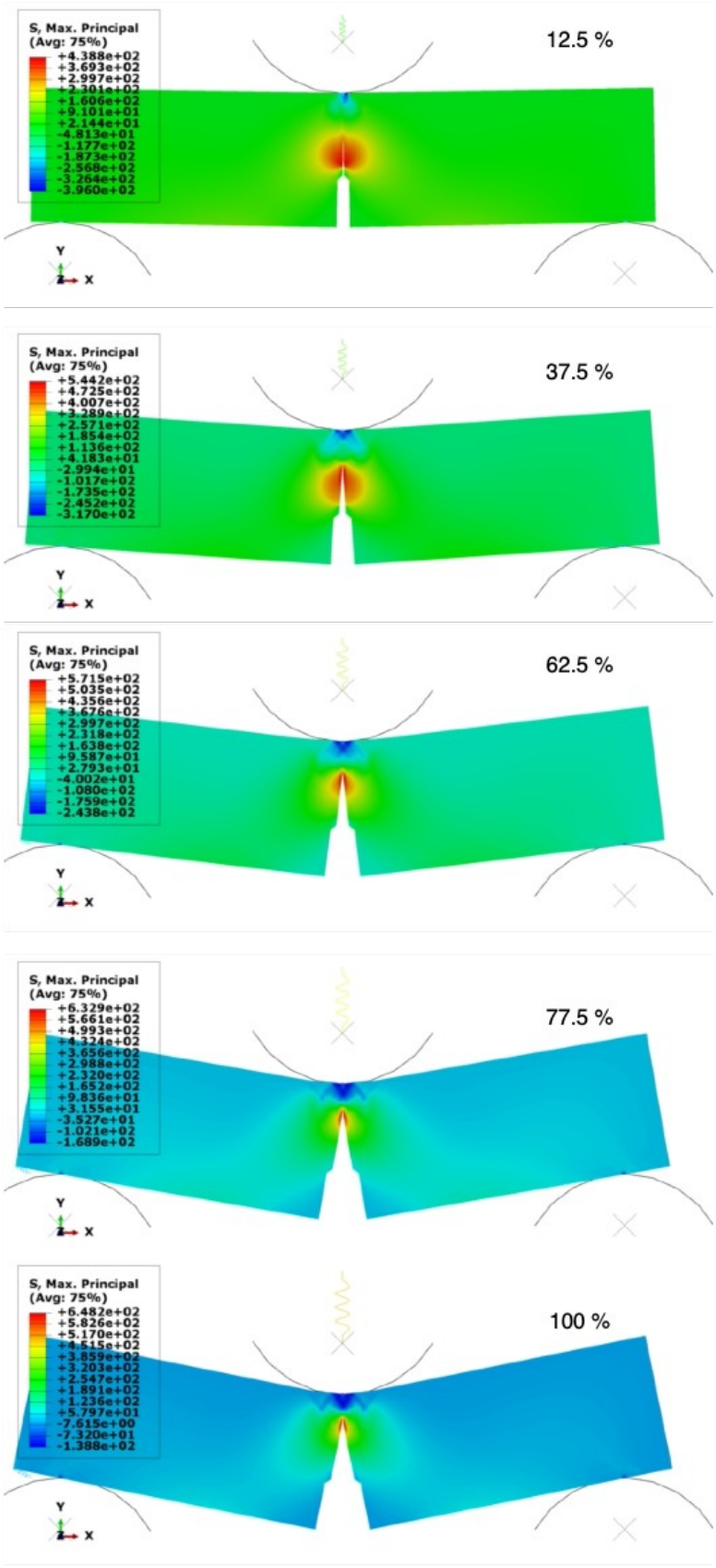

Figure 40. Crack propagation and stress distribution (max principal) of monotonic tests in several stages of loading.

Table 5. Experimental and numerical results of crack growth tests in SENB specimens.

Stress Ratio	Max Force (kN)	Min Force (kN)	Crack length (mm)	Simulated cycles	Experimental cycles	Calculated cycles
0.1	6.5 - 4.5	0.65 - 0.45	2.5	14	244,304	239,907
0.1	3.0	0.3	4.4	22	831,480	796,500

It should be mentioned again that the calibration of the models has been performed using only the data from the compact specimens, presented in the previous section. Therefore this two cases can be consider as blind predictions. The last experimental point of Figure 41 corresponds to a crack with extensive plastic deformation at the crack tip, a case which is out of the capabilities of the presented methodology. In Figure 43 the crack length is shown for the first test in different stages of the experiment, while the extensive plastic deformation of the area in frond of the crack tip is obvious.

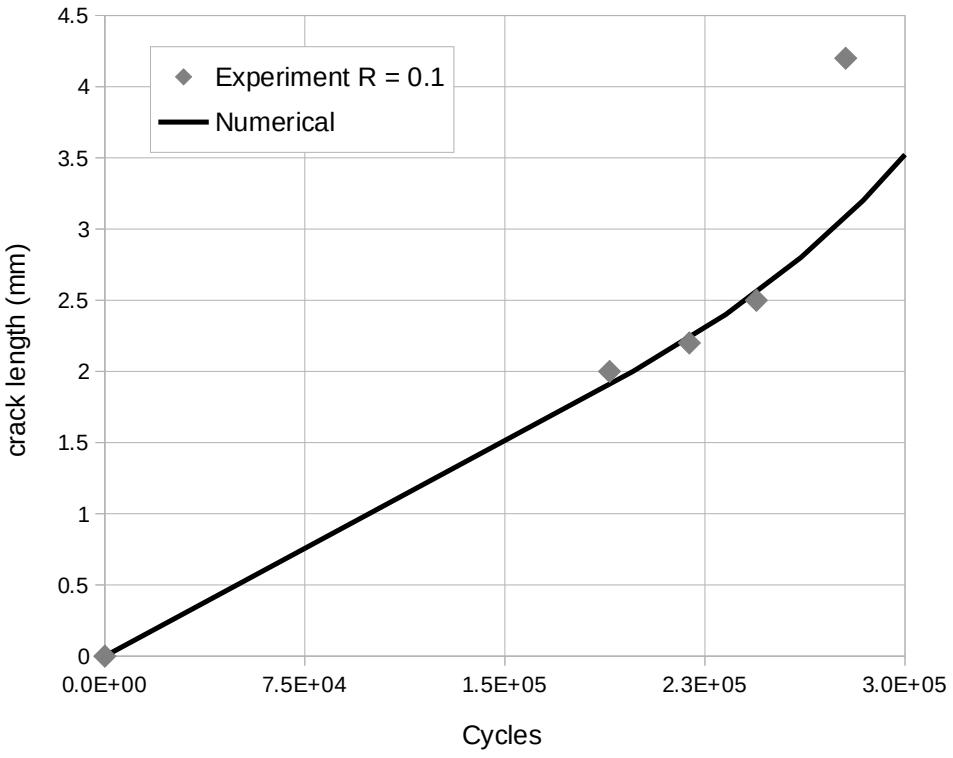

Figure 41. Loading cycles respect to crack length, the parameters of the model are $\sigma_{max}=8800$, $\varphi_{n}=5000$, $\alpha=1.4$, $\lambda_{n}=0.016$.

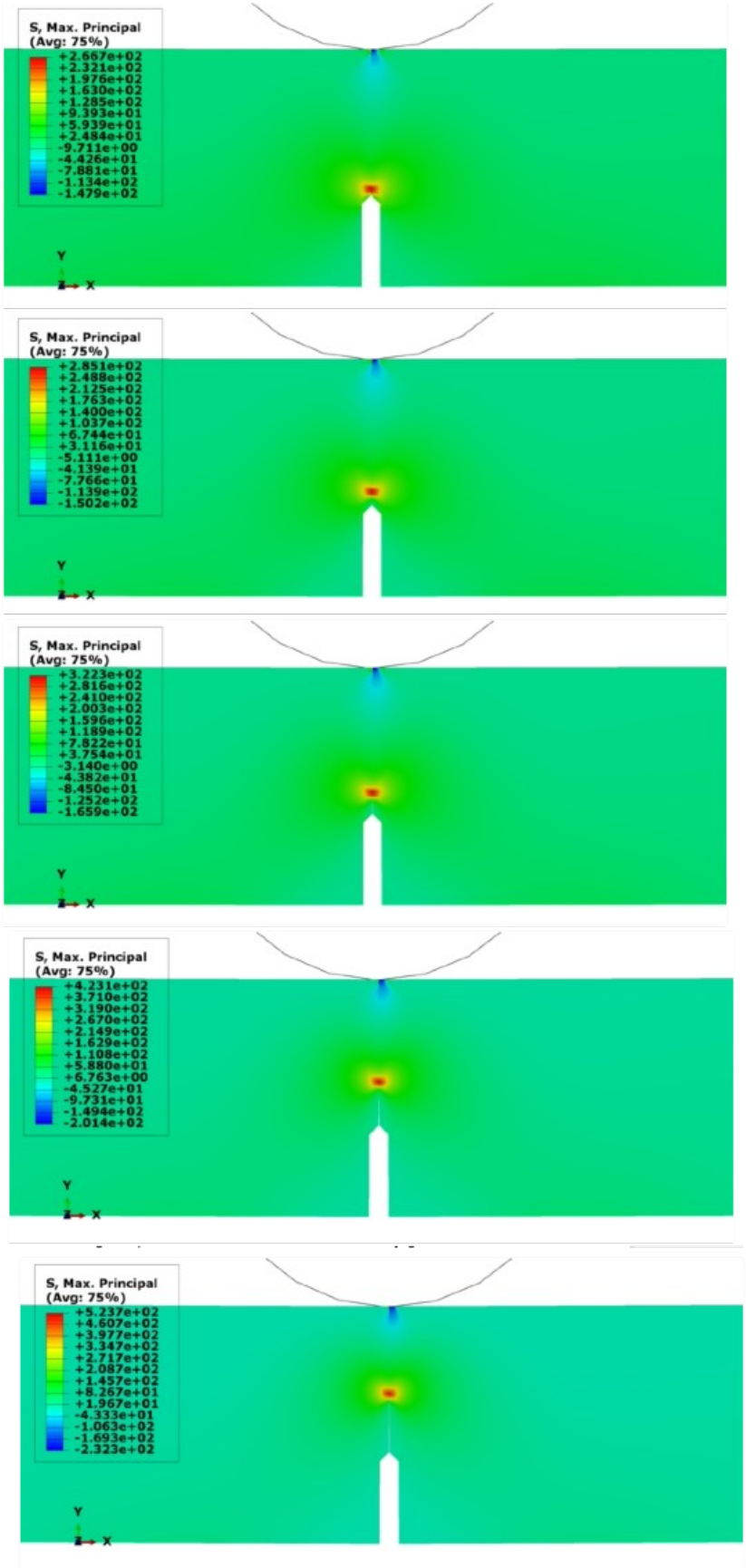

Figure 42. Crack propagation and stress distribution (max principal) for cyclic test of SENB speciment and R=0.1.

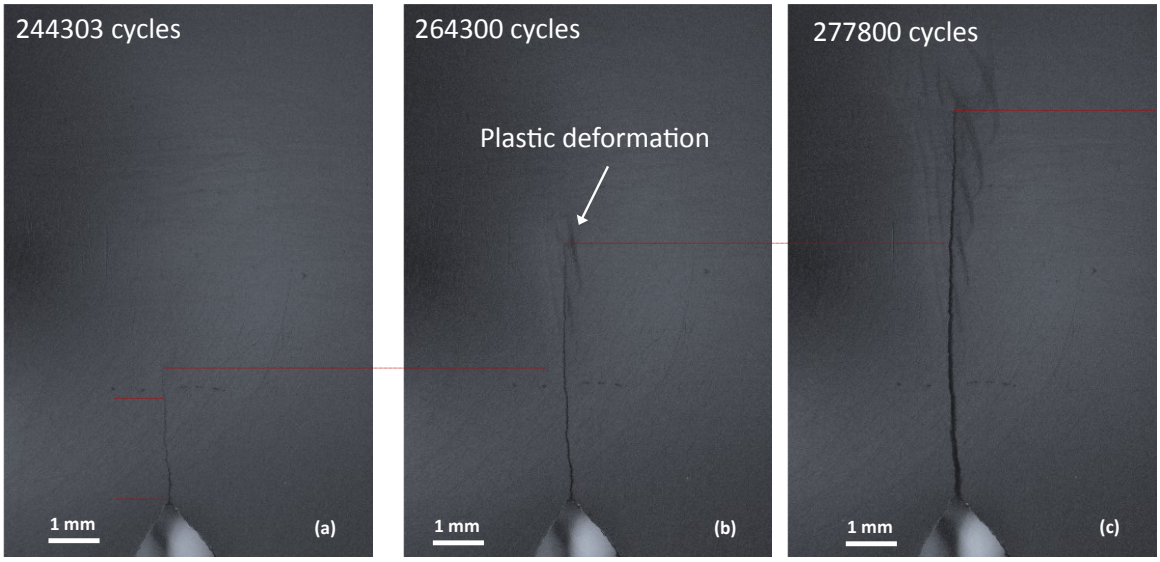

Figure 43. Plastic deformation in the area in frond of the crack tip.

8 Experimental results for future research

A series of experimental testing have also been performed in the framework of this dissertation in order to be used as validation and benchmark tests in future development of the proposed methodology in the study of welded steel components. Towards this purpose, five full-penetration welded specimens with a T-joint configuration, shown in Figure 44, are tested under high-cycle fatigue loading. Supplementary testing have also been performed in order to determine all the necessary material parameters which will be used for the development of the numerical model. More specific, uniaxial tensile tests, fracture toughness tests, crack growth rate tests, micro-hardness measurements, and metallurgical characterization of the welds have been performed for the material under consideration. It has to be mentioned that all the experimental results presented in previous sections were actually part of this experimental program, and all the specimens, compact and SENB, were extracted from the T-joints.

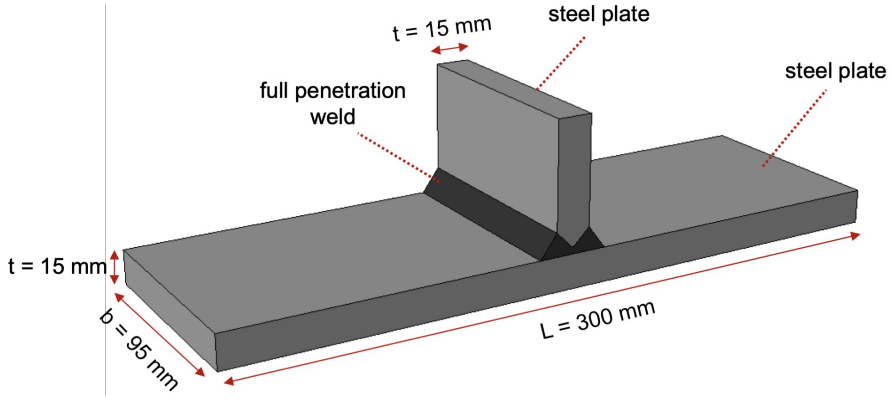

Figure 44. Geometrical properties of T-joint specimens.

The examined specimens consist of two perpendicular welded plates of equal thickness (15 mm) in a T-profile scheme, as shown in Figure 44. The experiments were conducted in the Laboratory of Mechanics Strength of Materials, Department of Mechanical Engineering. The specimens were subjected to four-point bending (see Figure 45).with a bending moment arm, e, equal to 67.5 mm. The experimental setup shown in Figure 46 uses an MTS 318.10 loading frame, with an MTS FlexTest SE digital controller and the test were conducted under a load control scheme.

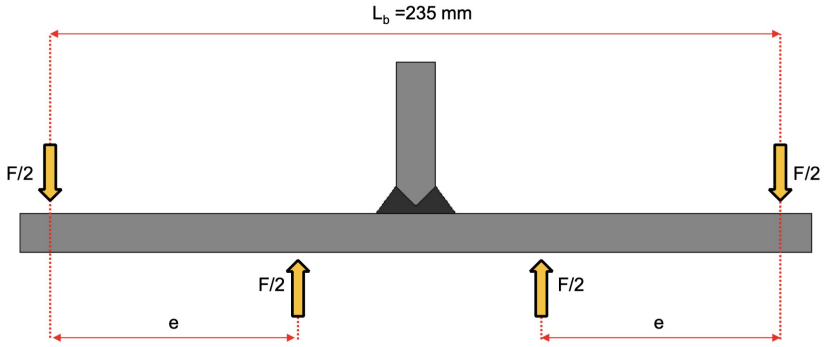

Figure 45. Schematic representation of the four-point bending scheme test setup.

Figure 46. Experimental setup for welded T-joints.

In addition to the fatigue test specimens, six single-edge-notched bend (SENB) specimens were machined from the parent plate, as shown in Figure 47. Three (3) specimens were notched in a direction transverse to the rolling direction (TDR) and three (3) specimens were notched parallel to the rolling direction (PDR). The nominal cross-sectional dimensions of the SENB specimens were 13 mm (width) and 26 mm (height), with a span of 109 mm. The tests were conducted at room temperature (RT) in accordance to [64, 65]. The specimens were tested under 3-point bending, as described in section 6, in order to determine the CTOD critical displacement and fracture toughness of the material under consideration.

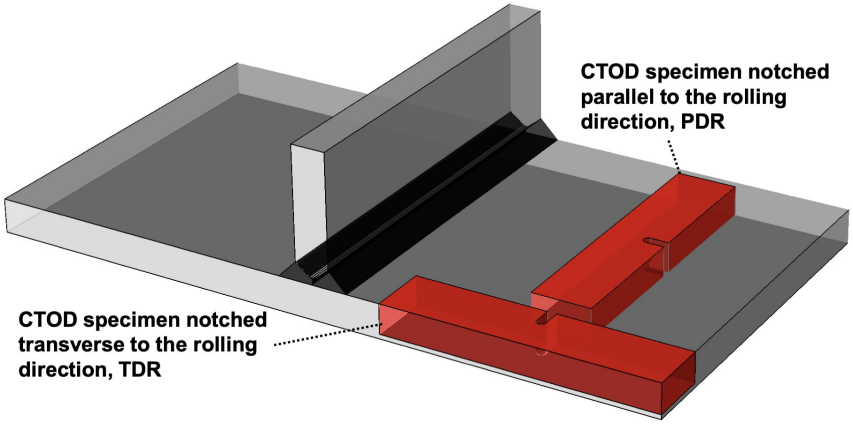

Figure 47. Schematic representation of CTOD specimen orientation with respect to the weld direction of the T-joint.

The gas metal arc weld (GMAW) method was employed for welding the specimens. The edge preparation and the dimensions of the weld presented in Figure 48, while the welding parameters are included in Table 6. Stereo-optical microscopy was performed on the transverse section of the T-joint specimens. Etching was performed with a solution of 30% HNO3 in distilled water. The macrostructure of the welds is shown in Figure 49a, where the weld metal and the HAZ are highlighted. The weld is free of macro-defects, and there is a smooth transition from the weld metal to the HAZ and the base metal. A detail of the weld root is shown in Figure 49b, where a small region of incomplete penetration can be observed.

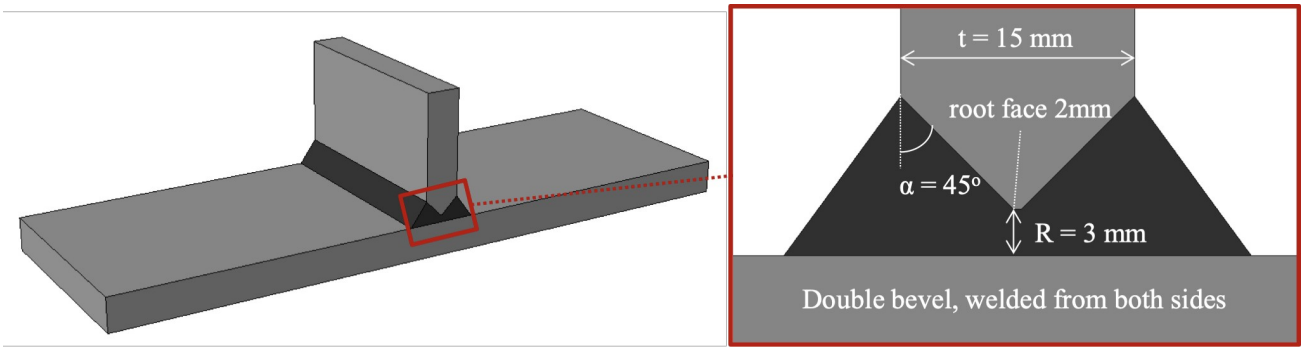

Figure 48. Weld configuration of T-joint specimens.

Table 6. Welding parameters.

Weld type	Number of passes on each side	Volts	Amp
GMAW	2	8.8	200

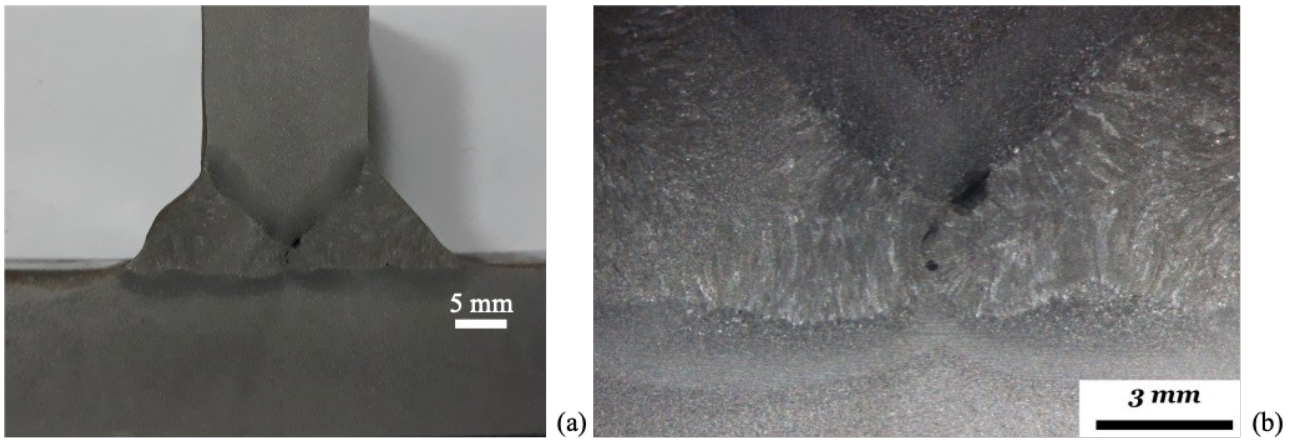

Figure 49. Macrostructure of the welds (a), Incomplete penetration (b).

Full metallographic analysis on the cross section was performed at the Laboratory of Materials, University of Thessaly. Etching was performed with Nital 2%. The base plate consists of ferrite and pearlite, while the HAZ consists of Widmanstätten ferrite and bainite. The weld metal exhibits a fine microstructure of Widmanstätten ferrite and acicular ferrite as shown in Figure 50. Moreover, micro-hardness measurements has also been performed according to the profiles shown in Figure 51a, with a load of 2 kgf on the Vickers scale. The results are presented in detail in Figure 51b.

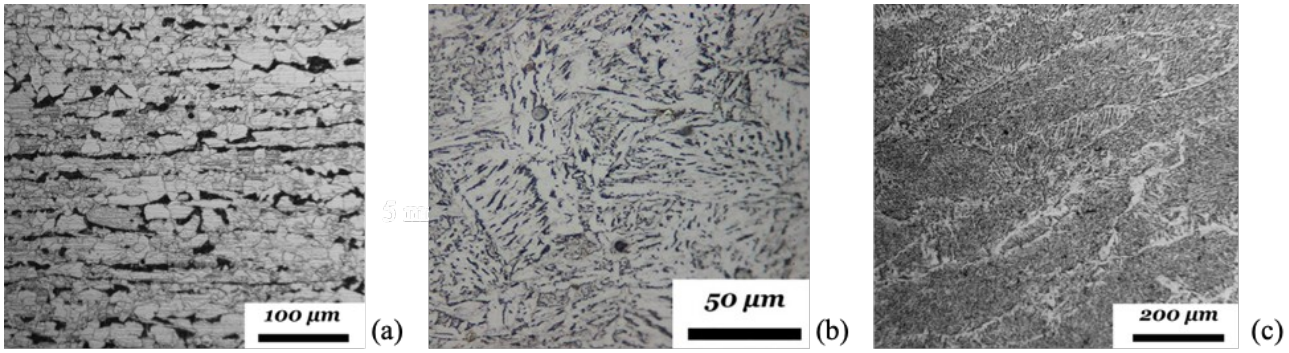

Figure 50. Microstructure of the base metal Ferrite and pearlite (a), HAZ adjacent to WM (grain growth area), Widmanstätten ferrite and bainite (b), Weld metal Widmanstätten ferrite and acicular ferrite (c).

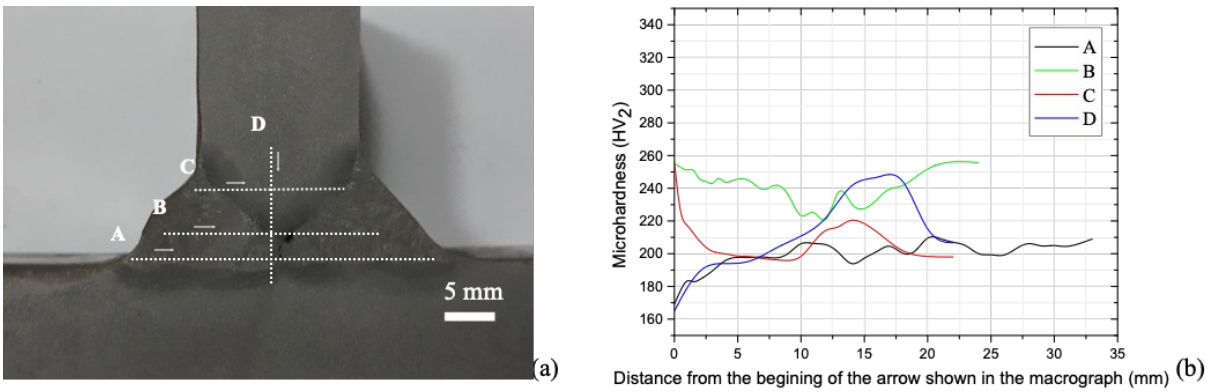

Figure 51. Profiles of micro-hardness measurements (a), Micro-hardness measurements across the lines (b).

The welded T-joints steel plates were tested under cyclic loading at a constant nominal stress ratio, $R=\sigma_{min}/\sigma_{max}=0.5$, and a frequency ranging from $f=6\,Hz$ to $f=8\,Hz$. Given the nominal stress range, the minimum and the maximum applied load were determined for each fatigue test (Table 7). The number of loading cycles to failure, N_f, are presented in Table 7 for all stress ranges considered. In these tests, failure refers to the loss of the specimen ability to sustain the maximum applied load, F_{max}. It is worth mentioning that the S355 steel was able to sustain a relative large number of cycles, more than 700,000, under fatigue loading with a maximum applied stress higher than the nominal yield strength of the material.

Table 7. Applied stress range and corresponding load for the fatigue tests of T-joint.

Specimen	$\Delta\sigma$ *(MPa)*	σ_{max} *(MPa)*	σ_{min} *(MPa)*	F_{max} *(kN)*	F_{min} *(kN)*	***Cycles to Failure, N_f***	$log\Delta\sigma$	$logN_f$
T-200	200	400	200	42.4	21.2	742,396	2.30	5.87
T-180	180	360	180	38.0	19.0	875,677	2.26	5.94
T-160	160	320	160	33.8	16.9	989,513	2.20	5.99
T-140	140	280	140	29.6	14.8	3,517,504	2.15	6.55
T-120	120	240	120	25.3	12.6	5,528,002	2.08	6.74

Crack propagation along the weld toe have been monitored using the MPI technique and were recorded by camera 1, as shown in Figure 52a. Both the left (LS) and right side (RS) of the weld (see Figure 52) were inspected during testing after a specified applied number of load cycles. Moreover, crack propagation through the thickness of the steel plate was monitored and recorded by camera 2 in each specimen (FL, FR, LS and BL), as shown in Figure 52a. The actual experimental setup with the camera at the FL position is shown in Figure 52b. The crack length, α, along the weld toe (RS location) for selected number of load cycles at a stress range of $\Delta\sigma=120\,MPa$ and $\Delta\sigma=200\,MPa$ is shown in Figure 53. The crack depth, d, through the thickness of the plate is presented in Figure 54 for the same selected number of load cycles of T-120(m) as in Figure 53a. The total crack length along

the weld toe at either RS or LS for each level of applied stress range with respect to the number of load cycles is presented in Figure 55. It is important to underline that the crack length, α, in Figure 55 corresponds to the crack length at the free surface of the specimen side along the weld toe, whereas no information related to the crack depth through the plate thickness is depicted in this figure. The coalescence of parallel cracking is presented in Figure 56 for the specimens subjected to $\Delta\sigma = 120\,MPa$ and $\Delta\sigma = 180\,MPa$. The stress fields in front of the tips of parallel cracks interact, resulting in the coalescence of cracks. This has been observed in most of the specimens tested.

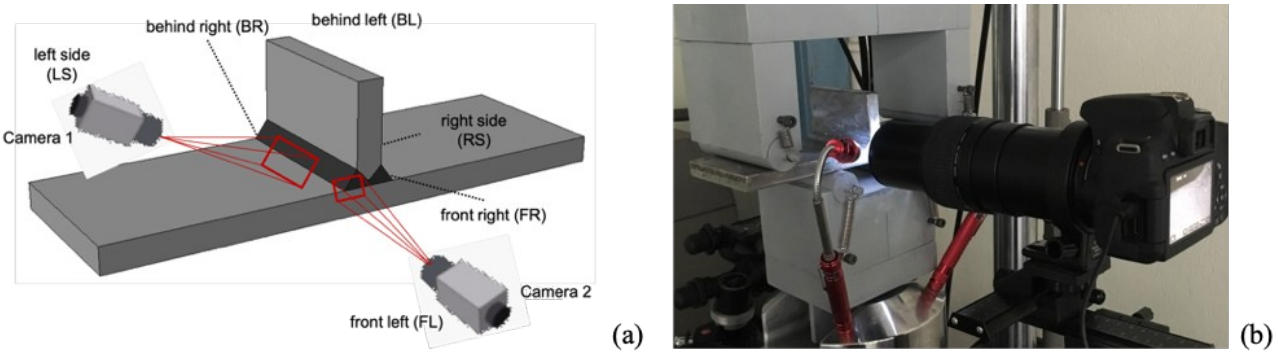

Figure 52. Schematic representation of camera positions, with respect to the weld of the T-joint plate specimens (a), Camera at the front left (FL) position (b).

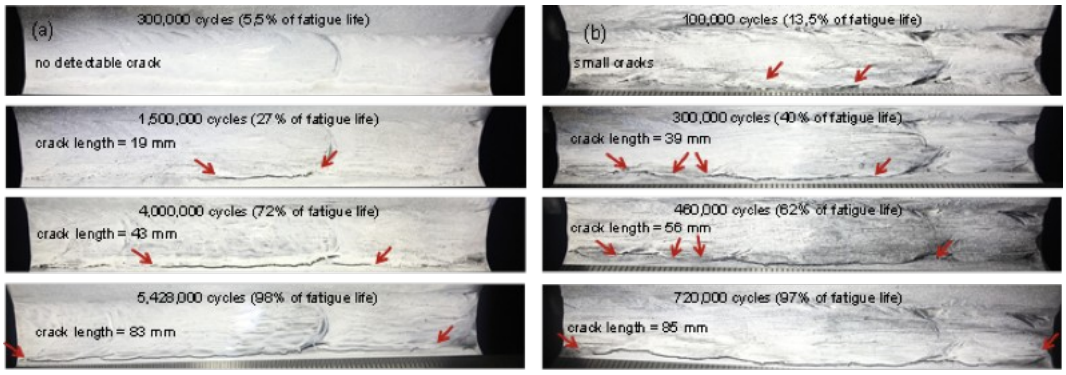

Figure 53. Crack development along the weld toe (crack length, α) at the RS weld region of T-joint plate specimen, subjected to nominal stress range of (a) $\Delta\sigma = 120\,MPa$ and (b) $\Delta\sigma = 200\,MPa$.

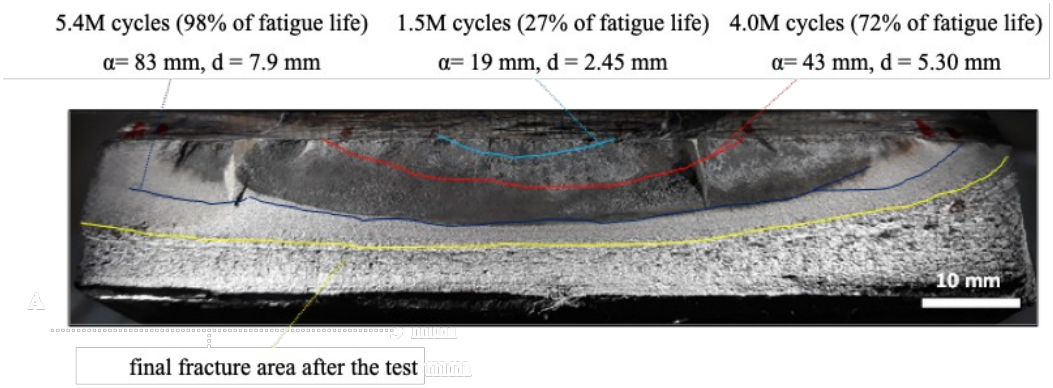

Figure 54. Crack development through the thickness of the plate (crack depth, d) of T-joint plate specimen, subjected to nominal stress range of $\Delta\sigma = 120\,MPa$.

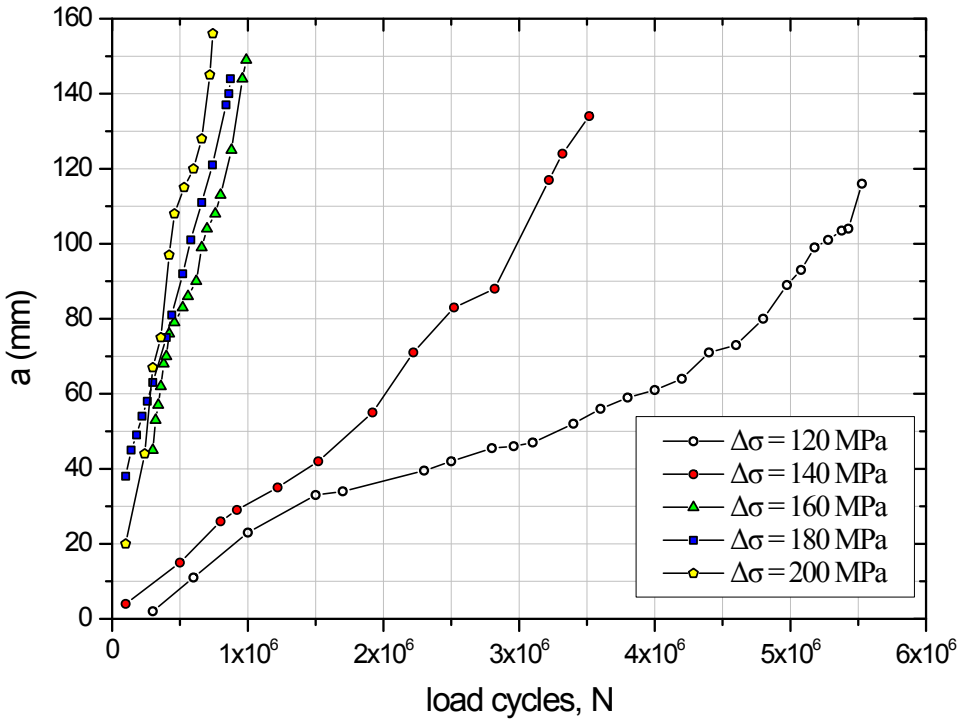

Figure 55. Total crack length, α, along the weld-toe of T-joint plate specimens (either LS or RS location) with respect to number of applied load cycles.

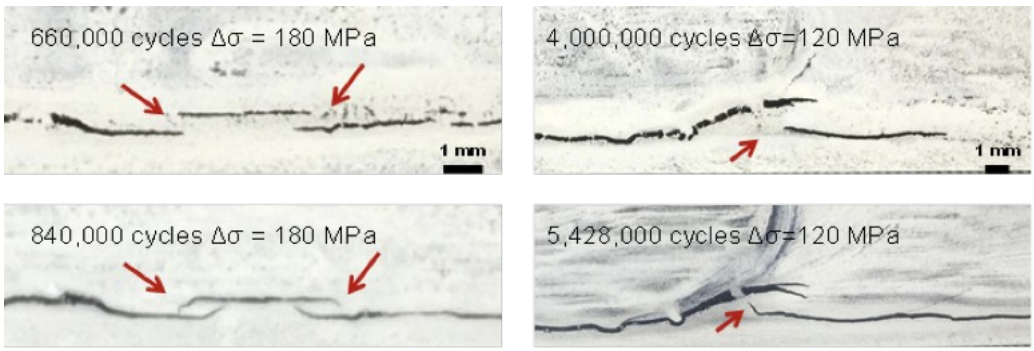

Figure 56. Coalescence of parallel cracking at the RS weld-toe of T-joint S355 steel plate specimens subjected to $\Delta\sigma = 180\,MPa$ (left), and $\Delta\sigma = 120\,MPa$ (right).

To determine the level of damage, a static loading cycle up to the maximum load, F_{max}, was performed after a specific number of load cycles was completed. The applied load with respect to the load-point displacement (LPD) diagrams, are presented in Figure 57 for several stages of fatigue testing of specimen T-160. Stiffness is reduced by about 30% from $28.2\,kN/mm$ to about $19.9\,kN/mm$ at failure after approximately 990.000 cycles. The stiffness drop is the result of the crack increase both along the width and thickness of the plate during fatigue. The flexural stiffness reduction of all specimens with respect to fatigue cycling as percentage of their fatigue life is presented in Figure 58. The specimens exhibited stiffness reduction of $4-8\%$ at approximately 90% of their fatigue life, while an approximately $7-10\%$ of stiffness drop was observed at 95% of their fatigue life for all specimens. Moreover, all specimens failed when their stiffness was reduced by 10% to 30%. The fact that the specimens maintained their stiffness until the last stage of their fatigue life indicates that the crack depth is relatively small for up to about 90% of their fatigue life.

The CTOD fracture toughness was determined based on experimental setup described in section 6. The force versus notch opening displacement was of type 6 (fully plastic behavior) according to [65]. The average maximum load, the crack mouth opening ($CMOD$) and the provisional fracture toughness K_{CTOD} obtained from the CTOD tests are summarized in Table 8.

Table 8. Average fracture toughness parameters from CTOD tests.

Steel material	Orientation	Number of tests	Test temperature	Maximum Load (kN)	$CMOD$, δ (mm)	K_{CTOD} $MPa\sqrt{m}$
S355JR	TDR	3	RT	16.07	1.791	38.6
S355JR	PDR	3	RT	15.79	1.827	41.4

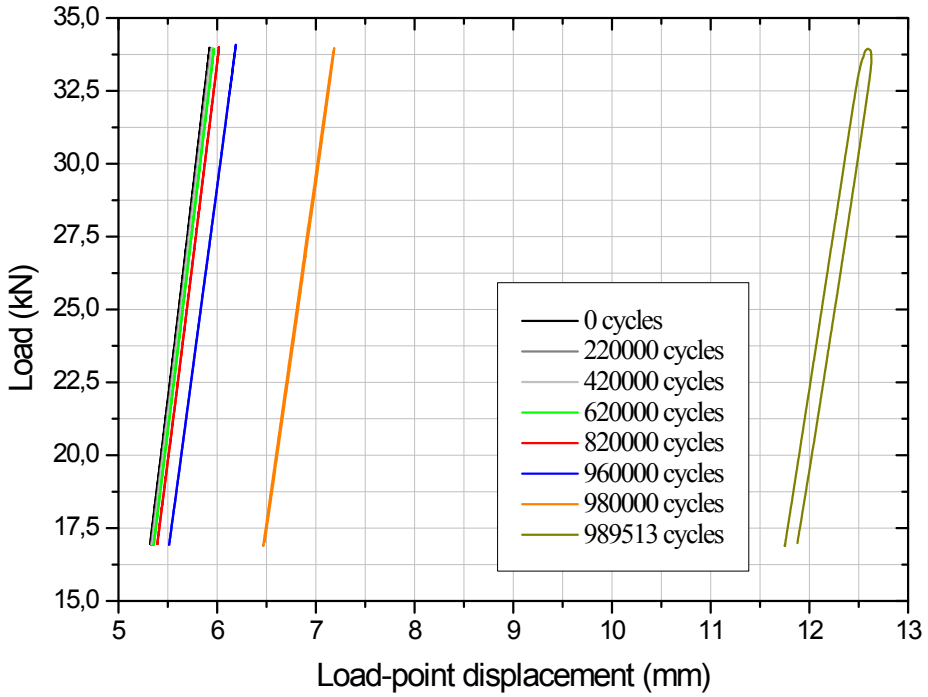

Figure 57. Load vs. LPD of T-joint mild-strength steel plate specimen at $\Delta\sigma = 160\,MPa$.

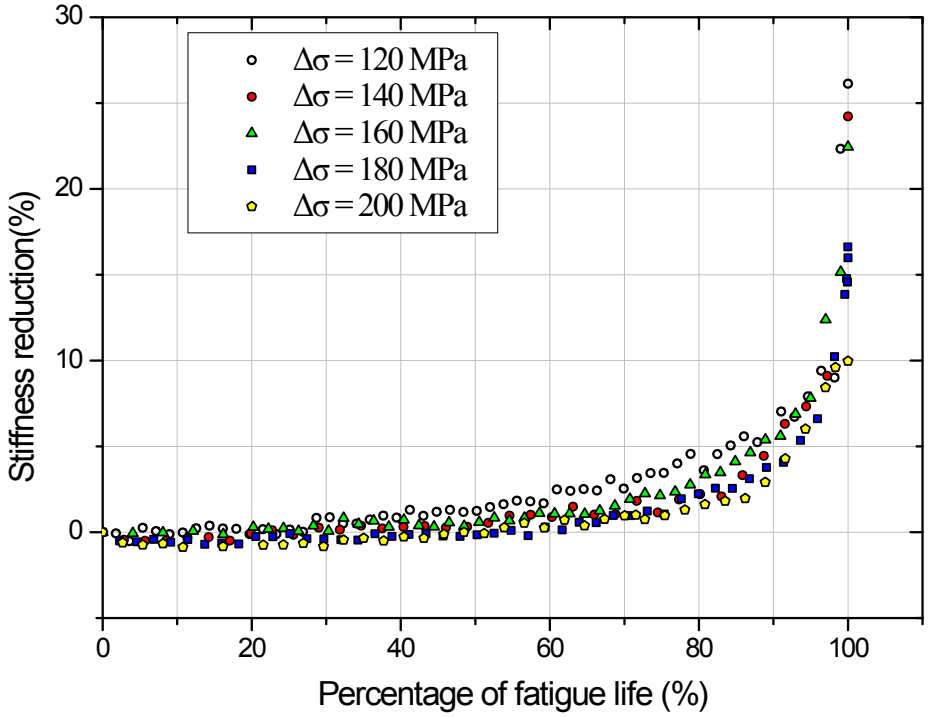

Figure 58. Flexural stiffness reduction (as percentage of initial stiffness) of T-joint steel plates with respect to percentage of fatigue life.

9 Conclusion

The numerical simulation of high cyclic fatigue of steels has been investigated in the framework of the present dissertation. A robust and effective algorithm was developed capable to simulate the crack propagation. More specific, the algorithm combines the cohesive element concept with the J-integral calculation in order to estimate the crack growth rate. The basic open issues of the use of cohesive elements in high cyclic fatigue of bulk materials, which are related to the calibration of the cohesive model and the cyclic skipping, are successfully addressed. The material calibration for cyclic loading can be achieved using a straight forward procedure, which simply requires the experimental data of a crack growth rate test (Δ**K**-da/dN) or the parameters of Paris' law. The cyclic skipping methodology uses the J-integral calculation for the estimation of stress intensity factor and the cohesive element length in order to calculate the loading cycles that the crack needs to propagate through all element length.

The developed algorithm is capable to perform a series of calculation automatically. More specific, it is capable to predict automatically the position of crack tip, the direction of crack propagation and the crack growth rate during the analysis, without the need of re-meshing or the need of consecutive analyses. The evaluation of stress intensity factor and the crack length is taking place at the end of each step and it is totally automatic.

Moreover experimental tests have been conducted in order to be used for the verification of the developed algorithm. More specifically, the proposed methodology has been verified against five crack growth tests, three in CT-specimens, two in SENB-specimens and one monotonic CTOD tests. A larger series of experimental testing was also performed, which concerns high cyclic fatigue tests of welded T-joints, in order to be used for further research and development of the proposed methodology.

It is also important to mention the limitations of the current work. The current algorithm has been developed for 2D analysis, and can be applied effectively only in problems that are under plain strain conditions. The expansion of the algorithm to 3D analysis is the first step of the planned future research, together with the development of a traction-separation law that takes into account the effect of the corrosion.

10 References

1. Miller, K., J., (1991), "Metal fatigue – past current and future." *Proc. Inst. Mech. Eng.*, **205**: 291–304.

2. Suresh S., (1998), "Fatigue of Material", *Cambridge University Press.*

3. Needleman, A., (2014), "Some issues in cohesive surface modeling", *23rd International Congress of Theoretical and Applied Mechanics,* 221-246.

4. Mann, J., Y., (1957), "The historical development of research on the fatigue of materials and structures", *Journal of Australian Institute of Metals*, **3**: 222-41.

5. Schutz, W., (1996), "A history of Fatigue", *Engineering Fracture Mechanics*, **54**: 263-300.

6. Schijve, J., (1973), " Effect of load sequences on crack propagation under random and program loading.: *Engineering Fracture Mechanics,* **5**: 269-80.

7. Ewing, J., A., Humfrey, J., C., (1903), "The fracture of metals under rapid alteration of stress", *Philosophical Transactions of the Royal Society, London,* **A200**: 241-50.

8. Irwin, G., R., (1957), "Analysis of stresses and strains near the end of a crack transversing a plate", *Journal of Applied Mechanics*, **24**: 361-4.

9. Paris, P., C., Gomez, M., P., Anderson, W., P., (1961), "A rational analytic theory of fatigue.", *The Trend in Engineering*, 13: 9-14.

10. Benzeggagh, M., Kenane, M., (1996), "Measurement of Mixed-Mode Delamination Fracture Toughness of Unidirectional Glass/Epoxy Composites with Mixed-Mode Bending Apparatus", *Composite Science and Technology*, **56**: 439.

11. Reeder, J., Kyongchan, S., Chunchu, P., B., Ambur, D., R., (2002), "Postbuckling and Growth of Delaminations in Composite Plates Subjected to Axial Compression", *43rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, Denver, Colorado*, **1746**: 10.

12. Wu, E. M., Reuter, R., C., (1965), "Crack Extension in Fiberglass Reinforced Plastics," *T and M Report, University of Illinois*, **275**.

13. Elguedj, T., Gravouil, A., Combescure, A., (2006), "Appropriate Extended Functions for X-FEM Simulation of Plastic Fracture Mechanics", *Computer Methods in Applied Mechanics and Engineering*, **195**: 501–515.

14. Sukumar, N., Prevost, J.-H., (2003), "Modeling Quasi-Static Crack Growth with the Extended Finite Element Method Part I: Computer Implementation," *International Journal for Solids and Structures*, **40**: 7513–7537.

15. Song, J. H., Areias, P. M. A., Belytschko, T. , (2006), "A Method for Dynamic Crack and Shear Band Propagation with Phantom Nodes," *International Journal for Numerical Methods in Engineering*, **67**: 868–893.

16. Barenblatt, G., I., (1959), "On equilibrium cracks forming during brittle fracture", *Prikl Mat Mekh (PMM)*, **23**: 434–444.

17. Barenblatt, G., I., (1962), "Mathematical theory of equilibrium cracks", *Adv Appl Mech*, **7**: 56–129.

18. Hillerborg, A,. Modeer, M., Petersson, P., E., (1976), "Analysis of crack formation and crack growth in concrete by means of fracture mechanics", *Cement and Concrete Research,* **6**: 773–782.

19. Andrews, D., J., (1976), "Rupture velocity of plane strain shear cracks", *J Geophys Res,* **81**: 5679–5687.

20. Rosakis, A., J., (2002), "Intersonic shear cracks and fault ruptures", *Adv Phys,* **51**: 1189–1257.

21. Rosakis, A., J., Xia, K., Lykotrafitis, G., Kanamori, H., (2007), "Dynamic shear rupture in frictional interfaces: speeds, directionality, and modes", *Treatise in Geophysics 4 - Earthquake Seismology*, 153–192.

22. Shi, Z., Needleman, A., Ben-Zion, Y., (2010), "Frictional slip modes and partition of energy during dynamical frictional sliding between identical elastic-viscoplastic solids", *Int J Fract,* **62**: 51–67.

23. Lu, X., Lapusta, N., Rosakis, A., J., (2010), "Pulse-like and crack-like dynamic shear ruptures on frictional interfaces: experimental evidence, numerical modeling, and implications", *Int J Fract,* **163**: 27–39.

24. Needleman, A., (1987), "A continuum model for void nucleation by inclusion debonding", *J Appl Mech,* **54**: 525–531.

25. Xu, X., P., Needleman, A., (1993), "Void nucleation by inclusion debonding in a crystal matrix", *Model Simul Mat Sci Engin,* **1**: 111–132.

26. Xu., X., P., Needleman, A., (1994), "Numerical simulations of fast crack growth in brittle solids", *J Mech Phys Solids,* **42**: 1397–1434.

27. Camacho, G., T., Ortiz, M., (1996), "Computational modeling of impact damage in brittle materials", *Int J Solids Struct,* **33**: 2899–2938.

28. Papoulia, K., D., Sam, S., H., Vavasis, S., A., (2003), "Time-continuous cohesive interface finite elements in explicit dynamics", *Int J Numer Meth Engin,* **58**: 679–701.

29. Falk, M., L., Needleman, A., Rice, J., R., (2001), "A critical evaluation of cohesive zone models of dynamic fracture", *Journal de Physique IV,* **11** (Pr5), 4350.

30. Needleman, A., (1990), "An analysis of tensile decohesion along an interface", *J Mech Phys Solids,* **38**: 289–324.

31. Beltz, G., E., Rice. J., R., (1991), "Dislocation nucleation versus cleavage decohesion at crack tips", *In: Lowe TC, et al. editors. Modeling the Deformation of Crystalline Solids, The Minerals, Metals and Materials Society (TMS)*, 457–480.

32. Sun, Y., Beltz, G., E., Rice, J., R., (1993), "Estimates from atomic models of tension-shear coupling in dislocation nucleation from a crack tip", *Mat Sci Engin,* **A170**: 67–85.

33. Park, K., Paulino, G., H., Roesler, J., R., (2009), "A unified potential-based cohesive model of mixed-mode fracture", *Journal of the Mechanics and Physics of Solids*, **57**: 891–908.

34. He, M., H., Xin., K., (2011), "Separation work analysis of cohesive law and consistently coupled cohesive law", *Appl Math Mech,* **32**: 1437–1446.

35. Ortiz, M., Pandolfi, A., (1999), "Finite-deformation irreversible cohesive elements for three-dimensional crack-propagation analysis", *Int J Num Meth Engin,* **44**: 1267–1282.

36. Nguyen, O., Repetto, E., A., Ortiz, M., Radovitzky, R., A., (2001), "A cohesive model of fatigue crack growth", *Int J Fract,* **110**: 351–369.

37. Roe, Siegmund, T., (2003), "An irreversible cohesive zone model for interface fatigue crack growth simulation", *Eng Fract Mech,* **70**: 209–232.

38. Mosler, J., Scheider, I., (2011), "A thermodynamically and variationally consistent class of damage-type cohesive models", *J Mech Phys Solids,* **59**: 1647–1668.

39. Van der Ven, A,. Ceder, G., (2004), "The thermodynamics of decohesion" *Acta Mater,* **52**: 1223–1235.

40. Yamakov, V., Saether, Phillips, D., R., Glaessgen, E., H., (2006), "Molecular-dynamics simulation-based cohesive zone representation of inter-granular fracture processes in aluminum", *J Mech Phys Solids,* **54**: 1899–1928.

41. Jiang, L., Y., (2010), "A cohesive law for carbon nanotube/polymer interface accounting for chemical covalent bonds", *Math Mech Solids,* **15**: 718–732.

42. Song, J., Curtin, W., A., Bhandakkar, T., K., Gao, H., J., (2010), "Dislocation shielding and crack tip decohesion at the atomic scale", *Acta Mater,* **58**: 5933–5940.

43. He, M., Li, S., (2012), "An embedded atom hyperelastic constitutive model and multiscale cohesive nite element method", *Comput Mech* **49**: 337–355.

44. Tvergaard, V., Hutchinson, J., W., (1992), "The relation between crack growth resistance and fracture process parameters in elastic-plastic solids", *J Mech Phys Solids*, **40**: 1377–1397.

45. Nielsen, K., L., Hutchinson, J., W., (2012), "Cohesive traction-separation laws for tearing of ductile metal plates", *Int J Impact Engin,* **48**: 15–23.

46. Kulkarni, M., G., Matous, K., Geubelle, P., H., (2010), "Coupled multi-scale cohesive modeling of failure in heterogeneous adhesives", *Int J Numer Meth Engin,* **84**: 916–946.

47. Mohammed, I., Liechti, K., M., (2000), "Cohesive zone modeling of crack nucleation at bimaterial corners", *J Mech Phys Solids,* **48**: 735–764.

48. Chalivendra., V., B., Hong, S., Arias, I., Knap, J., Rosakis, A., Ortiz, M., (2009), "Experimental validation of large-scale simulations of dynamic fracture along weak planes", I*nt J Impact Engin,* **36**: 888–898.

49. Hong, S., Kim, K., S., (2003), "Extraction of cohesive zone laws from elastic far fields of a cohesive crack tip: a field projection method", *J Mech Phys Solids,* **51**: 1267–1286.

50. Kim, H., G., Chew, H., B., Kim, K., S., (2012), "Inverse extraction of cohesive zone laws by field projection method using numerical auxiliary fields", *Int J Numer Meth Eng,* **91**: 516–530.

51. Bazant, Z., P., Yu, Q., (2011), "Size-effect testing of cohesive fracture parameters and nonuniqueness of work-of-fracture method", *J Engin Mech,* **137**: 580–588.

52. Maier, G., Bocciarelli, M., Bolzon, G., Fedele, R., (2006), "Inverse analyses in fracture mechanics", *Int J Fract,* **138**: 47–73.

53. Park, K., Paulino, G., H., (2012), "Computational implementation of the PPR potential-based cohesive model in ABAQUS: Educational perspective", *Engineering Fracture Mechanics*, **93**: 239–262.

54. Spring, D., W., Giraldo-Londono, O., Paulino, G., H., (2016), "A study on the thermodynamic consistency of the Park–Paulino–Roesler (PPR) cohesive fracture model", *Mechanics Research Communications* , **78**: 100–109.

55. Coleman, B., D., Gurtin, M., E., (1967), "Thermodynamics with internal state variables", *J. Chem. Phys.,* **47**: 597–613.

56. Lemaitre, J., (1992), "A Course on Damage Mechanics", *Springer-Verlag, Berlin.*

57. Simo, J., C., Hughes, T., J., R., (1998), "Computational Inelasticity", *Springer-Verlag, NewYork.*

58. Lemaitre, J., Chaboche, J., L., (1990), "Mechanics of Solid Materials", *Cambridge University Press.*

59. Ural, A., Krishnan, V., R., Papoulia, K., D., (2009), "A cohesive zone model for fatigue crack growth allowing for crack retardation", *International Journal of Solids and Structures,* **46**: 2453–2462.

60. Rice J., R., (1968), "Mathematical Analysis in the Mechanics of Fracture", *Academic Press, N.Y.*, pp. 191-311.

61. Li, F., Z., Shih, F., Needleman, A., (1985), "A comparison of methods for calculating energy release rates", *Engineering Fracture Mechanics*, **21 – 2**: 405-421.

62. ASTM E399, (1997), "Standard Test Method for Plane-Strain Fracture Toughness of Metallic Materials".

63. ASTM E647, "Standard Test Method for Measurement of Fatigue Crack Growth Rates".

64. BS 7608, (1993), "Code of practice for fatigue design and assessment of steel structures", *British Standards Institute*, London, UK

65. BS 7448, (1997), "Fracture Toughness tests – Parts 1 and 2: Method for the determination of KIC, critical CTOD and critical J values of welds in metallic materials", *British Standards Institute*, London, UK.

66. Matsuishi, M., Endo, T. (1968). Fatigue of metals subjected to varying stress. *Japan Society of Mechanical Engineers*, Fukuoka, Japan.

67. ASTM E1049-85. (2017). Standard practices for cycle counting in fatigue analysis. *ASTM International*.

11 Appendix

11.1 Cycle counting

Cyclic loading, such as wind loads on a turbine blade or vibrations on a bridge, subjects materials to stress variations that cause fatigue damage over time. The amplitude, mean, and number of these stress cycles are key parameters that affect the material fatigue life. However, real-world stress signals are often irregular and can contain various combinations of cycles of differing amplitudes and means. Therefore, cyclic counting algorithms are essential tools in fatigue analysis for determining the number and amplitude of stress cycles in a fluctuating signal. These algorithms convert complex, irregular stress signals into representative load cycles, which are crucial for estimating the lifetime of materials under cyclic loading. Among the most widely used of these algorithms is the Rainflow Counting Method, originally developed for fatigue analysis in mechanical structures, particularly in the aerospace and automotive industries. The rainflow method was not used in this dissertation but it will be analyzed briefly since it is the main methodology used in fatigue analyses.

The rainflow counting method was developed by [66] and gets its name from the analogy of raindrops flowing down a pagoda roof, where larger drops meet and merge with smaller drops as they fall. This metaphor describes how the algorithm counts stress cycles by identifying peaks and valleys in the stress-time signal and pairing them into half or full cycles.

The basic steps of the rainflow counting algorithm are as follows, while in [67] a description of its step can be found.

1. Identify peaks and valleys: The algorithm first identifies the local maxima and minima in the stress-time signal.
2. Cycle extraction: The method pairs the peaks and valleys to form full cycles. It uses the turning points of the stress signal to determine closed cycles based on the relative magnitudes of stress changes.
3. Closure of cycles: When a cycle is closed (i.e., the stress returns to a previously recorded peak or valley), the algorithm counts it as a full cycle and removes it from further analysis.
4. Residual stresses: Remaining unclosed cycles, or half-cycles, are left at the end of the process and are counted as such, since they also contribute to fatigue damage.
5. Cycle counting and histogram construction: The rainflow method eventually constructs a cycle histogram, categorizing the cycles by their range and mean values.

The rainflow method offers several advantages over other cycle counting techniques, such as level-crossing counting or peak counting:

- Accuracy in fatigue damage estimation: Rainflow counting provides a realistic representation of how actual stress cycles damage materials, correlating well with empirical fatigue life data.
- Suitability for irregular signals: Many real-world load signals are non-periodic and irregular. The rainflow algorithm handles such irregularity efficiently by focusing on stress reversals.
- Comprehensive treatment of different cycle types: The algorithm captures not only full cycles but also half-cycles, which can be important for situations where stress does not fully reverse.

Other cycle counting algorithms, such as the Level Crossing Counting and Range-Pair Counting methods, are also used in signal analysis but tend to be less effective at capturing the full picture of fatigue damage compared to the rainflow method. The Level Crossing technique, for instance, only counts the instances where the signal crosses predefined levels, missing the detailed stress cycle patterns that are vital in fatigue analysis.

In the present study a much simpler algorithm used for the counting of cycles due to the fact that only constant amplitude loading was used. The algorithm recognizes the peaks and the valleys of the signal simply by passing all the values and marking the $value_i$ as peak or a valley bases on inequalities 88 and 89, respectively. The range is defined as the difference of a peak from the valley.

$$value_{i-1} < value_i > value_{i+1} \tag{88}$$

$$value_{i-1} > value_i < value_{i+1} \tag{89}$$

11.2 Algorithm for J-integral calculation per element

```
c ######################################################################
      SUBROUTINE K_J_ELEM ( J_VALUE, Bc, XI, ETA, I, Q, STRESS)
      USE GLOBAL
c
      INCLUDE 'ABA_PARAM.INC'
c
      DIMENSION Bc(3,8), Bc2(4,2), Q(4,1), DQDX(2,1), DUDX(2,1),
     & SDUDX(2,1), H(2,1), STRESS(3)
c
      INTEGER I
c
      DOUBLE PRECISION Q, DQDX, DUDX, SDUDX, H, J_VALUE, STRESS
c
c      WRITE(7,*) '######## START K_J_elem ###########'
      CALL K_INIT_MAT(Bc2, 4, 2)
      CALL K_INIT_MAT(DQDX, 2, 1)
      CALL K_INIT_MAT(DUDX, 2, 1)
      CALL K_INIT_MAT(SDUDX, 2, 1)
      CALL K_INIT_MAT(H, 2, 1)
c
      Bc2(1,1) = Bc(1,1)
      Bc2(2,1) = Bc(1,3)
      Bc2(3,1) = Bc(1,5)
      Bc2(4,1) = Bc(1,7)
c
      Bc2(1,2) = Bc(2,2)
      Bc2(2,2) = Bc(2,4)
      Bc2(3,2) = Bc(2,6)
      Bc2(4,2) = Bc(2,8)

c      WRITE(7,*) 'Bc2 =', Bc2
c *******************************************************
c Derivative of Smooth function q at the direction of crack propagation
      DQDX(1,1) = Q(1,1)*Bc2(1,1) + Q(2,1)*Bc2(2,1)
     & + Q(3,1)*Bc2(3,1) + Q(4,1)*Bc2(4,1)
      DQDX(2,1) = Q(1,1)*Bc2(1,2) + Q(2,1)*Bc2(2,2)
     & + Q(3,1)*Bc2(3,2) + Q(4,1)*Bc2(4,2)
c      WRITE(7,*) 'DQDX(1,1) = ', DQDX(1,1)
c      WRITE(7,*) 'DQDX(2,1) = ', DQDX(2,1)
c *******************************************************
c Derivative of displacements at the direction of crack propagation
      DUDX(1,1) = UTR(1,1)*Bc2(1,1) + UTR(1,3)*Bc2(2,1)
     & + UTR(1,5)*Bc2(3,1) + UTR(1,7)*Bc2(4,1)
      DUDX(2,1) = UTR(1,2)*Bc2(1,1) + UTR(1,4)*Bc2(2,1)
     & + UTR(1,6)*Bc2(3,1) + UTR(1,8)*Bc2(4,1)
c      WRITE(7,*) 'DUDX(1,1) = ', DUDX(1,1)
c      WRITE(7,*) 'DUDX(2,1) = ', DUDX(2,1)
c
      SDUDX(1,1)=(STRESS(1)*DUDX(1,1)+STRESS(3)*DUDX(2,1))
      SDUDX(2,1)=(STRESS(3)*DUDX(1,1)+STRESS(2)*DUDX(2,1))
```

```
c
      H(1,1)= - SDUDX(1,1) + STRDENS
      H(2,1)= - SDUDX(2,1)
c
      J_VALUE = -( H(1,1) * DQDX(1,1)
     & + H(2,1) * DQDX(2,1) ) * JACOBDET * 4.0D0
c       WRITE(7,*) 'J_VALUE = ', J_VALUE
c
c       WRITE(7,*) '######## END K_J_elem ###########'
c
      RETURN
      END
```

11.3 Algorithm for cohesive element

```
c ************************************************************************
      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)
       USE GLOBAL
       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(*)
c
       DIMENSION Sc(ndofel,ndofel), Fc(ndofel,nrhs),
     & T(mcrd,nrhs), T_d(mcrd,mcrd), U_l(ndofel), R(mcrd, mcrd),
     & Bc(mcrd,ndofel), Bct(ndofel,mcrd), ShapeN(nnode), DU_l(ndofel),
     & del(mcrd), GP(2), GP_w(2), tmp(ndofel,mcrd), ddel(mcrd)
c
       DOUBLE PRECISION Gn, Gt, Tn_m, Tt_m, alph, beta, ln, lt, th,
     & dn, dt, m, n, Gam_n, Gam_t, dGnt, dGtn, DAMAGE, kt1, kt, kn1, kn,
     & N1, N2, del1, del2, del3, del4, deln_max, delt_max,el_length,Tt1,
     & ddel1, ddel2, ddel3, ddel4, ddeln, ddelt, Tn1_max, Tt1_max, Tn1,
     & PARAMn, PARAMt, FATn_NEW, FATt_NEW, DAMAGE1, el_length_un,
     & sin_a_un, cos_a_un, DAMAGE_PR, kn1_pre, kt1_pre
c ----------------------------------------------------------------------
      IF (COHESIVE_INFO(1,1) .EQ. 0) THEN
        IF (KINC .EQ. 1)  THEN
          IF (KSTEP .EQ. 1) THEN
            CALL K_FIND_CONNECT ( )
            CALL K_INIT_MAT_INT(COHESIVE_INFO,KTOTALELC,6)
            DO I = 1, KTOTALELC
              COHESIVE_INFO(I,1) = ELEMENTS_COH(I, 1)
            END DO
          END IF
        END IF
c         WRITE(7,*) 'ELEMENTS_COH', ELEMENTS_COH
c         WRITE(7,*) 'COHESIVE_INFO', COHESIVE_INFO
      END IF
c ----------------------------------------------------------------------
c Read input data & Initialize
      WRITE(7,*) 'STEP', KSTEP
      WRITE(7,*) 'INCREMENT', KINC
      WRITE(7,*) 'ELEMENT', JELEM
      Gn   = PROPS(1)
      Gt   = PROPS(2)
      Tn_m = PROPS(3)
      Tt_m = PROPS(4)
      alph = PROPS(5)
      beta = PROPS(6)
      ln   = PROPS(7)
```

```
      lt   = PROPS(8)
      th   = PROPS(9)
      n_GP = 2
      DATA GP   / 0.577350269189626 , -0.577350269189626 /
      DATA GP_W / 1.0 , 1.0 /
      CALL K_INIT_MAT (RHS,ndofel,nrhs)
      CALL K_INIT_MAT (AMATRX,ndofel,ndofel)
c Determine the PPR parameters
      m = (alph-1)*alph*ln**2/(1-alph*ln**2)
      n = (beta-1)*beta*lt**2/(1-beta*lt**2)
      dn = alph*Gn/(m*Tn_m)*(1-ln)**(alph-1)
     &     * (alph/m*ln+1)**(m-1)*(alph+m)*ln
      dt = beta*Gt/(n*Tt_m)*(1-lt)**(beta-1)
     &     * (beta/n*lt+1)**(n-1)*(beta+n)*lt
      IF (Gt .GT. Gn) THEN
        dGnt = 0
        dGtn = Gt - Gn
      ELSEIF (Gt .LT. Gn) THEN
        dGnt = Gn - Gt
        dGtn = 0
      ELSE
        dGnt = 0
        dGtn = 0
      END IF
c
      IF (Gn .EQ. Gt) THEN
        Gam_n = -Gn*(alph/m)**m
        Gam_t = (beta/n)**n
      ELSE
        Gam_n = (-Gn)**(dGnt/(Gn-Gt))*(alph/m)**m
        Gam_t = (-Gt)**(dGtn/(Gt-Gn))*(beta/n)**n
      END IF
c
      CALL k_Coords_Transform (R, el_length, COORDS, U, ndofel,
     & nnode, mcrd, el_length_un, cos_a_un, sin_a_un)
      DO i = 0, nnode-1
        U_l(1+i*mcrd) = R(1,1)*U(1+i*mcrd) + R(1,2)*U(2+i*mcrd)
        U_l(2+i*mcrd) = R(2,1)*U(1+i*mcrd) + R(2,2)*U(2+i*mcrd)
        DU_l(1+i*mcrd) = R(1,1)*DU(1+i*mcrd,1) + R(1,2)*DU(2+i*mcrd,1)
        DU_l(2+i*mcrd) = R(2,1)*DU(1+i*mcrd,1) + R(2,2)*DU(2+i*mcrd,1)
      END DO
c      WRITE(7,*) 'R', R
c      WRITE(7,*) 'U', U
c      WRITE(7,*) 'U_l', U_l
      del1 = U_l(7) - U_l(1)
      del2 = U_l(8) - U_l(2)
      del3 = U_l(5) - U_l(3)
      del4 = U_l(6) - U_l(4)
      ddel1 = DU_l(7) - DU_l(1)
      ddel2 = DU_l(8) - DU_l(2)
      ddel3 = DU_l(5) - DU_l(3)
      ddel4 = DU_l(6) - DU_l(4)
c
      DO i = 1, n_GP
c        WRITE(7,*) 'INTEGRATION POINT', i
```

```
c         WRITE(7,*) 'n_GP', n_GP
c         WRITE(7,*) 'svars', SVARS(n_GP*(i-1)+2)
c
          N1 = 0.5*(1 - GP(i))
          N2 = 0.5*(1 + GP(i))
          del(1) = N1*del1 + N2*del3
          del(2) = N1*del2 + N2*del4
          ddel(1) = N1*ddel1 + N2*ddel3
          ddel(2) = N1*ddel2 + N2*ddel4
          delt_max = SVARS(n_GP*(i-1)+1)
          deln_max = SVARS(n_GP*(i-1)+2)
c         WRITE(7,*) 'deln_max', deln_max
          kt1      = SVARS((i-1)+5)
          kn1      = SVARS((i-1)+7)
          PARAMn   = SVARS((i-1)+9)
          PARAMt   = SVARS((i-1)+11)
          DAMAGE   = SVARS((i-1)+13)
          kn1_pre  = SVARS((i-1)+15)
          kt1_pre  = SVARS((i-1)+17)
c Initialize values
          IF ((KSTEP .EQ. 1).AND.(KINC .EQ. 1)) THEN
            kt1    = 0.0D0
            kn1    = 0.0D0
            PARAMn = 0.0D0
            PARAMt = 0.0D0
            FATn   = 1.0E-8
            FATt   = 1.0E-8
            kn1_pre= 0.0D0
          END IF
c
          IF ((KINC .EQ. 1)) THEN
            KFLAGSTEP = 1
            DO j = 1, KTOTALELC
              IF (INT(COHESIVE_INFO(j,1)) .EQ. JELEM) THEN
                DAMAGE_PR = COHESIVE_INFO(j, 2)
                WRITE(7,*) 'DAMAGE PREVIOUS STEP', DAMAGE_PR
                DAMAGE = DAMAGE_PR
                GOTO 901
              END IF
            END DO
901   CONTINUE
          END IF
c
          WRITE(7,*) 'FATn arxi',FATn
          WRITE(7,*) 'FATt arxi',FATt
          WRITE(7,*) 'GP',GP
          WRITE(7,*) 'del', del
          WRITE(7,*) 'ddel', ddel
          WRITE(7,*) 'kt1', kt1
          CALL k_Cohesive_PPR (T,T_d,Gam_n,Gam_t,alph,beta,m,n,
     &  dn, dt, dGtn, dGnt, del, deln_max, delt_max, kt1, kn1, ddel,
     &  PARAMn, PARAMt, DAMAGE, Tn_m, Tt_m, kn1_pre, kt1_pre, JELEM)
          ShapeN(1) = -N1
          ShapeN(2) = -N2
          ShapeN(3) = N2
```

```
        ShapeN(4) = N1
        DO j = 1, nnode
          DO k = 1, mcrd
           DO l = 1, mcrd
             Bc(k,l+(j-1)*mcrd) = ShapeN(j)*R(k,l)
           END DO
          END DO
        END DO
        CALL K_TRAN_MAT (Bc,Bct,mcrd,ndofel)
        CALL K_MULTI_MAT (Bct,T_d,tmp,ndofel,mcrd,mcrd)
        CALL K_MULTI_MAT (tmp,Bc,Sc,ndofel,mcrd,ndofel)
        CALL K_MULTI_MAT (Bct,T,Fc,ndofel,mcrd,nrhs)
        thick = 0.5 * el_length * GP_w(i) * th
        CALL K_ADD_MAT (AMATRX,Sc,thick,ndofel,ndofel)
        CALL K_ADD_MAT (RHS,-Fc,thick,ndofel,nrhs)
c
c        WRITE(7,*) 'del ', del
        IF((delt_max.LT.ABS(del(1)))) THEN
c          WRITE(7,*) 'save delt_max'
          SVARS(n_GP*(i-1)+1) = ABS(del(1))
        END IF
        IF ((deln_max .LT. del(2))) THEN
c          WRITE(7,*) 'save deln_max'
          SVARS(n_GP*(i-1)+2) = del(2)
        END IF
        WRITE(7,*) 'KT1, KN1', kt1, kn1
        SVARS((i-1)+5)  = kt1
        SVARS((i-1)+7)  = kn1
        SVARS((i-1)+9)  = PARAMn
        SVARS((i-1)+11) = PARAMt
        SVARS((i-1)+15) = kn1_pre
        SVARS((i-1)+17) = kt1_pre
c        IF (kt1 .LT. kn1) THEN
c          DAMAGE1 = kn1
c        ELSE
c          DAMAGE1 = kt1
c        END IF
        DAMAGE1 = SQRT(kn1*kn1 + kt1*kt1)
        IF ((INT(DAMAGE1) .EQ. 1) .OR. (INT(DAMAGE).EQ.1)) THEN
          SVARS((i-1)+13) = 1.0D0
        ELSE
          SVARS((i-1)+13) = DAMAGE1
        END IF
c
      END DO
c
      dopen1=SQRT(del1*del1+del2*del2)
      dopen2=SQRT(del3*del3+del4*del4)
c     WRITE(7,*) 'dopen', dopen1, dopen2
c     WRITE(7,*) 'del', del2, del4
c
      IF (KINC .EQ. 1) THEN
        DO i = 1, KTOTALELC
          IF (INT(COHESIVE_INFO(i,1)) .EQ. JELEM) THEN
            IF (dopen1 .GE. dopen2) THEN
```

```
                COHESIVE_INFO(i, 4) = cos_a_un
                COHESIVE_INFO(i, 5) = sin_a_un
                COHESIVE_INFO(i, 6) = 0.0D00
              ELSE
                COHESIVE_INFO(i, 4) = -cos_a_un
                COHESIVE_INFO(i, 5) = -sin_a_un
                COHESIVE_INFO(i, 6) = 0.0D00
              END IF
              GOTO 902
            END IF
          END DO
902   CONTINUE
      END IF
      IF (REAL(TIME(1)) .EQ. 1.0) THEN
c
        DO i = 1, KTOTALELC
c           WRITE(7,*) INT(COHESIVE_INFO(i,1)), JELEM
          IF (INT(COHESIVE_INFO(i,1)) .EQ. JELEM) THEN
            COHESIVE_INFO(i, 2) = 0.5*(SVARS(13)+SVARS(14))
            COHESIVE_INFO(i, 3) = el_length_un
            COHESIVE_INFO(i, 7) = PARAMn
            COHESIVE_INFO(i, 8) = PARAMt
c             WRITE(7,*) 'COHESIVE INFO UEL', COHESIVE_INFO
            GOTO 1001
          END IF
        END DO
c
      END IF
c
1001   CONTINUE
       RETURN
       END
c
c ######################################################################
       SUBROUTINE k_Cohesive_PPR (T,T_d,Gam_n,Gam_t,alph,beta,m,n,
     & dn, dt, dGtn, dGnt, del, deln_max, delt_max, kt1, kn1, ddel,
     & PARAMn, PARAMt, DAMAGE, Tn_m, Tt_m, kn1_pre, kt1_pre, JELEM)
       USE GLOBAL
       INCLUDE 'ABA_PARAM.INC'
       DIMENSION T(2,1), T_d(2,2), del(2), ddel(2)
       DOUBLE PRECISION Gam_n, Gam_t, alph, beta, m, n, dn, dt, del,kdn,
     & dGtn, dGnt, deln_max, delt_max, Tn, Tt, deln, delt, sign_dt,kdt,
     & kn1, kt1, kn, kt, ddelt, ddeln, Tn1, ddel, Tn1_max, Tt1, Tt1_max,
     & PARAMn, PARAMt, DAMAGE, Tn_m, Tt_m, kn1_pre, kt1_pre
       delt = abs(del(1))
       deln = del(2)
       ddelt = abs(ddel(1))
       ddeln = ddel(2)
       WRITE(7,*) 'ENTER COHESIVE LAW'
c
       IF (del(1) .GE. 0) THEN
         sign_dt = 1
       ELSE
         sign_dt = -1
       END IF
```

```
c
      Tn = 0
c
      IF (deln_max .EQ. 0.D0) THEN
         Tn1     = 0.0D0
         Tn1_max = 0.0D0
      ELSE
      Tn1 = (Gam_t*(1-delt_max/dt)**beta*(delt_max/dt+n/beta)**
     & n+dGtn)*Gam_n/dn*(m*(1-deln_max/dn)**alph*(m/alph+deln_max/dn)**
     & (m-1)  -alph*(1-deln_max/dn)**(alph-1)*(m/alph+deln_max/dn)**m) *
     & (deln/deln_max)*(1-kn1_pre)
      Tn1_max = (Gam_t*(1-delt_max/dt)**beta*(delt_max/dt+n/beta)**
     & n+dGtn)*Gam_n/dn*(m*(1-deln_max/dn)**alph*(m/alph+deln_max/dn)**
     & (m-1)  -alph*(1-deln_max/dn)**(alph-1)*(m/alph+deln_max/dn)**m)*
     & (1-kn1_pre)
      END IF
c
      IF (delt_max .EQ. 0.D0) THEN
         Tt1     = 0.0D0
         Tt1_max = 0.0D0
      ELSE
      Tt1 = (Gam_n*(1-deln_max/dn)**alph*(deln_max/dn+m/alph)**
     & m+dGnt)*Gam_t/dt*(n*(1-delt_max/dt)**beta*(delt_max/dt+n/beta)**
     & (n-1)  -beta*(1-delt_max/dt)**(beta-1)*(delt_max/dt+n/beta)**n) *
     & (delt/delt_max)*(1-kt1_pre)
      Tt1_max = (Gam_n*(1-deln_max/dn)**alph*(deln_max/dn+m/alph)**
     & m+dGnt)*Gam_t/dt*(n*(1-delt_max/dt)**beta*(delt_max/dt+n/beta)**
     & (n-1)  -beta*(1-delt_max/dt)**(beta-1)*(delt_max/dt+n/beta)**n)*
     & (1-kt1_pre)
      END IF
c
      CALL k_cyclic_damage(kn1, kt1, deln, dn, ddeln, delt, dt,
     & ddelt, Tn1, Tn1_max, Tt1, Tt1_max, deln_max, delt_max,
     & PARAMn, PARAMt, Tn_m, Tt_m, kn1_pre, kt1_pre, JELEM)
c
      IF (deln .LT. 0) THEN
        deln = 0
      ELSEIF ((deln.GE.dn).OR.(delt.GE.dt)) THEN
        Tn = 0
        WRITE(7,*) 'Tn', Tn
      ELSEIF (deln .GE. deln_max) THEN
        Tn = (Gam_t*(1-delt/dt)**beta*(delt/dt+n/beta)**n+dGtn) *
     & Gam_n/dn*(m*(1-deln/dn)**alph*(m/alph+deln/dn)**(m-1)
     &       -alph*(1-deln/dn)**(alph-1)*(m/alph+deln/dn)**m)*(1-kn1)
      ELSE
        Tn=(Gam_t*(1-delt_max/dt)**beta*(delt_max/dt+n/beta)**n+dGtn)*
     & Gam_n/dn*(m*(1-deln_max/dn)**alph*(m/alph+deln_max/dn)**(m-1)
     &   -alph*(1-deln_max/dn)**(alph-1)*(m/alph+deln_max/dn)**m) *
     & deln/deln_max *(1-kn1)
        WRITE(7,*) 'Tn', Tn
      END IF
c
      IF ((deln .GE. dn).OR.(delt .GE. dt))  THEN
        Tt = 0
        WRITE(7,*) 'Tt', Tt
```

```
      ELSEIF (delt .GE. delt_max) THEN
        Tt = (Gam_n*(1-deln/dn)**alph*(deln/dn+m/alph)**m+dGnt) *
     & Gam_t/dt*(n*(1-delt/dt)**beta*(delt/dt+n/beta)**(n-1)
     &       -beta*(1-delt/dt)**(beta-1)*(delt/dt+n/beta)**n) *(1-kt1)
      ELSE
        Tt=(Gam_n*(1-deln_max/dn)**alph*(deln_max/dn+m/alph)**m+dGnt)*
     & Gam_t/dt*(n*(1-delt_max/dt)**beta*(delt_max/dt+n/beta)**(n-1)
     &   -beta*(1-delt_max/dt)**(beta-1)*(delt_max/dt+n/beta)**n) *
     & delt/delt_max *(1-kt1)
        WRITE(7,*) 'Tt', Tt
      END IF
c
      IF (del(2). LT. 0) THEN
        T_d(2,2) = -Gam_n/dn**2*(m/alph)**(m-1)*(alph+m)*
     & (Gam_t*(n/beta)**n + dGtn)
        T_d(2,1) = 0
        T(2,1) = T_d(2,2)*del(2)
      ELSE IF ((deln.LT.dn).AND.(delt.LT.dt).AND.(Tn.GE.-1.0E-5)) THEN
        T(2,1) = Tn
c
        IF (deln .GE. deln_max) THEN
          T_d(2,2) = (1-kn1)*
     & (Gam_t*(1-delt/dt)**beta*(delt/dt+n/beta)**n+dGtn) *
     & Gam_n/dn**2 *
     & ((1-deln/dn)**(alph-2)*(alph-1)*alph*(deln/dn+m/alph)**m -
     & 2*(1-deln/dn)**(alph-1)*alph*(deln/dn+m/alph)**(m-1)*m +
     & (1-deln/dn)**alph*(deln/dn+m/alph)**(m-2)*(m-1)*m)
          T_d(2,1) =(1-kn1)*
     & Gam_t/dt*(-(1-delt/dt)**(beta-1)*beta*(delt/dt+n/beta)**n +
     & (1-delt/dt)**beta*(delt/dt+n/beta)**(n-1)*n) * sign_dt *
     & Gam_n/dn*(-(1-deln/dn)**(alph-1)*alph*(deln/dn+m/alph)**m +
     & (1-deln/dn)**alph*(deln/dn+m/alph)**(m-1)*m)
c   (3) Unloading/reloading condition
        ELSE
          T_d(2,2) =(1-kn1)*
     & (Gam_t*(1-delt_max/dt)**beta*(delt_max/dt+n/beta)**n+dGtn) *
     & Gam_n/dn*((1-deln_max/dn)**alph*(deln_max/dn+m/alph)**(m-1)*m
     & -(1-deln_max/dn)**(alph-1)*alph*(deln_max/dn+m/alph)**m)
     & / deln_max
          T_d(2,1) =(1-kn1)*
     & Gam_t/dt*(-(1-delt_max/dt)**(beta-1)*beta*(delt_max/dt+n/beta)**n
     & +(1-delt_max/dt)**beta*(delt_max/dt+n/beta)**(n-1)*n) * sign_dt *
     & Gam_n/dn*(m*(1-deln_max/dn)**alph*(m/alph+deln_max/dn)**(m-1)
     &   -alph*(1-deln_max/dn)**(alph-1)*(m/alph+deln_max/dn)**m) *
     & deln/deln_max
        END IF
c
      ELSE
        T(2,1) = 0
        T_d(2,2) = 0
        T_d(2,1) = 0
      END IF
c
      IF ((delt.LT.dt) .AND. (deln.LT.dn) .AND. (Tt.GE.-1.0E-5)) THEN
        T(1,1) = Tt*sign_dt
```

```
c
        IF (delt .GE. delt_max) THEN
          T_d(1,1) =(1-kt1)*
     & (Gam_n*(1-deln/dn)**alph*(deln/dn+m/alph)**m+dGnt) *
     & Gam_t/dt**2 *
     & ((1-delt/dt)**(beta-2)*(beta-1)*beta*(delt/dt+n/beta)**n -
     & 2*(1-delt/dt)**(beta-1)*beta*(delt/dt+n/beta)**(n-1)*n +
     & (1-delt/dt)**beta*(delt/dt+n/beta)**(n-2)*(n-1)*n)
          T_d(1,2) =(1-kt1)*
     & Gam_t/dt*(-(1-delt/dt)**(beta-1)*beta*(delt/dt+n/beta)**n +
     & (1-delt/dt)**beta*(delt/dt+n/beta)**(n-1)*n) * sign_dt *
     & Gam_n/dn*(-(1-deln/dn)**(alph-1)*alph*(deln/dn+m/alph)**m +
     & (1-deln/dn)**alph*(deln/dn+m/alph)**(m-1)*m)
c
        ELSE
          T_d(1,1) =(1-kt1)*
     & (Gam_n*(1-deln_max/dn)**alph*(deln_max/dn+m/alph)**m+dGnt) *
     & Gam_t/dt*(n*(1-delt_max/dt)**beta*(delt_max/dt+n/beta)**(n-1)
     &      -beta*(1-delt_max/dt)**(beta-1)*(delt_max/dt+n/beta)**n)
     & / delt_max
          T_d(1,2) =(1-kt1)*
     & Gam_n/dn*(-(1-deln_max/dn)**(alph-1)*alph*(deln_max/dn+m/alph)**m
     & +(1-deln_max/dn)**alph*(deln_max/dn+m/alph)**(m-1)*m) * sign_dt *
     & Gam_t/dt*(n*(1-delt_max/dt)**beta*(delt_max/dt+n/beta)**(n-1)
     &   -beta*(1-delt_max/dt)**(beta-1)*(delt_max/dt+n/beta)**n) *
     & delt/delt_max
        END IF
c
      ELSE
        T(1,1) = 0
        T_d(1,1) = 0
        T_d(1,2) = 0
      END IF
c
      IF (INT(DAMAGE).EQ.1) THEN
        T(2,1) = 0
        T(1,1) = 0
        T_d(1,1) = 0
        T_d(1,2) = 0
        T_d(2,2) = 0
        T_d(2,1) = 0
c       WRITE(7,*) 'COMPLETE FAILURE'
c       WRITE(7,*) 'Td',T_d
c       WRITE(7,*) 'T',T_
      END IF
c
      RETURN
      END
c ######################################################################
      SUBROUTINE k_calibrate_fat(FAT_NEW, PARAM, FAT)
      USE GLOBAL
      INCLUDE 'ABA_PARAM.INC'
      DOUBLE PRECISION FAT_NEW, PARAM, DAM, DDAM, FACTOR
c
      WRITE(7,*) 'ENTER CALIBRATE FAT'
```

```
      WRITE(7,*) 'N_CYCLES', N_CYCLES
      WRITE(7,*) 'DAMAGE_CALIB', DAMAGE_CALIB
      FAT_NEW  = 0.00001*FAT
      WRITE(7,*) 'FAT', FAT
      WRITE(7,*) 'FAT_NEW', FAT_NEW
      N_ITER = 0
      WRITE(7,*) 'PARAM', PARAM
1001   CONTINUE
c
      DAM    = 0.0D0
      IF (INT(NUM_CYCLES) .EQ. 0) THEN
         J = 1
      ELSE
         J = INT(NUM_CYCLES*0.5D0)
      END IF
      DO I = 1, J
c
        DDAM = FAT_NEW*PARAM
        DAM = DAM + DDAM
      END DO
      WRITE(7,*) 'DDAM', DDAM
      WRITE(7,*) 'DAM', DAM
c
      FACTOR = DAMAGE_CALIB / DAM
      WRITE(7,*) 'FACTOR', FACTOR
     IF ((ABS(DAMAGE_CALIB-DAM)>FAT_TOLERANCE).AND.(N_ITER<100*J)) THEN
         FAT_NEW = FACTOR * FAT_NEW
         N_ITER = N_ITER + 1
         WRITE(7,*) 'FAT_N', FAT_NEW
         GOTO 1001
      END IF
     IF (PARAM .LT. 10E-10) THEN
        FAT_NEW = FAT
     END IF
      WRITE(7,*) 'FAT_NEW', FAT_NEW
c
      RETURN
      END
c ######################################################################
      SUBROUTINE k_cyclic_damage(kn1, kt1, deln, dn, ddeln, delt, dt,
     & ddelt, Tn1, Tn1_max, Tt1, Tt1_max, deln_max, delt_max, PARAMn,
     & PARAMt, Tn_m, Tt_m, kn1_pre, kt1_pre, JELEM)
      USE GLOBAL
      INCLUDE 'ABA_PARAM.INC'
      DOUBLE PRECISION  kn1, kt1, kt, kn, deln, dn, ddeln, delt, dt,
     & ddelt, Tn1, Tn1_max, Tt1, Tt1_max, deln_max, delt_max, PARAMn,
     & PARAMt, Tn_m, Tt_m, kn1_pre, kt1_pre
c
      WRITE(7,*) 'ENTER CYCLIC DAMAGE'
      WRITE(7,*) 'kt1 start ', kt1
      WRITE(7,*) 'kn1 start ', kn1
      WRITE(7,*) 'FATn ', FATn
      WRITE(7,*) 'FATt ', FATt
      WRITE(7,*) 'deln ', deln
      WRITE(7,*) 'deln_max ', deln_max
```

```
      WRITE(7,*) 'ddeln ', ddeln
      WRITE(7,*) 'delt ', delt
      WRITE(7,*) 'delt_max ', delt_max
      WRITE(7,*) 'ddelt ', ddelt
      WRITE(7,*) 'Tn1 ', Tn1
      WRITE(7,*) 'Tn1_max ', Tn1_max
      WRITE(7,*) 'Tt1 ', Tt1
      WRITE(7,*) 'Tt1_max ', Tt1_max
      WRITE(7,*) 'BE ', BE
      WRITE(7,*) 'Tn1-BE*Tn1_max ', Tn1-BE*Tn1_max
      WRITE(7,*) 'Tt1-BE*Tt1_max ', Tt1-BE*Tt1_max
      kt   = kt1
      kn   = kn1
      dkt1 = 0.0D0
      dkn1 = 0.0D0
      kn1_pre = kn1_pre
      kt1_pre = kt1_pre
c
      IF (deln .GE. deln_max) THEN
        WRITE(7,*) 'Monotonic loading No fatigue Damage normal'
      ELSE
        IF ((ddeln > 0) .AND. (Tn1-BE*Tn1_max)>0) THEN
          WRITE(7,*) 'Reloading fatigue Damage normal'
c
          dkn1  = ddeln*FATn*(Tn1_max/deln_max)*(dn/Tn_m)
          WRITE(7,*) 'PARAMn', ddeln*(Tn1_max/deln_max)*(dn/Tn_m)
          WRITE(7,*) 'dkn1', dkn1
          kn1 = kn + dkn1
          PARAMn = PARAMn + ddeln*(Tn1_max/deln_max)*(dn/Tn_m)
          WRITE(7,*) 'dkn1, kn1, PARAMn', dkn1, kn1, PARAMn
        ELSE
          WRITE(7,*) 'Unloading No fatigue Damage normal'
          kn1_pre = kn1
        END IF
      END IF
      IF (kn1 .GE. 1.0) THEN
        kn1 = 1.0
        kt1 = 1.0
      END IF
      WRITE(7,*) 'kn1 end', kn1
c
      IF (delt .GE. delt_max) THEN
        WRITE(7,*) 'Monotonic loading No fatigue Damage shear'
      ELSE
        IF ((ddelt > 0) .AND. (Tt1-BE*Tt1_max)>0) THEN
          WRITE(7,*) 'Reloading fatigue Damage shear'
c           kt1 = (kt + (ddelt*FAT*(Tt1-BE*Tt1_max))/Tt1_max)/
c     & (1 +(ddelt*FAT*(Tt1-BE*Tt1_max))/Tt1_max)
c           dkt1  =(ddelt*FATt*(Tt1-beta*Tt1_max)*(1-kt)/Tt1_max)/
c     & (1+ ddelt*FATt*(Tt1-beta*Tt1_max)/Tt1_max)
          dkt1  = ddelt*FATt*(Tt1_max/delt_max)*(dt/Tt_m)
          WRITE(7,*) 'dkt1', dkt1
          kt1 = kt + dkt1
c           PARAMt = PARAMt + ddelt*(Tt1-beta*Tt1_max)/Tt1_max
          PARAMt = PARAMt + ddelt*(Tt1_max/delt_max)*(dt/Tt_m)
```

```
        ELSE
          WRITE(7,*) 'Unloading No fatigue Damage shear'
          kt1_pre = kt1
        END IF
      END IF
c
      IF (kt1 .GE. 1.0) THEN
        kt1 = 1.0
        kn1 = 1.0
      END IF
      IF (kn1 .GE. 1.0) THEN
        kt1 = 1.0
        kn1 = 1.0
      END IF
c
      IF (JELEM.EQ.100042) THEN
      WRITE(7,*) 'kt1 end', kt1
      WRITE(7,*) 'kn1 end', kn1
      END IF
c
      RETURN
      END
```

11.4 Algorithm for bulk element calculations

```
c ######################################################################
      SUBROUTINE K_UEL (Bc, XI, ETA, I, STRESS, STRAN)
      USE GLOBAL
c
      INCLUDE 'ABA_PARAM.INC'
c
      DIMENSION Cc(3,3), Bc(3,8), B1(3,4), B2(4,8), STRESS(3), STRAN(3)
c
      DOUBLE PRECISION XI, ETA, B1, B2, Bc, STRESS, STRAN
c
      INTEGER I
c
c     WRITE(7,*) '######## START K_UEL ##########'
c
      XI        = 0.0D0
      ETA       = 0.0D0
      JACOB(1,1) = 0.0D0
      JACOB(1,2) = 0.0D0
      JACOB(2,2) = 0.0D0
      JACOB(2,1) = 0.0D0
      JACOBDET   = 0.0D0
      CALL K_INIT_MAT(Bc,3,8)
      CALL K_INIT_MAT(B1,3,4)
      CALL K_INIT_MAT(B2,4,8)
      CALL K_INIT_MAT(Cc,3,3)
c *******************************************************
c Material Stifness matrix
      Cc(1,1)=(Ee*(1.0D0 -ni))/((1.0D0 - 2.0D0*ni)*(1.0D0 + ni))
      Cc(1,2)=(Ee*ni)/((1.0D0 - 2.0D0*ni)*(1.0D0 + ni))
      Cc(1,3)=0.0D0
      Cc(2,1)=(Ee*ni)/((1.0D0 - 2.0D0*ni)*(1.0D0 + ni))
      Cc(2,2)=(Ee*(1.0D0 - ni))/((1.0D0 - 2.0D0*ni)*(1.0D0 + ni))
      Cc(2,3)=0.0D0
      Cc(3,1)=0.0D0
      Cc(3,2)=0.0D0
      Cc(3,3)=Ee/(2.0D0*(1.0D0 + ni))
c *******************************************************
c Jacobian
c     WRITE(7,*) '########## START K_JACOBIAN #########'
c *******************************************************
c Derivatives of actual coordinates with respect to physical coordinates
      !dxdxi
      JACOB(1,1) = -(0.25D0)*(1-ETA)*XTR(1,1)+(0.25D0)*(1-ETA)*XTR(1,3)
     & +(0.25D0)*(1+ETA)*XTR(1,5)-(0.25D0)*(1+ETA)*XTR(1,7)
      !dydxi
      JACOB(1,2) = -(0.25D0)*(1-ETA)*XTR(1,2)+(0.25D0)*(1-ETA)*XTR(1,4)
     & +(0.25D0)*(1+ETA)*XTR(1,6)-(0.25D0)*(1+ETA)*XTR(1,8)
      !dxdeta
      JACOB(2,1) = -(0.25D0)*(1-XI)*XTR(1,1)-(0.25D0)*(1+XI)*XTR(1,3)
     & +(0.25D0)*(1+XI)*XTR(1,5)+(0.25D0)*(1-XI)*XTR(1,7)
      !dydeta
```

```
      JACOB(2,2) = -(0.25D0)*(1-XI)*XTR(1,2)-(0.25D0)*(1+XI)*XTR(1,4)
     & +(0.25D0)*(1+XI)*XTR(1,6)+(0.25D0)*(1-XI)*XTR(1,8)
c *********************************************************
c Determinant of Jacobian matrix
      JACOBDET = JACOB(1,1)*JACOB(2,2) - JACOB(1,2)*JACOB(2,1)
c       WRITE(7,*) 'JACOBDET = ', JACOBDET
c       WRITE(7,*) 'JACOB = ', JACOB
c       WRITE(7,*) '########## END K_JACOBIAN #########'
c *********************************************************
c Stain transformation from physical to actual coordinates
      B1(1,1)=  JACOB(2,2)/JACOBDET
      B1(1,2)= -JACOB(1,2)/JACOBDET
      B1(1,3)= 0.0D0
      B1(1,4)= 0.0D0
      B1(2,1)= 0.0D0
      B1(2,2)= 0.0D0
      B1(2,3)= -JACOB(2,1)/JACOBDET
      B1(2,4)=  JACOB(1,1)/JACOBDET
      B1(3,1)= -JACOB(2,1)/JACOBDET
      B1(3,2)=  JACOB(1,1)/JACOBDET
      B1(3,3)=  JACOB(2,2)/JACOBDET
      B1(3,4)= -JACOB(1,2)/JACOBDET
c     WRITE(7,*) 'B1 = ', B1
c *********************************************************
c Matrix B2 for strain calculation at physical coordinates
      B2(1,1) = (0.25D0)*(-1.0D0 + ETA) !dN1dxi
      B2(1,2) = 0.0D0
      B2(1,3) = (0.25D0)*( 1.0D0 - ETA) !dN2dxi
      B2(1,4) = 0.0D0
      B2(1,5) = (0.25D0)*( 1.0D0 + ETA) !dN3dxi
      B2(1,6) = 0.0D0
      B2(1,7) = (0.25D0)*(-1.0D0 - ETA) !dN4dxi
      B2(1,8) = 0.0D0
      B2(2,1) = (0.25D0)*(-1.0D0 + XI ) !dN1deta
      B2(2,2) = 0.0D0
      B2(2,3) = (0.25D0)*(-1.0D0 - XI ) !dN2deta
      B2(2,4) = 0.0D0
      B2(2,5) = (0.25D0)*( 1.0D0 + XI ) !dN3deta
      B2(2,6) = 0.0D0
      B2(2,7) = (0.25D0)*( 1.0D0 - XI ) !dN4deta
      B2(2,8) = 0.0D0
      B2(3,1) = 0.0D0
      B2(3,2) = (0.25D0)*(-1.0D0 + ETA) !dN1dxi
      B2(3,3) = 0.0D0
      B2(3,4) = (0.25D0)*( 1.0D0 - ETA) !dN2dxi
      B2(3,5) = 0.0D0
      B2(3,6) = (0.25D0)*( 1.0D0 + ETA) !dN3dxi
      B2(3,7) = 0.0D0
      B2(3,8) = (0.25D0)*(-1.0D0 - ETA) !dN4dxi
      B2(4,1) = 0.0D0
      B2(4,2) = (0.25D0)*(-1.0D0 + XI ) !dN1deta
      B2(4,3) = 0.0D0
      B2(4,4) = (0.25D0)*(-1.0D0 - XI ) !dN2deta
      B2(4,5) = 0.0D0
      B2(4,6) = (0.25D0)*( 1.0D0 + XI ) !dN3deta
```

```
      B2(4,7) = 0.0D0
      B2(4,8) = (0.25D0)*( 1.0D0 - XI ) !dN4deta
c     WRITE(7,*) 'B2 = ', B2
c *******************************************************
c Calculation of matrix B
      CALL K_MULTI_MAT (B1,B2,Bc,3,4,8)
c     WRITE(7,*) 'Bc =', Bc
c *******************************************************
c Calculation of Strains
      DO ki=1,3
        STRAN(ki)=0.0D0
      END DO
c
      DO ki=1,3
        DO kj=1,KNDOFEL
          STRAN(ki)=STRAN(ki)+UTR(1,kj)*Bc(ki,kj)
        END DO
      END DO
c *******************************************************
c Calculation of Stresses
      DO ki=1,3
        STRESS(ki)=0.0D0
      END DO
c
      DO ki=1,3
        DO kj=1,3
          STRESS(ki)=STRESS(ki)+Cc(ki,kj)*STRAN(kj)
        END DO
      END DO
c *******************************************************
c      WRITE(7,*) '######## END K_UEL ##########'
      RETURN
     END
c ************************************************************************
```

11.5 Algorithm for finding q values of J-integral

```
c ######################################################################
      SUBROUTINE K_Q_FACTOR (Q, I, PATH)
      USE GLOBAL
c
      INCLUDE 'ABA_PARAM.INC'
      DIMENSION Q(4,1), VE1(2), VE2(2), VEL1(2), VEL2(2), DIRE(1,4)
      DOUBLE PRECISION Q, CENTERX, CENTERY, VE1, VE2, COSTH,
     & VLENGTH1, VLENGTH2, CENTERX1, CENTERY1, CENTERX2, CENTERY2,
     & SIDE1, SIDE2
      INTEGER NEIB1, NEIB2, I1, I2
c
      CALL K_FIND_NEIBO (I, NEIB1, NEIB2, PATH, I1, I2)
      CENTERX  = 0.0D0
      CENTERY  = 0.0D0
      CENTERX1 = 0.0D0
      CENTERY1 = 0.0D0
      CENTERX2 = 0.0D0
      CENTERY2 = 0.0D0
      CALL K_INIT_MAT(Q,4,1)
      IF (NEIB1 .EQ. 0) THEN
         I1 = I2
      END IF
      IF (NEIB2 .EQ. 0) THEN
         I2 = I1
      END IF
      CALL K_FIND_ELEM_DIR (I, I1, I2, DIRE)
c
      I1 = DIRE(1,1)
      I  = DIRE(1,2)
      I2 = DIRE(1,3)
c
      DO J=1, 8, 2
         CENTERX  = CENTERX + X(I,J)
         CENTERY  = CENTERY + X(I,J+1)
         CENTERX1 = CENTERX1 + X(I1,J)
         CENTERY1 = CENTERY1 + X(I1,J+1)
         CENTERX2 = CENTERX2 + X(I2,J)
         CENTERY2 = CENTERY2 + X(I2,J+1)
      END DO
c
      CENTERX  = 0.25D0*CENTERX
      CENTERY  = 0.25D0*CENTERY
      CENTERX1 = 0.25D0*CENTERX1
      CENTERY1 = 0.25D0*CENTERY1
      CENTERX2 = 0.25D0*CENTERX2
      CENTERY2 = 0.25D0*CENTERY2
c
c     WRITE(7,*) 'I', I1,I,I2
      DO J = 1, 8, 2
        SIDE1 = (CENTERX1-CENTERX)*(X(I,J+1)-CENTERY)-(X(I,J)-CENTERX)*(CENTERY1-CENTERY)
        SIDE2 = (CENTERX-CENTERX2)*(X(I,J+1)-CENTERY2)-(X(I,J)-CENTERX2)*(CENTERY-CENTERY2)
```

```
c       WRITE(7,*) 'SIDE', SIDE1, SIDE2
        K = INT(0.5*(J+1))
        IF ((SIDE1 .GT. 0.0D0) .AND. (SIDE2 .GT. 0.0D0))  THEN
          Q(K,1) = 1.0D0
        ELSE
          Q(K,1) = 0.0D0
        END IF
c
      END DO
c
      RETURN
      END
c ######################################################################
      SUBROUTINE K_FIND_ELEM_DIR (I, I1, I2, DIRE)
      USE GLOBAL
c     ORDER THE POINTS CLOCKWISE, USING POLAR ANGLE OF EACH POINT WITH RESPECT TO A REFERENCE POINT
      INCLUDE 'ABA_PARAM.INC'
      DIMENSION DIRE(1,3)
      DOUBLE PRECISION CENTERX, CENTERY, CENTERX1, CENTERY1, CENTERX2,
     & CENTERY2, COSTH, TH
      INTEGER I, I1, I2
c
c     WRITE(7,*) 'ELEM_DIR I' , I1, I, I2
      DO J=1, 8, 2
        CENTERX  = CENTERX + X(I,J)
        CENTERY  = CENTERY + X(I,J+1)
        CENTERX1 = CENTERX1 + X(I1,J)
        CENTERY1 = CENTERY1 + X(I1,J+1)
        CENTERX2 = CENTERX2 + X(I2,J)
        CENTERY2 = CENTERY2 + X(I2,J+1)
      END DO
c
      CENTERX  = 0.25D0*CENTERX
      CENTERY  = 0.25D0*CENTERY
      CENTERX1 = 0.25D0*CENTERX1
      CENTERY1 = 0.25D0*CENTERY1
      CENTERX2 = 0.25D0*CENTERX2
      CENTERY2 = 0.25D0*CENTERY2
c
      TH  = ATAN2(CENTERY  - CRACK(2), CENTERX  - CRACK(1))
      TH1 = ATAN2(CENTERY1 - CRACK(2), CENTERX1 - CRACK(1))
      TH2 = ATAN2(CENTERY2 - CRACK(2), CENTERX2 - CRACK(1))
c
c     WRITE(7,*) 'TH', TH, TH1, TH2
      IF (TH .LT. 0.0D0) THEN
        IF (ABS(REAL(TH1)) .LT. ABS(REAL(TH2))) THEN
          DIRE(1,1) = I1
          DIRE(1,2) = I
          DIRE(1,3) = I2
        ELSE
          DIRE(1,1) = I2
          DIRE(1,2) = I
          DIRE(1,3) = I1
        END IF
      ELSE
```

```
        IF (ABS(REAL(TH1)) .GT. ABS(REAL(TH2))) THEN
          DIRE(1,1) = I1
          DIRE(1,2) = I
          DIRE(1,3) = I2
        ELSE
          DIRE(1,1) = I2
          DIRE(1,2) = I
          DIRE(1,3) = I1
        END IF
      END IF
c
      RETURN
      END
c ######################################################################
      SUBROUTINE K_FIND_NEIBO (ELEM, NEIB1, NEIB2, PATH, I1, I2)
      USE GLOBAL
      INTEGER ELEM, NEIB1, NEIB2, I, J, FLAG, PATH, M, COMM,
     & I1, I2
c find neighbors of an element
      FLAG  = 1
      NEIB1 = 0
      NEIB2 = 0
      DO I = 1, SIZE_PATH
        DO M = 1, NUM_ELEM
          IF (EL_PATHS(PATH, I) .EQ. ELEMENTS_NOT_COH(M,1)) THEN
            IF (ELEM .NE. M) THEN
            COMM = 0
            DO J = 2, KXNODEL + 1
              DO K = 2, KXNODEL + 1
                IF (ELEMENTS_NOT_COH(ELEM,J) .EQ. ELEMENTS_NOT_COH(M, K)) THEN
                  COMM = COMM + 1
c                  WRITE(7,*) 'COMM', COMM
c                  WRITE(7,*) 'ELEMENTS(M, 1)', ELEMENTS(M,1)
                  IF (COMM .EQ. 2) THEN
                    IF (FLAG .EQ. 1)  THEN
                      NEIB1 = ELEMENTS(M,1)
                      I1    = M
                      FLAG = FLAG + 1
                    ELSEIF (FLAG .EQ. 2) THEN
                      NEIB2 = ELEMENTS(M,1)
                      I2    = M
                    END IF
                    GOTO 610
                  END IF
                END IF
              END DO
            END DO
            END IF
          END IF
        END DO
610     CONTINUE
      END DO
c     WRITE(7,*) 'NEIB1, ELEM, NEIB2', NEIB1, ELEMENTS(ELEM,1), NEIB2
c
      RETURN
```

11.6 Algorithm for finding paths for J-ntegral

```
 ######################################################################
      SUBROUTINE K_FIND_PATH ( )
      USE GLOBAL
      DIMENSION EL_NODES(1,4), CANDI_ELEM(1,16),
     & CANDI_ELEM_2(1,SIZE_PATH)
      INTEGER EL_NODES, CANDI_ELEM, SEARCH_ELEMENT,i , j,
     & CANDI_ELEM_2, COUNT
c
      CALL K_INIT_MAT_INT(EL_PATHS,N_PATHS,SIZE_PATH)
c find first path
      DO i = 1, KTOTALNOD
        IF (NODE_TO_EL(i,1) .EQ. TIP_ID) THEN
            EL_PATHS(1,1) = NODE_TO_EL(i,2)
            EL_PATHS(1,2) = NODE_TO_EL(i,3)
            EL_PATHS(1,3) = NODE_TO_EL(i,4)
            EL_PATHS(1,4) = NODE_TO_EL(i,5)
        END IF
      END DO
c     WRITE(7,*) 'EL_PATHS', EL_PATHS
c find rest paths
      DO i = 1, N_PATHS - 1
        COUNT = 0
c        WRITE(7,*) 'i', i
c        WRITE(7,*) 'EL_PATHS', EL_PATHS(i,:)
        DO j = 1, SIZE_PATH
          IF (EL_PATHS(i,j) .NE. 0) THEN
            SEARCH_ELEMENT = EL_PATHS(i,j)
            CALL K_FIND_NODES_FR_ELEM (SEARCH_ELEMENT, EL_NODES)
            CALL K_FIND_ELEM_FR_NODES (EL_NODES, CANDI_ELEM)
            CALL K_ADD_ELEM_TO_PATHS (i, j, CANDI_ELEM, COUNT)
            COUNT = COUNT - 1
          END IF
        END DO
      END DO
c     WRITE(7,*) 'i', N_PATHS
c     WRITE(7,*) 'EL_PATHS', EL_PATHS(N_PATHS,:)
c
      RETURN
      END
c ######################################################################
      SUBROUTINE K_ADD_ELEM_TO_PATHS (i, j, CANDI_ELEM, COUNT)
      USE GLOBAL
      DIMENSION CANDI_ELEM(1,16)
      INTEGER FLAG, i, j, CANDI_ELEM, COUNT, k, l, m
c remove elements that have already be added to paths
      DO k = 1, 16
        FLAG = 1
        DO l = 1, N_PATHS
          DO m = 1, SIZE_PATH
            IF (CANDI_ELEM(1,k) .EQ. EL_PATHS(l,m)) THEN
              FLAG = 0
            END IF
          END DO
```

```
        END DO
        IF (FLAG .EQ. 1) THEN
          EL_PATHS(i + 1,j + COUNT) = CANDI_ELEM(1,k)
          COUNT = COUNT + 1
        END IF
      END DO
c
      RETURN
      END
c ######################################################################
      SUBROUTINE K_FIND_NODES_FR_ELEM (ELEMENT_ID, EL_NODES)
      USE GLOBAL
      DIMENSION EL_NODES(1,4)
      INTEGER EL_NODES, ELEMENT_ID, k
c
      CALL K_INIT_MAT_INT(EL_NODES,1,4)
c
      DO k = 1, KTOTALEL
        IF (ELEMENTS_NOT_COH(k,1) .EQ. ELEMENT_ID) THEN
          EL_NODES(1,1) = ELEMENTS_NOT_COH(k,2)
          EL_NODES(1,2) = ELEMENTS_NOT_COH(k,3)
          EL_NODES(1,3) = ELEMENTS_NOT_COH(k,4)
          EL_NODES(1,4) = ELEMENTS_NOT_COH(k,5)
        END IF
      END DO
c
c     WRITE(7,*) 'ELEMENT_ID', ELEMENT_ID
c     WRITE(7,*) 'EL_NODES', EL_NODES
      RETURN
      END
c ######################################################################
      SUBROUTINE K_FIND_ELEM_FR_NODES (EL_NODES, CANDI_ELEM)
      USE GLOBAL
      DIMENSION CANDI_ELEM(1,16), EL_NODES(1,4)
      INTEGER CANDI_ELEM, m, l, EL_NODES, k
c
      CALL K_INIT_MAT_INT(CANDI_ELEM,1,16)
      k = 0
c
      DO l = 1, 4
        DO m = 1, KTOTALNOD
          IF (NODE_TO_EL(m,1) .EQ. EL_NODES(1,l)) THEN
            CANDI_ELEM(1,k + 1) = NODE_TO_EL(m,2)
            CANDI_ELEM(1,k + 2) = NODE_TO_EL(m,3)
            CANDI_ELEM(1,k + 3) = NODE_TO_EL(m,4)
            CANDI_ELEM(1,k + 4) = NODE_TO_EL(m,5)
            k = k + 4
          END IF
        END DO
      END DO
c     WRITE(7,*) 'CANDI_ELEM', CANDI_ELEM
c
      RETURN
      END
c ######################################################################
```