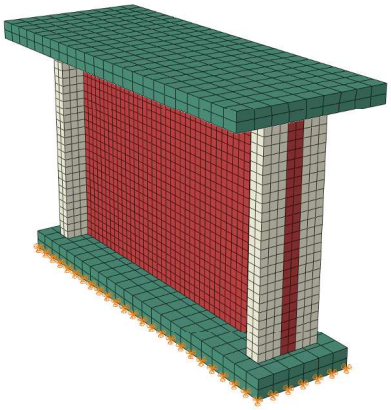
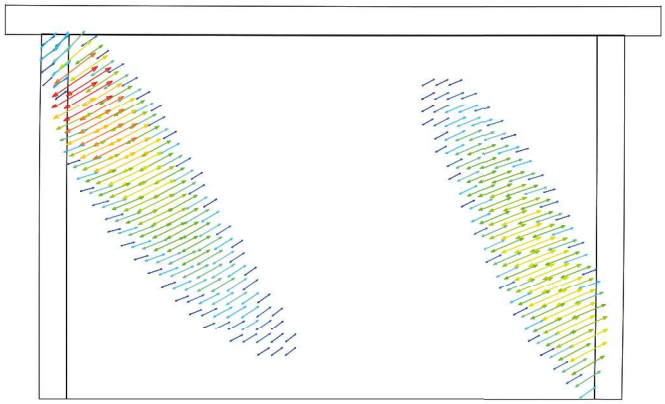
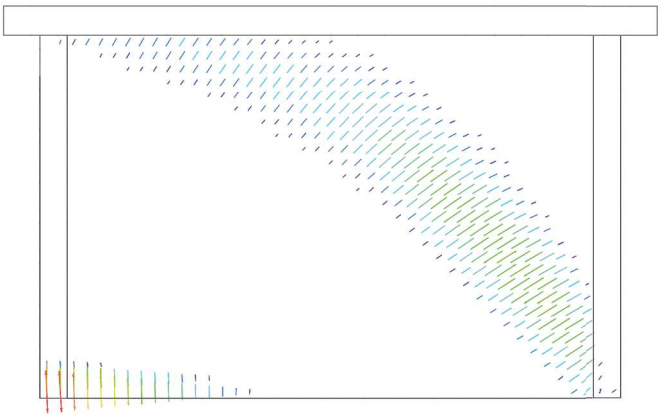
Homogenious Orthotropic Masonry Material Model

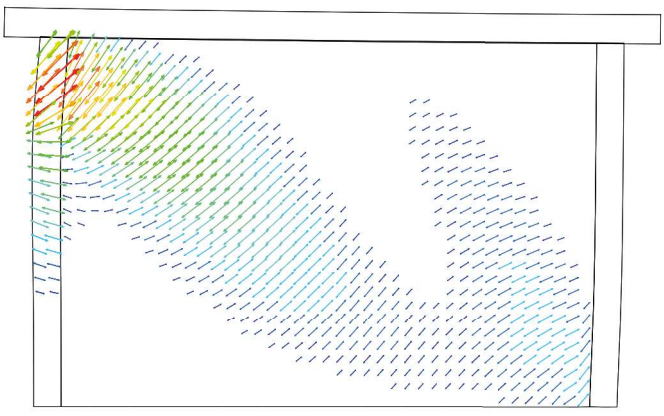
Research, development and implementation for explicit analysis

Aivaras Aukselis









TU Delft Delft University of Technology

zonneveld ingenieurs®

Homogeneous Orthotropic Masonry Material Model

Research, development and implementation for explicit analysis

by

A. Aukselis

to obtain the degree of

Master of Science
in Structural Engineering

at the Delft University of Technology,

to be defended publicly on Wednesday January 18th, 2017 at 15:00.

Student number:	4415779	
Project duration:	March 01, 2016 – December 14, 2016	
Thesis committee:	Prof. dr. ir. J. G. Rots,	TU Delft, Chairman
	Dr. ir. M. A. N. Hendriks,	TU Delft
	Dr. ir. Y. Yang,	TU Delft
	Ing. H. Hoorn,	Zonneveld ingenieurs b.v.
	Ir. A. Koot,	Zonneveld ingenieurs b.v.

TU Delft

zonneveld ingenieurs®

Abstract

The Groningen region, in the Netherlands, experiences earthquakes since 1986. These earthquakes are new to the region and the structures built there are not designed for it. Therefore, they pose a great risk for the historical structures and people living in the area. Most of the buildings there are built using masonry and to analyze such structures on how they will be affected by earthquakes using currently available tools and methods is rather complicated.

In this report, the author investigates the use of available continuum damage mechanics models for the purpose of simulating orthotropic masonry behavior during an explicit integration analysis. The material behavior is described in Fortran programming language and is used as a custom user subroutine (VUMAT) in Simulia Abaqus finite element analysis software. The developed material model exhibits 3D elasticity and 2D plane plasticity. Furthermore, it is assumed that two general failure mechanisms are present. One associated with tensile and shear brittle fracture represented by Rankine type yield surface and other with distributed crushing of a material represented by Hill type yield surface. The model exhibits uncoupled damage evolution in the tension regime and coupled in compression. Additionally, the model supports tensile crack closure, while in compression it accumulates the plastic deformations and if an element is crushed it can be flagged for deletion from the mesh. The model is formulated in such a way that most of the properties in material directions are independent of one another.

The developed model was tested by examining its behavior in analyses where numerical models were composed out of one or few elements. Additionally, for experimental comparison, four shear walls were modeled, three subjected to monotonic loading and one to cyclic. The analyses closely agree to experimental results even when using raw test data. The material model is stable due to the explicit approach and provides qualitative results as it is flexible enough to be used for various types of analyses, either static or cyclic.

In the final part of the report, further developments are considered, including improvements to the code base, additional testing, and development of a custom element.

Acknowledgments

The thesis was carried out as a part of the Civil Engineering program at Delft University of Technology, for the master track of Structural Engineering with a specialization in Concrete Structures. The research and development were carried out in cooperation with Zonneveld ingenieurs B.V. from March 2016 to December 2016.

Here, I would like to express my gratitude, first, to the members of my assessment committee: Max Hendriks for referencing me to Zonneveld ingenieurs B.V. for an internship, that later led to the start of this thesis; Harm Hoorn, for securing me a position in the company that completely reflected my field of interest; Jan Rots, for suggesting to use 2D Plasticity for solid elements instead of trying to define 3D orthotropic plasticity; Yuguang Yang, for the useful end relevant feedback; and Arie Koot, for all the interesting discussions together with the guidance and feedback you provided through my thesis period. All of it helped me to improve the quality of my work. I would further like to thank Mark Slotboom for suggesting the thesis topic during the internship at Zonneveld ingenieurs B.V., without you this thesis would not have commenced.

In addition, I would like to thank all my friends, both the ones in the Netherlands and back home, who directly or indirectly helped me with my thesis and kept me company during my master studies. Without you, the two and a half years spent here in the Netherlands would not have been so unforgettable. And I would like specifically to thank Milda, for providing me a place to live during the last few months of my thesis, it was a great favor for me and it allowed me to better focus on my work instead of spending most of my time for hunting of apartments; Madina, for finding time to proofread my work to eliminate as much as possible little mistakes and further improve the quality of my work; and Paul, for all of the interspersing discussions and for accompanying me to all of the travels I participated or organized during my studies.

Last but not least, I would like to thank my family in Lithuania, for supporting me in various ways during this time. Also, for letting me leave to the Netherlands for my studies, no matter how hard it was for you.

Thank you.

A. Aukselis
Delft, January 2017

Contents

1

Introduction

In Europe masonry is one of the most used construction materials. Most of the historic heritage is built using such means of construction. In many places in Europe, these buildings are built in earthquake-prone locations. This causes a threat to the historic heritage. It is most prevalent in the locations that just start to experience earth shakes as all of the buildings there were built without taking into account seismic activities. One of such locations is Groningen region in the Netherlands.

In Groningen, seismic activities started quite recently, in 1986. At that time first heavier earthquakes were experienced. The second big quake happened in 1997 and had a magnitude of 3.4 in Richter scale. In 2003 The Royal Netherlands Meteorological Institute (KNMI) admitted the link between the natural gas extraction and the earthquakes in the region. Since then the earthquakes only increase in magnitude. The third quake happened in 2006 of a magnitude of 3.5 and the fourth in 2012, and had a magnitude of 3.6 in Richter scale. Not only that the big earthquakes get stronger but also smaller rumbles increase in magnitude and frequency.[1]

Gas extraction produces earthquakes that are relatively near the surface. From such seismic activity, more damages are observed compared to natural earthquakes that have a deeper point of origin and the same magnitude.

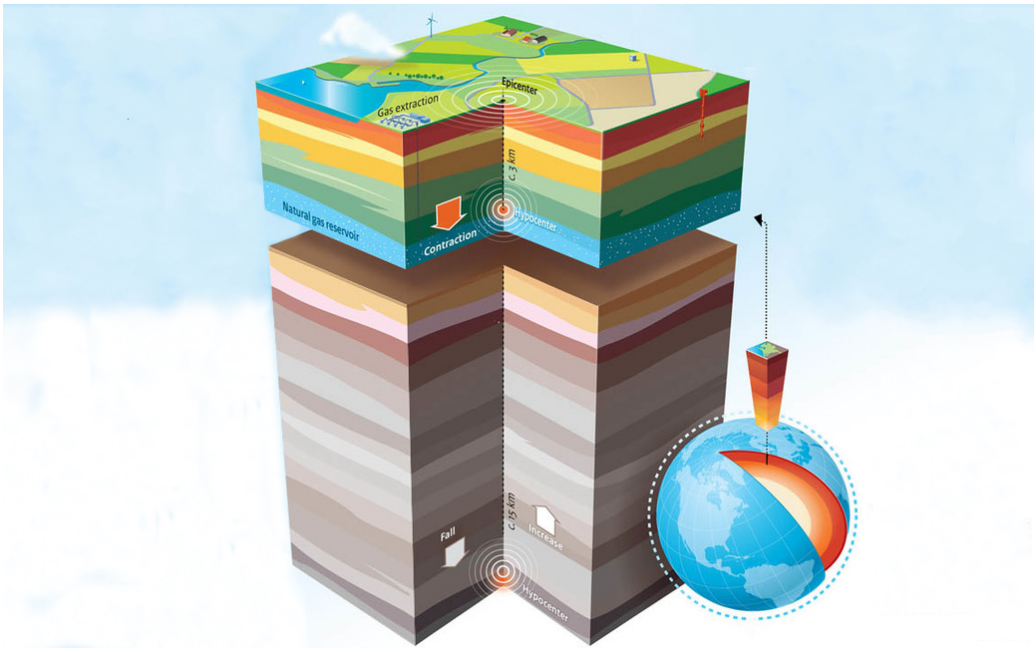


Figure 1.1: Earthquake depth due to gas extraction (~3 km) and natural causes (~15 km) (Source: [44])

[1] Data used in this paragraph is obtained from The Royal Netherlands Meteorological Institute [44]

1.1. Effects and solutions

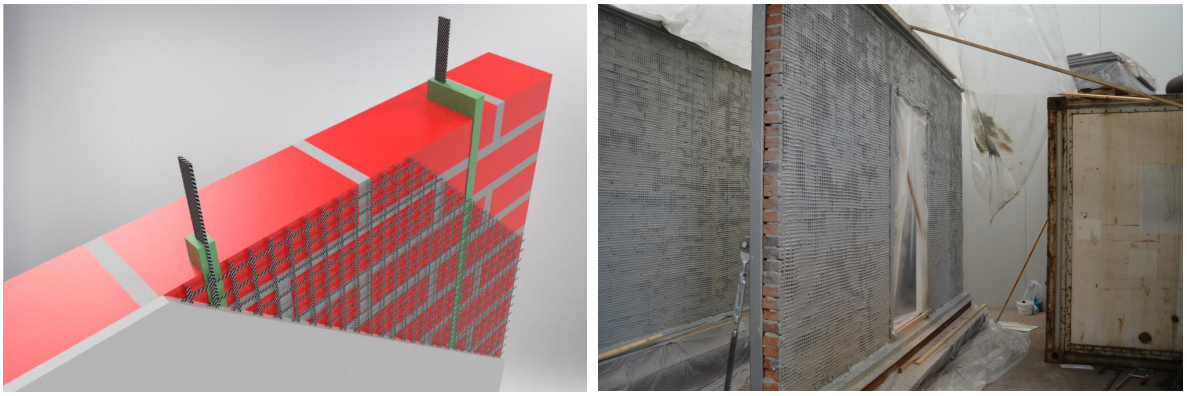
The earthquakes produced by natural gas extraction do not pose an immediate risk to the structures as their magnitude is low. To the contrary of earthquake-prone regions that exhibit the seismic activity of moderate strengths (5.0 or more in Richter scale) [10]. But rather they damage buildings slowly. Cracks that appeared during one earthquake, can contribute to the collapse of the building during the other one, even if the next quake is several magnitudes weaker. Particularly considering that the buildings in the region were not designed to resist such type of load. This poses a great risk to historical heritage and safety of the people living in the area.

(a) (b)

Figure 1.2: Damage on buildings in Groningen. (a) mortar failure, already filled cracks under the windows and (b) mortar failure, loose bricks are taken out. *Photos by Harm Hoorn*.

It can be prevented by taking measures. One of the possible solutions is to reinforce the buildings in the area. It can be accomplished by renovating load bearing elements or installing dampers at the foundations of such buildings. So that the seismic loading is taken into account. The solutions to the problem are yet to be implemented in the area. One of the currently potential wall reinforcement techniques is milling deep and shallow grooves into the masonry wall, then embedding carbon strips in the grooves using a special visco-elastic adhesive. The curing is done on only one side of the wall. After the strips are placed, a carbon net is placed on the surface. Finally, a cement based or a polymer based layer is applied. [90]

However, the problem arises when it has to be analyzed which buildings will need reinforcing or if the designed solutions are sufficient.

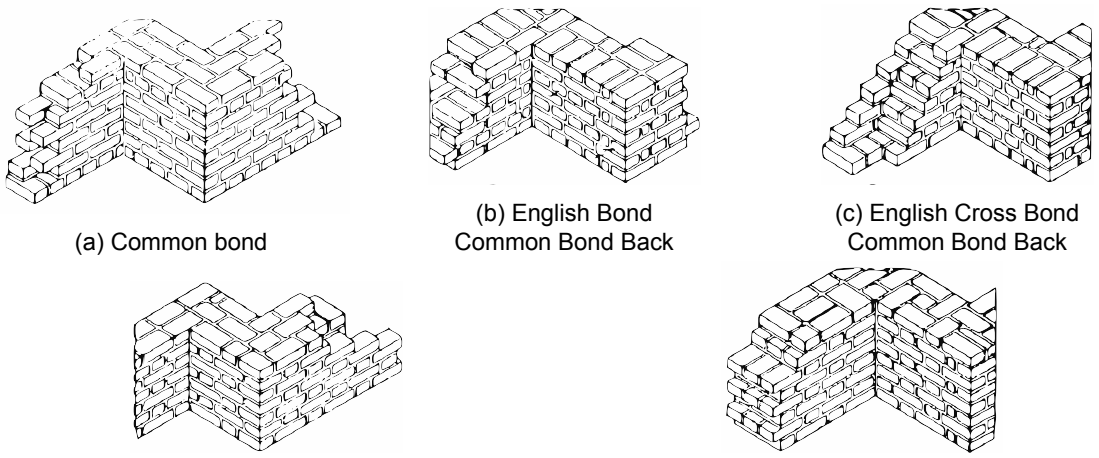


(a) Visualization of used retrofit system (from [90]) (b) Retrofit system applied to a wall (from [1])

Figure 1.3: Reinforcing masonry building from earthquakes

1.2. Material properties of masonry

Masonry is a highly orthotropic composite material with strengths and properties dependent on workmanship, materials, and layouts used. It consists of large units and joints between them. The arrangement of these units and mortar joints determine anisotropy of the composite material. While the type of units and composition of mortar determines overall strength and elasticity. Mortar joints behave as the weakest plane, thus their orientation has the governing effect on the load bearing capacity of masonry.



(a) Common bond

(b) English Bond
Common Bond Back

(c) English Cross Bond
Common Bond Back

(d) Single Flemish Bond
Dutch Corner

(e) Double Flemish Bond
English Corner

Figure 1.4: Examples of bond types in brick masonry

The failure behavior is determined by 5 basic failure mechanisms (see Figure 1.5). Two of these mechanisms are governed by the properties of the bond (a and b) and three by unit strengths. However, mechanism (c) is only likely [5] to form if tensile strength of masonry unit is weak, it would allow a crack passing along head mortar joints and through the center of the units (see Figure 1.6a) to form. But if masonry unit strengths are far greater than the strengths of mortar, a zigzag like pattern (see Figure 1.6b) would form instead.

The difference in elastic properties between a joint and a unit can cause failure [38] as well. This difference, particularly if the unit is stiffer than the joint, leads to a state of triaxial compression in the joint and compression or biaxial tension in the unit (see Figure 1.7). The cause is that the joint lateral extension is confined by the unit, this, in turn, causes a crack in the unit itself. Increased deformation produces additional vertical cracks until the unit fails.

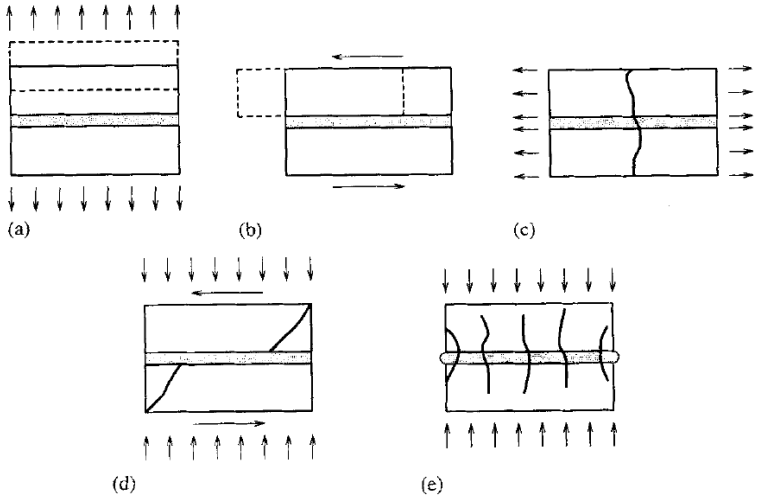
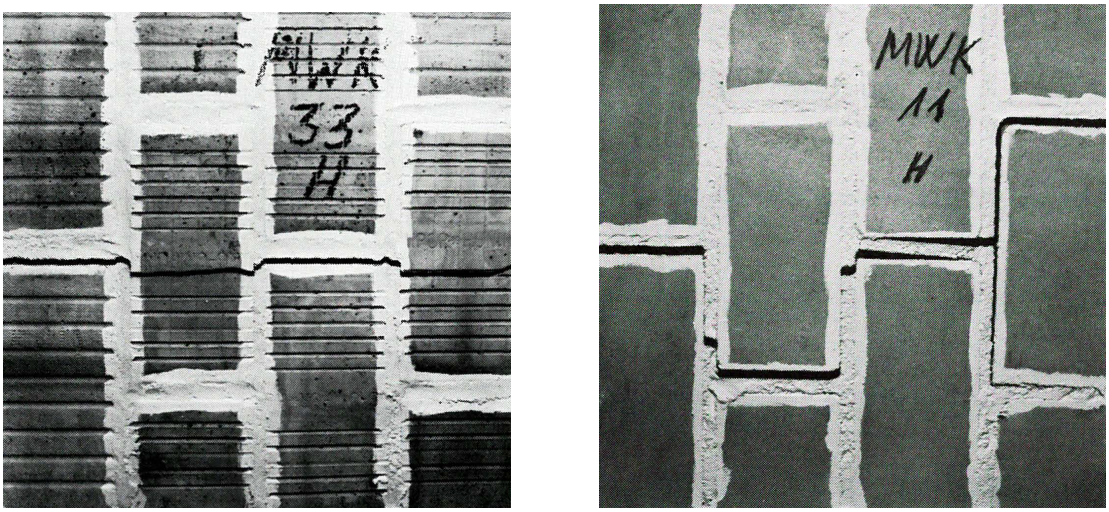


Figure 1.5: Masonry failure mechanisms (from [58]): (a) joint tensile cracking; (b) joint slipping; (c) unit direct tensile cracking; (d) unit diagonal tensile cracking; (e) masonry crushing.



(a) (b)

Figure 1.6: Modes of tension failure of masonry walls under direct tension (from [5]): (a) through type, (b) zigzag type

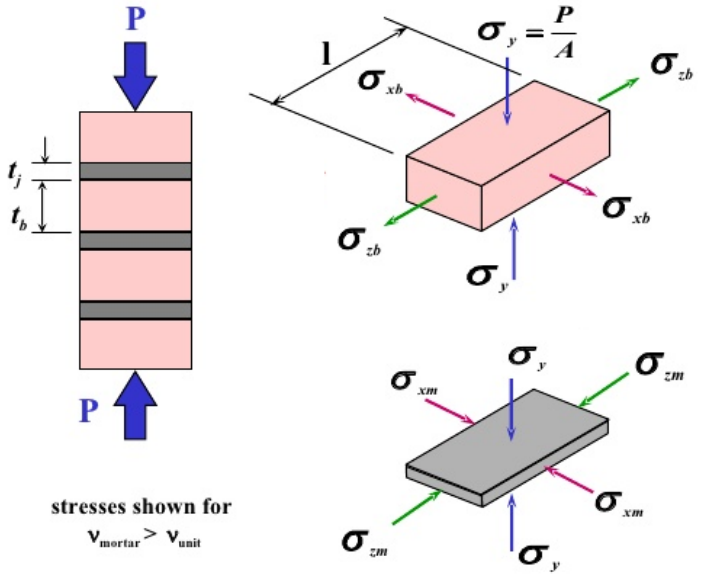


Figure 1.7: Stress state in masonry units and joints in uniaxial compression

The strengths and failure mode change when different orientations of bed joints and loadings are used. In the case of uniaxial tension (see Figure 1.8) the mode of failure always remains the same, cracking of a joint. Therefore, for the tensile strength of masonry as the whole, it is safe to assume the strengths of the bond between masonry and the mortar. However, this assumption is only valid for loading direction perpendicular to bed joint as for other directions friction between units and joints cause additional strengths.

In the case of masonry with high strengths mortar and low tensile strengths in units e.g. bricks with a high number of perforations, failure might occur when stresses exceed tensile strengths of a unit. Then tensile strengths of the masonry can be considered to be the tensile strengths of a unit.

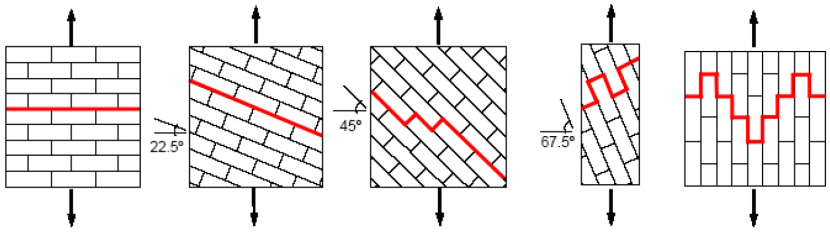


Figure 1.8: Modes of failure of solid clay unit masonry under uniaxial tension (from [73]).

During uniaxial compression (see Figure 1.9) the failure modes alternate between crushing of masonry when loading is applied perpendicular to the bed joints and tensile splitting of joints when loading is applied parallel. While at loading angles in between, combination of failure modes are obtained.

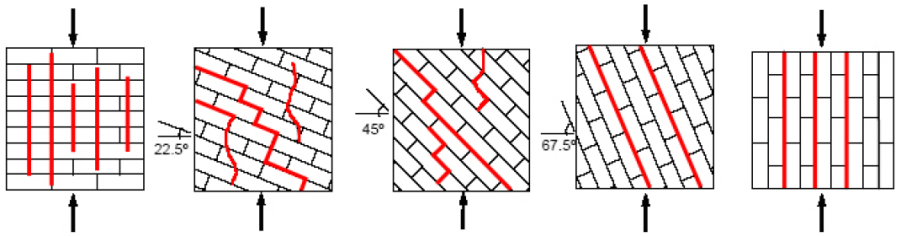


Figure 1.9: Modes of failure of solid clay unit masonry under uniaxial compression (from [72, 73]).

The behavior of masonry under biaxial stress cannot be described only by means of principal stresses. Masonry is an orthotropic material, thus, the strengths are fixed to the material axes and cannot rotate together with principle stresses. Therefore, the strength envelope has to be described in terms of full stress vector in a fixed set of material axes or in terms of principle stresses and the rotation angle θ between principal stresses and material axes.

Considering the test results obtained by Page [72, 73] (see Figure 1.10), that were carried out with half scale solid clay units. It can be seen that the failure strengths are heavily influenced by the orientation of the principle stresses in respect to the material directions.

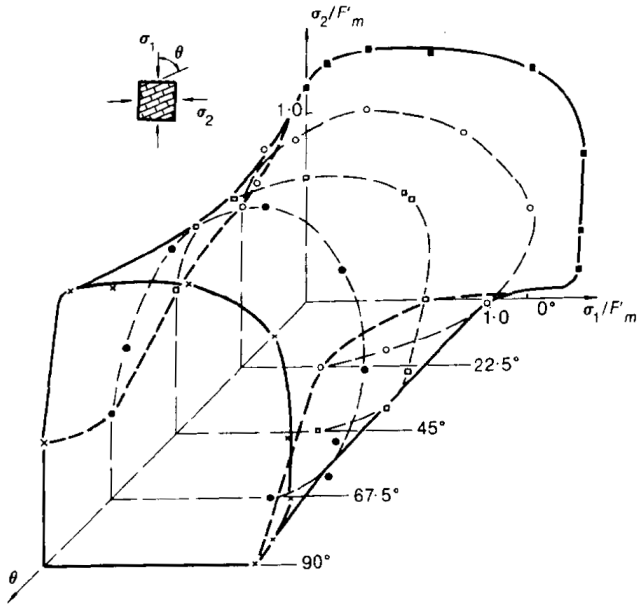


Figure 1.10: Failure surface of brickwork loaded in biaxial compression (from [72]).

A lateral compressive stress decreases the tensile strengths. The lowest strength is reached when the tension is perpendicular to the bed joints. In biaxial tension-compression tests failure generally occurs by cracking/sliding of the joints or a combined mechanism involving both – joints and units (see Figure 1.11). However, there is no available experimental results that determine the influence of lateral tensile stress to the tensile strengths of the masonry. Consequently, assumptions have to be made and no exact behavior can be derived.

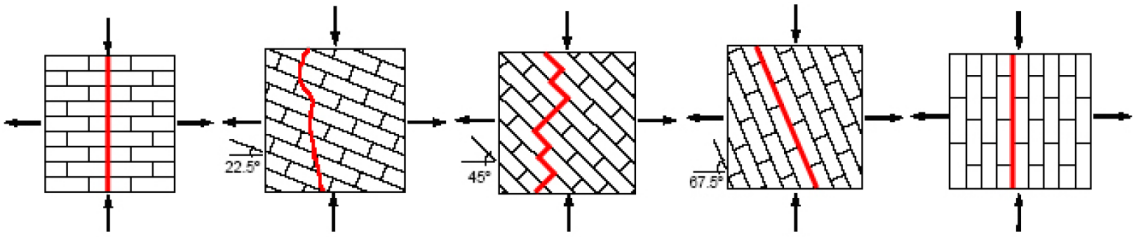


Figure 1.11: Modes of failure of solid clay unit masonry under biaxial tension–compression (from [73]).

In biaxial compression typically failure occurs by splitting of a specimen in mid-thickness (see Figure 1.12). The orientation of principal stresses does not influence the failure mechanism. However, if the ratio between principal stresses is not equal to one, the orientation has a significant influence to the formation of the failure mechanism in the specimen. In that case, failure occurs as a combined mechanism of both – joint failure and lateral splitting.

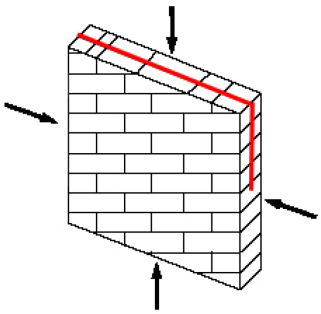


Figure 1.12: Mode of failure of solid clay unit masonry under biaxial compression (from [72]).

The strengths envelope obtained by Page [72, 73] is of limited applicability in masonry. Different types of masonry composed of different materials, unit shapes and bonds will likely produce different failure modes and envelopes. There are further studies on characterization of biaxial strengths of masonry [27, 34, 60] that are advised to refer when designing masonry.

The determination of the shear response [4, 39, 92] depends on the ability of the test set-ups to generate a uniform state of stress in joints. The confinement stress increases the shear strengths due to frictional behavior of masonry in shear. Furthermore, the behavior of joint is non-associative, i.e. $\delta_n \neq \delta_t \tan\phi$, where δ_n and δ_t are respectively the normal and tangential relative displacements between sliding surfaces at a masonry joint and ϕ is the angle of friction.

Although some dilatation is likely to occur when two rough units pass over each other, experiments indicate that the real joint behavior is quite complex. The dilation of the masonry subjected to shear depends on the micro-scale geometrical and mechanical features of the masonry joint [93]. Furthermore its is observed that the angle of dilation tends to reduce by both means of increasing isotropic pressure and increasing tangential relative displacement (see Figure 1.13 and 1.14).

Another feature of masonry is softening behavior. This kind of behavior is prominent for quasi-brittle materials e.g. clay bricks, ceramics, rock or concrete. Softening is a gradual decrease of mechanical resistance under a continuous increase of deformation forced upon a material specimen or a structure. Materials exhibiting softening behavior fail due to a process of progressive internal crack growth. Such mechanical behavior is commonly attributed to the heterogeneity of the material, due to the presence of different phases and material imperfections, like defects and voids.

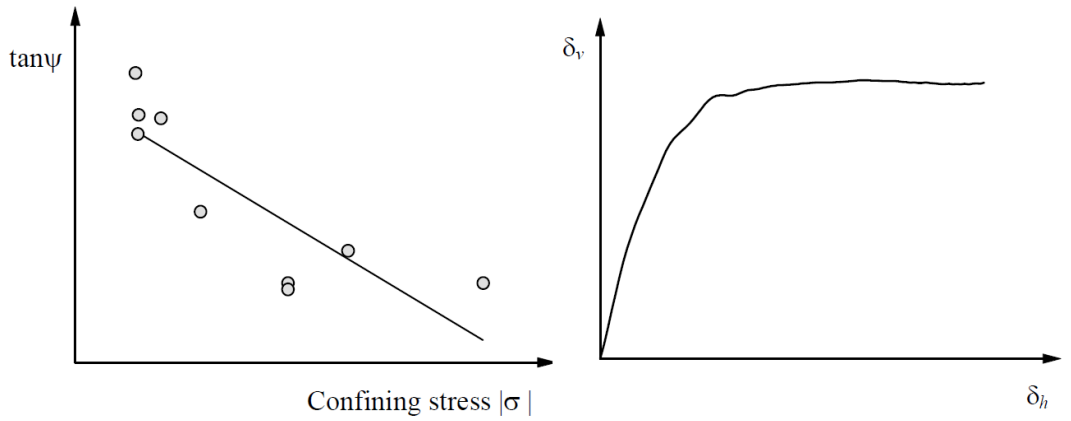


Figure 1.13: Typical shear bond behavior of the joints for solid clay units (from [92]): (a) tangent of the dilatation angle Ψ as a function of the normal stress level; (b) relation between the normal and the shear displacement upon loading.

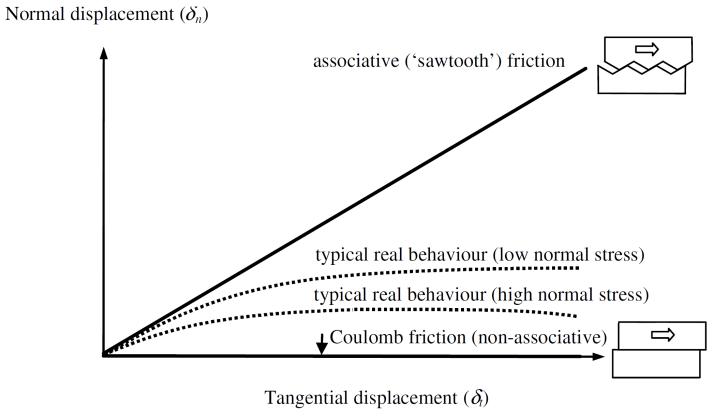


Figure 1.14: Masonry joint behavior (from [29]): relationship between associative & Coulomb friction (non-associative) idealizations and typical real behavior.

Initially, mortar and units contain micro-cracks and inclusions. The initial stresses and cracks, as well as variations of internal stiffness and strength, cause progressive crack growth when the material is subjected to progressive deformation. At the start of the loading, the micro-cracks are stable which means that they grow only when the load is increased. Around peak load an acceleration of crack formation takes place and the formation of macro-cracks starts. In a deformation controlled test, the macro-crack growth results in softening and localization of cracking in a small zone while the rest of the specimen unloads.

Characteristic stress-displacement diagrams for quasi-brittle materials in uniaxial tension, uniaxial compression and pure shear can be seen in Figure 1.17. The fracture energy , denoted by G_f and G_c, is defined as the integral of the $\sigma - \delta$ diagram for tension and compression, respectively. In case of mode II failure mechanism, i.e. slip of the unit-mortar interface under shear loading, the inelastic behavior in shear can be described by the mode II fracture energy $G_{II,f}$, defined by the integral of the $\tau - \delta$ diagram. Figure 1.17c shows brittle behaviors in shear. The value of the fracture energy depends on the level of the confining stress. Shear failure is a salient feature of masonry behavior which must be incorporated in a micro-modeling strategy. However, for continuum macro-models, this failure cannot be directly included because the unit and mortar geometries are not discretized. Shear failure is then associated with tension and compression modes in a principal stress space.

The consideration of the cyclic behavior of masonry adds another layer of complexity. Furthermore, the exact behavior is not sufficiently documented. However, there are tests done regarding cyclic shear loading on masonry walls [4, 28, 84], but cyclic uniaxial and biaxial tests are very rare. In addition, most

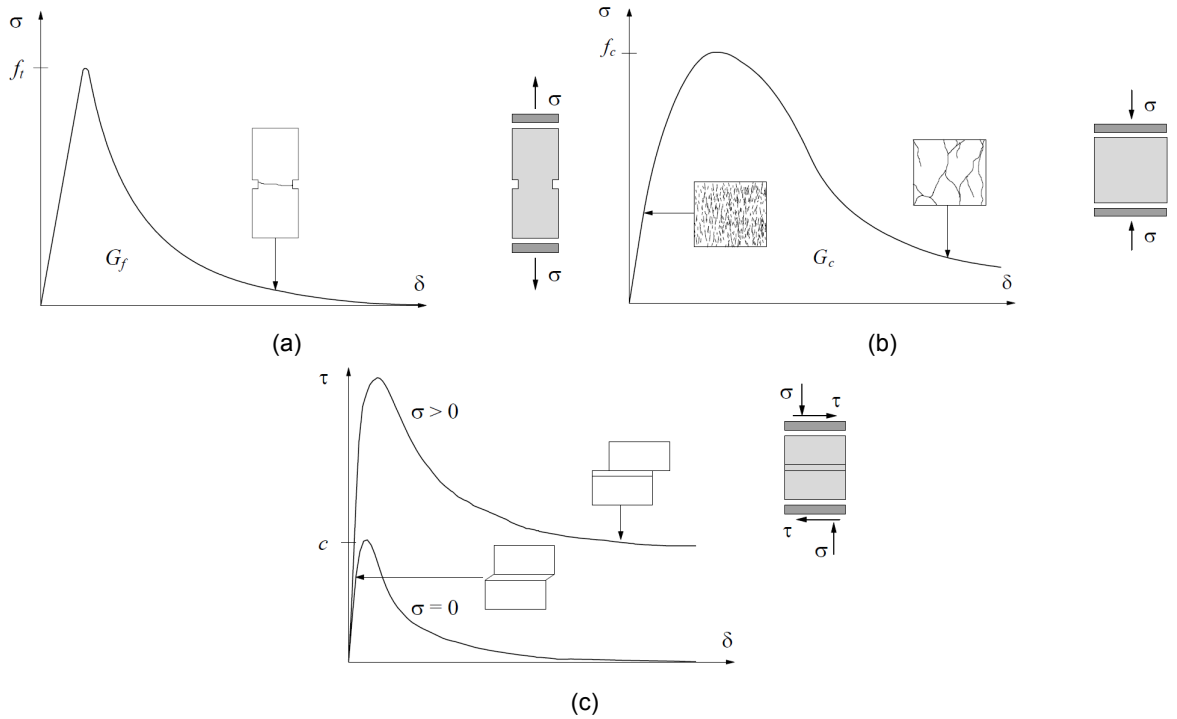


Figure 1.15: Typical behavior of quasi-brittle materials under uniaxial and shear loading and definition if fracture energy (from [57]): (a) tensile loading (f_t denotes the tensile strength); (b) compressive loading (f_c denotes the compressive strength); (c) shear loading (c denotes the cohesion).

of the masonry material models available in the market do not simulate cyclic behavior and are designed only for static pushover tests.

Tensile unloading of the material when it is already in the plastic phase causes reduction of initial stiffness. Such behavior occurs due to separation of the units and cracking of the joints while specimen is loaded, and units coming back to initial positions while unloading.

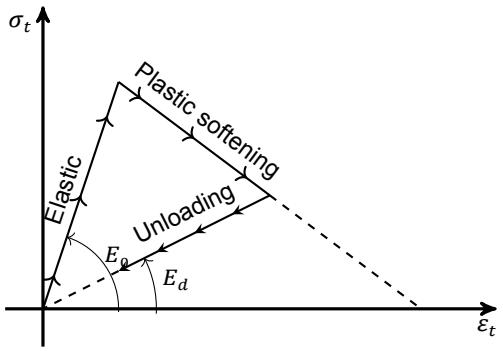


Figure 1.16: Tensile cyclic behavior.

On the contrary, when material is loaded in compression, during softening or hardening of masonry specimens no damage to the stiffness is produced (see Figure 1.17). In such loading situation when hardening or crushing of the material occurs the specimen deforms and plastic strain will be generated as a result. When the load is removed, the specimen does not regain its initial form and remains deformed.

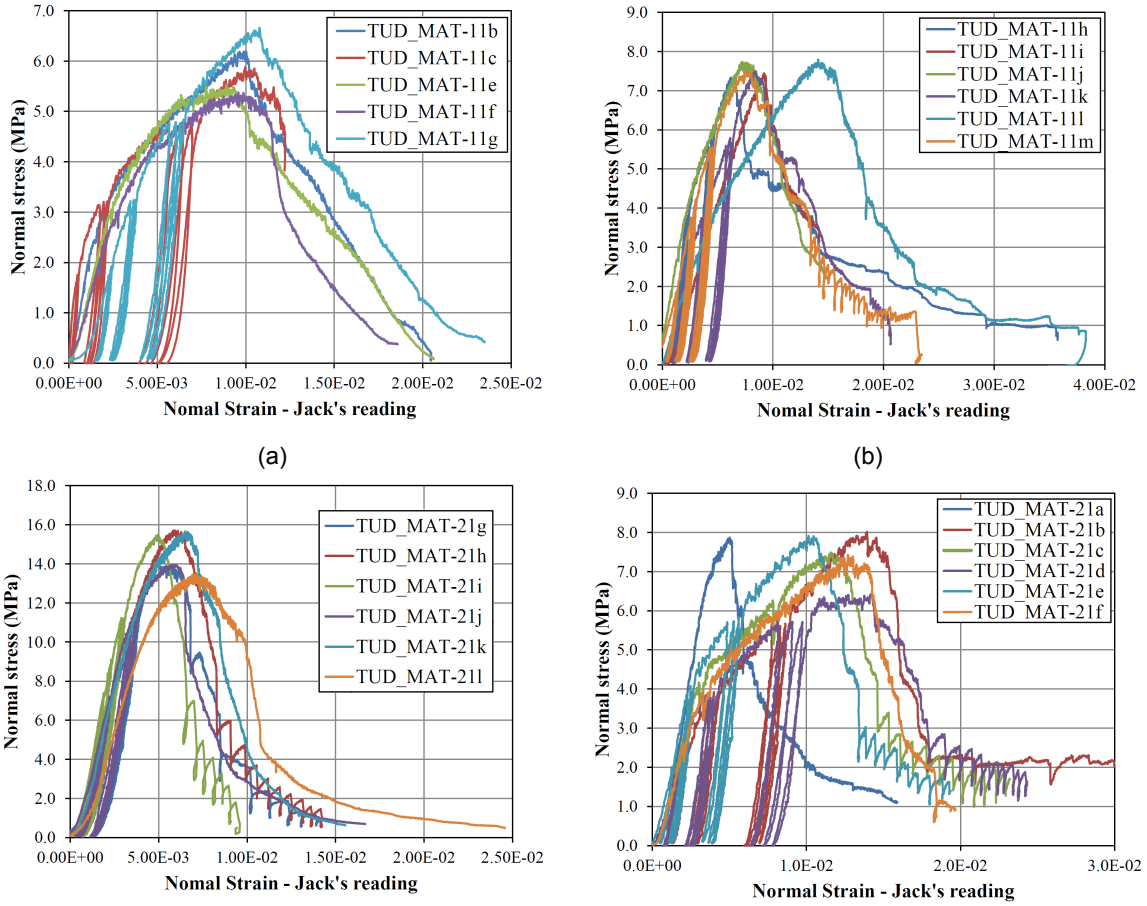


Figure 1.17: Compression tests on masonry specimens (from [21]): (a) Calcium silicate units, vertical compression (perpendicular to the bed joint); (b) Calcium silicate units, horizontal compression (parallel to the bed joint); (c) Clay units, vertical compression (perpendicular to the bed joint); (d) Clay units, horizontal compression (parallel to the bed joint) .

The cyclic shear behavior in the bed joints is quite simple (see Figure 1.18a). After the joint is fully softened only stress due to friction is remaining and no damage to shear stiffness is present. However, when a shear wall is subjected to a lateral load (see Figure 1.18b), various second order effects take place in the definition of cyclic shear response.

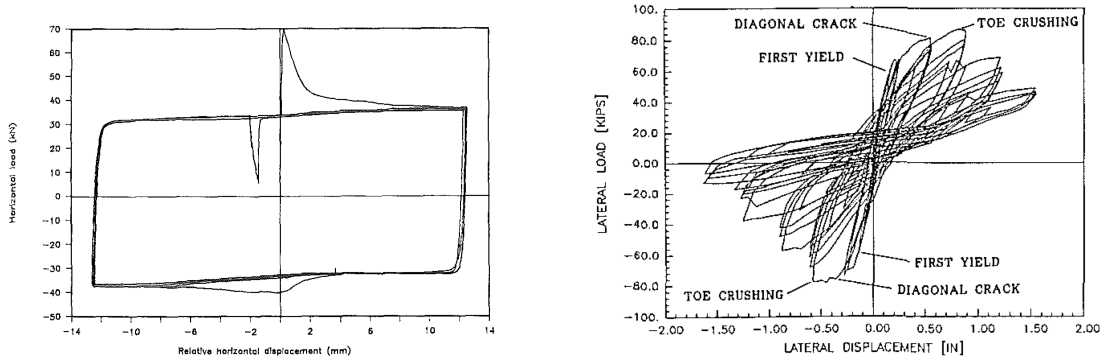


Figure 1.18: Typical cyclic shear response: (a) of the joints for solid clay units (from[4]); (b) of the masonry walls (from [84])

1.3. Current methods

As described in the previous section, it can be clearly seen that the masonry is a very complex composite material. This means that analyzing big masonry structures using empirical methods are highly ineffective and cost inefficient. Consequently, other methods should be used i.e. Finite Element Method (FEM). To model masonry successfully with FEM, simplifications are needed. Whereas, how much to simplify the model is up to the engineer. There are mainly two distinctively different modeling approaches commonly used in FEM, namely, micro- and macro-modeling (see Figure 1.19). However, a combination of the two approaches is also possible.

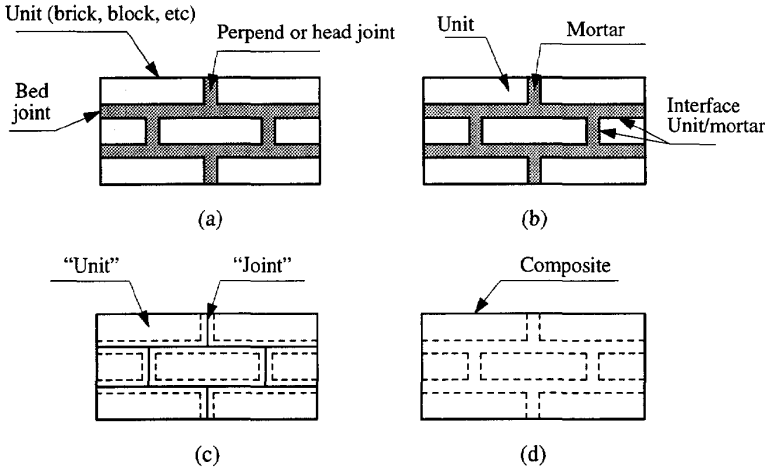
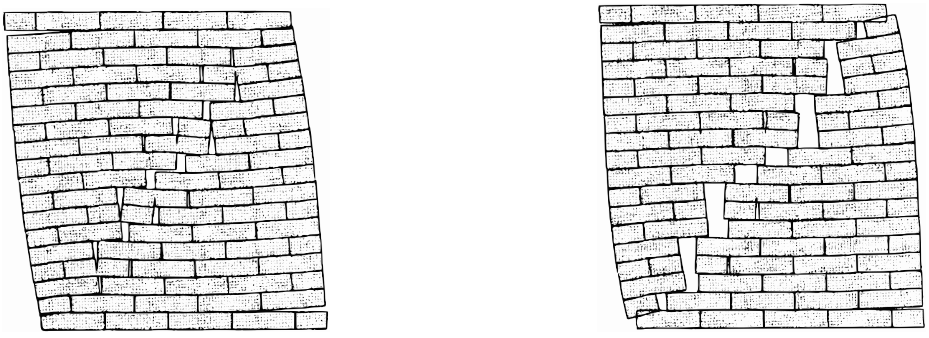


Figure 1.19: Modelling strategies for masonry structures (from [56]): masonry sample (a); detailed (b) and simplified (c) micro-modelling; macro-modelling (d).

1.3.1. Micro-modeling

Micro-modeling is so far the most accurate method available to simulate the behavior of masonry. Since the material is separated into units of continuous elements and interfaces (discontinuous elements) between them (see Figure 1.19b). Accordingly, only well established and accurate isotropic material models can be used to simulate the anisotropic behavior of masonry. Therefore, the results of such analyses are the most reliable (see Figure 1.21). On the contrary, the models require an extensive amount of time to be prepared and the analyses can be computationally taxing when analyzing big and complex structures.

Nevertheless, this shortcoming could be minimized by removing continuous elements that represent mortar and replacing them by interfaces with mortar-like properties (see Figure 1.19c). This way simplified micro-model is obtained [2, 25, 26, 52, 54, 58, 83, 87]. However, the high level of detail required to represent masonry accurately still makes this approach only suitable for analyses of small elements and small structural details.



(a) Deformed mesh at peak load (b) Deformed mesh at collapse

Figure 1.20: Micro-modelling of masonry shear walls (from [58])

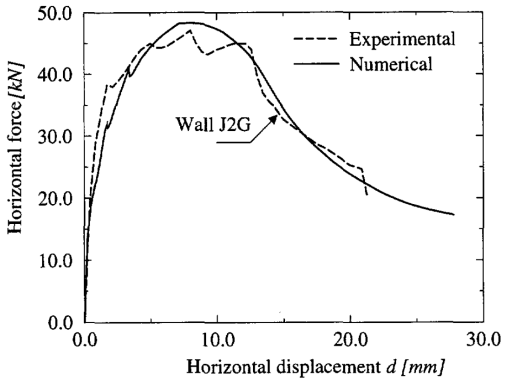


Figure 1.21: Force – displacement graph. Comparison experimental shear wall and numerical micro modeling analysis (from [56]).

Even though micro-modeling is computationally taxing it is not limited to simple masonry walls (Figure 1.21), but also bigger structures or parts of buildings containing walls, columns and/or arches can be modeled. The Figure 1.22 shows a finite element representation with interface elements that was used in [52] to study a pillar-arch stone structure, analyzed under pseudo-dynamic loading.

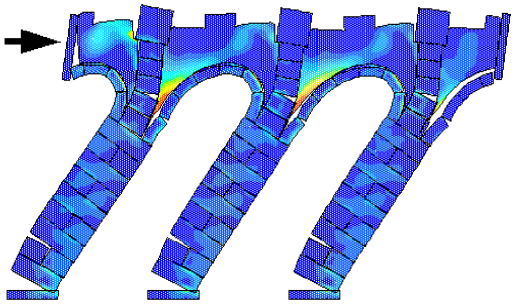


Figure 1.22: Monastery S. Vicente de For a, in Lisbon. Deformation pattern using interface elements (from [52]).

As the computation power of computers increase, more and more complex models can be analyzed. A very good example of such analysis was presented by Alexandris et al. [2]. They investigated the collapse mechanisms of traditional one- and two-story houses under earthquake loading in 2D and 3D (see Figure 1.23). The models were used to evaluate alternative intervention options.

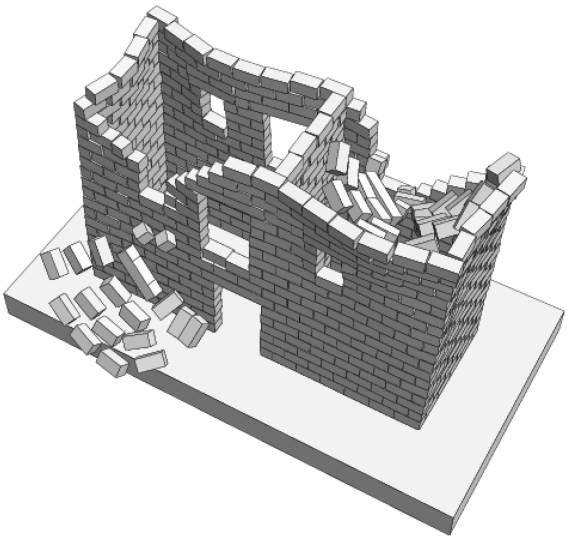
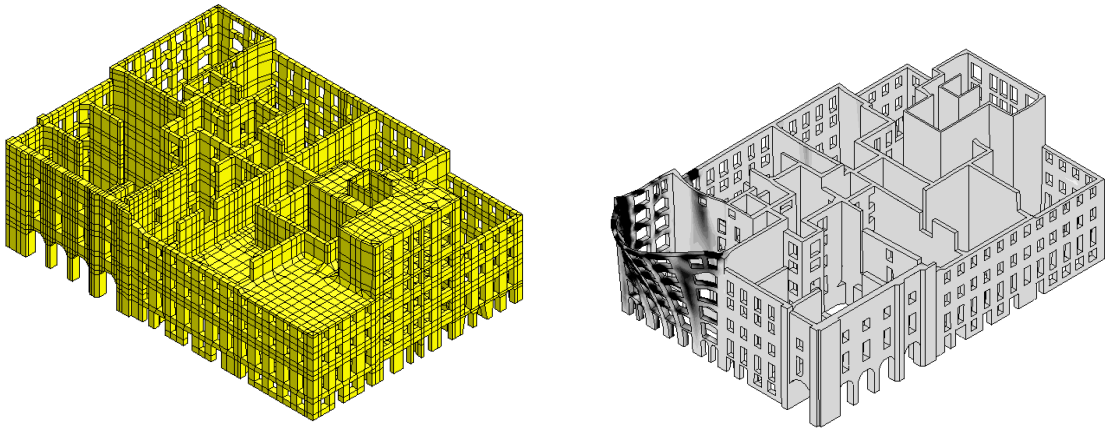


Figure 1.23: Collapse of a two-story house under seismic loading (from [2]).

1.3.2. Macro-modeling

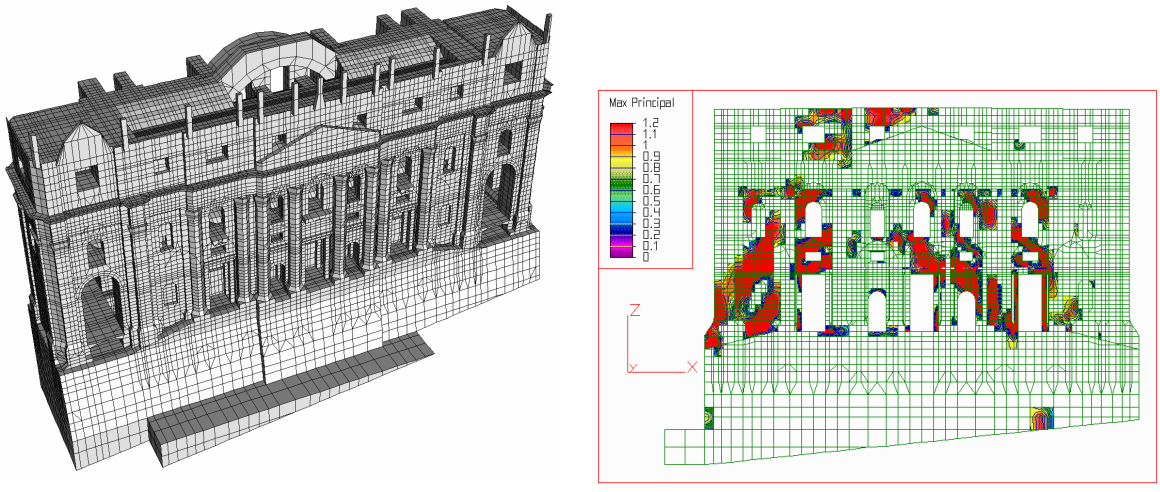
As demonstrated in the previous subsection the power of modern numerical tools to represent the complex interaction is sufficient in specific cases. However, when the structure becomes larger the interactions between masonry components (units and joints) start to merge in a homogeneous behavior (see Figure 1.19c). Therefore for large-scale analyses, a faster material model can be assembled. However, making further simplifications reduces the accuracy of the simulation. Consequently, this type of analysis is only relevant when robustness and ease of use (no need to model complex structures consisting of lots of different elements and interactions) are more important than the slight reduction in accuracy of the behavior of the structure.

One of the good examples of such approach was presented in [80]. Finite element model for a block compound in Lisbon (see figure 1.24) was modeled in order to perform a seismic analysis. The model of this scale currently would not be possible to assemble using micro-modeling. Another complex model used for masonry analysis was assembled by Macchi [61]. There a St. Peter's Basilica was modeled and analyzed (see Figure 1.25) in order to aid the studies of its restoration.



(a) (b)

Figure 1.24: Finite element model for a block compound in Lisbon (from[80]): (a) Finite element mesh (b) results for seismic (shading indicates damage levels).



(a) (b)

Figure 1.25: Studies for the St Peter's Basilica restoration (from [61]): (a) Model of the entire Facade; (b) tensile stresses on the central section under the effect of the settlement.

Masonry macro models started from the usage of isotropic material description, because of its simplicity. Isotropic materials require few material parameters, therefore, it is easier to perform analyses on historic structures as not all needed material properties can be obtained. Papa [74] derived unilateral damage model for an orthotropic case from a material model originally developed for isotropic materials. The outcome included a homogenized technique to take into account the texture of brick and mortar. A year later Lourenco et al. [59], developed an orthotropic material model for masonry and demonstrated that the homogeneous material model is sufficiently accurate when modeling shear walls under monotonic loading. It was able to predict cracking patterns, peak and ultimate strengths of the structures (see Figure 1.26).

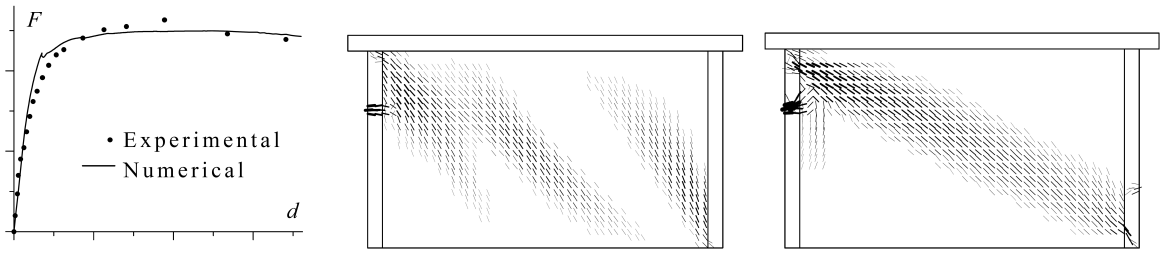


Figure 1.26: Results for an analysis of a masonry shear wall (from [59]): (a) load displacement diagram; (b,c) predicted cracking pattern at peak and ultimate load.

In all of the above examples, smeared damage models were adopted, even if they only provide general information about the level of damage expected on the structure. To point out, the damage simulated this way is unrealistic. It propagates trough significant volumes and spreads over large regions of the structure. However, Clemente et al. [12] proposed an enhancement to traditional damage approaches. Their model was based on smeared-crack scalar damaged model and it was modified to reproduce localized (discrete) cracks. To achieve it a local crack-tracking algorithm was used. This model enables the simulation of more realistic damage distributions than the original smeared-crack model.

The localized cracks predicted by the crack tracking model reproduce consistently a set of expectable plastic hinges developing gradually in the structure and leading to the full collapsing mechanism. The model has been used to analyze the response of the structure of Mallorca Cathedral under gravity and seismic forces (see Figure 1.27).

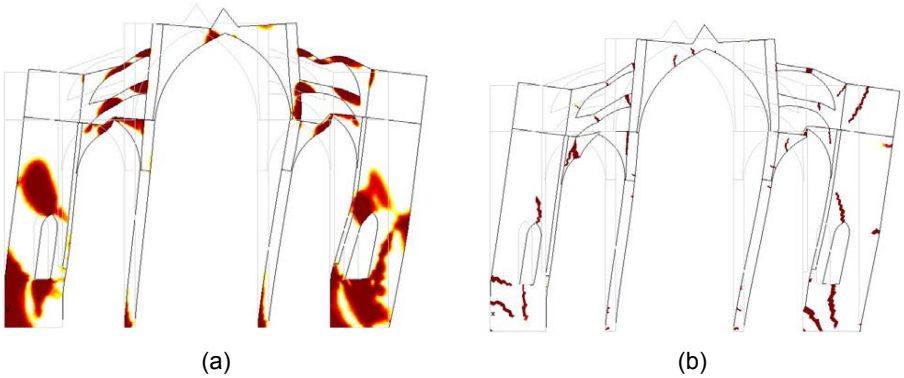


Figure 1.27: Seismic analysis of Mallorca Cathedral (from [12]):(a) smeared damage approach versus (b) localized damage approach.

An interesting approach was used by Pela et al. [76], where they composed a material model from isotropic failure envelopes by mapping them from fictitious isotropic stress and strain space to a real orthotropic space. This allowed them to use well defined isotropic failure criteria for orthotropic material behavior simulation. Additionally, they used a local crack-tracking algorithm derived by Clemente et al. [12] in order to localize cracks produced by damage due to tension (see Figure 1.30).

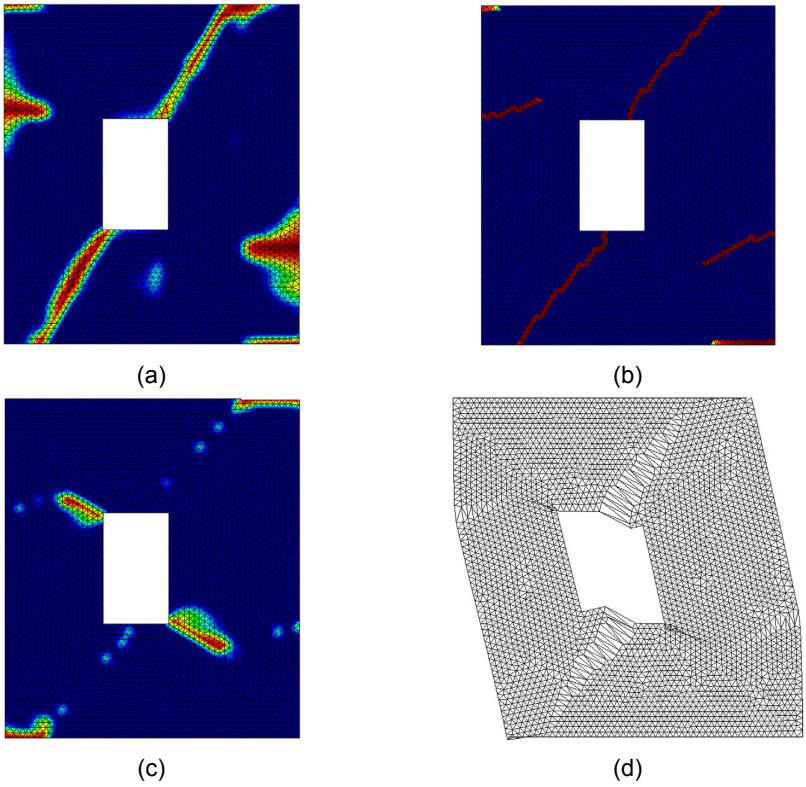


Figure 1.28: Analysis of shear walls with opening (from [76]):(a) smeared tensile damage; (b) localized tensile damage; (c) smeared compression damage; (d) deformed mesh (x50).

1.3.3. Limit-modeling

Previously mentioned modeling techniques can be combined by modeling the structure subdivided into homogeneous elastic or rigid blocks that are connected with interfaces simulating most common failure mechanisms (see Figure 1.29). These modeled collapse mechanisms are then analyzed by applying kinematic limit analysis. The approach was first proposed by Giuffré [32, 33] where he observed a pattern in failure modes of historical and traditional buildings in Italy. This approach is particularly interesting as a tool for seismic analysis of buildings which do not conform to box behavior because of lack of stiff floor slabs or because of weaker partial collapses affecting the façade or inner walls. For more upper and lower bound limit analyses and techniques reader is referred to [30–33, 53, 62–66, 87].

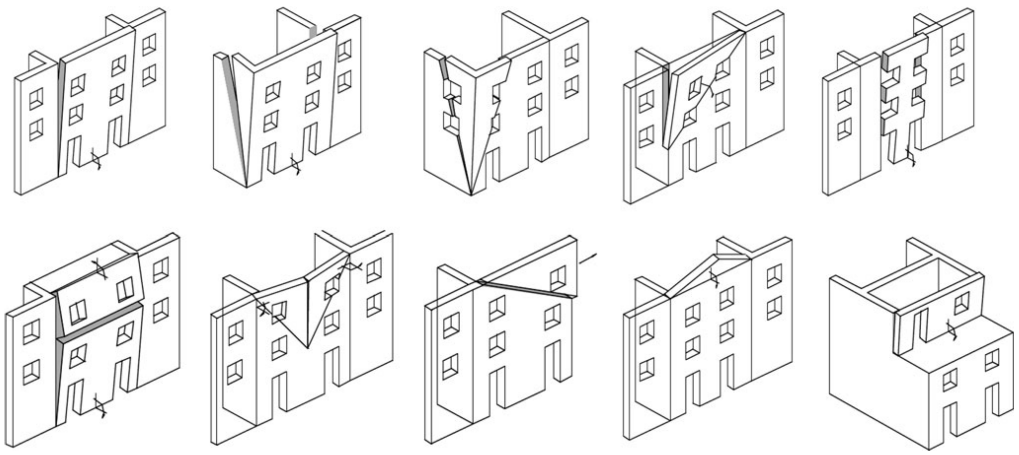


Figure 1.29: Failure mechanisms for buildings embedded within urban texture (from [83] according to [16]).

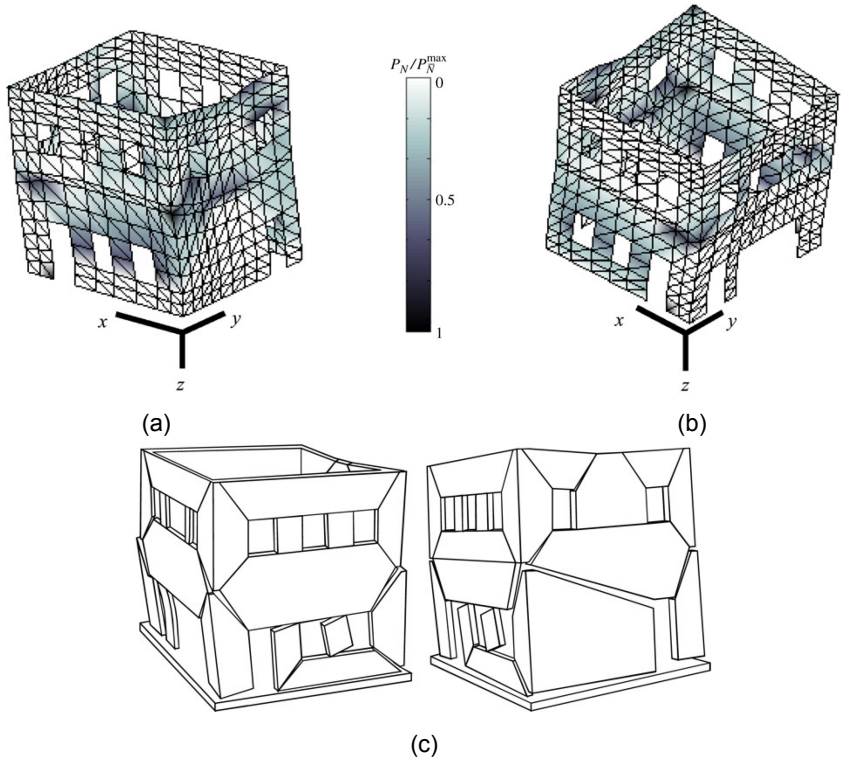


Figure 1.30: 3-D homogenized limit analysis of a masonry building (from[66]).

1.4. Current numerical methods used in the Netherlands

Currently, there are several finite element software packages that are being used in the Netherlands to analyze masonry structures. Mainly *Simulia Abaqus*, *TNO Diana* and *LS-DYNA*. *Simulia Abaqus* and *LS-DYNA* both support implicit and explicit integration while *TNO Diana* has only an implicit solver. The implicit solvers are very difficult to work with when analyzing highly nonlinear problems that involve large deformations and large amounts of plasticity. Therefore, for masonry analyses under earthquake loading they are not an ideal solution and explicit solvers are preferred.

Even though *LS-DYNA* has a masonry material model it is developed and used only by *ARUP* and the model is not available for third parties. Because of the fact the thesis is sponsored by *Zonneveld ingenieurs b.v.* the use of *LS-DYNA* masonry model is not possible as *Zonneveld ingenieurs b.v.* uses *Simulia Abaqus* as their finite element analysis software.

For masonry analyses with *Simulia Abaqus* an adapted model of *Concrete Damaged Plasticity* is being used by *Zonneveld ingenieurs b.v.*. This model is one of the default materials that comes integrated in *Simulia Abaqus* and it is developed to be used as a tool to analyze reinforced concrete. *Concrete Damaged Plasticity* is an asymmetric isotropic continuum damage model that supports different behaviors in tension and compression, strengths hardening/softening, and damage to stiffness.

Although, by adapting the model to be used with an orthotropic material such as masonry causes several issues. E.g. due to the isotropic nature of the material model, engineer using it has to predict a critical failure mechanism in order to adapt real orthotropic properties of a material to the isotropic model. Therefore, to simulate masonry behavior more precisely a better material model has to be developed. Such model should exhibit asymmetric orthotropic plastic behavior.

1.5. Scope of the thesis

The main aim of the thesis is to develop a nonlinear material model based on the Continuum Damage Mechanics, that can be used in Simulia Abaqus Finite Element Analysis software. The material model should exhibit asymmetric orthotropic plastic behavior and it should be robust and accurate in small scale tests as well as large-scale complex analyses.

The development is focused on three-dimensional structures that are loaded by static as well as dynamic loading. The model is a macro type and it simplifies masonry as continuous homogeneous material, on the contrary to the micro modeling where units and mortar are modeled separately.

The objectives of the study are:

- To assemble information on the existing knowledge about Continuum Damage Mechanics models, through a comprehensive literature review;
- To develop an orthotropic masonry model that is able to accurately predict elastic and plastic behavior of masonry structures and incorporates the knowledge of nonlinear fracture mechanics used in crack propagation problems;
- To test the model by comparing the predicted behavior with the behavior observed in experiments on masonry. The developed model should be able to predict the failure modes and the ultimate load with a reasonable agreement with the experimental evidence.
- To demonstrate the applicability of the verified model in engineering practice case-studies i.e. in analysis of shear walls.

It must be mentioned, however, that results of masonry tests that are large and complex or small and simple, typically shows a wide scatter. Thus, the main concern is not too sharply reproduce the experimental results in the form of load-displacement curve but to demonstrate the ability of models to capture the behavior observed in the experiments.

It is to be noted that the model developed in this study can be used in a broader field than just masonry. It is applicable to any other anisotropic material like plastics, wood, and fiber-reinforced composites.

1.6. Outline of the Thesis

This thesis consists of six Chapters.

Chapter 1 contains a brief introduction of the seismic situation in the Netherlands and effects of it on the masonry structures. This chapter also describes masonry and it's properties as well as the overview of current approaches to masonry analysis. The scope of the thesis and its outlines are also included.

Chapter 2 presents the review of several Continuum Damage Mechanics models and aspects corresponding to their numerical implementation.

Chapter 3 describes the formulation of a damage model for masonry. Such a model accounts for different orthotropic behaviors in tension and compression. Individual damage criteria are considered for tension and compression, according to different failure mechanisms. The former is associated with cracking phenomenon, while the latter is associated with the crushing of the material. Entirely different elastic and inelastic behaviors can be predicted along the material axes, both in tension and compression. For compression, the model can predict residual plastic strain and it can simulate the cyclic behavior. The resulting formulation is implemented in a nonlinear finite element code for Simulia Abaqus Explicit software package.

Chapter 4 validates the damage model developed in Chapter 3 by means of the FE analysis of cube tests in order to portray the pure theoretical characteristics of the model.

Chapter 5 furthermore validates the damage model by means of the FE analysis of engineering practice case studies, e.g. shear walls with monotonic or cyclic loading. A smeared crack approach is considered.

Chapter 6 presents an extended summary and the final conclusions together with suggestions for future work which can be derived from this study.

2

Basic concepts and theoretical formulations

This Chapter presents various approaches to material modeling using Continuum Mechanics, the possible descriptions of failure envelope and their comparability with available tests. Basic concepts are defined, together with the theoretical formulation. Then, a comparative discussion concerning different failure envelopes and damage models is carried out, in order to emphasize the implications arising from the different backgrounds. Furthermore, the chapter will also describe the relation between different analysis techniques and explanations where does the custom material models belong in Simulia Abaqus analysis procedure.

2.1. Implicit and Explicit algorithms

The Finite element method (FEM) is one of the most popular methods in both research and industrial numerical simulations. In the FEM codes, several different algorithms are implemented, these algorithms can have varying computational costs, accuracy or ease of use. Understanding the nature, advantages and disadvantages of these algorithms are very helpful for choosing the right algorithm for the particular problem.

Finite element algorithms can be classified into two categories: implicit and explicit algorithms. In implicit algorithms, a matrix system has to be solved one or more times per step through an iterative procedure in order to obtain a force equilibrium and advance the analysis. This type of technique generally has an advantage regarding numerical stability. In many cases, an “Unconditional Stability” may be obtained, resulting in no time step restrictions caused by stability considerations [40].

However, in explicit algorithms, the solution can be advanced without solving a system of equations. It generally requires that small time steps need to be taken to ensure numerical stability. Although, there are cases when step-size restrictions are more stringent than accuracy considerations might require. Though, on the other hand, due to the lack of equation solving the computational cost per step is generally much less for explicit algorithms than for implicit. None of the algorithms are perfect for all analysis situations, therefore an optimal one has to be chosen for a specific case.

Implicit algorithms are mostly used for linear and non-linear static or quasi-static analyses where slow loading conditions are applied. For non-linear analyses where a model is subjected to large amounts of plasticity or a complicated model with a high number of interactions is used, the unconditionally stable implicit method will encounter some difficulties [86]. In such cases, in order for the system to converge time increment has to be reduced and as the reduction of the time increment continues, the computational cost in the tangent stiffness matrix is dramatically increased and can even cause divergence. Furthermore, local instabilities cause force equilibrium difficult to be achieve. Therefore, the implicit algorithm loses its advantages and as the time increment for convergence approaches the stable time of an explicit analysis, the explicit method becomes more viable to use.

Explicit algorithms are mostly used for large, fine mesh, high phased and short analyses. For an explicit solution, the CPU cost per increment is approximately proportional to the size of the model. There is no drastic increase in memory or processing time as the problem size or complexity increases

such as that associated with an implicit method [78]. However, the time increment in the explicit analysis depends only on the dimension of the elements and the properties of the materials used (see eq. 2.2). Therefore long static or quasi-static analyses even if not complex, become less feasible as it would take an extensive amount of computational time to complete them.

The stable time increment for explicit algorithm is defined as:

$$\Delta t \leq \frac{2}{\omega_{max}} \tag{2.1}$$

where ω_{max} is the element maximum eigenvalue. A conservative estimate of the stable time increment is given by the minimum taken over all the elements. The above stability limit can be rewritten as

$$\Delta t = \min\left(\frac{L_e}{c_d}\right) \tag{2.2}$$

where L_e is the characteristic element dimension and c_d is the current effective, dilatational wave speed of the material.

Regarding modeling and analyzing masonry, the most widely used method is implicit, mostly due to a large number of FEM software that supports such method. Despite this, masonry FEM analysis would greatly benefit from an explicit algorithm as big models, earthquake loading, extensive amounts of damage and cracking heavily slow down implicit analyses as well as produces extensive amounts of convergence problems. However, in quasi-static analyses due to infeasibility to simulate long periods of time, the simulated time period has to be shortened, this induces undesirable kinematic effects onto the structure. On the other hand, these effects can be neglected if kinetic energy is less than 5% of total strain energy [40].

For further reading about implicit and explicit algorithms reader is referred to [7, 40, 41, 78, 82, 86].

2.2. Theory of plasticity

A fundamental difference between elastic and inelastic behavior is that in the elastic analysis the total stress can be evaluated from total strain alone, however, in inelastic solutions the total stress at time t also depends on stress and strain history (see Figure 2.1). There are a very large number of developed material models that simulate the distinctive phenomena of plasticity i.e. elastoplasticity, creep, and viscoplasticity [6].

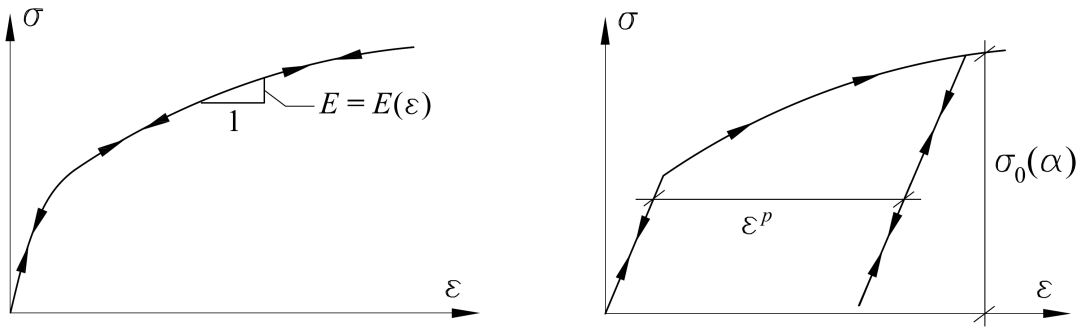


Figure 2.1: Material nonlinearity (from[45]): (a) non-linear elasticity (b) elasto-plasticity.

Generally, for metals the theory required to describe plastic flow is simplistic since metals are insensitive to isotropic pressure, are generally incompressible and with approximation follow associated flow rules. For other materials such as concrete, rocks, masonry, fiber reinforced composites etc. the conditions are more complicated, but regardless, falling within the frame work of plasticity theory.

The theory of plasticity employs some of the fundamental concepts such as the yield criterion, the flow rule, and the consistency conditions. The yield criterion describes the limit at which material becomes plastic and the consistency conditions prevent stresses from exceeding that prescribed limit, while the flow rule describes the relationship between strains and stresses once material enters plasticity.

2.2.1. Yield criteria

One of the first proposals for criteria of yielding of plastic solids, mainly soils, was made by Coulomb [13] in 1776, and had been applied by Poncelet [77] in 1840 and Rankine [81] in 1853 to problems such as the calculation of earth-pressure on retaining walls. At the end of 19th century, Mohr generalized the criterion and it became widely used, and known as Mohr-Coulomb yield criterion [95].

The criterion is pressure sensitive and is best used in describing the materials whose behavior is strongly depending on yield limit and hydrostatic pressure i.e. materials like soils, rocks or concrete. The Mohr–Coulomb criterion is based on the assumption that the phenomenon of macroscopic plastic yielding is, essentially, the result of frictional sliding between material particles [17]. The general formulation of Coulomb's friction law states that critical combination (see eq. 2.3) of normal stress σ_n and shear stress τ triggers a plastic yielding.

$$\tau = c - \sigma_n \tan\varphi \tag{2.3}$$

where σ_n is tensile positive normal stress, c is the cohesion and φ is the angle of internal friction (see Figure 2.2).

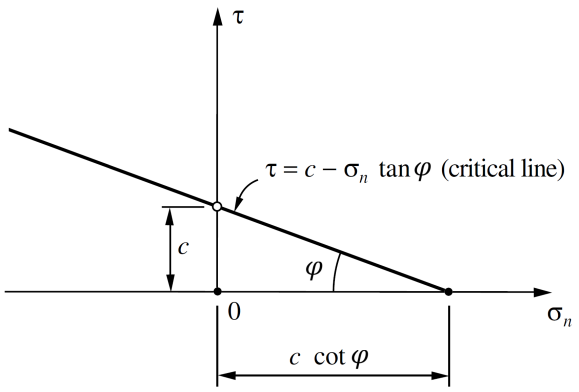


Figure 2.2: The Mohr–Coulomb criterion. Mohr plane representation.

The Mohr-Coulomb multi-surface representation (see Figure 2.9) can be derived from a yield function expressed in terms of the principle stresses (see eq. 2.4). Such representation consists of 6 yield surfaces described by every combination of σ_{max} and σ_{min} in the yield function, whose roots are $\Phi_i(\boldsymbol{\sigma}) = 0$.

$$\Phi(\boldsymbol{\sigma}) = (\sigma_{max} - \sigma_{min}) + (\sigma_{max} + \sigma_{min})\sin\varphi - 2c\cos\varphi \tag{2.4}$$

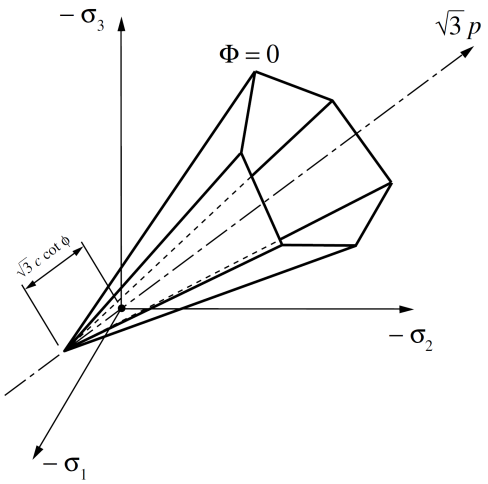


Figure 2.3: The Mohr–Coulomb criterion. Multi-surface representation in principal stress space (from [17]).

The first scientific study of the plasticity started from a study of plastic behavior in metals in 1884. At that time Tresca [89] published a preliminary account of experiments on punching and extrusion, which led him to state that a metal yielded plastically when the maximum shear stress attained a critical value. Unlike in the Mohr-Coulomb model in Tresca's findings metals were not hydrostatic pressure sensitive, therefore a new yield criterion in principal stress space was assembled (see eq. 2.5) .

$$\Phi(\boldsymbol{\sigma}) = \sigma_{max} - \sigma_{min} - \sigma_y \tag{2.5}$$

Nevertheless, like Mohr-Culoumb, Tresca criterion can be represented as 6 surfaces in principle stress space (see Figure 2.4) composed from all possible combinations of σ_{max} and σ_{min} in the yield function (eq. 2.5) whose roots are $\Phi_i(\boldsymbol{\sigma}) = 0$.

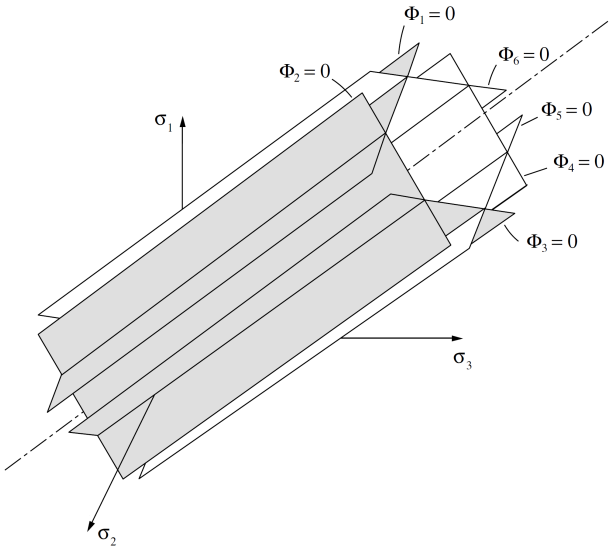


Figure 2.4: The Tresca criterion. Multi-surface representation in principal stress space (from [17]).

After the findings of Tresca, there were numerous suggestions of the yield criteria for metals, however, more accurate works were yet to be presented. In 1913 von Mises [67] made an advancement in the definition of yield criterion for metals purely from a mathematical point of view. He described the failure in terms of second stress invariant J_2 reaching a critical value.

$$J_2 = k(\alpha) \tag{2.6}$$

where k is critical value assumed to be a function of internal hardening variable α and mostly used with a relation $\tau_y = \sqrt{k}$, where τ_y is the shear yield stress. Therefore, a yield function (see Figure 2.5) for the von Mises criterion can be defined as

$$\Phi(\boldsymbol{\sigma}) = \sqrt{J_2(\mathbf{s}(\boldsymbol{\sigma}))} - \tau_y \tag{2.7}$$

In 1952 a smooth approximation to Mohr-Coulomb law (see Figure 2.6) was proposed by Drucker and Prager [20]. The new criterion was derived from von Mises criterion by adding an extra term to introduce pressure sensitivity. The Drucker–Prager criterion states that the critical combination of hydrostatic pressure p and second deviatoric stress invariant J_2 triggers the yielding of a material. Therefore, the yielding is initiated when the following equation is satisfied

$$\sqrt{J_2(\mathbf{s})} + \eta p = \bar{c} \tag{2.8}$$

where η and $\bar{c}$ are material parameters. However, in order to approximate the Mohr-Coulomb yield surface, Drucker-Prager's yield function can be defined as follows

$$\Phi(\boldsymbol{\sigma}) = \sqrt{J_2(\mathbf{s}(\boldsymbol{\sigma}))} + \eta p(\boldsymbol{\sigma}) - \xi c, \tag{2.9}$$

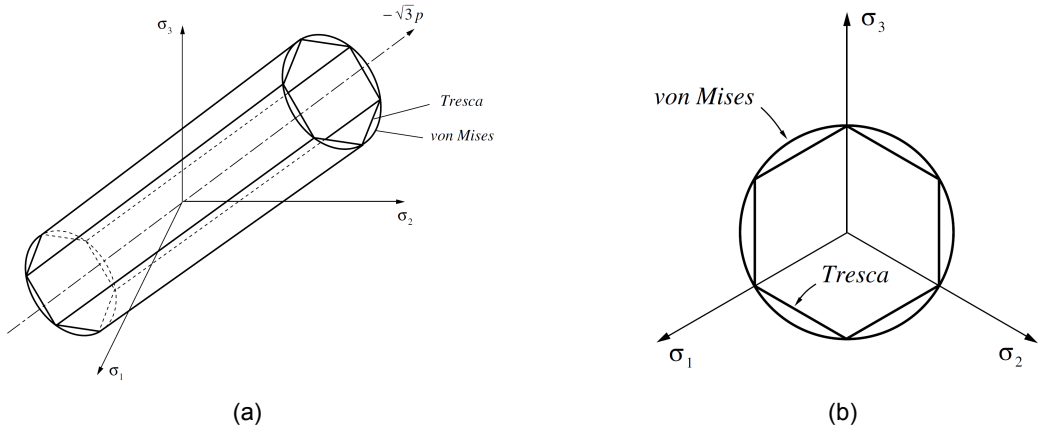


Figure 2.5: The Tresca and von Mises yield surfaces in principal stress space (adapted from [17]): (a) 3d view and (b) view along hydrostatic axis.

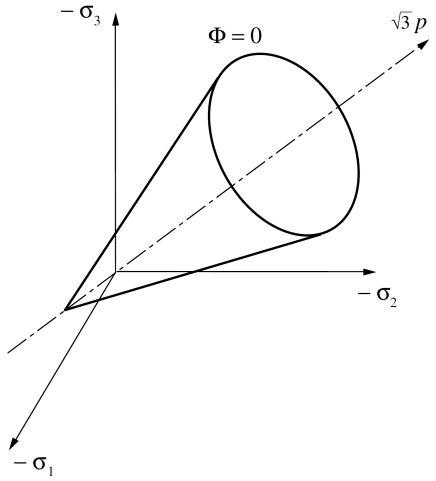


Figure 2.6: The Drucker–Prager yield surface in principal stress space (from [17]).

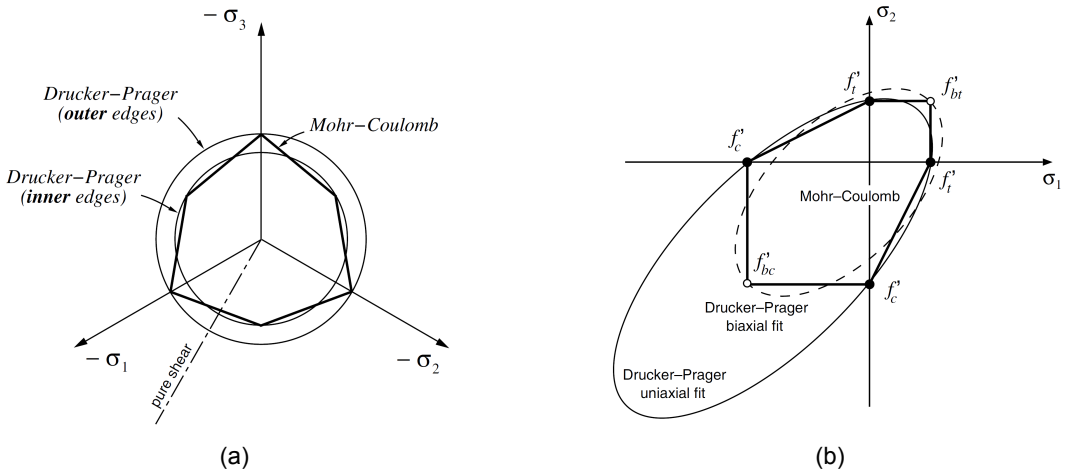


Figure 2.7: Drucker–Prager approximations (adapted from [17]): (a) approximations in principle stress plane (b) approximations matching the Mohr–Coulomb surface in uniaxial tension and uniaxial compression.

where c is the cohesion and the parameters η and ξ define the required approximation to the Mohr-Coulomb criterion.

The stated yield criteria form the basis of yield descriptions in material mechanics. From these, other criteria are derived and described. It must be pointed out that all of the criteria previously mentioned are of isotropic nature. Hill [37] in 1948 described an anisotropic version of von Mises criterion, a modification of it will be used in this thesis as a part of the description of the failure envelope. For further information on various other yield criteria reader is referenced to [8, 9, 11, 18, 23, 35]

2.2.2. The plastic flow and stress return

The plastic flow is classified according to the relation of plastic strain increment and deviatoric stress directions. If direction of plastic strain increment is equivalent to the direction of the deviatoric stress it is called associated flow rule. However, if they are not equivalent it is called non-associated flow rule. The former is usually observed in metals, while the latter is valid for other materials such as rock, concrete and masonry. From here the yield function $f(\boldsymbol{\sigma})$, and the flow function $g(\boldsymbol{\sigma})$ can be defined. For associated flow rule $f(\boldsymbol{\sigma}) = g(\boldsymbol{\sigma})$ and for non-associated flow rule $f(\boldsymbol{\sigma}) \neq g(\boldsymbol{\sigma})$.

The direction of the plastic strain is perpendicular to the flow surface and is equal to the direction of the flow vector for the isosurface $\Psi(\mathbf{A})$ (see Figure 2.8), where $\Psi(\mathbf{A})$ is $f(\boldsymbol{\sigma})$ or $g(\boldsymbol{\sigma})$ depending on the type of the flow:

$$N = \frac{\partial \Psi(\boldsymbol{\sigma})}{\partial \boldsymbol{\sigma}} \tag{2.10}$$

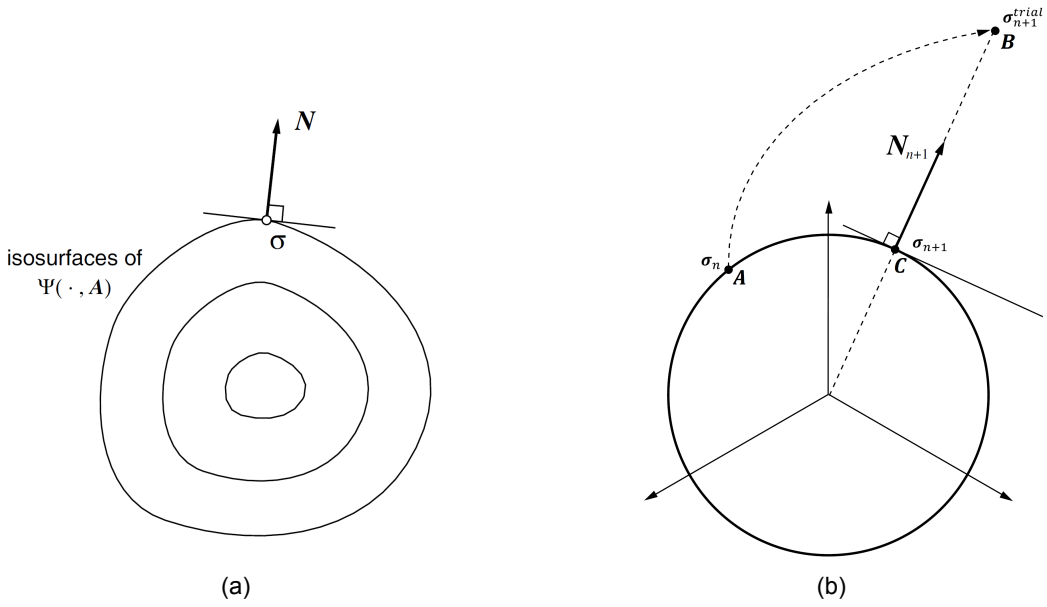


Figure 2.8: Smooth potential (from [17]). (a) The flow vector and (b) stress return represented in a plane perpendicular to the hydrostatic pressure line in principal stress space

For finite element calculation, not only the direction of the flow is important but also the magnitude $d\lambda$ of the stress return. Let $\boldsymbol{\sigma}_n$ be a current stress, $\boldsymbol{\sigma}_{n+1}^{trial}$ be trial stress for the current calculation increment and $\boldsymbol{\sigma}_{n+1}$ be the final stress. The trial stress increment can then be defined as follows:

$$d\boldsymbol{\sigma}_n^{trial} = \boldsymbol{\sigma}_{n+1}^{trial} - \boldsymbol{\sigma}_n = \mathbf{D} d\boldsymbol{\varepsilon} \tag{2.11}$$

The “real” stress increment is equal to

$$d\boldsymbol{\sigma}_n = d\boldsymbol{\sigma}_n^{trial} - \mathbf{D} d\boldsymbol{\varepsilon}_{pl} = \mathbf{D}\left(d\boldsymbol{\varepsilon} - d\boldsymbol{\varepsilon}_{pl}\right) \tag{2.12}$$

In order to return back to the yield surface, the plastic strain increment ($d\boldsymbol{\varepsilon}_{pl}$) has to be found. Which can be expressed as:

$$d\boldsymbol{\varepsilon}_{pl} = d\lambda \frac{\partial g}{\partial \boldsymbol{\sigma}} \tag{2.13}$$

The plastic strain is determined by two quantities the scalar $d\lambda$ and the gradient of the loading surface $(\partial g/\partial \boldsymbol{\sigma})$ giving the direction. To determine the magnitude $d\lambda$, the following condition must be satisfied:

$$f(\boldsymbol{\sigma}_B - d\boldsymbol{\sigma}_p) = 0 \tag{2.14}$$

Where plastic corrector $d\boldsymbol{\sigma}_p$ can be expressed as:

$$d\boldsymbol{\sigma}_p = d\lambda \mathbf{D} \frac{\partial g}{\partial \boldsymbol{\sigma}} \tag{2.15}$$

Considering Taylor expansion of the yield surface around the point B we have:

$$f(\boldsymbol{\sigma}_B - d\boldsymbol{\sigma}_p) \simeq f(\boldsymbol{\sigma}_B) - \left(\frac{\partial f}{\partial \boldsymbol{\sigma}}\right)_B^T d\boldsymbol{\sigma}_p = 0 \tag{2.16}$$

Inserting eq. 2.15 to eq. 2.16 we obtain the following:

$$d\lambda = \frac{f(\boldsymbol{\sigma}_B)}{\left(\frac{\partial f}{\partial \boldsymbol{\sigma}}\right)_B^T \mathbf{D} \left(\frac{\partial g}{\partial \boldsymbol{\sigma}}\right)_C} \tag{2.17}$$

One should note that this solution is only valid if flow direction at point B and the flow direction at point C is the same. Otherwise, iterations are necessary. However, if the flow function has discontinuities or flow is described by several functions, the trial stress that falls in the sub-differential set of Ψ (see Figure 2.9) should always return to the point of discontinuity $\boldsymbol{\sigma}$. The sub-differential set of Ψ is bounded by a set of normals as follows:

$$N_i = \frac{\partial \Psi_i}{\partial \boldsymbol{\sigma}} \tag{2.18}$$

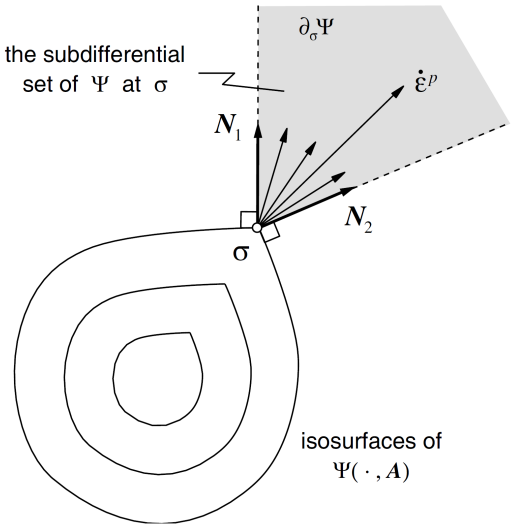


Figure 2.9: The flow vector (from [17]). Non-smooth potential.

2.2.3. Stress and strain hardening

Most materials exhibit some degree of hardening as an addition to plastic straining. The hardening of the material manifests as the change of the size and the shape of the yield surface during plastic loading. This change is complex and an accurate description of it is rather difficult to obtain. Therefore, hardening is often described as a combination of two different types of hardening, namely isotropic hardening and kinematic hardening (see Figure 2.10).

A material is said to experience isotropic hardening when the evolution of the yield surface is such that, at any state of hardening, it can be described as an isotropic (uniform) expansion, without translation, of the initial yield surface. In the case of von Mises yield criterion the elastic domain expands

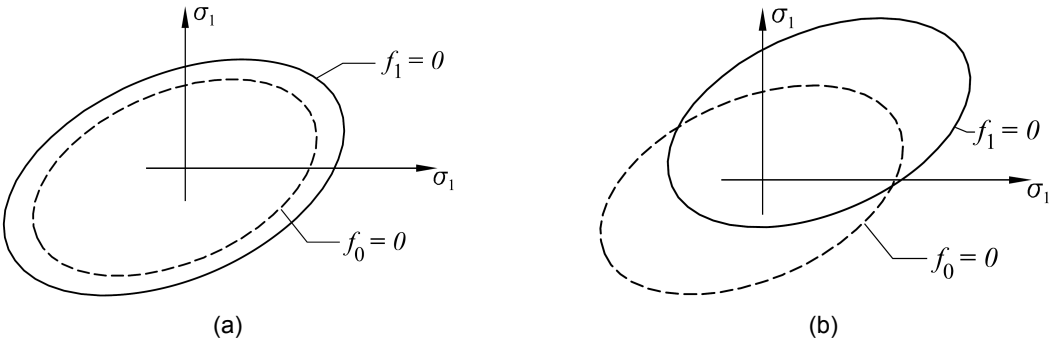


Figure 2.10: Types of hardening. (a) Isotropic and (b) Kinematic hardening

equally in tension and compression during plastic flow, i.e. isotropic hardening corresponds to the increase in the radius of the von Mises cylinder in principal stress space. Therefore, the hardening can be described by the following:

$$f(\boldsymbol{\sigma}) - \sigma_0(\alpha) = 0 \tag{2.19}$$

where $\sigma_0(\alpha)$ is the material strengths depending on dynamic hardening parameter α when the following is always true $\dot{\alpha} \geq 0$. On the other hand, for orthotropic materials, isotropic hardening is controlled by changing each strength parameters in yield criteria $\boldsymbol{\sigma}_0(\alpha)$.

Kinematic hardening takes place when the shape and size of a yield surface is preserved but it can translate in the stress space. Lemaitre and Chaboche [50] showed that many materials when being loaded and hardened in one direction, exhibit a decreased resistance to plastic yielding in the opposite direction. The phenomenon is called Bauschinger-effect and it can be simulated by introducing kinematic hardening.

In the case of von Misses yield criterion kinematic hardening is introduced by replacing stress vector in the criterion by the relative stress tensor (see eq. 2.21) that depends on stress deviator $\mathbf{s}(\boldsymbol{\sigma})$ and the symmetric deviatoric stress-like tensor, $\boldsymbol{\beta}$, also known as backstress tensor.

$$f(\boldsymbol{\sigma}, \boldsymbol{\beta}) = \sqrt{3J_2(\boldsymbol{\eta}(\boldsymbol{\sigma}, \boldsymbol{\beta}))} - \sigma_0 \tag{2.20}$$

where

$$\boldsymbol{\eta}(\boldsymbol{\sigma}, \boldsymbol{\beta}) = \mathbf{s}(\boldsymbol{\sigma}) - \boldsymbol{\beta} \tag{2.21}$$

There are a number of studies that propose a constitutive model for the definition of the elasto-plastic behavior under cyclic loading conditions [50, 69, 85]. However, only the isotropic hardening behavior will be implemented in this thesis as kinematic hardening behavior for composite material such as masonry is poorly documented and any effort to implement it would be purely phenomenological. Furthermore, considering explicit integration analysis, it is likely that kinematic hardening behavior in the simulations of structures would appear naturally to some extent, without extra considerations in the material model.

2.3. Continuum Damage Mechanics

In 1958 Kachanov [43] introduced a theory of continuum damage mechanics that provided a powerful and general framework for the derivation of consistent material models suitable for many engineering fields and the term itself was defined by Janson and Hult [42]. The advantage of continuum damage mechanics is that it can be used to simulate a wide range of materials such as ceramics, plastics, metals, rock, concrete, and masonry. It is due to the simplistic approach in modeling damaged materials. Such materials are assumed to remain continuous and the effect of cracks is modeled by changing the mechanical properties, such as strength and stiffness.

Kachanov [43] defined the collective effect of deterioration by means of a field variable named continuity and denoted ψ. Completely deterioration free material was given the condition $\psi = 1$ and completely damaged continuous material with no remaining load caring capacity was defined as $\psi = 0$. As continuity ψ defines absence of defects, the variable $D = 1 - \psi$ defines state of deterioration or damage [70, 79].

Continuum damage mechanics is a counterpart of fracture mechanics, where the fracture is simulated by embedding a fixed crack in a usually non-degradable material. However, both of these approaches can be combined to simulate crack growth and degradation of load carrying capacity [42, 47].

The damage itself in solids is considered as discontinuities in the medium that on a macro scale is continuous. These discontinuities are generally caused by micro-cracks and micro-voids. The biggest volume over which such material can be treated as homogeneous is called representative elementary volume (REV) or unit cell (see 2.11). The size of this unit cell can vary from about $0.1\ mm^2$ for metals and ceramics to about $100\ mm^2$ for concrete (see Figure 2.12).

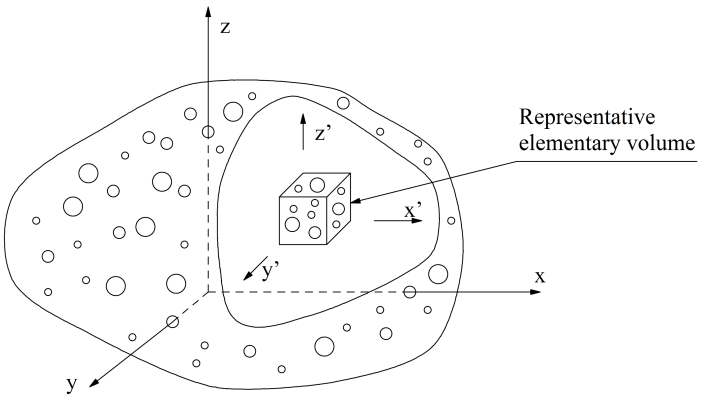


Figure 2.11: Representative elementary volume in a composite body (from [55])

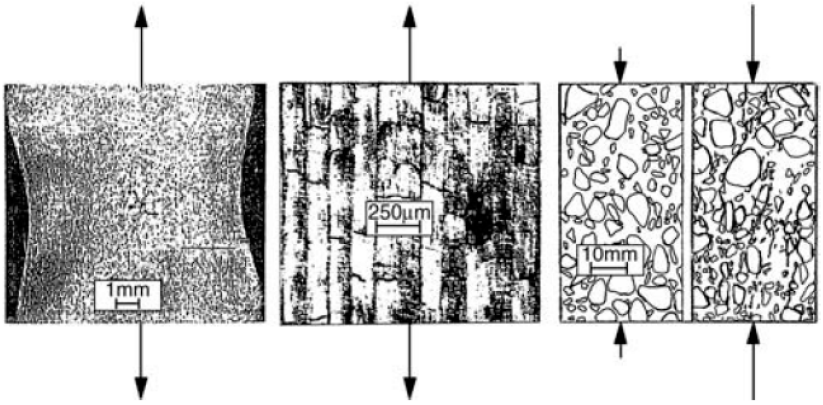


Figure 2.12: Examples of damage (from [51]) in a metal (micro-cavities in copper), in a composite (micro-cracks in carbon-fiber/epoxy resin laminate), and in concrete (crack pattern).

The damage of the material is always related to the plastic strain (permanent strain) and more generally to a strain dissipation in the scale of REV (mesolevel) or the scale of discontinuities (microlevel). In mesolevel three types of damage can occur: the nucleation and growth of voids in mesofield of plastic strains under static loading causes ductile damage; low cycle fatigue damage occurs under repeated high-level loadings and elevated temperatures causes creep damage by intergranular decohesion in metals.

Regarding microlevel, two different kinds of damage can be classified: quasi-brittle and high cycle fatigue damage. The former is when brittle failure is caused during monotonic loading. The latter is when a material is subjected to a loading of a large number of repeated cycles. Materials like ceramics, concrete, and metals exhibit this type of damage.

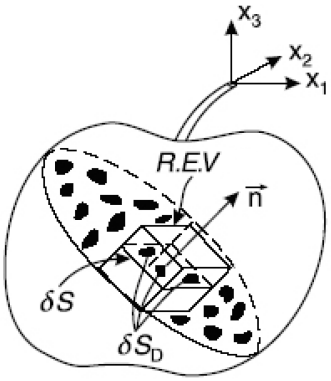


Figure 2.13: Damaged element and interpretation of the damage variable (from [75]).

Considering an isolated unit cell (REV) in a damaged solid and letting dS be the area of the section of the unit cell identified by it's normal $\vec{n}$. On the section, cracks and voids leave traces and reduce the effective area of resistance $d\bar{S}$ where ($d\bar{S} < dS$). Let dS_D be the difference between the area and the effective area:

$$dS = dS - d\bar{S} \tag{2.22}$$

The Damage variable D can be defined as the surface density of micro-cracks [51] therefore it can be written as a ratio between the area of the voids and cracks, and the area of the section:

$$D(\vec{n}) = \frac{dS_D}{dS} \tag{2.23}$$

The above expression provides the measure of local damage relative to the direction $\vec{n}$. $0 \leq D(\vec{n}) \leq 1$ characterizes the damaged state, where $D(\vec{n}) = 0$ is equivalent to initial undamaged state and $D(\vec{n}) = 1$ – to fully damaged material.

However, if the damage is isotropic, the crack and cavities are distributed evenly in all directions. Therefore, the damage variable $D(\vec{n})$ can be completely described by the scalar intrinsic variable d:

$$D(\vec{n}) = d \qquad \forall \vec{n} \tag{2.24}$$

On the other hand, in the general case of anisotropic damage the variable $D(\vec{n})$ depends on the orientation of the normal. It was shown that during loading history, the micro cracks undergo irreversible growth in the direction perpendicular to the maximal tensile strain [24, 46]. The concept of the damage variable can further be advanced to the formulation of effective stress.

If F is the applied force onto the cross section of the unit cell, the uniaxial case is considered due to simplicity. The stress in the cross section is $\sigma = F/S$ satisfying the equilibrium conditions. Therefore, if the isotropic damage d is present, the effective area of resistance is

$$\bar{S} = S - S_D = S(1 - d) \tag{2.25}$$

hence effective stress is defined by

$$\bar{\sigma} = \sigma \frac{S}{\bar{S}} = \frac{\sigma}{1 - d} \tag{2.26}$$

In case of multi-axial isotropic damage, the damage does not depend on the orientation of the normal and the operator $(1 - d)$ can be applied to all of the components. Therefore, we can consider the tensorial form

$$\bar{\boldsymbol{\sigma}} = \frac{\boldsymbol{\sigma}}{1 - d} \tag{2.27}$$

or the inverse expression

$$\bar{\boldsymbol{\sigma}} = \boldsymbol{\sigma}(1 - d) \tag{2.28}$$

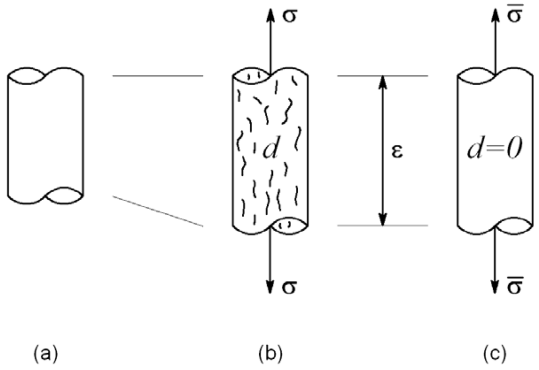


Figure 2.14: Effective stress and equivalence in strain (from [75]): virgin material (a), damaged material (b) and equivalent virgin material (c).

The hypothesis of the strain equivalence states that the strain associated with a damaged state under applied stress $\boldsymbol{\sigma}$ is equivalent to the strain associated with its undamaged state under effective stress $\bar{\boldsymbol{\sigma}}$ [49] (see Figure 2.14). Therefore, by using simple relation $\bar{\sigma} = E\varepsilon$ we obtain:

$$\sigma = (1-d)\bar{\sigma} = (1-d)E\varepsilon \tag{2.29}$$

Where E is Young's modulus. From which can be derived that actual tension stress σ is related to strain by means of a damaged stiffness:

$$E_d = (1-d)E \tag{2.30}$$

The damage is irreversible, therefore

$$\dot{S}_d \geq 0, \quad \dot{d} \geq 0 \quad \rightarrow \quad \dot{E}_d \leq 0 \tag{2.31}$$

The damage is only initiated if when the stress or strain exceeds initial failure threshold σ_0 (or ε_0):

$$d = 0 \;\; if \;\; \begin{cases} \sigma < \sigma_0 \\ \varepsilon < \varepsilon_0 \end{cases} \tag{2.32}$$

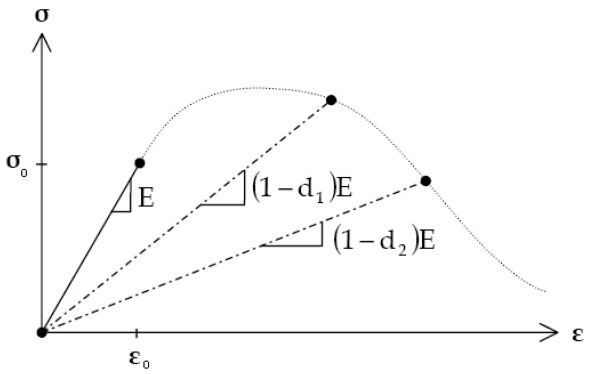


Figure 2.15: Damaged Young's modulus during increasing uniaxial load (from [75]).

In case of unloading we have:

$$\dot{\varepsilon} < 0 \quad \rightarrow \quad \dot{S}_d = 0 \quad and \quad \dot{d} = 0 \tag{2.33}$$

and, therefore,

$$\dot{\sigma} = (1-d)E\dot{\varepsilon} - dE\dot{\varepsilon} = E_d\dot{\varepsilon} \tag{2.34}$$

Unloading does not increase the damage, it will, however, follow the unloading branch according to the damaged stiffness until the point of origin. A further reloading follows the same loading paths until a failure criterion is reached once more. The damage constitutive law does not allow any occurrence of irreversible plastic deformation as after unloading all the deformations are recovered. However, the actual behavior of materials, especially masonry, do not fully comply with this law. As described in section 1.2 plasticity occurs in actual physical tests and the behavior of such mediums as masonry is not isotropic. Therefore, some additional considerations are necessary.

2.4. Damage models and failure envelopes for masonry

There were only a few attempts to describe the general failure criteria for masonry, due to the difficulties involved in developing a representative biaxial test and the fact that a large number of such tests is required. This problem was discussed by Yokel and Fattal [94] with reference to the failure of shear walls. Further efforts were made by Dhanasekar et al. [19] where they interpolated the test data of Page [72, 73] by means of three elliptic cones (see Figure 2.16). However, as it is described in the paper, the cones do not correspond with the observed distinct modes of failure. The elliptic cones have been expressed by a second-order tensor polynomial. A significant review of the subject can be found in [3, 36, 68].

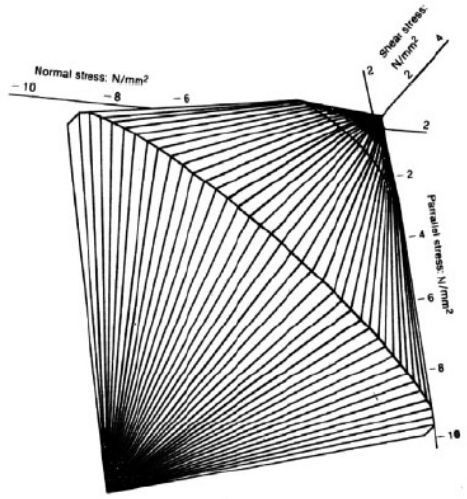


Figure 2.16: Failure surface idealized by Dhanasekar et al. [19].

There were several proposals to use existing yield criteria of available composite materials for the expression of analytical failure models of masonry. For example, Syrmakesis and Asteris [88], used a Tsai and Wu [91] cubic tensor polynomial. Nevertheless, the adaptation did not return a satisfactory approximation of Page's [72] experimental data (see Figure 2.17)

The further advancements in masonry behavior modeling were made by Lourenco [56]. He proposed a multi-surface plane-stress softening plasticity model for a masonry failure simulation. The model consists of two yield criteria: one Rankine-like for failure prediction of joints and one Hill-like for failure prediction of units (see Figure 2.18). He described different stress-strain behavior laws in tension and compression. In tension exponential softening law was adopted along material axes and in compression, he applied an isotropic parabolic hardening law followed by a parabolic/exponential softening law with different compressive fracture energies along each material axes. Therefore, the principal directions of damage are fixed and aligned with the initial orthotropy axes. However, although the model adopted two different fracture energies, a single internal scalar variable controlled the plasticity algorithm in order to determine the softening in two material axes. Furthermore, the damage was only described in terms of softening of the material strengths and the degradation of stiffness was not present.

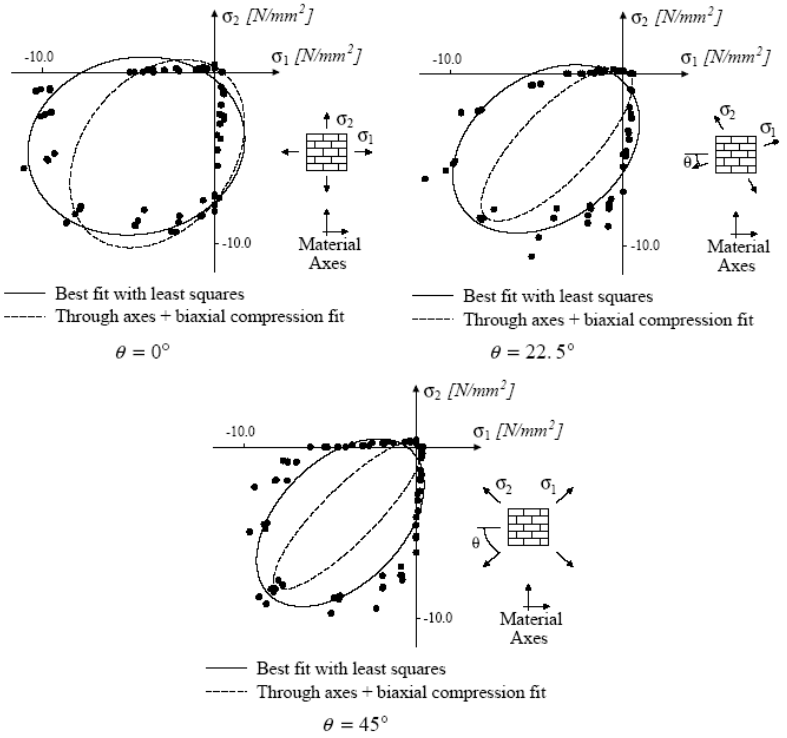


Figure 2.17: Comparison between experimental results from Page [72, 73] and a Hoffman type yield surface (from [56]).

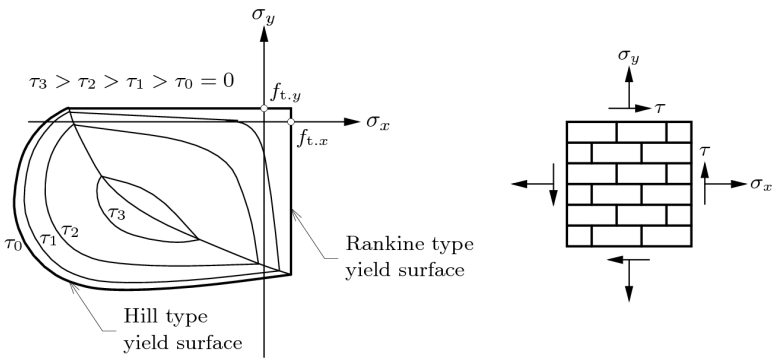


Figure 2.18: Multi-surface failure envelope in Cauchy stress space (from [56]).

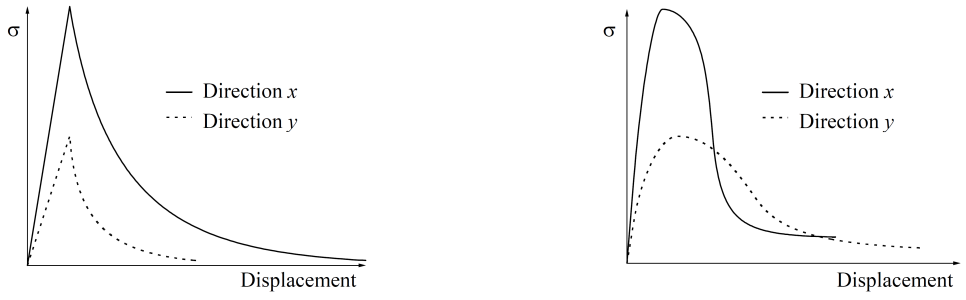


Figure 2.19: Behavior of the model (from [57]) for (a) tension and (b) compression, along two orthogonal directions.

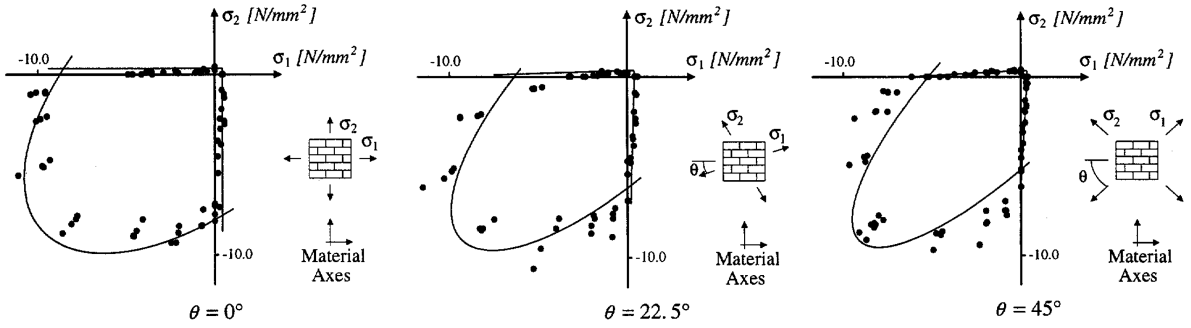


Figure 2.20: Comparison between Lourenco model and experimental results from Page [72, 73] (from [56]).

Since the works of Lourenco there was no further significant development of masonry constitutive material models until Pela [75] proposed the use of isotropic yield surfaces mapped on to orthotropic space as viable analysis approach for masonry structures. His model consisted of isotropic Faria [23] and isotropic Rankine-type failure envelopes (see Figure 2.21a). Furthermore, he used a modified method proposed by Oller et al.[71] where authors multiplied the transformation tensor by a “shape adjustment tensor”, whose purpose is to adjust the isotropic criterion to the desired orthotropic one. But, the shape adjustment tensor must be derived by means of an iterative procedure. Thus, the non-linear solution of a quadratic system by the Newton-Rapson method is required which is not an easy task and is quite costly, since the shape adjustment tensor depends on the stress state at the point at each instant of the mechanical process. To circumvent this, Pela [75] simplified Oller’s et al. method to a standard form of a stress transformation tensor and obtained similar results to Lourenco [56] (see Figure 2.22).

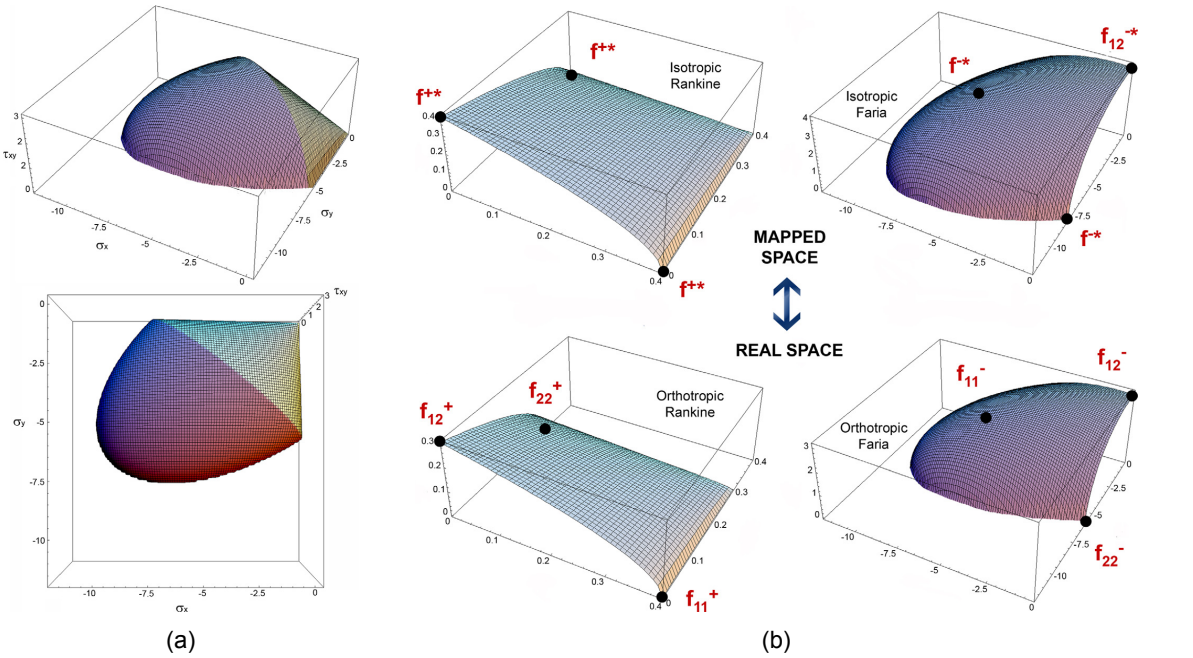


Figure 2.21: Yield surfaces used by Pela [75]: (a) Composite yield surfaces in Cauchy stress space and (b) mapping each surface from fictitious isotropic stress space to real stress space.

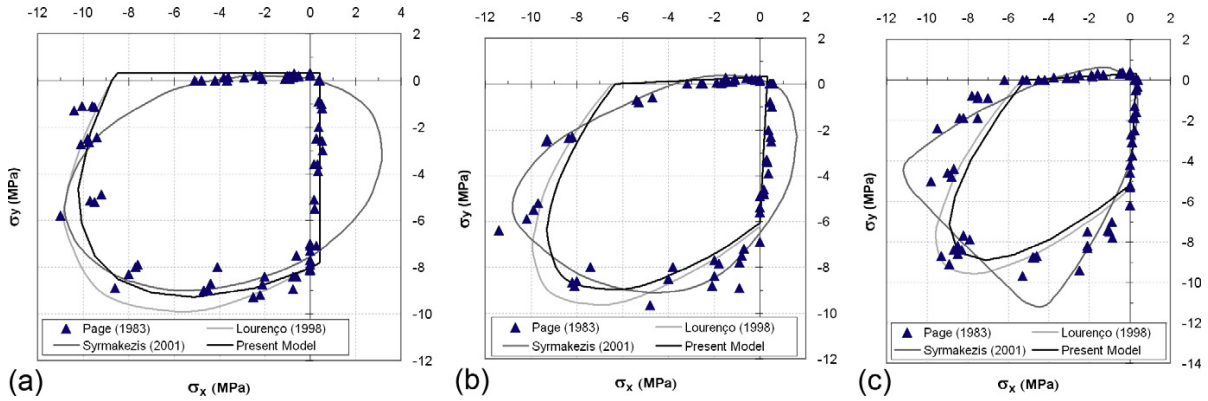


Figure 2.22: Comparison between Pela, Lourenco, Symakezis models and experimental results from Page [72, 73] (from [75]).

Like in Lourenco's [56] model in Pela's [75] work, the damage was modeled by softening of the material strengths, but additionally, Pela used stiffness damage model proposed by Papa [74]. The model allows stiffness recovery when loading changes from tension to compression which is in agreement with experimental results. On the other hand, it does not allow any permanent plastic strains to occur, while it can be true for tensile behavior, it is not in agreement with the cyclic compression test results [21].

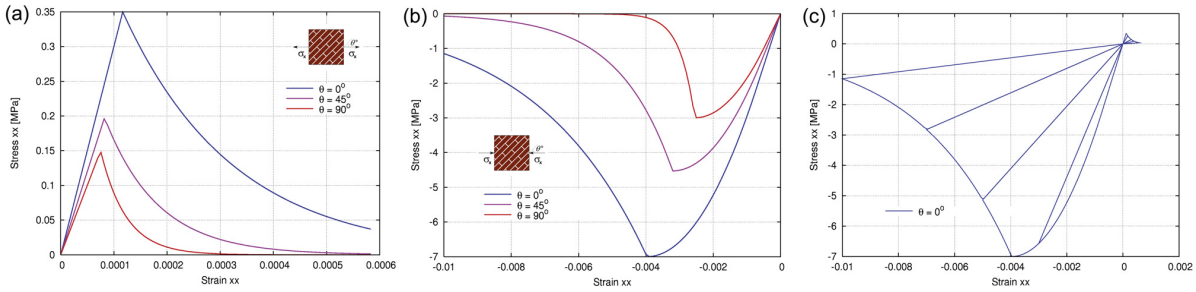


Figure 2.23: Behavior of the model (from [75]) for (a) tension and (b) compression, along two orthogonal directions. (c) represents cyclic behavior

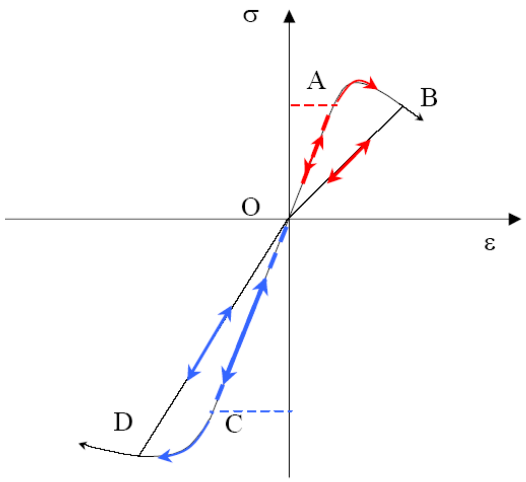


Figure 2.24: Damage model proposed by Papa [74] (from [75]).

2.5. Simulia Abaqus user material interface

Before writing a material model it is important to understand how custom user material (VUMAT) interfaces with Simulia Abaqus. This section will discuss the capabilities of user subroutines for Abaqus / Explicit package.

Figure 2.25 portrays the location of execution of VUMAT. The scheme is composed out of the experience gathered when working with various Abaqus subroutines and can differ from the actual execution procedure to some extent. The Abaqus manual does not discuss the actual global flow of explicit analysis.

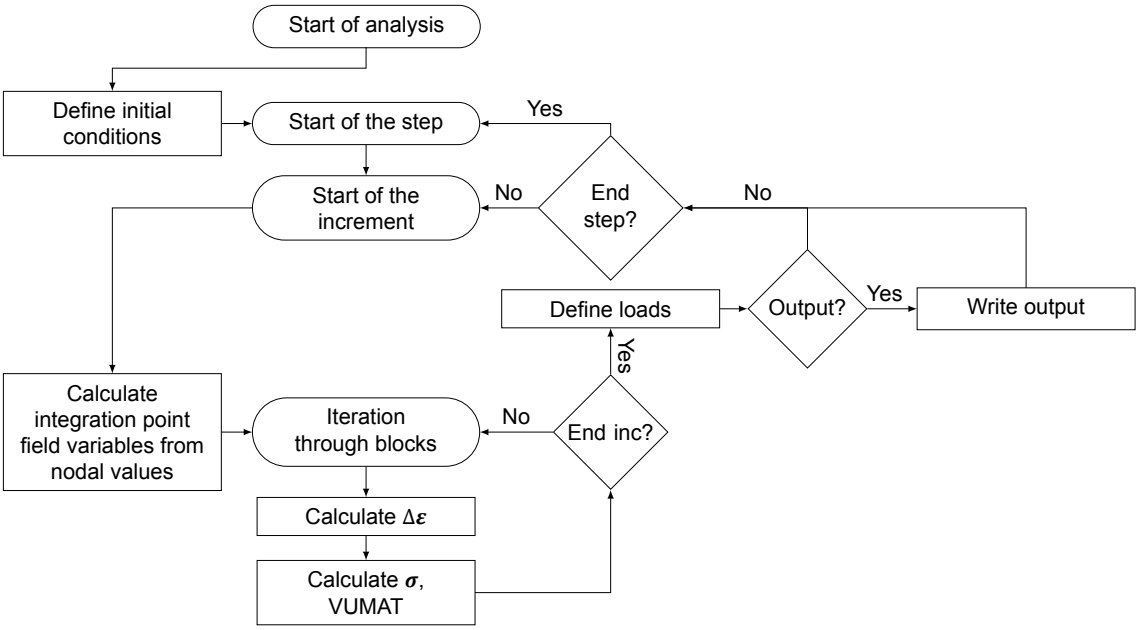


Figure 2.25: Global flow of Abaqus/Explicit

When the whole model is prepared for the analysis it's integration points are subdivided into blocks containing the maximum number of 136 points. For each of the blocks, the user subroutine is called. Each block only contains the integration points of the same material, and inside the subroutine, these points have to be iterated to provide the output for all of them. Furthermore it is important to note, that there is only one integration point per element in the explicit analysis. The exact procedure of user subroutine used for the thesis is described in Figure 3.1.

The User subroutine must output new stress vector and can output optional vectors such as state variables, stretch tensors, deformation gradients, temperature values, total internal energies, inelastic energies and field variables. The material properties for the calculation are passed to the subroutine through the `props` array, this array should be defined in the input file of the analysis. The strain is passed to the subroutine as strain increment and together with stresses have a direction corresponding to the local material orientation. The orientation can be defined in the input file and by default, it corresponds to the global axes of the analysis.

The state variables internally denoted as `SDV`s, act as information storage for a specific element and at the same time they can be used as output for the end user. Element deletion can be controlled by one of the variables and the `SDV` number for deletion control should be specified in the input file.

The energy output is beneficial as large sudden energy gain or loss in the system can signify a problem with the analysis. Most often such problem is caused by mesh distortion, interface elements, damping or drift of the solution and in order to let Abaqus calculate energies correctly, total internal energy and inelastic internal energy values have to be outputted from the VUMAT.

The other output possibilities such as temperature values, stretch tensors, deformation gradients and field variables fall outside the scope of this thesis and will not be used in the definition of the material model. For a full description of the input and output of the VUMAT subroutines the reader is advised to refer Abaqus Manual [15] Chapter 1.2.22.

As for the control of the time incrementation, there is no possibility to do it directly in VUMAT, however, Abaqus uses the subroutine indirectly to determine the stable time. It is done by dividing the analysis into two phases: phase one when the step time is zero and phase two when $t > 0$. During the time zero dummy values for strain ($\Delta\varepsilon_{1,2,3} = -0.001$ and $\Delta\varepsilon_{12,23,31} = 0.001$) are passed into the subroutine. The subroutine is expected to output stresses as if the material would be perfectly elastic. Abaqus uses these stresses to assemble internal stiffness matrix for calculation of stable time, internal wave speed, eigenmodes etc. During the phase two user subroutine is expected to simulate the material as it was programmed for.

On the other hand, when using VUMAT for a material, some of the analysis functions stop working, one of these is the β component of Rayleigh damping. It is not applied and if such damping is wanted during analysis, it has to be calculated manually inside the subroutine. But this poses a problem, β damping heavily effects stable time increment, therefore special measures have to be taken to counteract and correct the stable time. These measures will be described in subsection 3.5.2 Stable Time.

Although Abaqus provides extensive flexibility and freedom to model custom material as the user wants, there are still shortcomings and behaviors that are not possible to model within VUMAT interface. E.g. in user material subroutine environment it is impossible to know adjacent elements or integration points, thus such algorithms as the one described by Clemente et al. [12], the local crack-tracking algorithm, are impossible to implement without developing a custom element.

2.6. Summary

In this Chapter, basic theoretical considerations were discussed. It was determined that the explicit algorithm is superior to implicit when the complex, highly plastic and rapidly loaded models have to be analyzed, yet, the opposite is true when static or quasi-static analysis are required. Furthermore, a brief overview of Plasticity was presented including the most common yield criteria such as Tresca, Mohr-Coulomb, von Mises and Drucker-Prager. The relation between the deviatoric stress tensor and plastic strain increment were defined together with the types of material hardening.

Followed by an overview of Continuum damage mechanics. Where, the basic concepts of damage variable, effective stress and strain-equivalence have been discussed. Further, available material models for masonry were reviewed. Two main distinctive works were identified, namely the work Lourenco [56] and the work of Pela [75]. Even though Pela used an interesting approach in modeling orthotropic behavior the results were not far from what was obtained by Lourenco. However, the damage model used by Pela is beneficial for the accuracy of cyclic analysis, although some improvements to the damage model could be made. These additional considerations will be further discussed in section 3.4.

And last but not least, the Abaqus user subroutine interface was analyzed and the structure of the analysis procedure was presented. It was determined that the user subroutine must output only new stress vectors each increment in order for the analysis to run. Ofcourse, for better inetgartion and clearer output more considerations are required.

3

Description of the material model

This model uses a simplified approach for 3d space. It means that material strengths in out of plane direction are ignored. The model exhibits plastic behavior in plane and elastic out of the plane. Such way of describing the material allows a fast and easy preparation of structural model for Finite Element Analysis, while in turn providing an easier calibration of the material properties and a faster calculation comparing it to full 3d orthotropic plasticity.

This chapter will present formulations used in describing the material model. The formulation includes orthotropic stiffness matrix, yield and flow surfaces, damage in elasticity and strengths and additional considerations for better integration with Abaqus software. For the simplified scheme of the material subroutine procedure see Figure 3.1.

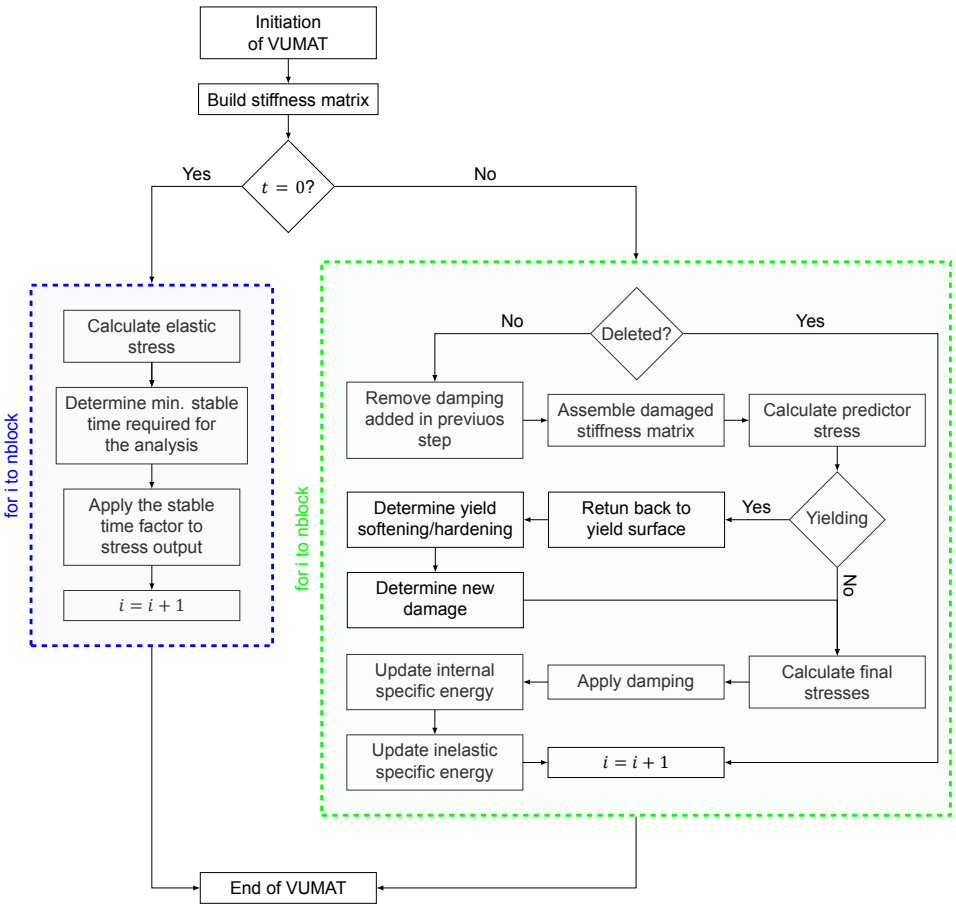


Figure 3.1: Scheme of material subroutine procedure.

3.1. Stiffness matrix

The stiffness matrix ($\mathbf{K}$) is an inverse of compliance tensor ($\mathbf{D}$) which is derived for an element from an elastic stress–strain relationship ($\boldsymbol{\varepsilon} = \mathbf{D}\boldsymbol{\sigma}$). For a purely anisotropic material, such compliance matrix consists of 36 elastic constants. Because of the symmetry of constitutive tensor, only 21 of 36 constants are independent.

$$\mathbf{D} = \begin{bmatrix} D_{11} & D_{12} & D_{13} & D_{14} & D_{15} & D_{16} \\ & D_{22} & D_{23} & D_{24} & D_{25} & D_{26} \\ & & D_{33} & D_{34} & D_{35} & D_{36} \\ & & & D_{44} & D_{45} & D_{46} \\ & \text{Symmetry} & & & D_{55} & D_{56} \\ & & & & & D_{66} \end{bmatrix} \tag{3.1}$$

However, the material would require 21 independent tests for all of its parameters to be determined, which is impractical. Nevertheless, the amount of independent constant can be reduced if the material in question has planes of symmetry. The plane of symmetry is defined as a plane through a material that divides the material into two parts in which the elastic constants are a mirror image of each other. A material is called orthotropic if at every point it has 3 planes of symmetry. The intersections of these planes are called principle axes of orthotropy.

In case a material is orthotropic the amount of independent elasticity constants is reduced to 9 (see eq. 3.2). These constants can be expressed in 12 engineering therms, i.e. 3 Young moduli E_1, E_2, E_3, 6 Poisson's ratios ν_{12}, ν_{13}, ν_{21}, ν_{23}, ν_{31}, ν_{32} and 3 shear moduli G_{12}, G_{13}, G_{23} (see eq. 3.3)

$$\mathbf{D} = \begin{bmatrix} D_{11} & D_{12} & D_{13} & 0 & 0 & 0 \\ & D_{22} & D_{23} & 0 & 0 & 0 \\ & & D_{33} & 0 & 0 & 0 \\ & & & D_{44} & 0 & 0 \\ & \text{Symmetry} & & & D_{55} & 0 \\ & & & & & D_{66} \end{bmatrix} \tag{3.2}$$

$$\mathbf{D} = \begin{bmatrix} \frac{1}{E_1} & \frac{-\nu_{12}}{E_2} & \frac{-\nu_{13}}{E_3} & 0 & 0 & 0 \\ \frac{-\nu_{21}}{E_1} & \frac{1}{E_2} & \frac{-\nu_{23}}{E_3} & 0 & 0 & 0 \\ \frac{-\nu_{31}}{E_1} & \frac{-\nu_{32}}{E_2} & \frac{1}{E_3} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{G_{12}} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{G_{23}} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{G_{31}} \end{bmatrix} \tag{3.3}$$

where due to the symmetry of the matrix, the following holds true $-\nu_{21}/E_2 = -\nu_{12}/E_1$, $-\nu_{31}/E_3 = -\nu_{13}/E_1$ and $-\nu_{23}/E_2 = -\nu_{32}/E_3$. This way number of engineering constants is reduced to the number of elasticity constants.

Due to the fact the three–dimensional behavior is poorly documented and additionally that the masonry in practice is usually appears as walls, out of plane characteristic of such material can be ignored. Therefore, a 2d case of compliance matrix can be assembled by setting $\sigma_3 = \sigma_5 = \sigma_6 = 0$:

$$\mathbf{D} = \begin{bmatrix} \frac{1}{E_1} & \frac{-\nu_{12}}{E_2} & 0 \\ \frac{-\nu_{21}}{E_1} & \frac{1}{E_2} & 0 \\ 0 & 0 & \frac{1}{G_{12}} \end{bmatrix} \tag{3.4}$$

if $\boldsymbol{\sigma} = (\,\sigma_1\ \sigma_2\ \tau_{12}\,)^T$ and $\boldsymbol{\varepsilon} = (\,\varepsilon_1\ \varepsilon_2\ \gamma_{12}\,)^T$ the compliance relationship takes form:

$$\begin{pmatrix}\varepsilon_1\\ \varepsilon_2\\ \gamma_{12}\end{pmatrix} = \begin{bmatrix}\frac{1}{E_1} & \frac{-\nu_{12}}{E_2} & 0\\ \frac{-\nu_{21}}{E_1} & \frac{1}{E_2} & 0\\ 0 & 0 & \frac{1}{G_{12}}\end{bmatrix}\begin{pmatrix}\sigma_1\\ \sigma_2\\ \tau_{12}\end{pmatrix} \tag{3.5}$$

by inverting the stress – strain equation a stiffness relationship ($\boldsymbol{\sigma} = \mathbf{K}\boldsymbol{\varepsilon}$) can be obtained, where $\mathbf{K} = \mathbf{D}^{-1}$:

$$\begin{pmatrix}\sigma_1\\ \sigma_2\\ \tau_{12}\end{pmatrix} = \begin{bmatrix}E_1\Delta & E_1\nu_{12}\Delta & 0\\ E_2\nu_{21}\Delta & E_2\Delta & 0\\ 0 & 0 & G_{12}\end{bmatrix}\begin{pmatrix}\varepsilon_1\\ \varepsilon_2\\ \gamma_{12}\end{pmatrix} \tag{3.6}$$

where $\Delta = 1/(1-\nu_{12}\nu_{21})$.

The determined relationship is only valid for 2D elements, e.g. shell elements. However, for practical purposes it is useful to model the structure in 3D. Therefore, an out of plane stiffness should be added, for 3D case stress and strain tensors respectively are equal to $\boldsymbol{\sigma} = (\,\sigma_1\ \sigma_2\ \sigma_3\ \tau_{12}\ \tau_{23}\ \tau_{31}\,)^T$ and $\boldsymbol{\varepsilon} = (\,\varepsilon_1\ \varepsilon_2\ \varepsilon_3\ \gamma_{12}\ \gamma_{23}\ \gamma_{31}\,)^T$, so the matrix form of stress strain relationship can be written as:

$$\begin{pmatrix}\sigma_1\\ \sigma_2\\ \sigma_3\\ \tau_{12}\\ \tau_{23}\\ \tau_{31}\end{pmatrix} = \begin{bmatrix}E_1\Delta & E_2\nu_{21}\Delta & 0 & 0 & 0 & 0\\ E_1\nu_{12}\Delta & E_2\Delta & 0 & 0 & 0 & 0\\ 0 & 0 & E_3 & 0 & 0 & 0\\ 0 & 0 & 0 & G_{12} & 0 & 0\\ 0 & 0 & 0 & 0 & G_{23} & 0\\ 0 & 0 & 0 & 0 & 0 & G_{31}\end{bmatrix}\begin{pmatrix}\varepsilon_1\\ \varepsilon_2\\ \varepsilon_3\\ \gamma_{12}\\ \gamma_{23}\\ \gamma_{31}\end{pmatrix} \tag{3.7}$$

Note, that off-diagonal terms were removed from the third row and column in stiffness matrix, this is done to simulate the behavior of plane stress elements using solid elements, as any deformations appearing in out of plane direction should not influence the deformation in the plane of material simulation. This simplification is not realistic, however, because the out of plane direction is elastic small displacements would introduce large forces in the damaged in-plane behavior.

Furthermore, even though the shear moduli is independent of other stiffness constants, Lekhnitskii [48] proposed an approximation for practical purposes, that was obtained by analyzing tests results from 45 rocks:

$$G_{ij} \simeq \frac{E_i E_j}{E_i(1+\nu_{ji}) + E_j} \tag{3.8}$$

3.2. Failure envelope

A failure envelope describes a surface in a stress space. When stresses in the material reach this surface plastic yielding occurs. For masonry deriving such yield surface is a complex task. However, as described in section 2.4 there already are failure surfaces that were composed with masonry in mind. For this model the yield surface that was proposed by Lorenço et al. [59] was chosen. The chosen surface is comparatively well documented and has good agreement with experimental results. Furthermore, the author of the surface, proposed various methods to calibrate the surface with the experimental test results.

The surface is described by two yield criteria. These criteria simulate failure of a brick and mortar separately. For the description of a mortar failure, Rankine-like surface is used:

$$f(\sigma) = \frac{\sigma_x - f_{tx} + \sigma_y - f_{ty}}{2} + \sqrt{\left(\frac{(\sigma_x - f_{tx}) - (\sigma_y - f_{ty})}{2}\right)^2 + \alpha\tau_{xy}^2} \tag{3.9}$$

and for a brick – Hill-like:

$$f(\sigma) = \sqrt{\sigma_x \cdot \left(\frac{\beta \cdot \sigma_y}{2} + \frac{f_{cy}}{f_{cx}} \cdot \sigma_y\right) + \sigma_y \cdot \left(\frac{\beta \cdot \sigma_x}{2} + \frac{f_{cx}}{f_{cy}} \cdot \sigma_y\right) + \gamma \cdot \tau_{xy}^2} - \sqrt{f_{cx} \cdot f_{cy}} \tag{3.10}$$

Where parameters α, β, γ describe the shape of the composite yield surface. These parameters could be obtained by performing least squares fit method for the yield surfaces on to experimental data or calculating them from following tests:

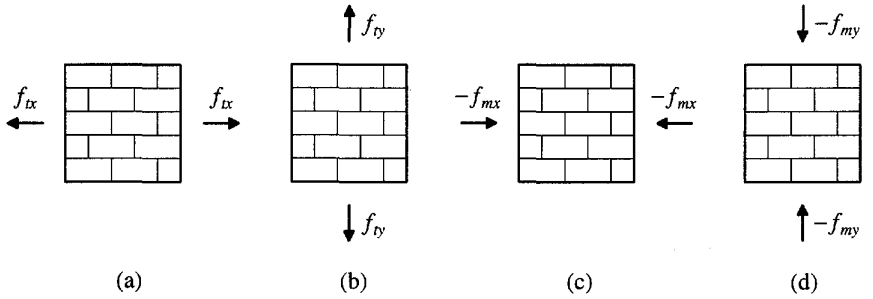


Figure 3.2: Natural tests to calibrate the composite model (from [59]): uniaxial tension (a) parallel to the bed joints and (b) normal to the bed joints; uniaxial compression (c) parallel to the bed joints and (d) normal to the bed joints

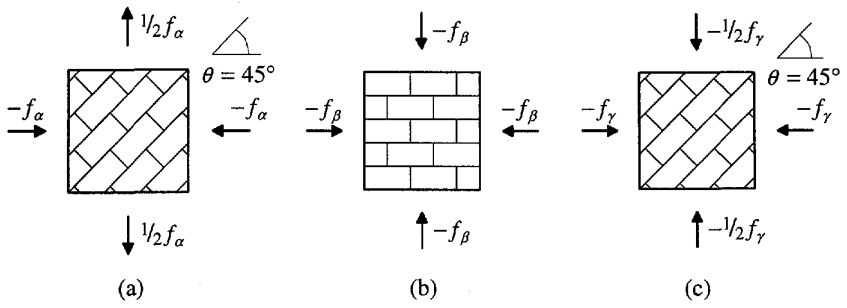


Figure 3.3: Possible non-standard tests to calibrate the composite model and calculate (a) parameter α, (b) parameter β and (c) parameter γ (from [59]).

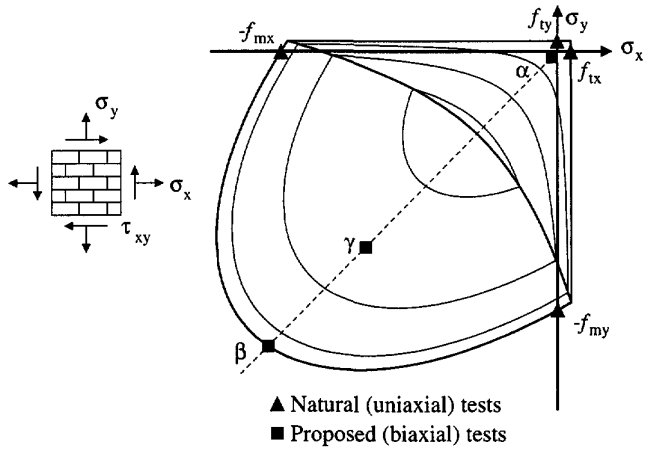


Figure 3.4: Typical position of the natural tests and proposed non-standard tests with respect to composite model (from [59]).

With these tests, the model parameter α, β and γ, read:

$$\alpha = \frac{1}{9}\left(1 + 4 \cdot \frac{f_{tx}}{f_\alpha}\right)\left(1 + 4 \cdot \frac{f_{ty}}{f_\alpha}\right) \tag{3.11}$$

$$\beta = \left(\frac{1}{f_\beta^2} - \frac{1}{f_{mx}^2} - \frac{1}{f_{my}^2}\right) f_{mx} f_{my} \tag{3.12}$$

$$\gamma = \left(\frac{16}{f_\gamma^2} - 9\left(\frac{1}{f_{mx}^2} - \frac{\beta}{f_{mx} f_{my}} - \frac{1}{f_{my}^2}\right)\right) f_{mx} f_{my}. \tag{3.13}$$

Furthermore, if the specific tests are unavailable an approximation could be made by defining them from the constituents of the masonry. Parameter α is defined by the shear strengths and tensile strengths of a joint:

$$\alpha = \frac{f_{tx} f_{ty}}{\tau_u^2}; \tag{3.14}$$

parameter γ can be defined by the shear and compression strengths of a unit, the same way as α was defined:

$$\gamma = \frac{f_{cx} f_{cy}}{\tau_u^2}; \tag{3.15}$$

while for parameter β a biaxial compression test must be performed in order to determine the strengths of the units of masonry (where $\sigma_x = \sigma_y = -f_{45°}$):

$$\beta = \left(\frac{1}{f_{45°}^2} - \frac{1}{f_{cx}^2} - \frac{1}{f_{cy}^2}\right) f_{cx} f_{cy}. \tag{3.16}$$

Alternatively, the factor α is also related to the friction angle (φ) (see eq. 3.17). The friction angle is linear and true to its formulation in an isotropic pressure situation, however, the behavior is different in case the stresses are not isotropic, e.g. vertical load is applied onto the specimen while no horizontal pressure is present. In such cases, the friction angle starts at infinity when stress is zero and approaches zero when stress approaches infinity.

$$\alpha = \frac{1}{\mu^2} \tag{3.17}$$

where, μ is the tangent of the firction angle ($\mu = \tan\varphi$).

By fitting the surfaces to the experimental data from Lurati et al. [60], the following shape can be obtained:

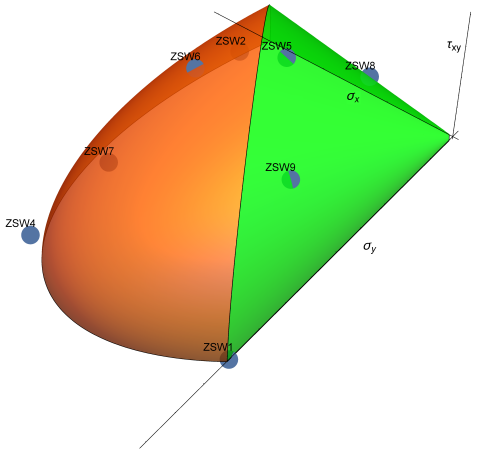


Figure 3.5: Composite surface render in Wolfram Mathematica. It is presented in Cauchy stress space. Where the horizontal plane represents σ_x and σ_y, while vertical – τ_{xy}. The fit was made according to test results from Lurati et al. [60]

3.3. Stress-Strain relation

In FEM analyses the yielding is initiated when the trial stress ($\boldsymbol{\sigma}^{trial}$) overpasses the yield surface ($f(\boldsymbol{\sigma}) = 0$). Theoretically, the stresses cannot exceed the yield surfaces therefore a corrector stress ($\boldsymbol{\sigma}^{cor}$) has to be determined in order to return the trial stress back to the surface. The plastic corrector ($\boldsymbol{\sigma}^{cor} = d\lambda \mathbf{D} \frac{\partial g}{\partial \boldsymbol{\sigma}}$) was derived in section 2.2.2. The magnitude ($d\lambda$) of the plastic corrector is determined by the yield function $f(\boldsymbol{\sigma})$, the direction – by flow function $g(\boldsymbol{\sigma})$. When the flow function is equal to the yield function ($f(\boldsymbol{\sigma}) = g(\boldsymbol{\sigma})$), the stress–strain relation is called associated, otherwise – non associated. The Hill-type part of the yield surface will be treated as associated flow and Rankine-type – as non-associated.

The flow function for Rankine-like criterion can be written as follows:

$$g(\sigma) = \frac{\sigma_x - f_{tx} + \sigma_y - f_{ty}}{2} + \sqrt{\left(\frac{(\sigma_x - f_{tx}) - (\sigma_y - f_{ty})}{2}\right)^2 + \xi \tau_{xy}^2}; \tag{3.18}$$

where ξ represents parameter inversely related to the square of the tangent of the dilation angle (ψ), therefore:

$$\xi = \frac{1}{\tan^2 \psi}. \tag{3.19}$$

The dilation angle is defined as the angle between isotropic pressure line and the flow function. This angle determines the volume change of an element during yielding. If dilation angle is positive, at yielding element will expand, if negative – it will contract, but if the dilation angle is zero the element will retain its volume. The change of volume can be described as follows:

$$\frac{\Delta V}{V} = (1 + \varepsilon_x)(1 + \varepsilon_y)(1 + \varepsilon_z) - 1 \simeq \varepsilon_x + \varepsilon_y + \varepsilon_z; \tag{3.20}$$

and the dilation angle relation to the change of volume can be expressed as:

$$\frac{dV}{V} = d(\varepsilon_x + \varepsilon_y + \varepsilon_z) = d\lambda \tan \psi; \tag{3.21}$$

therefore,

$$\frac{dV}{V d\lambda} = \tan \psi. \tag{3.22}$$

However, as in the case with the angle of friction, the angle of dilation in uniaxial loading situation changes from infinity to zero. As it can be observed in the figure 3.6 that in the case of yielding from point B, the dilation angle is much greater than when yielding occurs from point A.

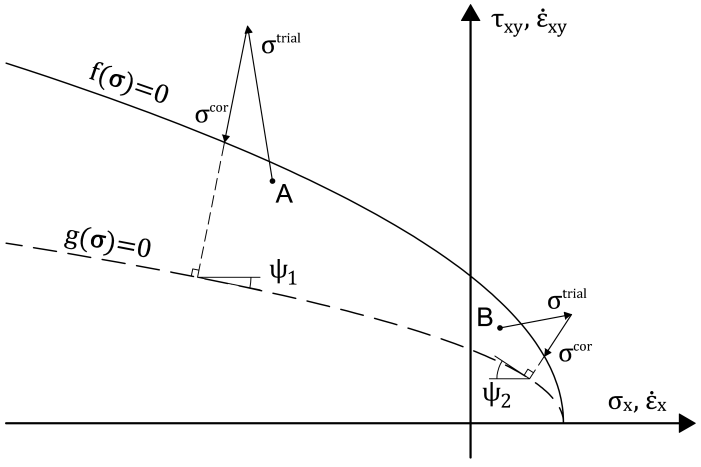


Figure 3.6: Yield and flow functions together with dilation angles at $\sigma_y = 0$.

This poses a problem in cyclic analyses where only shear load and uniaxial compression is present. In such case, the analyzed structure would extensively dilate introducing unrealistic lift and in case of

confinment – acessive internal pressure in the structure. To circumvent this it is possible to set the dilation angle to zero by making the flow surface flat in σ_x–σ_y plane at the start of the analysis or at a later point when some volumetric increase is accumulated. There is also a possibility to decrease the dilation angle gradually with the acumulated equivalent plastic strain. In the later case, it would be best to relate the dilation of an element to the maximal possible volumetric change in terms of equivalent plastic strain. Such relation can be expressed as:

$$\frac{\Delta V}{V}(\varepsilon_{pl}) = \frac{2\arctan\left(\frac{\pi\varepsilon_{pl}^{eq}}{2d_{\max}}\right)}{\pi d_{\max}} \tag{3.23}$$

where $d_{\max}$ represents maximal possible volume increase and ε_{pl}^{eq} equivalent plastic strain in the element.

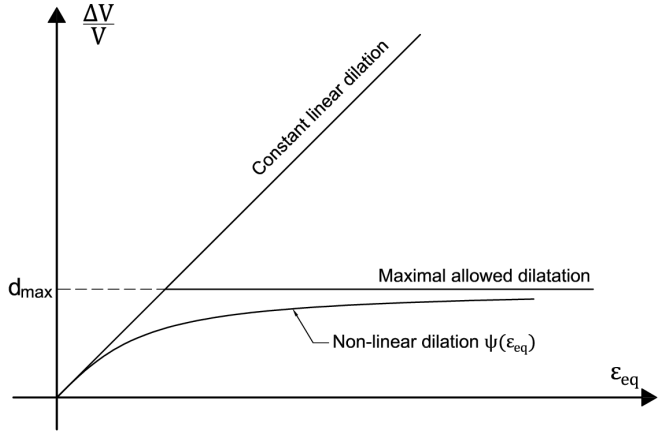


Figure 3.7: Non-linear dilation vs. constant linear dilation at $\psi = 45°$

By differentiating eq. 3.23 in respect to ε_{pl}, the relation between tangent of the dilation angle and equivalent plastic strain can be derived:

$$\tan\psi = \frac{4d_{\max}^2}{\pi^2\varepsilon_{pl,eq}^2 + 4\,d_{\max}^2} \tag{3.24}$$

by substituting the obtained relationship into eq. 3.19 and then into eq. 3.18 the plastic strain dependent flow function is obtained. However, it has to be noted that this relation is purely phenomenological as exact dilatory behavior of masonry is poorly documented and furthermore due to the shape of the flow function the dilation angle in orthotropic stress situation will always be different than the set value, and that the imposed maximal volume change constraint will only be valid in isotropic loading conditions.

As discussed in the Section 2.2.2, if the flow surfaces have discontinuities a subdifferential zone of flow directions have to be defined. The most simplistic approach would be is to iterate the return stress until the wanted solution is obtained. However, such iteration would heavily slow down the explicit analysis which is not a wanted side effect. Therefore a faster, but potentially less accurate method has to be derived.

First, yield condition is checked for both surfaces (Figure 3.8). If both of these surfaces are yielding it means that the predictor stress is in a stress space where advanced return algorithm has to be initiated. The algorithm determines if the predictor stress is in the subdifferential zone by checking the yield conditions once more, but this time using the returned stresses of both yielding surfaces determined in first conditional check. In such case, only the other yield surface has to be checked for the obtained values of the returned stresses (e.g. if Rankine-like surface returned to stress point A then only Hill-like surface has to be checked at that point A and vice versa). If after this check both surfaces are yielding,

then the initial predictor stress lies in the subdifferential stress zone. Otherwise, a returned stress is assumed to the one returned by a yield surface which the second check resulted in a not yielding state (see Figure 3.9).

In case the initial predictor stress falls into the subdifferential stress zone, then the stresses have to be returned to the intersection curve of both yield surfaces so that the return direction is perpendicular to the curve in strain space. However, finding a return to the intersection of two implicit surfaces is an extremely computationally taxing procedure, therefore an approximation of the intersection point is used. Such approximation involves defining a plane that is tangent to the Hill-like surface at a point of first stress return (point B) and a line passing trough points A and B*. The intersection of the plane and the line would estimate the intersection of the yield surfaces. This approach is only accurate if the strain and stress increments are infinitesimal. In that case, both Hill-like and Rankine-like surfaces act as flat planes and the error from the approach is minimal.

In figures 3.8 and 3.9 the stresses are presented in $\zeta - \chi$ plane. It is a fictitious plane that goes trough the points AOB. The rest of the vectors and the points presented in the schemes are only the projections to the plane. The scale of the plane is adjusted so that the stress return would appear to be perpendicular to the yield or flow surfaces. This is done in order to simplify the description and presentation of the return algorithm.

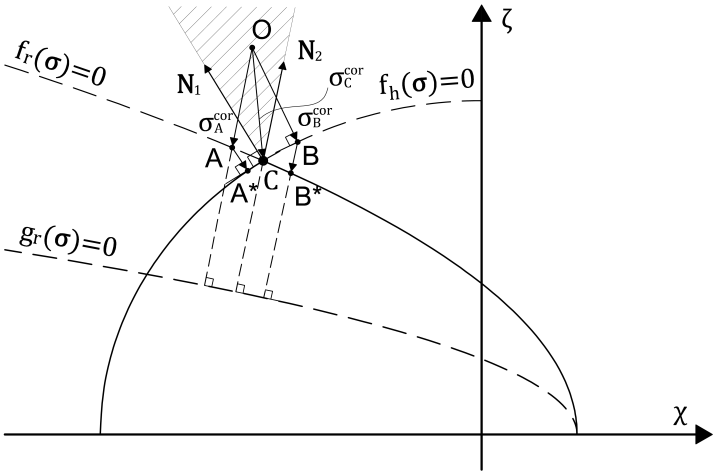


Figure 3.8: The stress return if the predictor stress originates in a subdifferential stress zone. Plotted in $\zeta - \chi$ stress plane, vectors N_1 and N_2 represents the projections of the subdifferential zone boundaries on to the plane.

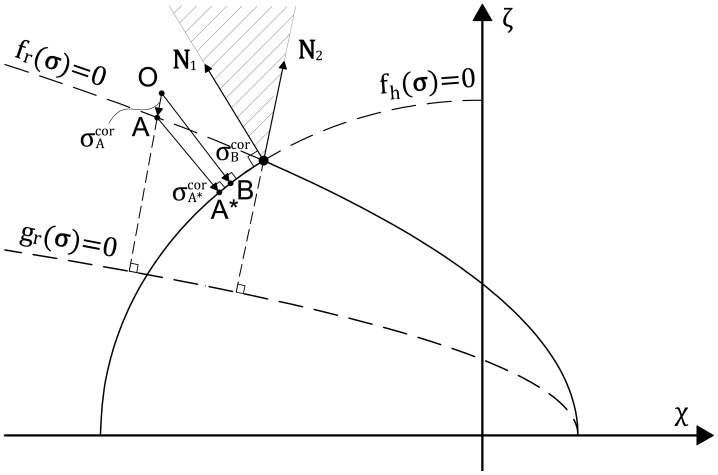


Figure 3.9: The stress return if the predictor stress does not originate in a subdifferential stress zone. Plotted in $\zeta - \chi$ stress plane, vectors N_1 and N_2 represents the projections of the subdifferential zone boundaries on to the plane.

Another region when flow is not defined by the general algorithm is at the apex of Rankine-like surface cone when the shear stress is zero. There, two issues have to be taken into account: stress at the subdifferential zone and stresses when the numerator of the partial derivative of the flow function in respect to stress tensor is zero (see Figure 3.10). Let's call the latter case "Line of undefined flow". If the trial stress point does not fall on to the line but when returned has at least one of its stress components higher than the corresponding yield limit the returned stress has to be corrected to be on the appex of the cone. Alternatively, if the trial stress point does fall on the line of undefined flow it has to be immediately returned to the apex of the cone as otherwise there are no solutions. Therefore, it can be concluded that in this situation the subdifferential zone can be defined by the outcome of the first return and the line of undefined flow by the numerator of the flow function.

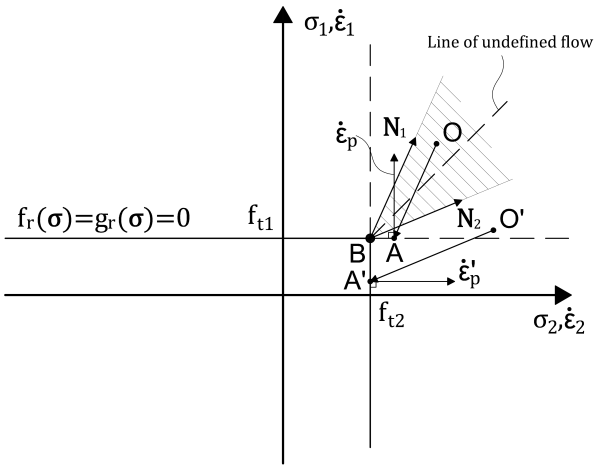


Figure 3.10: The stress return if the predictor stress does (Point O) and does not (point O') originate in a subdifferential stress zone. Plotted in $\sigma_1 - \sigma_2$ stress plane, when $\tau_{12} = 0$. Vectors N_1 and N_2 represents the projections of the subdifferential zone boundaries on to the plane and point B is the apex of Rankine-like yield/flow surface cone.

3.4. Damage

The damage in the model is described by means of reduction in stiffness and strengths of the material. The strengths reduction (softening) is subdivided into two regions: tension and compression. For former fracture energy based linear softening (Figure 3.11a) is adopted. Because the behavior is fracture energy based it is mesh-insensitive. The evolution of softening curve can be expressed as:

$$f_{yt1}(\varepsilon) = f_{yt0} - \frac{f_{yt0}^2 h(\varepsilon_{max} - \varepsilon_y)}{2G_f} \tag{3.25}$$

where f_{yt0} is initial tensile strengths of the material, ε_{max} is the maximal total strain in the analysis, G_f is the fracture energy, h is the element height and ε_y is strain at initial yield expressed as $\varepsilon_y = f_{yt0}/E$ with E the stiffness of the material.

Material tensile strengths together with softening behavior are uncoupled in both material directions. This means that when the material is subjected to tension along one of its axes the strengths do not get reduced along the other one. Furthermore, it is important to note that the lowest possible strengths are set to 1% of the initial material strengths in that direction. This is done to ensure numerical stability during the analysis.

For compression (see Figure 3.11b) a mesh sensitive softening/hardening curve is used, the curve does not have a predefined shape and it is specified in terms of equivalent plastic strain and compressive strengths as a tabular input by the end user. Such approach was chosen in order to not restrict the definition of different compressive behaviors as strain and compressive strengths relationship vary extensively from one masonry type to another. Unlike tension softening, compression hardening is coupled in material axes through equivalent plastic strain ($f_c(\varepsilon_{eq,c})$), where compressive equivalent plastic

strain is equal to the sum of scalar plastic multipliers ($d\lambda_h$) of corrector stress increments calculated due to the yielding in Hill-like surface:

$$\varepsilon_{eq,c} = \sum d\varepsilon_{eq,c} = \sum d\lambda_h \tag{3.26}$$

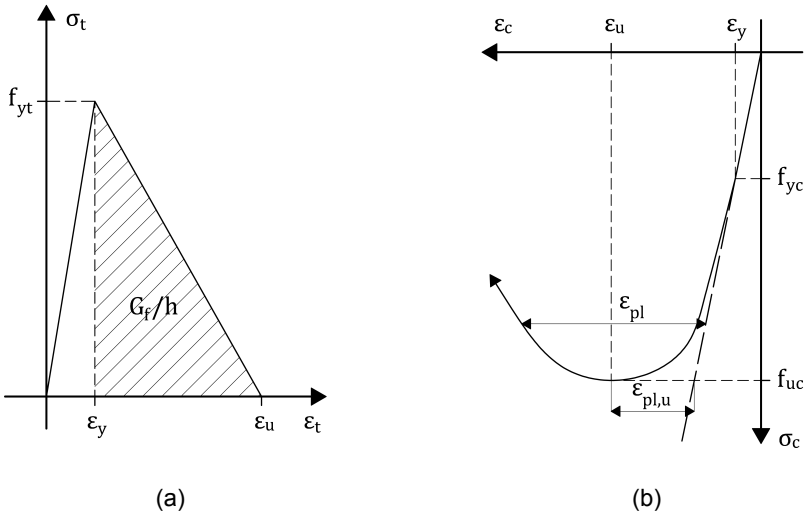


Figure 3.11: Tension (a) and compression (b) softening/hardening curves.

Due to homogenized approach, the shear softening behavior is directly related to the tensile strengths of the material. Therefore, yielding due to shear on Rankine-like yield surface will reduce the tensile strengths of the material. However, the tensile softening is controlled by total strain, therefore, it becomes difficult to introduce shear effects into the softening behavior. Two regions need to be declared in the Cauchy stress space, namely when the material is in compression and a region when at least one of the material axes are in tension. If yielding occurs in the first region the tensile strengths are controlled by equivalent plastic strain (see eq. 3.27), if in the second – by the combination of plastic shear and total axial strain.

$$f_{yt2}(\varepsilon_{eq,r}) = f_{yt0}\frac{f_{yt0}^2 h + 2EG_f - Ef_{yt0}h\varepsilon_{eq,r}}{f_{yt0}^2 h + 2EG_f} \tag{3.27}$$

Furthermore, the softening accumulated in compression causes additional softening in tension. E.g. during compression loading, compressive strengths got softened by 10%, the tensile strengths will get reduced by the same amount of percent as well. Additionally, because the softening behavior in compression is coupled in material axes, the reduction in tensile strengths is also coupled. The full tensile strengths relationship for a given material direction can be expressed as:

$$f_{yt}(\varepsilon_{eq,r}, \varepsilon_{s,r}, r, \varepsilon) = r \cdot \left(f_{yt1}(\varepsilon) - (2f_{yt0} - f_{yt2}(\varepsilon_{s,r}) - f_{yt2}(\varepsilon_{eq,r}))\right) \tag{3.28}$$

where r is the ratio between current compressive strengths in the softening regime and initial ultimate compressive strengths ($r = f_c/f_{uc0}$); $\varepsilon_{s,r}$ is plastic shear strain accumulated when stresses at least in one of the material axes are in tension.

As for the stiffness damage, the model exhibits a reduction of stiffness only when it is subjected to tension. Variable d controls the reduction in elasticity ($E_d = (1-d)E_0$). The damage serves purely a crack closure function, therefore it is based on total strain, it is uncoupled in material directions and is calculated with a rather simple expression:

$$d = 1 - \frac{f_y}{E_0 \varepsilon_{t,max}} \tag{3.29}$$

where $\varepsilon_{t,max}$ is maximal total tensile strain during the analysis and f_y is current yield strengths in the element.

The damage parameters are applied to the diagonal and off-diagonal terms of the stiffness matrix. The only exceptions are the stiffness parameters representing out of plane elasticity. The off-diagonal terms get full damage as soon at least one of the material directions get damaged. The damage recovers fully if the material becomes compressed. The damaged stiffness matrix can be represented as follows:

$$\mathbf{K}_d = \begin{bmatrix} (1-d_x)E_1\Delta & \lfloor\kappa\rfloor E_2\nu_{21}\Delta & 0 & 0 & 0 & 0 \\ \lfloor\kappa\rfloor E_2\nu_{12}\Delta & (1-d_y)E_2\Delta & 0 & 0 & 0 & 0 \\ 0 & 0 & E_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sqrt{\kappa}2G_{12} & 0 & 0 \\ 0 & 0 & 0 & 0 & 2G_{23} & 0 \\ 0 & 0 & 0 & 0 & 0 & 2G_{31} \end{bmatrix} \tag{3.30}$$

where $\kappa = (1-d_x)(1-d_y)$. The brackets $\lfloor\cdot\rfloor$ represent the floor function.

During the cyclic loading, the damaged stiffness does not transfer from compression to tension (Full recovery). During the compression, there is no damage to the elasticity of the material. Though the softening that occurs during compression get proportionally transferred to all tensile components.

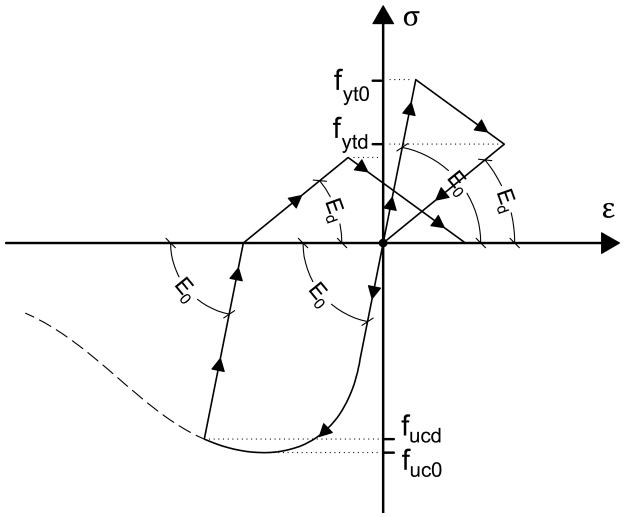


Figure 3.12: Stress – Strain curve. Damage to stiffness.

This model uses incremental return approach when the stresses are returned to the yield surface of yield initiation and only then the surface is softened (contracted) or hardened (expanded). This type of approach creates stress delay in terms of strain increment. The resultant stress is delayed behind actual stress by the magnitude of one strain increment. If time step is infinitesimal, the strain increment is infinitesimal as well, thus the deviation becomes negligible.

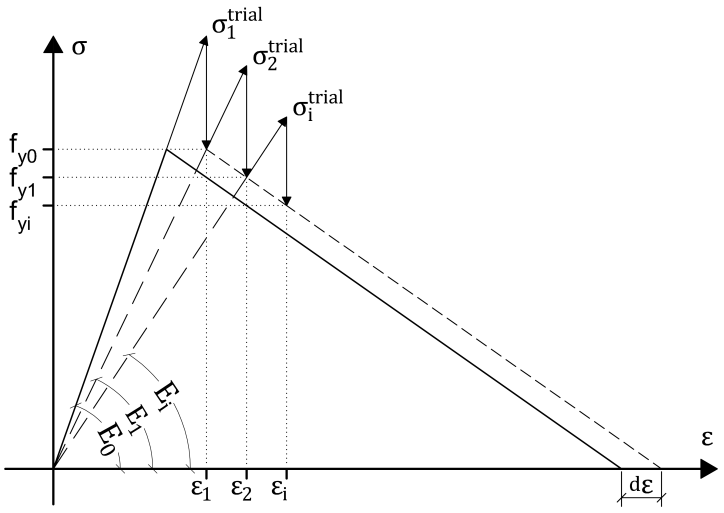


Figure 3.13: Incremental stepping algorithm, represented for tension softening case.

3.5. Additional considerations

3.5.1. Damping

If the damping of the material is desired it has to be explicitly implemented into the user subroutine. In Abaqus, generally, mass and stiffness proportional damping is used. Such damping is also known as Rayleigh damping and it consists of a mass proportional term α and a tangent stiffness proportional term β (see eq. 3.31). The mass proportional term can be specified in Abaqus and it will be applied to user subroutine. However, even though stiffness proportional therm can be specified it is not applied for user defined materials. Therefore, β damping factor has to be implemented in user subroutine.

$$\mathbf{C} = \alpha\mathbf{M} + \beta\mathbf{K} \tag{3.31}$$

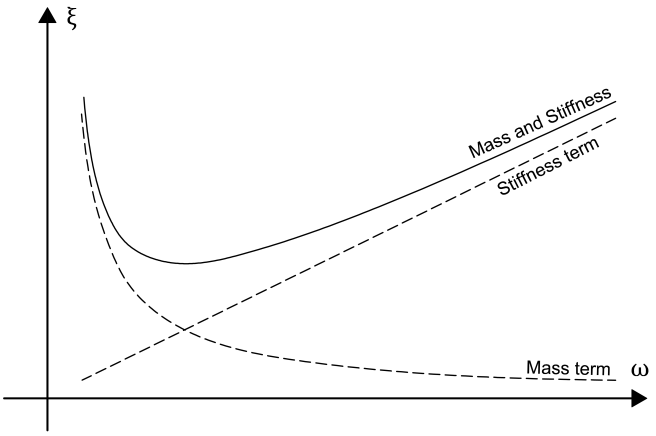


Figure 3.14: Representation of Rayleigh damping therms.

Numerically, β part of Rayleigh damping is implemented by multiplying stiffness proportional damping factor β by a stress rate and adding the result to the stress increment. Furthermore, this additional stress has to be removed at the beginning of next increment.

$$\boldsymbol{\sigma}_{new} = \boldsymbol{\sigma} + \beta\frac{d\boldsymbol{\sigma}}{dt} \tag{3.32}$$

3.5.2. Stable time estimation

During the analysis at $t = 0$ Abaqus determines the stable time increment. The calculations of it are based on the stress output from the subroutine. However, the stress output and calculated stable time increment relationship is quadratic, this means that added Beta damping to an element does not produce sufficient increment reduction. Therefore, the stresses have to be altered by a factor to represent the true influence of beta damping to the stable time.

The stable time increment condition is as follows:

$$\Delta t \leq \frac{2}{\omega_{max}} \left(\sqrt{1 + \xi^2} - \xi \right) \tag{3.33}$$

Where,

ω_{max} – Highest eigen-frequency of the element

ξ – Fraction of critical damping, can be also expressed as $\xi = \beta\omega/2$

Highest eigenfrequency of the element can be calculated by determining eigenmodes of an element, or it can be determined by the fallowing relationship:

$$\omega_{max} = \frac{2c_d}{L_{e,min}} \tag{3.34}$$

where $L_{e,min}$ is the smallest characteristic length in the element and c_d is dilation wave speed in the material. The c_d can be expressed trough the relation of Lamé's constants (λ and μ) and material density ρ as follows:

$$c_d = \sqrt{\frac{\lambda + 2\mu}{\rho}} \tag{3.35}$$

The Lamé's constants can be found:

$$\mu = \frac{1}{2} \frac{\Delta \mathbf{S} : \Delta \mathbf{e}}{\Delta \mathbf{e} : \Delta \mathbf{e}} \tag{3.36}$$

$$\lambda = \bar{K} - \frac{2}{3}\mu \tag{3.37}$$

where $\Delta\mathbf{S}$ is the deviatoric stress increment, $\Delta\mathbf{e}$ is the deviatoric strain increment and $\bar{K}$ is the effective bulk modulus expressed as:

$$\bar{K} = \frac{-\Delta p}{\Delta \varepsilon_{vol}} \tag{3.38}$$

where Δp is increment of equivalent pressure stress defined as $p = -\frac{1}{3}(\sigma_x + \sigma_y + \sigma_z)$, $\Delta\varepsilon_{vol}$ is the increment of the volumetric strain.

If we assume that the strain increments $\Delta\boldsymbol{\varepsilon}$ is:

$$\Delta\boldsymbol{\varepsilon} = \begin{bmatrix} -0.001 \\ -0.001 \\ -0.001 \\ 0.001 \\ 0.001 \\ 0.001 \end{bmatrix} \tag{3.39}$$

Then the stable time increment can be obtained:

$$t_{min} = \frac{1.5\, L_e \left(\sqrt{1 + \xi^2} - \xi \right)}{\sqrt{\frac{0.25\, \Delta\, (E_1 + 2\, E_2\, \nu + E_2) + 0.25\, E_3 + G_{12} + G_{23} + G_{31}}{\rho}}} \tag{3.40}$$

where $\Delta = 1/(1 - \nu_{12}\nu_{21})$.

3.5.3. Element deletion

Element deletion is implemented my flagging the element for deletion when the tabular input of compression hardening/softening curve runs out of values. Abaqus deletes the element by setting the stress and strain increments in the element equal to zero. By default when an element is deleted mesh does not contact with itself, however, it is possible to reinitialize a contact after the deletion using specific modeling strategies. For further reading on contact re-initiation after the deletion, the reader is referenced to Abaqus Analysis User's Guide, Section 36.4.1 [14].

3.6. Summary

The model consists of three-dimensional stiffness matrix that is compatible with two-dimensional plasticity model. The failure is initiated through one of the two yield surfaces, namely Rankine like and Hill like. The plastic flow is non-associated for the former and associated for the latter. For the Rankine like non-associated flow, three types of dilatory behavior were considered: always constant; constant until prescribed volume increase is reached and then dilation is disabled; dilation angle decreases gradually with plastic strain. The intersections of the yield surfaces and the apex of the Rankine-like surface cone cause a problematic stress return in the sub-differential zone. In this chapter, the underlying algorithms used to mitigate this issue were discussed.

Furthermore, the model exhibits tensile strengths softening based on total strain when the material is subjected to tension and on plastic strain when it is subjected to compression and shear. Compressive behavior supports both hardening and softening of the material strengths and the softening percentage is transferred to the tensile strengths. Additionally, in tension, crack closure is present while in compression plastic deformations are accumulated. The softening and hardening is applied in increments where the material strengths in current increment depend on the plastic strains and total strains generated in the previous one.

Finally, additional measures were taken to ensure stable and smooth analysis, stiffness proportional therm of Rayleigh damping was incorporated together with stable time estimation based on the effects of the added damping. To even further increase the stability of an explicit analysis an option to remove crushed element was considered.

4

Material model verification

For the model, verification tests several different kinds of cube setups were used. The tests were subdivided into two categories tension-compression and shear. Each category contains tests subjected to monotonic loading and cyclic loading. It is so to show that the model functions as expected in all loading situations.

4.1. Cube Setups

For tension–compression tests the size of the cube is $1x1x1m$ and it is composed of 64 single integration elements is used. It results in the $0.25x0.25x0.25m$ size of each element. The second order accuracy is enabled in the settings of Abaqus. The cube is fixed at the bottom plane in the vertical direction. Above the cube, there is a reference point tied with an equation to the top surface of the cube. On the reference point, depending on the nature of the test, either upwards or downwards displacement is applied. The analyses themselves were performed with the double precision setting.

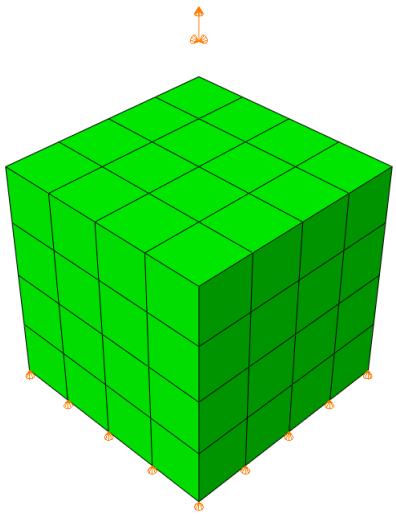


Figure 4.1: Cube set-up for tension-compression tests.

The cube for the shear test has the same dimensions as the cube for compression/tension tests, but it is only composed of one element. It was done like this due to the difficulty to obtain pure shear without over-constraining the cube.

There is a reference point tied with an equation to the top right edge of the cube. On the reference point, horizontal displacement is applied. The cube is restrained in the vertical direction at the bottom and in horizontal – on bottom right edge. Out of plane displacements were restricted in order to obtain a more stable output. The whole cube was exposed to isotropic pressure. The analyses themselves were performed with the double precision setting.

Note that this way of modeling induces extensive hourglass effect on the element. However, because the element has only one integration point, hourglassing does not have a significant effect on the results of analyses.

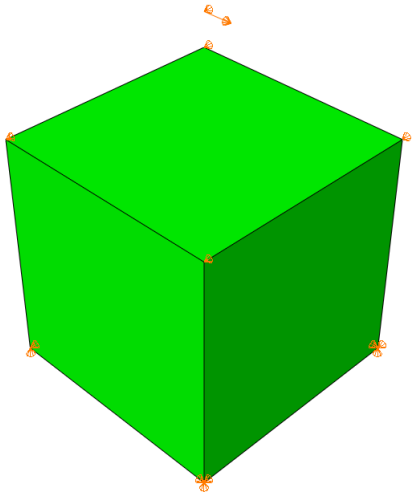


Figure 4.2: Cube set-up for shear tests.

In the tests material properties described in the table 4.1 are used. In some tests, some of these properties will be alternated in order to better verify the model, but in that case changing values will be explicitly stated. Note that the properties used are not usually found in real masonry specimens, such properties were chosen in order to easier check and display the results.

Table 4.1: Material properties used trough out the tests.

Elastic properties

$E_x\ [Pa]$	$E_y\ [Pa]$	$E_z\ [Pa]$	v_{xy}	$G\ [Pa]$
$10000 \cdot 10^6$	$5000 \cdot 10^6$	$10000 \cdot 10^6$	0.2	$3000 \cdot 10^6$

Inelastic properties

$f_{tx}\ [MPa]$	$f_{ty}\ [MPa]$	$G_{fx}\ [Nm/m]$	$G_{fy}\ [Nm/m]$	α
$1 \cdot 10^6$	$0.5 \cdot 10^6$	15	15	1.56
$f_{cx}\ [MPa]$	$f_{cy}\ [MPa]$	β	γ	$Soft.\ Type.$
$10 \cdot 10^6$	$5 \cdot 10^6$	-1	3	$hyp.\ f_{cc} = 30$

4.2. Uniaxial tension

In this test, the tensile behavior of the model will be examined. Initiation and softening of yield surface will be tested with various different fracture energies. The results will be verified by comparing them to hand calculations. Furthermore, plasticity localization will be examined as well as high strain situations and problems that can occur.

The analysis was performed with four different fracture energies (in the direction of loading), namely: $G_f = 1.5Nm^{-1}$, $G_f = 5.0Nm^{-1}$, $G_f = 10.0Nm^{-1}$ and $G_f = 15.0Nm^{-1}$.

The observed behavior was as expected and it can be verified by calculating the ultimate tensile strain ε_u and comparing it to obtained results:

$$\varepsilon_u = \frac{F_y}{E} + 2\,\frac{G_f}{hF_y}. \tag{4.1}$$

This yields strains $\varepsilon_u = 0.124‰$, $\varepsilon_u = 0.18‰$, $\varepsilon_u = 0.26‰$ and $\varepsilon_u = 0.34‰$ respectively. It can be observed from figure 4.3 that the values match the results.

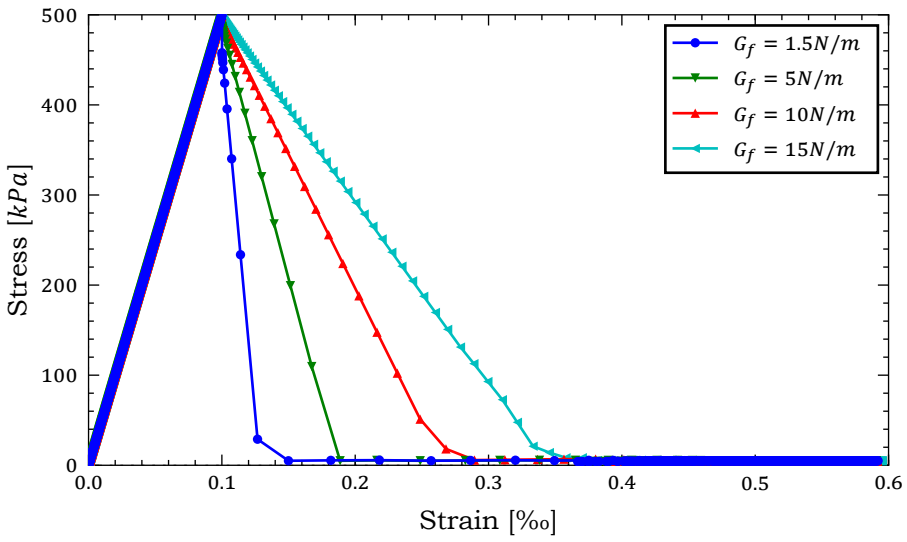


Figure 4.3: Stress – Strain curve. Tensile softening behavior.

Even though the smeared crack approach is used, the cracks localize (Figure 4.4). This behavior is present because unlike in static implicit analysis in dynamic explicit analysis displacements and forces propagate through the material at a specific speed. Because of this propagation in each increment, the stresses do not increase uniformly trough out the mesh, therefore there will be places in the model where stresses reach the yield values first. As soon as that happens, at that location, tensile softening of the material starts. Because of the softening, the affected elements will act as the weakest link in the model and all of the stresses will redistribute to them, therefore a localized crack will appear.

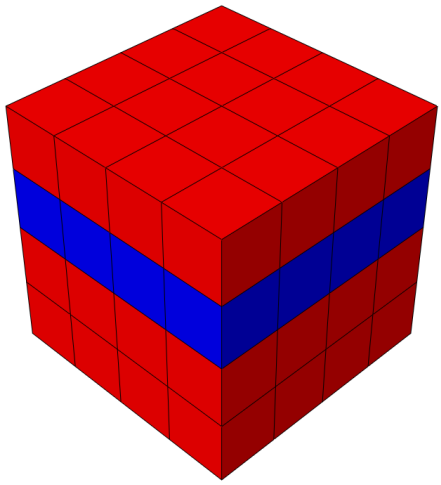


Figure 4.4: Damage localization. Blue color represents damaged elements, red – non-damaged.

Sometimes in analyses, a big crack openings can be observed. It is important for a material model to allow the separation of these cracks without additional excessive forces. When analyzing such situation with default Abaqus settings large ever increasing resistance forces can be observed with high strains (see Figure 4.5). It is due to the Linear viscosity setting that tries to suppress such deformations. It works by applying additional forces to the element when the volumetric strain rate is high. It is most apparent when the element is damaged and it's volume increases quite rapidly. Nevertheless, it can be avoided by setting linear viscosity to 0 (see [14], Section 27.1.4) but in that case stiffness damping should be applied as after appearance of each crack the structure will have high-frequency vibrations (Figure 4.6).

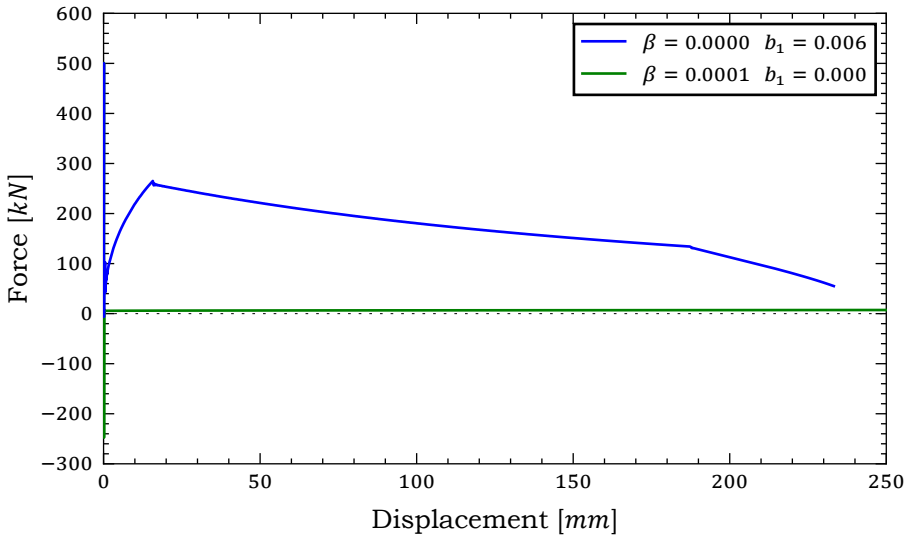


Figure 4.5: Force – Displacement curve. Tensile softening behavior in high strains.

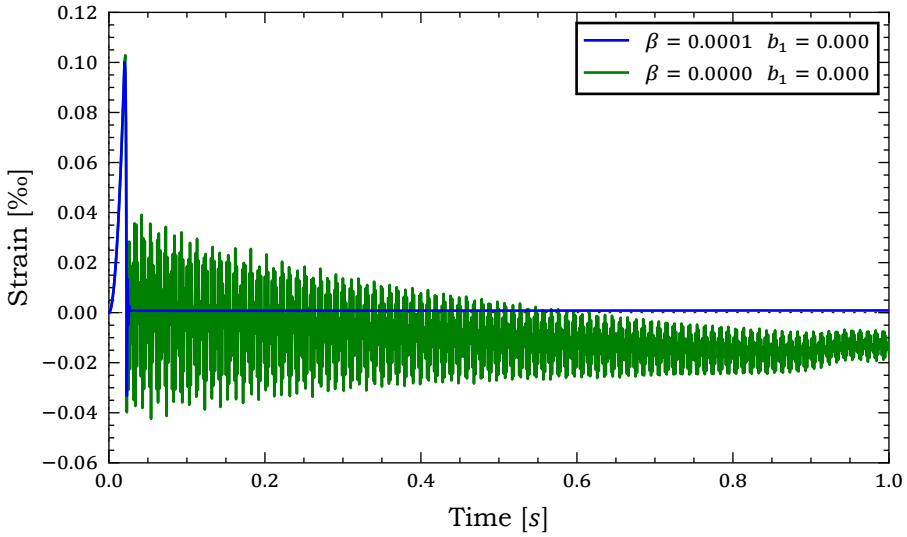


Figure 4.6: Oscillations of elastic element.

In the Figures above β represents stiffness proportional Rayleigh damping coefficient and b_1 – Linear viscosity coefficient.

4.3. Uniaxial compression

In this test, the compressive behavior of the model will be examined. Initiation and softening of yield surface will be tested. The results will be verified by comparing them to hand calculations. Furthermore, damage localization will be examined as well as high strain situations.

As an input for the compression hardening/softening behavior, Thornfeldt softening curve for concrete was used.

$$f(\varepsilon) = f_u \frac{\varepsilon}{\varepsilon_u} \left(\frac{n}{n - \left(1 - \left(\frac{\varepsilon}{\varepsilon_u}\right)^{n \cdot k}\right)} \right); \tag{4.2}$$

$$\varepsilon_u = f_u \frac{n}{E \cdot (n-1)}; \tag{4.3}$$

$$k = \begin{cases} 1, & \text{if } \varepsilon_u < \varepsilon < 0 \\ 0.67 + \frac{f_{cc}}{62}, & \text{if } \varepsilon \leq \varepsilon_u \end{cases} \tag{4.4}$$

$$n = 0.8 + \frac{f_{cc}}{17}. \tag{4.5}$$

Where,

- f_u – Ultimate compressive strength of the material
- ε_u – Strain at ultimate strengths
- E – Elasticity modulus of the material
- n, k – Shape parameters
- f_{cc} – Factor defining softening shape

This curve was generated for yield strength of $F_u = 5 \cdot 10^6 Pa$ and a varied f_{cc} factor, namely – $15, 20, 30$ (See Table 4.2).

Table 4.2: Properties used in different analyzes

No.	E_x	E_y	$f_{u,x}$	$f_{u,y}$	f_{cc}
1	$10 \cdot 10^9 Pa$	$5 \cdot 10^9 Pa$	$10 \cdot 10^6 Pa$	$5 \cdot 10^6 Pa$	15
2	$10 \cdot 10^9 Pa$	$5 \cdot 10^9 Pa$	$10 \cdot 10^6 Pa$	$5 \cdot 10^6 Pa$	20
3	$10 \cdot 10^9 Pa$	$5 \cdot 10^9 Pa$	$10 \cdot 10^6 Pa$	$5 \cdot 10^6 Pa$	30

During the tests, downwards displacement of 17.5mm was applied to the cube. The hardening/softening behavior was as expected, but it must be noted that the plastic strain localization only happens when the transition from hardening to softening is sudden ($f_{cc} = 30$).

By comparing the output to the input provided, it can be seen that curves do not match (see Figure 4.9). There is ever increasing drift of stresses from the expected result. This is because during analysis, the hardening/softening behavior for compression is being tracked using equivalent plastic strain and that the yield value of Rankine-like yield surface is not equal to the material strengths along each of the material directions. This means that a perfect equivalence in the stress–strain diagram is not achieved. It can be solved by formulating $\dot{\varepsilon}_{eq} \neq \dot{\lambda}_h$, but due to the usual lack of the experimental results current used simplification is sufficiently accurate.

By plotting the difference ratio between equivalent plastic strain and the strain obtained from the analysis (see Figure 4.10) it can be seen that the equivalent plastic strain is always $\sqrt{\frac{1}{2}}$ of the actual plastic strain. Therefore, one could adjust stress-strain curve depending on the type of analysis expected in order to minimize the deviation.

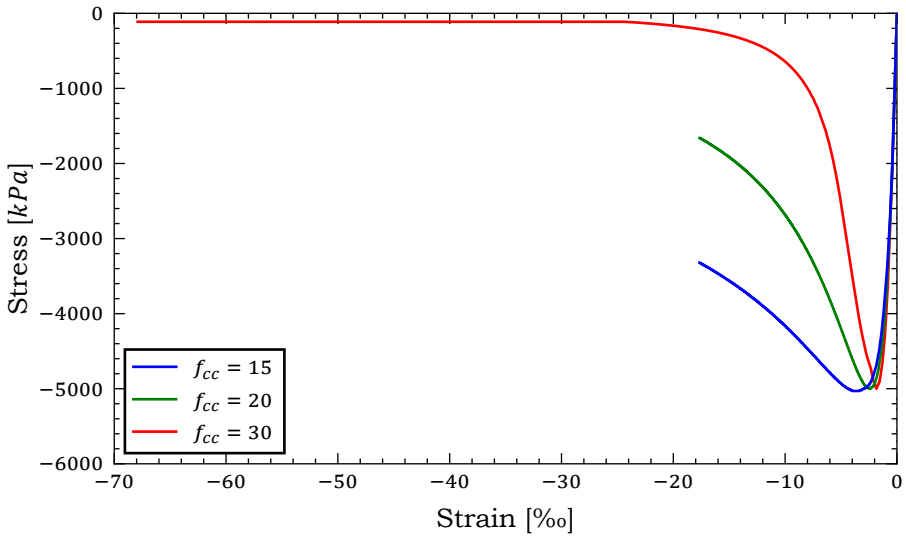


Figure 4.7: Stress – Strain curve. Compression hardening/softening behavior.

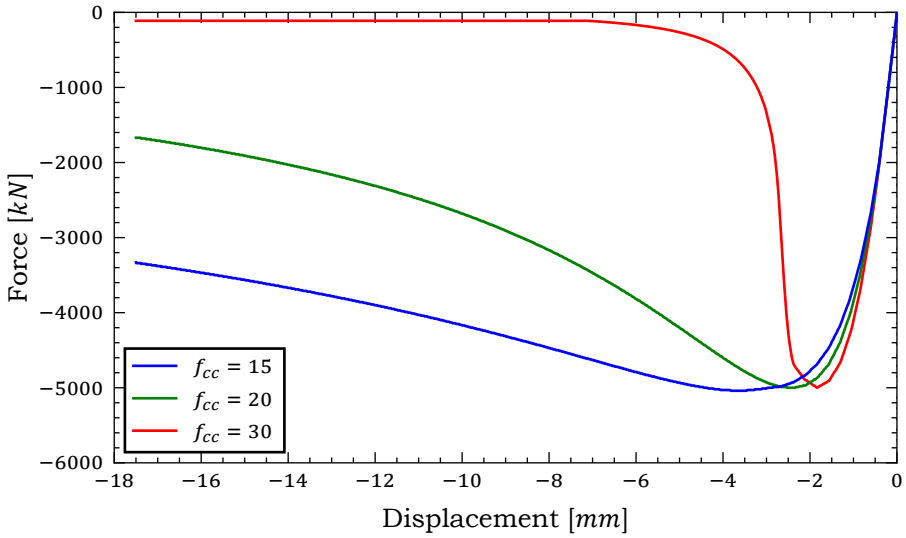


Figure 4.8: Force – Displacement curve. Compression hardening/softening behavior.

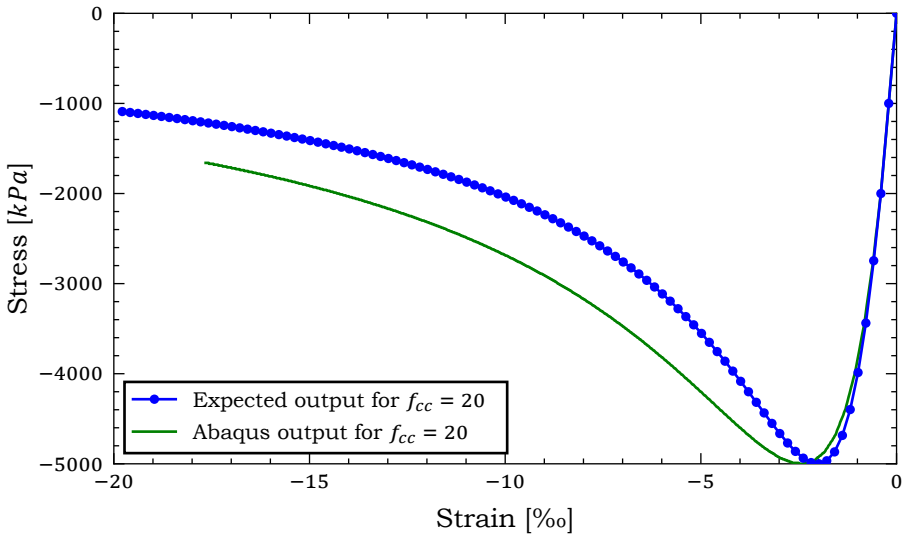


Figure 4.9: Stress – Strain curve. Comparison of compression softening to expected results.

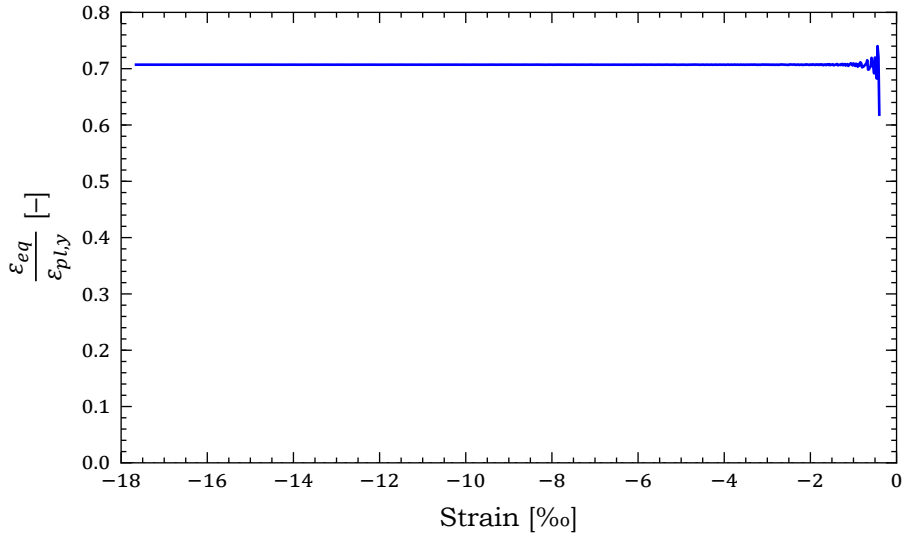


Figure 4.10: Equivalent plastic strain versus plastic strain in loading direction

4.4. Shear

In this test, the shear behavior of the material model will be tested. Shear and tension relation together with dilation angle will be examined. For material properties, the Tensile fracture energy, and the dilation angle will be alternated between the analyses. The fracture energies are set to be the same both for x and y directions. The properties used are presented in table 4.4.

Table 4.3: Properties used in different analyzes

No.	p	G_{ft}	$\tan\psi$
1	$1\,000\ Pa$	20.0	0.0
2	$100\,000\ Pa$	20.0	0.0
3	$200\,000\ Pa$	20.0	0.0
4	$400\,000\ Pa$	20.0	0.0
5	$1\,000\ Pa$	15.0	0.0
6	$1\,000\ Pa$	10.0	0.0
7	$1\,000\ Pa$	20.0	0.5
8	$1\,000\ Pa$	20.0	1.0

From the Figure 4.11 it can be seen that increased pressure results in increased shear. The difference from shear at low pressure can be described by:

$$\tau = \mu p \tag{4.6}$$

Friction coefficient μ in this case is 0.8 (from Eq. 3.17)

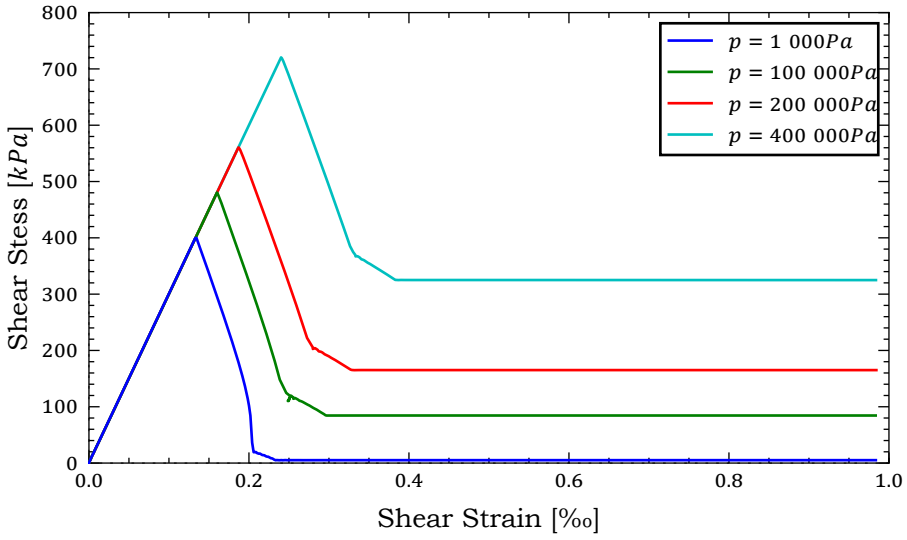


Figure 4.11: Stress – Strain curve. Influence of different isotropic pressures

This is only valid in isotropic loading situation. If the loading is orthotropic the shear strain – shear stress relation is parabolic meaning that with the lower uniaxial compressive force the friction would approach infinity and with a higher – zero.

From above figure it can be seen that the softening of tension influences the softening of shear, the retaliation comes through plastic strain. The fracture energy in tension defines ultimate plastic strain and that, in turn, defines the Mode-II fracture energy.

Next, the dilation angle will be examined. The results of the analysis can be checked by:

$$\frac{dV}{V} = \varepsilon_1 + \varepsilon_2 + \varepsilon_3 = d\lambda \tan\Phi \tag{4.7}$$

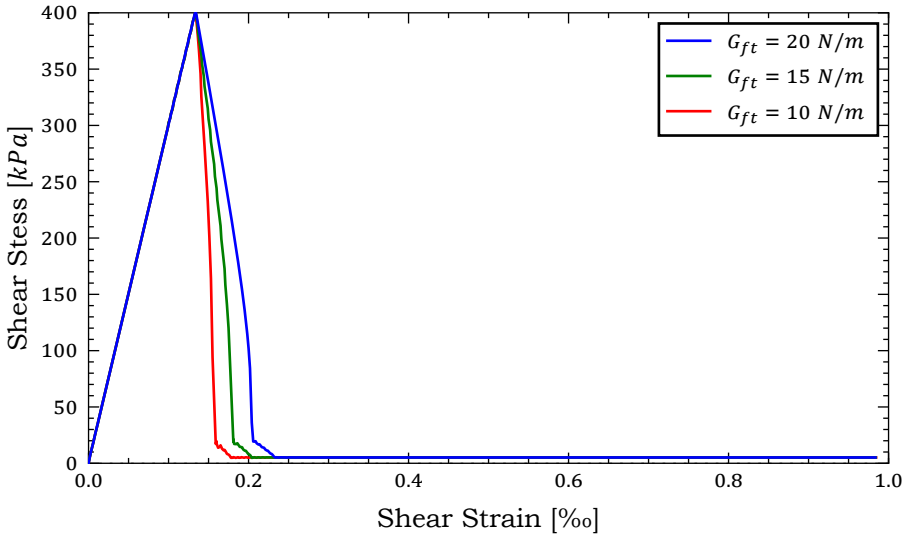


Figure 4.12: Stress – Strain curve. Influence of tensile fracture energy

$$\tan \Phi = \frac{\varepsilon_1 + \varepsilon_2 + \varepsilon_3}{d\lambda} \tag{4.8}$$

From Figure 4.13 it can be seen that the volume change is mostly uniform, although there is some fluctuation due to the shock-waves in the analysis. By differentiating the curves with respect to equivalent plastic strain we get rate of change of volumetric strain that further shows existence of two shock waves: one at the beginning of analysis when displacement gets to be applied and the second one during the softening of the material (see Figure 4.12, $G_{ft} = 20\ N/m$). These disturbances happen because during the softening the stress distribution fluctuates and because the flow surface is not linear, the direction of the flow changes with changing stress. It does not induce huge effects on the volumetric strain, though the derivative of it is effected substantially (see Figure 4.14).

It is also important to note that decreasing dilation angle increases the shear sensitivity to minor fluctuations. This is due to shear multiplier factor α used for the flow function. The lower the friction angle the higher is the ξ factor and therefore, the shear stress has more influence on the direction of the flow. This coupled with the nonlinear flow field will result in high instabilities around tension with low dilation angles.

4.5. Cyclic: Tension – Compression

In this test, cyclic behavior of the material will be analyzed. The effects of the change of the state (Tension/Compression) will be tested. The default material properties was used for the text, except for tensile fracture energy which is $G_{ft} = 30\ N/m$.

Cyclic vertical displacement (Figure 4.15) was applied to the test cube. The displacement was prescribed in such a way that specimen does not fully soften in the first cycle, but only in the second.

During tension, all of the plastic strain is converted to the damage of the stiffness. Thus, after unloading no plastic deformations are present. To the contrary, in compression, the plastic strain is accumulated and stiffness does not experience any damage (see Figure 4.16). When a specimen is compressed it's stiffness fully recovers and when put back to tension the damage is resumed.

4.6. Cyclic: Shear

In this test, a cyclic shear behavior of the material model will be examined. For this analysis, the same element as in the previous shear analysis is used. However, the boundary conditions were changed. The horizontal displacement is applied on the top face of the element instead of the rightmost edge, as wells as the bottom support was expanded from the edge of the element to the whole face. It was done so to prevent hourglassing in the element. Even though the hourglassing did not have a

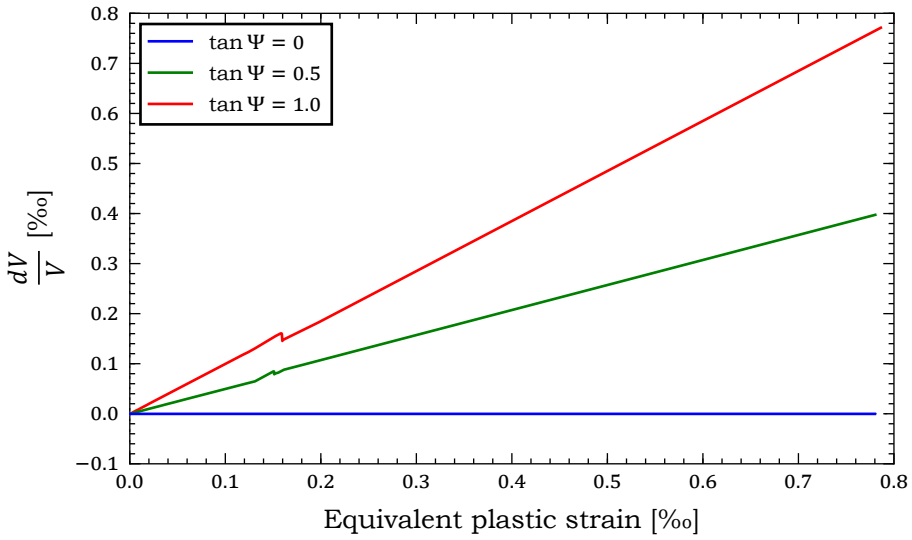


Figure 4.13: Volume change in respect to plastic strain

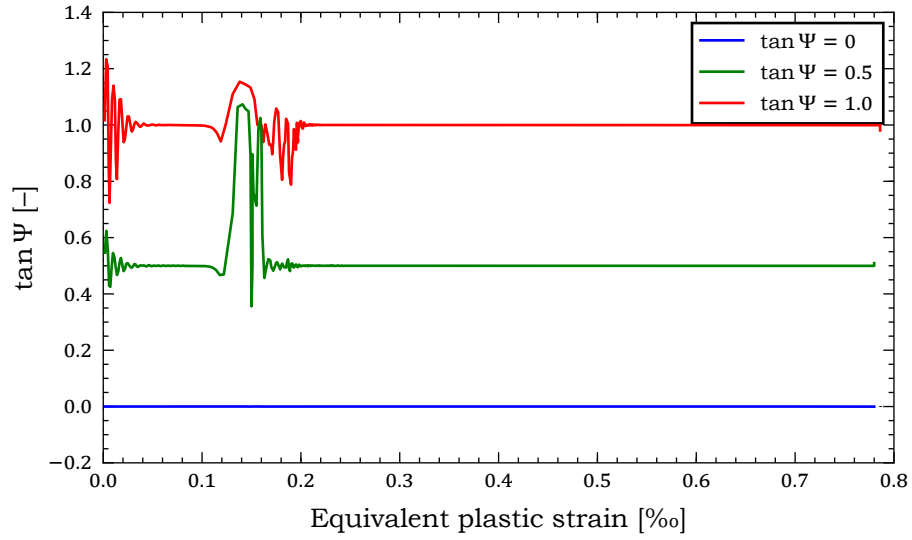


Figure 4.14: Dilation angle in respect to plastic strain

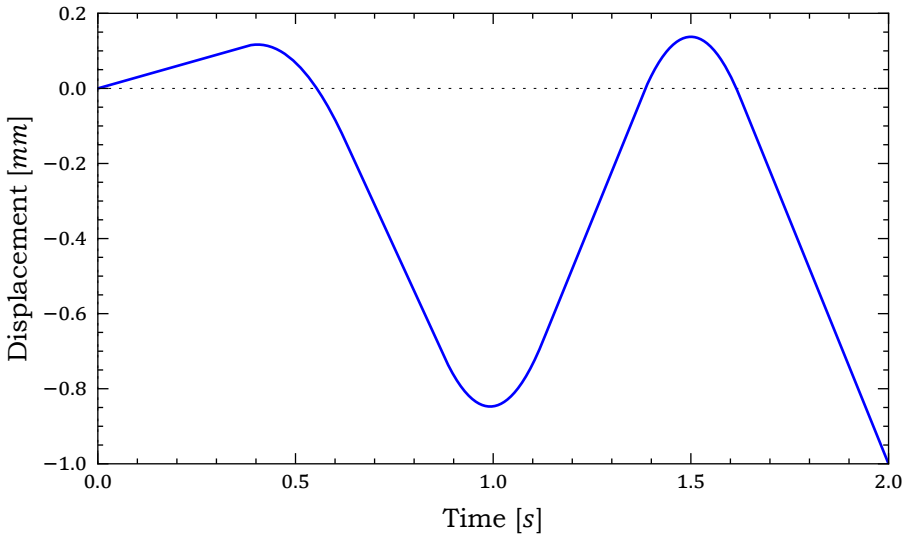


Figure 4.15: Dilation angle in respect to plastic strain

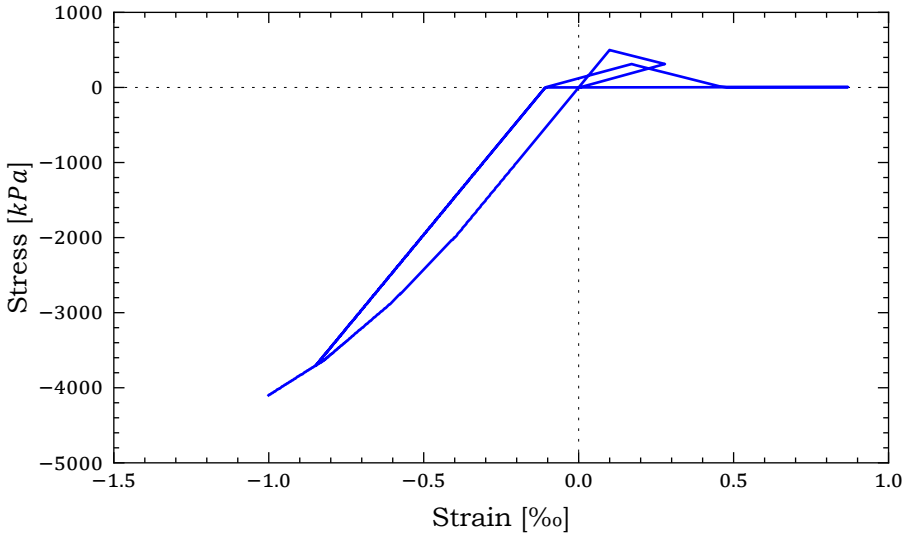


Figure 4.16: Stress – Strain curve. Tension – Compression cycles.

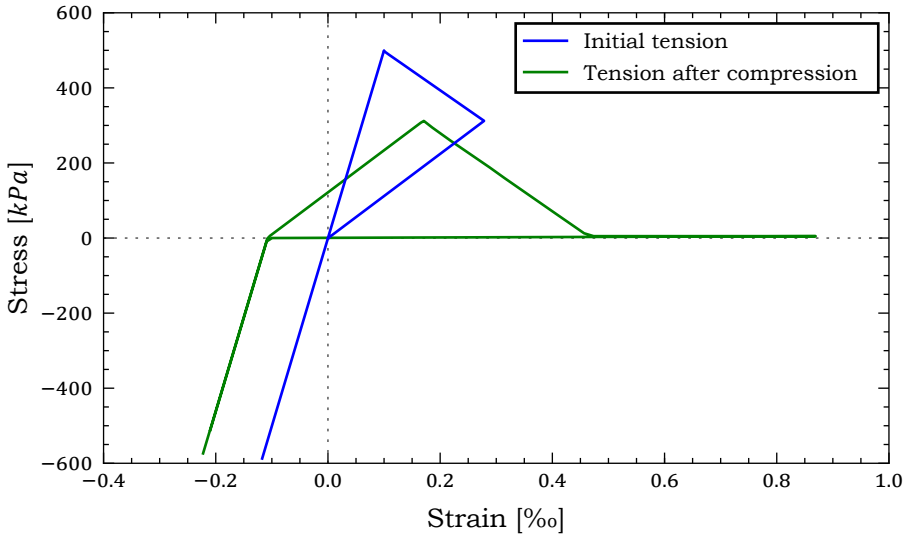


Figure 4.17: Stress – Strain curve. Zoomed region of tensile part.

negative effect on analysis results during the monotonic loading, during cyclic loading it had significant influence. These new boundary conditions over-constrain the element, therefore pure shear behavior is not obtained.

Table 4.4: Properties used in different analyzes

No.	d_{max}	$\tan\psi$
1	0	1.0
2	0.001	0.0
3	0	0.0

Harmonic cyclic shear loading was applied to the test elements (see Figure 4.18). The duration of the analyses was 10s in order to minimize the dynamic effects. During this period 10 full shear cycles were applied.

First, constant dilation was examined. From Figure 4.19 it can be seen that the dilation is much higher than the set dilatory angle of $\tan\psi = 1.0$. This is due to several reasons. First of all the element is over-constrained, therefore, stresses in the constrained material axes get accumulated over the time of the analysis (see Figure 4.20), furthermore, because of the shape of the flow function, the lower the uniaxial stress, the greater is the dilation angle. Such high dilation is unrealistic and it is never seen in experimental results.

From Figure 4.20 it can be also observed that even tough the dilation is ever increasing, the stress after reaching certain point starts softening. It is due to the fact that the confining stress grows so high that the material starts yielding and softening due to the stresses reaching the compression surface.

When examining the dynamic dilation behavior it can be seen that the volumetric strain change is significantly lower than in the previous analysis. This type of behavior would be expected from experimental results. However, in the material properties it was that the maximal possible volumetric change is 0.001 of total volume, but in the analysis 3 times the higher value was achieved (see Figure 4.21). This is, again, due to the shape of the flow function, does not matter how low the dilation angle will be set, at low values of uniaxial pressure, the dilation angle will still be high.

As in the previous analysis, the stress appears to be growing (see Figure 4.22) due to the confinement of the element. However, this time softening in compressive strengths was not present.

The final shear test is a control for the confining stresses. The dilation in this test was set to zero. Not only that but the flow surface was made to be completely flat. In this case, no dilation is experienced by the element (see Figure 4.23), though a sudden volume decrease can be observed at the start of

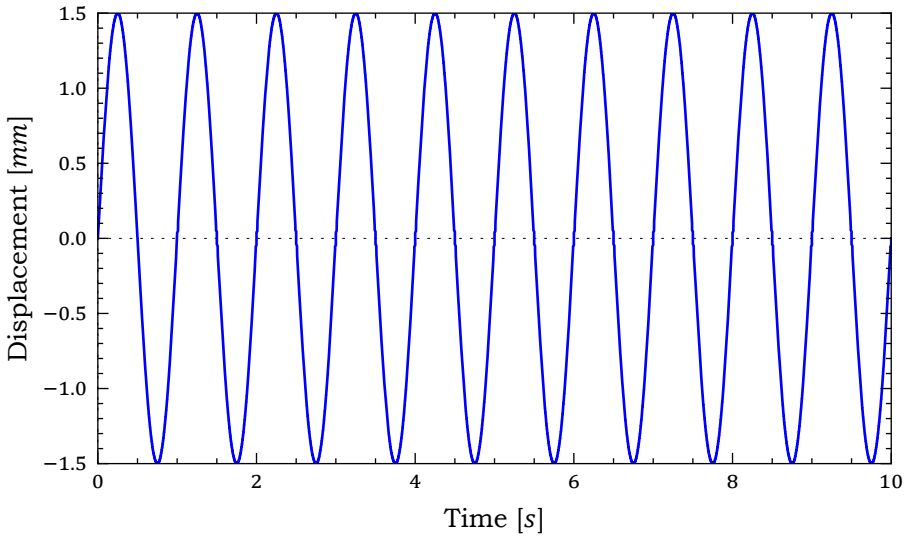


Figure 4.18: Input signal for the analyses.

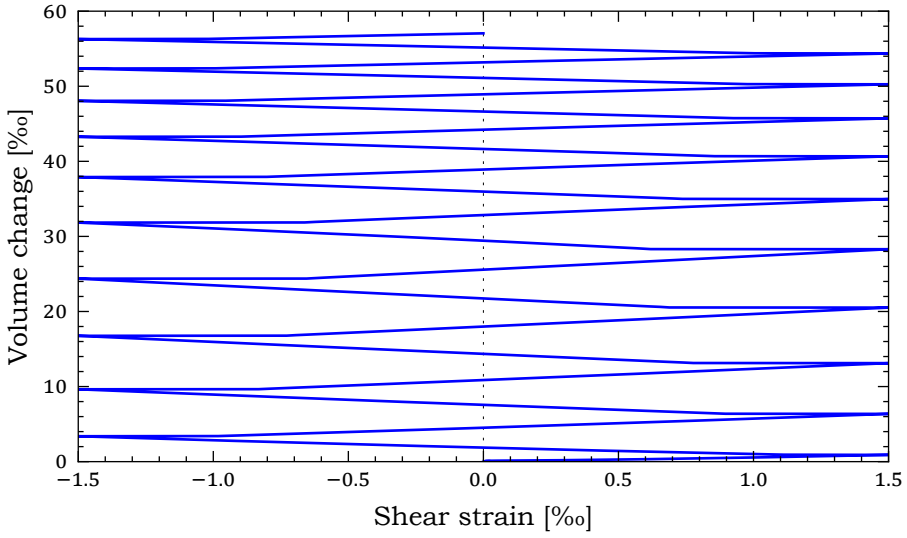


Figure 4.19: Volume change per shear strain, for an element with constant dilation angle.

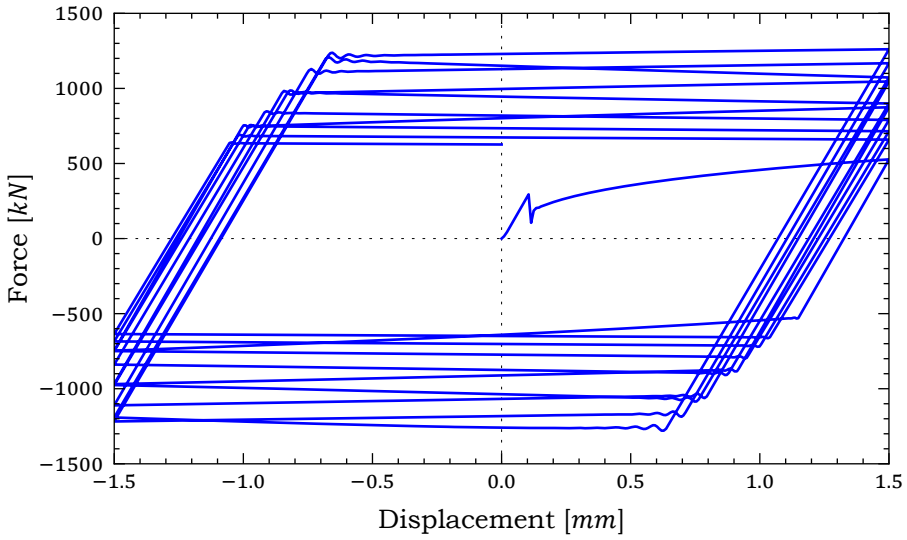


Figure 4.20: Force – Displacement curve. Shear force resistance in an element with a constant dilation.

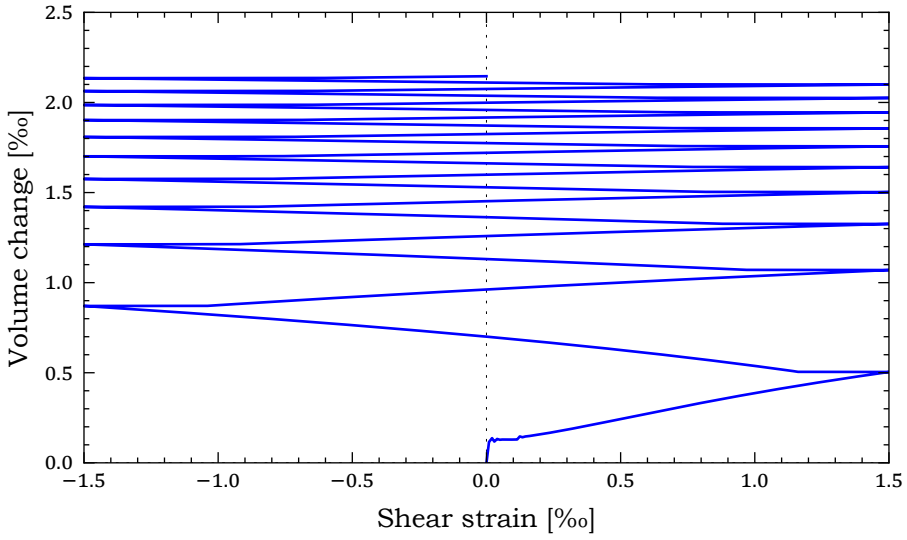


Figure 4.21: Volume change per shear strain, for an element with decreasing dilation angle.

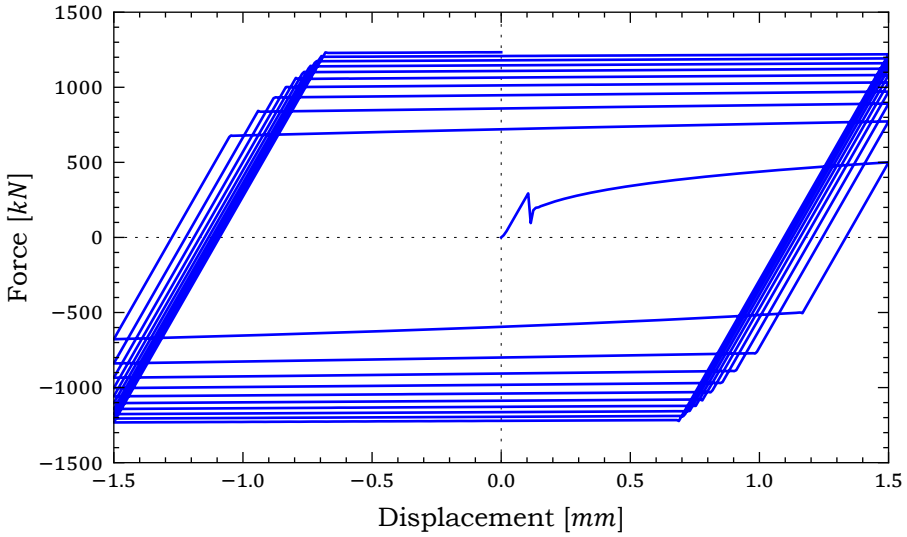


Figure 4.22: Force – Displacement curve. Shear force resistance in an element with a decreasing dilation.

the analysis. This decrease is due to elastic volume change due to the instantaneous application of vertical pressure.

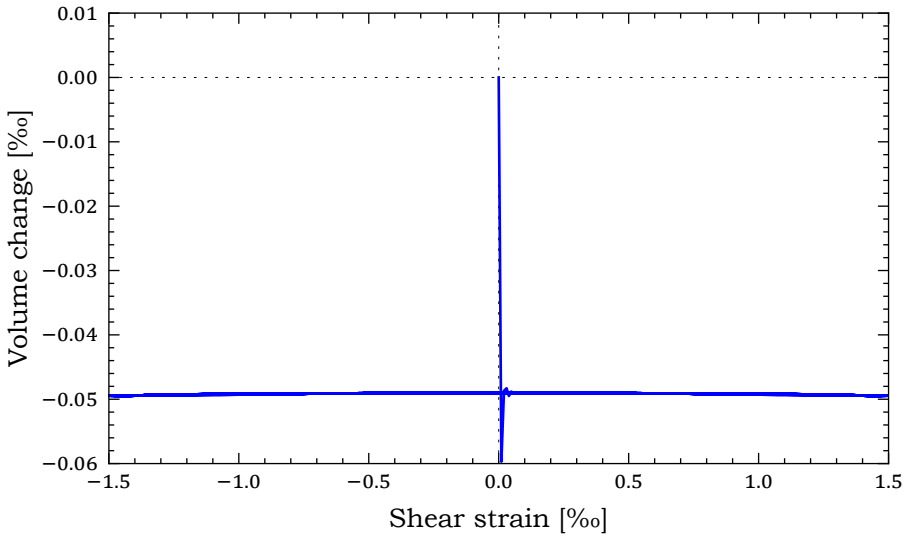


Figure 4.23: Stress – Strain curve. Zoomed region of tensile part.

Because there is no dilation, there are no confining stresses. From Figure 4.24 it can be observed a continuous cycle of the shear force after first tensile strength softening, during the analysis.

From the cyclic shear test, it was determined that the model behaves as expected.

4.7. Element deletion

In this test, the element deletion is analyzed. For the test, a masonry wall is placed between two platforms. The platforms are disconnected from the wall and a general contact is defined in the whole model with “Hard” normal behavior and a tangential behavior with the friction of 0.7. The top platform is loaded with imposed downward displacement. The bottom – clamped.

The input used for compressive behavior is depicted in Figure 4.26. The failure of an element is initiated by the last equivalent plastic strain entry in the input table. In such case output variable $SDV29$

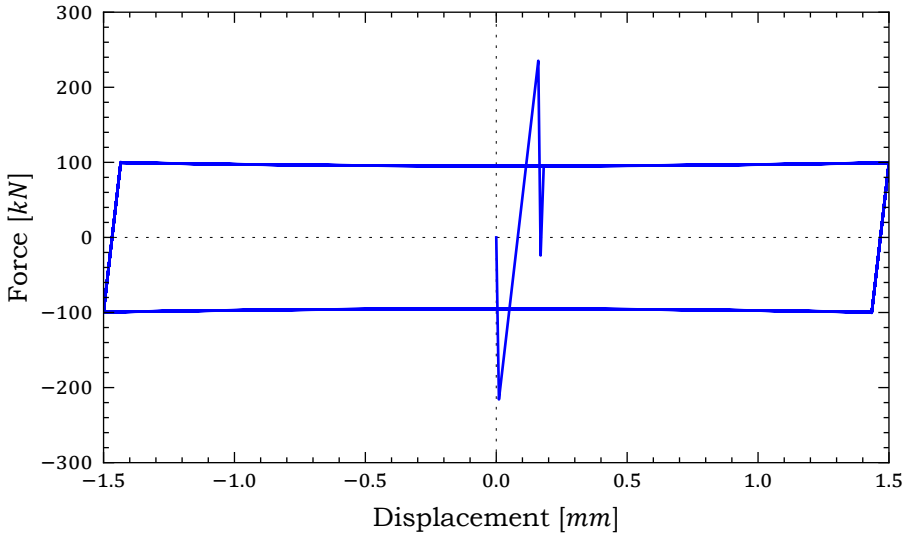
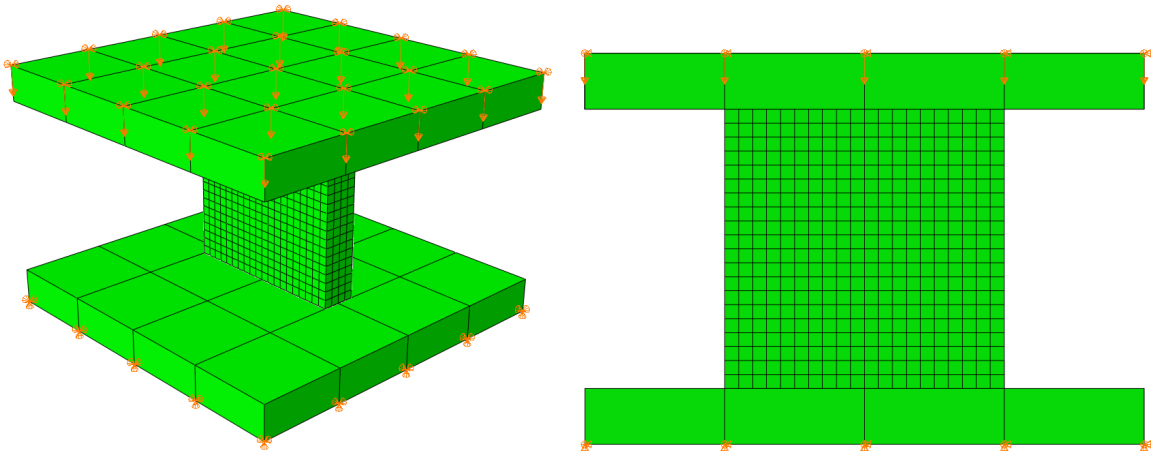


Figure 4.24: Stress – Strain curve. Zoomed region of tensile part.



(a) 3D view (b) Side view

Figure 4.25: Model used for element deletion testing

is set to 0 and the failed (crushed) element is deactivated. For such element strain increments and stress increments are always equal to zero.

During the analysis localization of crushing can be observed (See Figure 4.27b). Such localization introduces stress redistribution and propagation of crushed elements (See Figure 4.27c). After the whole line of elements is removed the top part of the wall drops onto the bottom part and closing the gap.

It must be noted that after element deletion Abaqus recalculates the contact surfaces and wall does not fall through. But for this to happen special measures has to be taken (Consult Abaqus User Guide 36.4.1 Modeling surface erosion).

By further examining the stress-strain relationship of a failed element, it can be seen that indeed the stress is set to zero as soon as the input is exceeded (Figure 4.28)

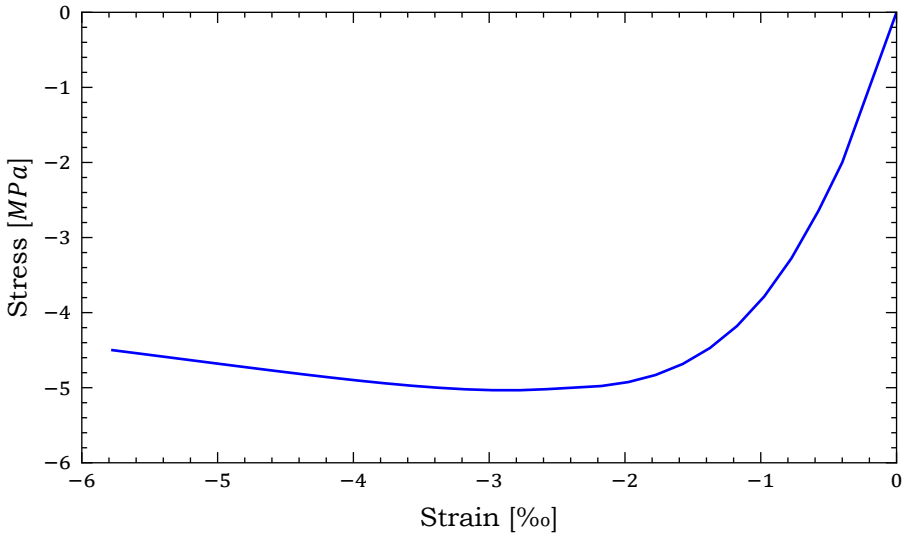
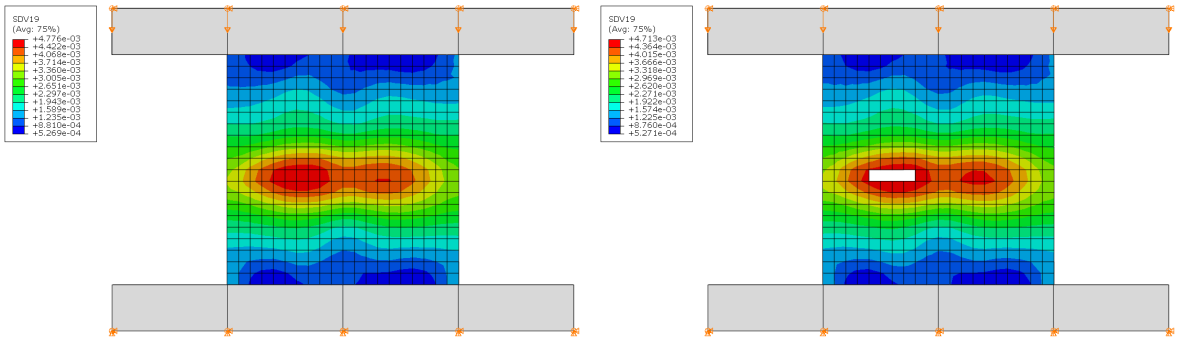
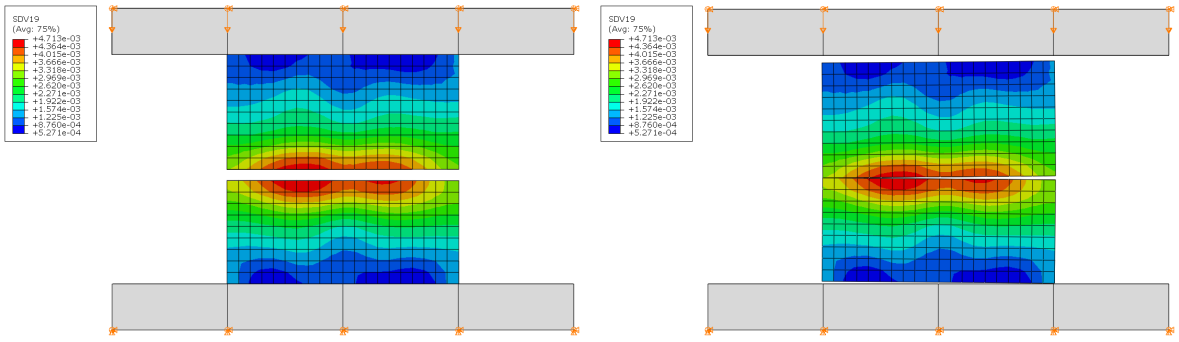


Figure 4.26: Stress – Strain relationship of input.



(a) Before crushing

(b) Crushing localized



(c) Crushing propagated

(d) Closure of crack

Figure 4.27: Equivalent compressive plastic strain throughout the analysis

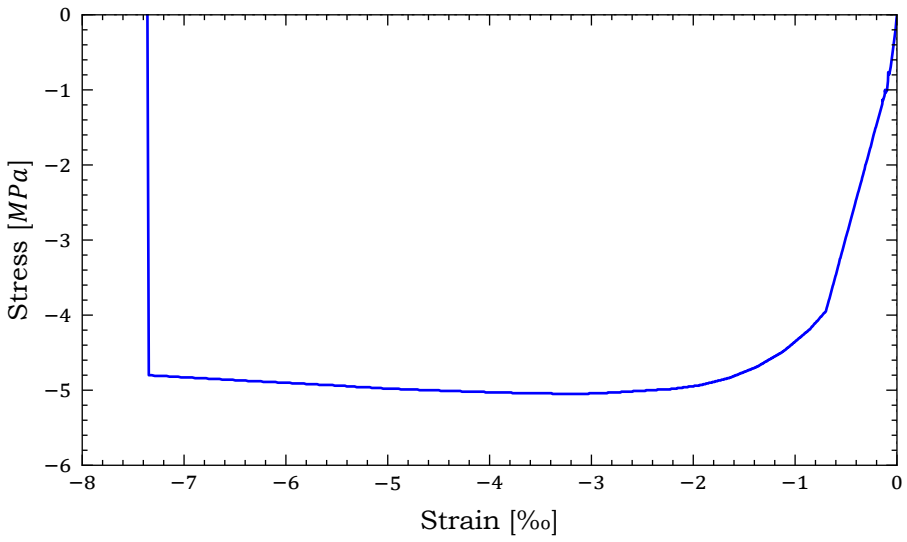


Figure 4.28: Stress – Strain relationship of a failed element.

4.8. Summary

In this chapter, tensile, compressive and shear behaviors subjected to both monotonic and cyclic loadings were tested. All of the obtained results were true to formulations described in Chapter 3. It has to be noted, that although the dynamic dilation angle worked as expected, it introduced some instabilities into the system. These instabilities can cause substantial problems in larger scale analyses, therefore it is recommended to only use this feature if confining pressure is always present in the affected elements or to avoid it at all. Alternatively, dilation can be set to zero immediately or after wanted volumetric change is reached as these settings provided much more stable results.

In the end, the capabilities of element deletion were presented. The contact between the elements can be re-initiated after elements are deleted and the appeared openings can be closed.

5

Comparison to experiments

In this chapter, developed material model will be compared to the results of experimental tests and its ability to simulate the behavior of different masonry tests will be assessed. It is important to note that the masonry macro-models will always include some degree of approximation as all of the failure mechanisms of masonry cannot be simulated with the smeared out approach.

The goal of this chapter is to verify the material model in different loading situations. Namely, shear behavior with low pressure so the failure mechanisms prescribed by the Rankine-like yield surface can be tested; shear behavior with high pressure so the failure mechanisms prescribed by Hill-like surface can be tested and shear behavior under cyclic loading so the cyclic capabilities of the model can be examined. Therefore, walls from experiments carried out by Ganz and Thürlimann [27, 28] will be analyzed. The tests will be referenced as ETH Zurich tests trough out the chapter.

ETH Zurich wall tests were chosen mainly because of two reasons. First of all the researchers that carried out the tests had provided sufficient amount of material data with which an accurate failure envelope can be mapped in Cauchy stress space. Second, the underlying failure surface was compared by Lourenço [56] to specifically these test results, therefore, the tests act as a good control for the developed material model.

5.1. Determination of the material properties

To determine the material properties 12 wall panels reported by Ganz and Thürlimann [27] of dimensions $1200\times1200\times150\ [mm^3]$ and denoted K were considered. The panels were loaded proportionally in principle stress directions σ_1 and σ_2 along different rotations θ of the bed joint. To determine the inelastic properties of the masonry the test results of these wall panels were mapped in Cauchy stress space. For the mapping panels, $K5$ and $K9$ were disregarded. The reason is that the boundary conditions affected the failure of the panel $K5$ and panel $K9$ included reinforcement.

The mapping of the results was done using a least square fit method. To determine the ratio between experimental and predicted failure was calculated by determining the stress vector $\sqrt{\sigma_x^2+\sigma_y^2+\tau_{xy}^2}$ and predicted failure envelope intersection point. The comparisons between the fit and experimental results can be seen in table 5.1.

The fit resulted in the following material properties: $f_{tx}=0.246\ [MPa]$; $f_{ty}=0.0\ [MPa]$; $\alpha=1.716$; $f_{cx}=1.730\ [MPa]$; $f_{cy}=7.505\ [MPa]$; $\beta=-1.171$; $\gamma=1.0$, however for numeric analysis it is more stable to have some amount of strengths in the material. Therefore, for further calculations tensile strength in y direction will be used as follows: $f_{ty}=0.05\ [MPa]$.

The elastic properties were determined directly from test data and they are as follows: $E_x=2460\ [MPa]$; $E_y=5460\ [MPa]$; $\nu_{xy}=0.18$; $G_{xy}=1130\ [MPa]$. However, data to determine some of the inelastic parameters was missing, therefore, these parameters: tensile fracture energy, compression softening curve and ultimate equivalent plastic strain for compression had to be assumed.

For the tensile softening behavior, the fracture energies of $20\ ^{Nm}/_{m}$ are used. For the compression regime, parabolic hardening/softening law are used (see Figure 5.1) in order to obtain a softening behavior that is not sensitive to the element size. However, as examined in the previous chapter the

Table 5.1: Comparison between plasticity model and the experimental results.

Panel	Regime	$\frac{\sigma_1}{\sigma_2}$	θ [°]	Plasticity model			Experimental results			Ratio
				σ_x $[MPa]$	σ_y $[MPa]$	τ_{xy} $[MPa]$	σ_x $[MPa]$	σ_y $[MPa]$	τ_{xy} $[MPa]$	
K1	Tens.	−0.092	22.5	−0.069	−0.825	0.389	−0.071	−0.850	0.401	1.030
K2	Tens.	−0.050	22.5	−0.142	−1.145	0.509	−0.138	−1.111	0.494	0.970
K3	Comp.	0.000	0.0	0.000	−7.505	0.000	0.000	−7.635	0.000	1.017
K4	Comp.	0.000	90.0	−1.730	0.000	0.000	−1.601	0.000	0.000	0.926
K6	Tens.	0.000	45.0	−0.344	−0.344	0.344	−0.317	−0.317	0.317	0.922
K7	Tens.	0.000	22.5	−0.344	−2.005	0.830	−0.351	−2.048	0.848	1.021
K8	Tens.	0.000	67.5	−0.337	−0.057	0.140	−0.217	−0.037	0.090	0.644
K10	Comp.	0.313	0.0	−2.097	−6.711	0.000	−2.015	−6.449	0.000	0.961
K11	Comp.	0.323	22.5	−2.019	−4.675	1.146	−2.063	−4.778	1.171	1.022
K12	Comp.	0.313	45.0	−1.897	−2.085	0.994	−2.038	−2.239	1.068	1.074

compression localization heavily depends on the softening rate of the material and the speed of stress propagation in the analyzed structure. Therefore the compressive fracture energy: $G_{fcx} = 3280\ ^{Nm}/_{m}$ and $G_{fcy} = 10000\ ^{Nm}/_{m}$, and the ultimate plastic strain $\kappa_p = 8 \cdot 10^{-4}$ is chosen. For the effective compressive region $h = 375\ [mm]$ was selected to cover the height of 5 elements.

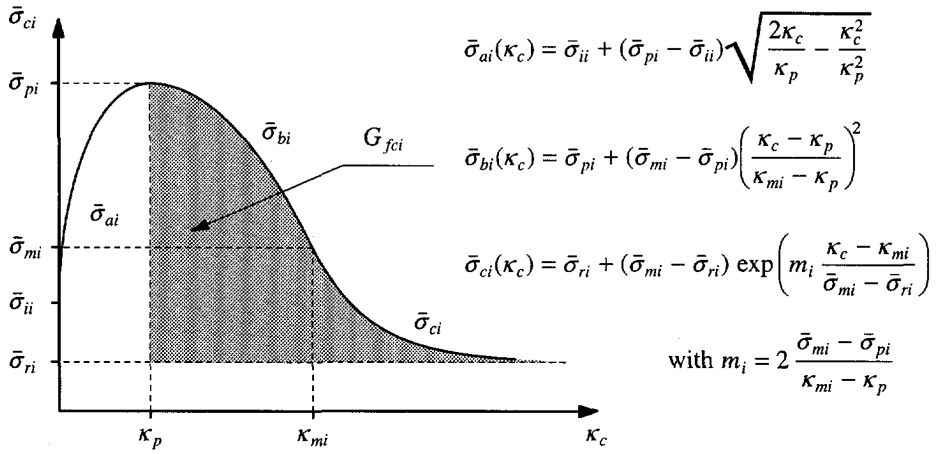


Figure 5.1: Parabolic hardening/softening law for compression (from [56])

5.2. Decription of the tests and analysis setup

The shear walls analyzed consist of hollow clay brick masonry. The tests were reported by Ganz and Thürlimann [28] and denoted W. There were in total seven walls tested. However, only four of them will be used for the comparison, namely walls $W1$, $W2$, $W4$ and $W6$. The walls $W3$ and $W5$ had reinforcement in them, therefore they fall outside of the scope of this thesis. The wall $W7$ was removed due to the indecisive crack patterns, while simulation could be performed it would be difficult to compare the results.

The geometry of the walls consist of a masonry panel of $3600 \times 2000 \times 150\ [mm^3]$ and flanges of $150 \times 2000 \times 600\ [mm^3]$ (see Figure 5.2). The wall is subjected to the boundary conditions by two reinforced concrete slabs at top and bottom of the wall. Additionally, the wall is given prescribed uniformly distributed load p over the length of the wall with a resultant P. Furthermore, a horizontal displacement d is applied at the top concrete slab. The magnitudes of P and d alternate between the walls tested. The walls $W1$, $W2$ and $W4$ are subjected to monotonic displacement increment, while wall $W6$ is loaded with a dynamic signal.

All of the numerical analyses are performed using dynamic explicit integration method. The continuum elements used are 3D brick elements consisting of 8 nodes and 1 integration point. The size of these elements is $75 \times 75 \times 75\ [mm^3]$, the mesh is regular throughout the whole model (see Figure 5.3). Aside from the loads mentioned before, self-weight of the wall and concrete slabs are also included in

the analysis. The monotonic analyses are performed in a period of 1 second while quasi-static in a 12s period.

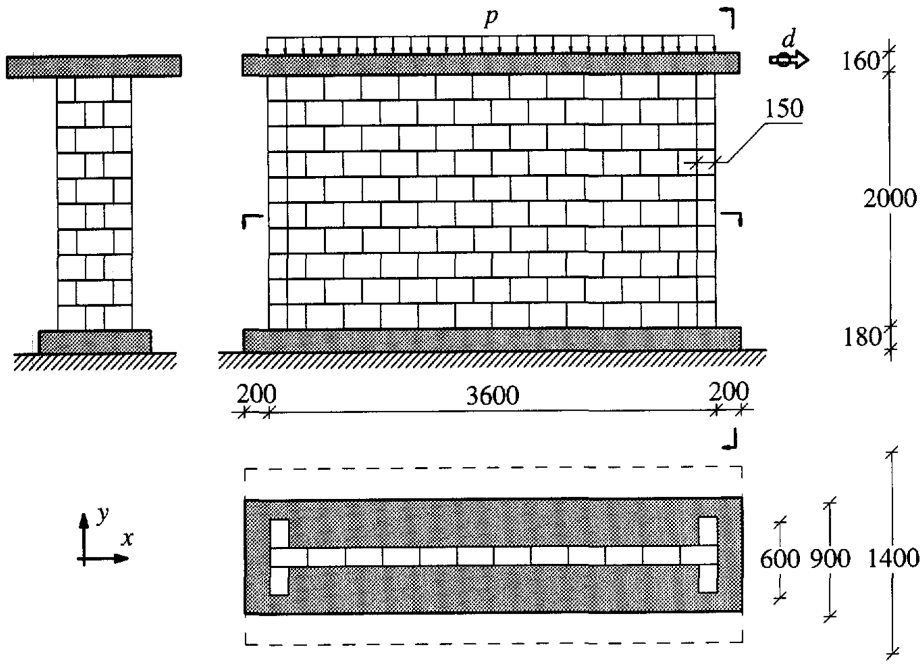


Figure 5.2: Geometry and loads for ETH Zurich shear walls (from [56])

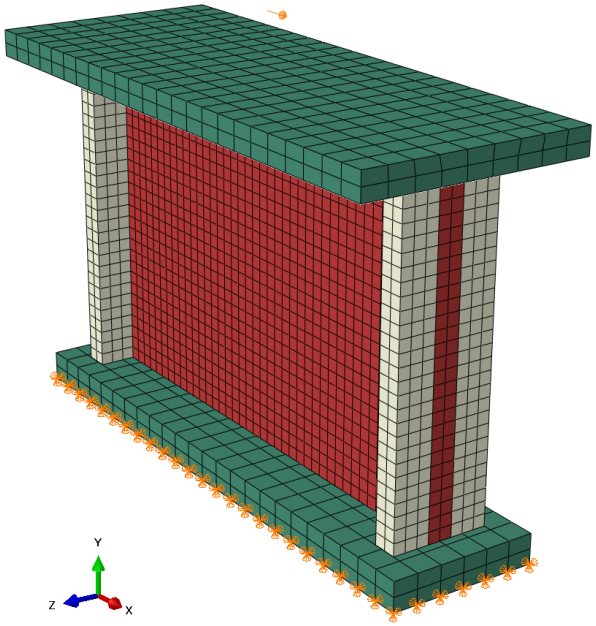
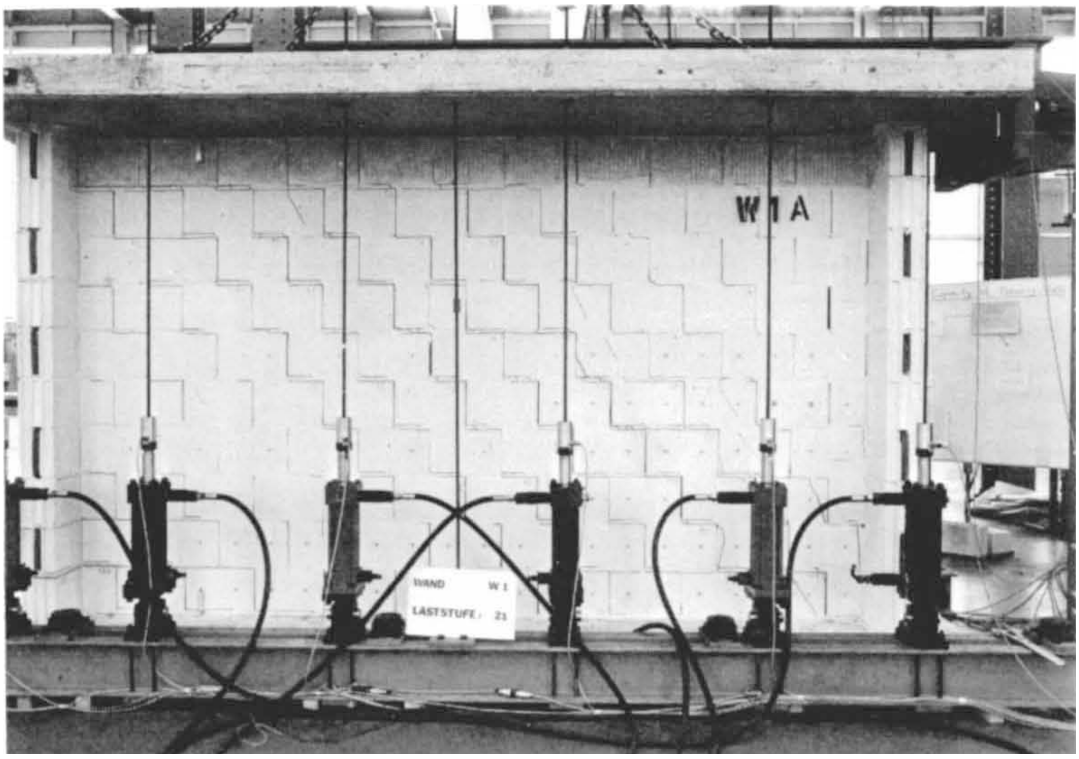
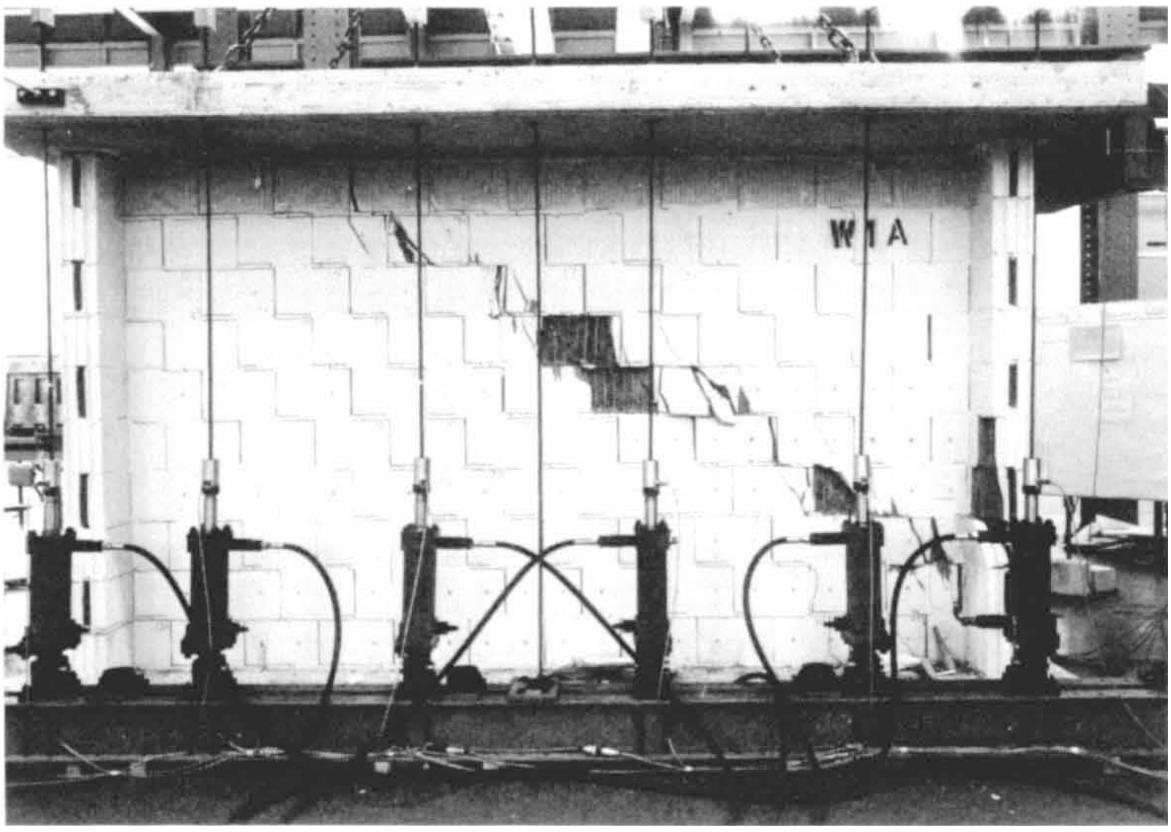


Figure 5.3: Mesh and boundary conditions of walls used in the numerical analyses (different colors denote different material properties applied to the elements)



(a)



(b)

Figure 5.4: Wall $W1$. Experiment failure patterns: (a) at peak load; (b) at end stage

5.3. Results of the analyses

First, the wall $W1$ is analyzed. The wall has an applied distributed load $P = 416\ [kN]$. The numerical results show good agreement to the experimental tests both in horizontal force – displacement curve and in crack patterns of the wall. Furthermore, the developed material model combined with an explicit integration analysis provides exceptional stability. The test was completed successfully without an extra effort or special modeling techniques, by contrast, where numerical analysis (implicit integration) reported by Lourenco diverged without reaching 15mm horizontal displacement point.

By analyzing the force – displacement diagrams (see Figure 5.5) it can be observed that the cracking of the wall starts at around 1mm of displacement. There are some fluctuations in the magnitude of the horizontal force, due to a sudden tensile failure of the elements. Furthermore, because the analysis is executed with 1s of simulation time, the dynamic effects further increase the fluctuations in the force-displacement curve.

It further has to be noted that up until 8mm of displacement the dilation throughout the whole wall is constant. After that, first elements surpass the set limit of 0.03 volumetric strain and the dilation angle for those elements is set to zero. It forces shear localization and strength degradation.

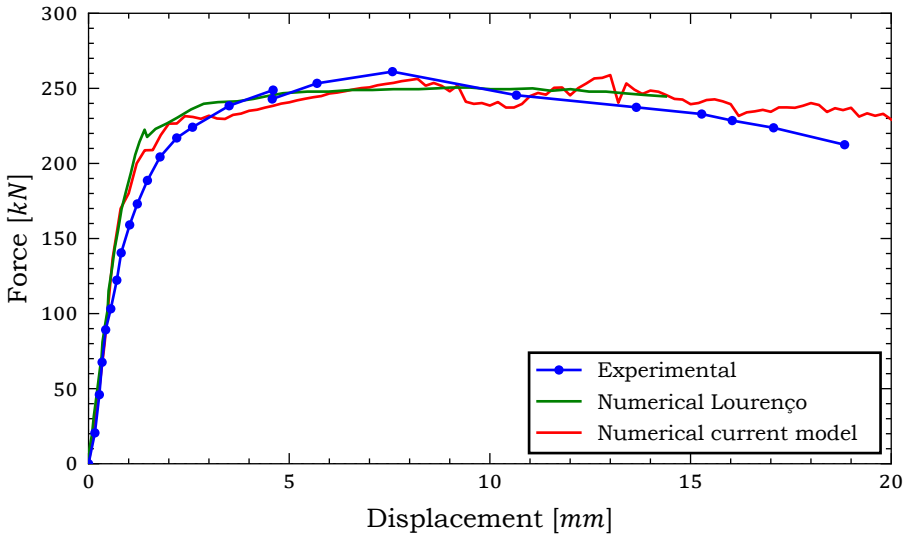
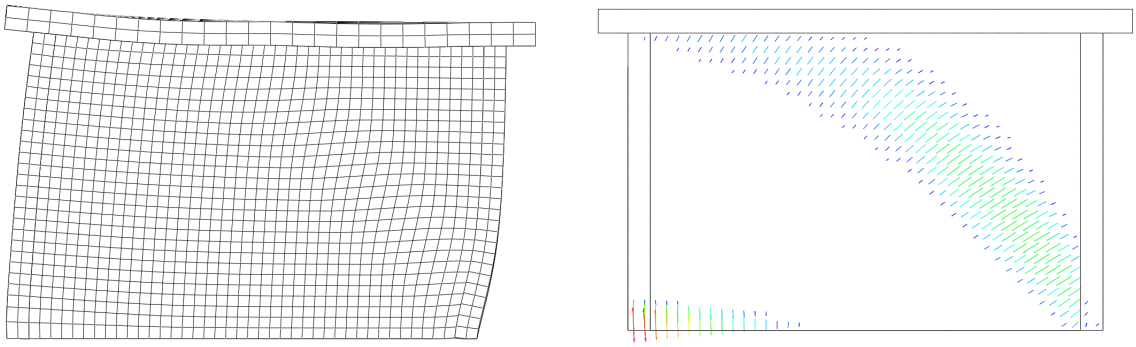


Figure 5.5: Wall $W1$. Horizontal force – displacement diagrams.

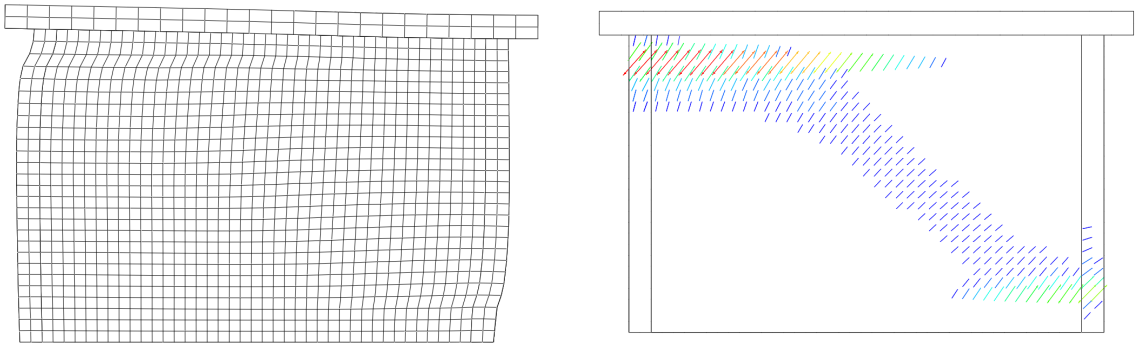
The behavior is the wall is shown in figures 5.6 and 5.7 in terms of deformed meshes and principal strain directions. For the plots of principle strain, the lowest 5% of the values are discarded in order to obtain more legible results. A reasonable agreement can be seen between numerical and experimental results. At the beginning, (see Figure 5.6b) of the analysis the right part of the wall starts to crack in a diagonal direction. This cracking is accompanied by the flexural cracks at the left toe of the wall. As seen later the later crack will close and will be replaced by flexural cracks at the upper part of the left flange.

Upon increasing of the loading (see Figure 5.7b), cracking starts to concentrate around the initially cracked diagonal strip. The cracking band angle and position coincides with the experimental results. Furthermore, after a point of 8mm of displacement shear localization starts appearing at top left and bottom right parts of the wall. This is due to the dilation angle being set to zero at the elements in those locations. However, the localization appears to propagate horizontally instead of diagonally due to it being sensitive to the element orientation. In this case, it does not alter the accuracy of the results significantly as in the experiment, flange flexural failure was observed at the approximate position. But it means that in cyclic shear analysis dilation angle cannot be set to zero as that would induce localized shear in a horizontal strip that might significantly alter the analysis results. On the other hand, non-zero dilation angle produces shear failure mode that seems to be determined by the loading conditions and the geometry of the wall rather than the orientation of the elements.



(a) (b)

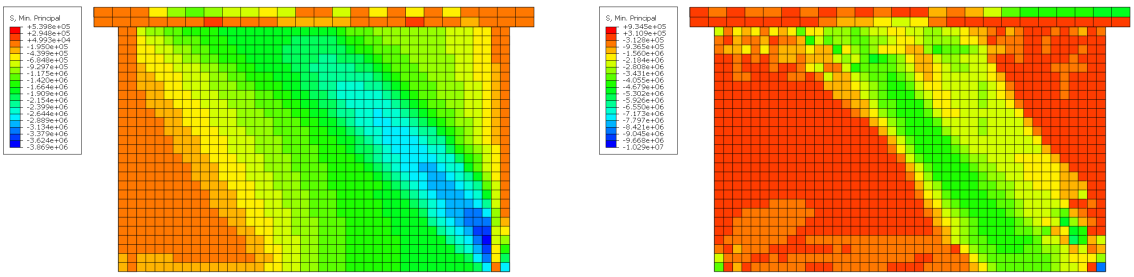
Figure 5.6: Wall W1. Results of the analysis at the displacement of 2mm: (a) deformed mesh (scale 100); (b) max. principle strain directions



(a) (b)

Figure 5.7: Wall W1. Results of the analysis at the displacement of 20mm: (a) deformed mesh (scale 5); (b) max. principle strain directions

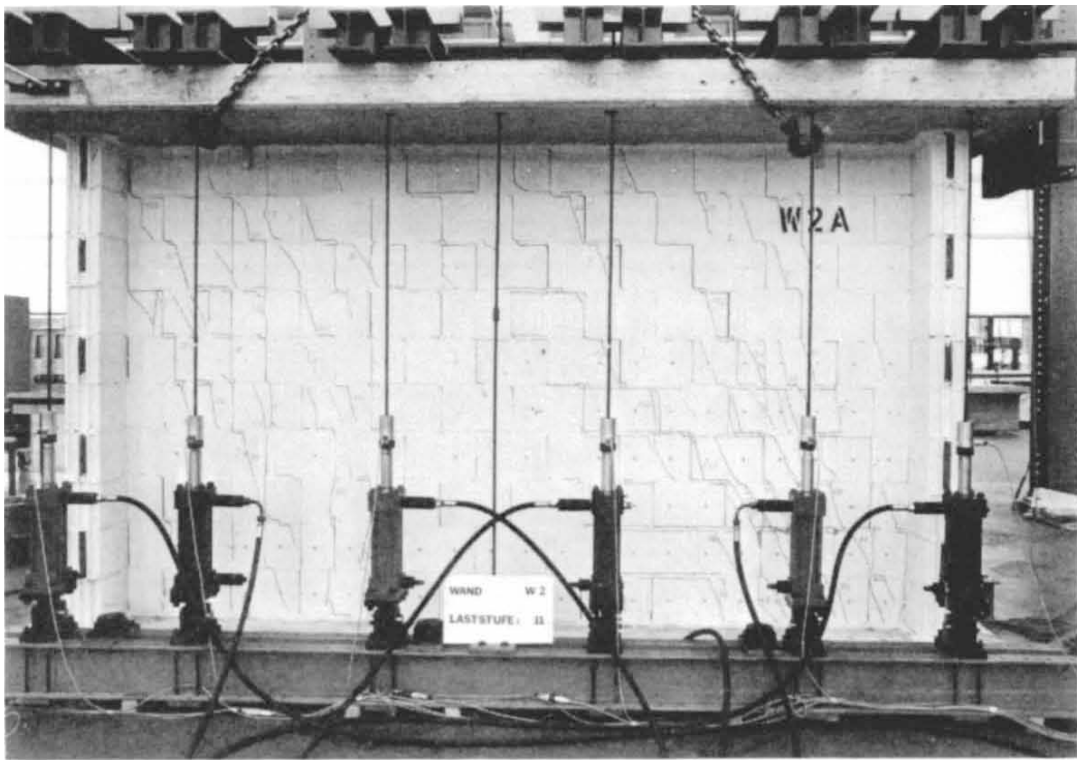
Figure 5.8 shows the contour of minimum principal stresses at the same stages of analysis. Upon the increasing of the loading, it can be observed that the stresses concentrate in the narrower band with peak stresses at the bottom right corner especially the flange. However even at the displacement of 20mm, it can be seen that the stresses do not exceed maximal compressive strengths, therefore it can be concluded that the failure is completely governed by the tension regime.



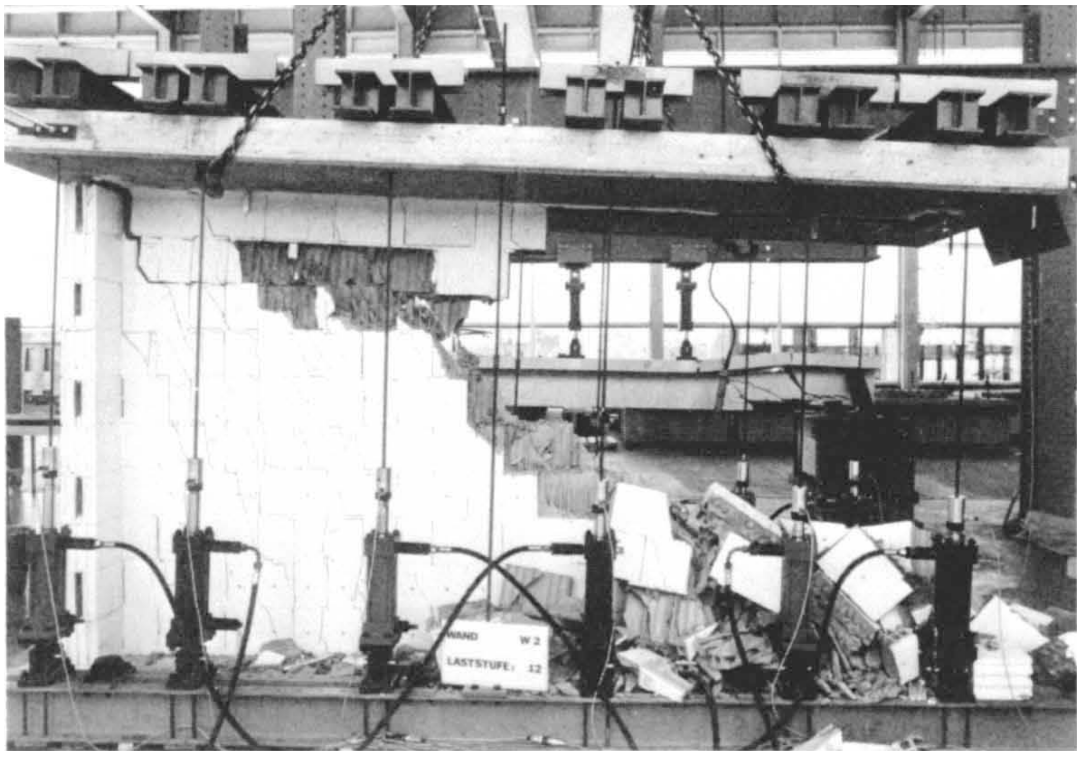
(a) (b)

Figure 5.8: Wall W1. Minimal principal stresses at the displacement of (a) 2 mm and (b) 20 mm.

Next wall $W2$ is analyzed. The wall is subjected to the vertical load P of $1287\ [kN]$. At the beginning of the analysis (see Figure 5.10) the behavior of the wall is ductile followed by a later brittle failure due to compression in the bottom right toe of the wall. As in analysis executed by Lourenço, the force displacement curve is around 10-20% higher than obtained in the experiments. The slight difference in peak strengths comes from slightly different fit obtained for the material properties. However, in general,



(a)



(b)

Figure 5.9: Wall W2. Experiment failure patterns: (a) at peak load; (b) at end stage

the good agreement is found between the analysis and the experiment. Even though the strengths of the wall are higher, the force-displacement curve follows parallel path after 3mm of displacement. At around 8mm of displacement, a sudden failure of the wall is observed.

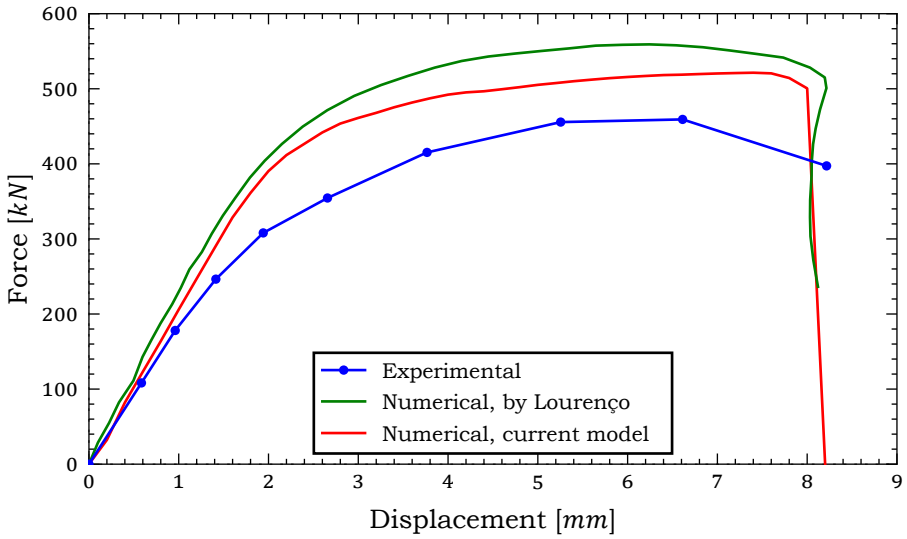
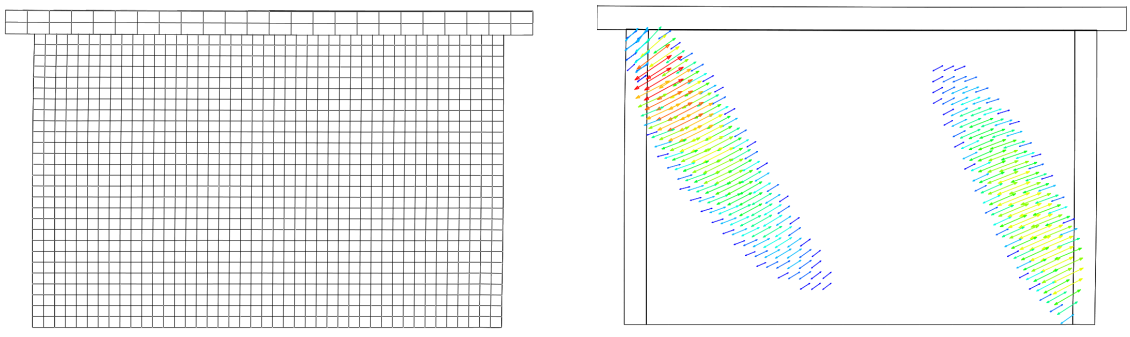


Figure 5.10: Wall $W2$. Horizontal force – displacement diagrams.

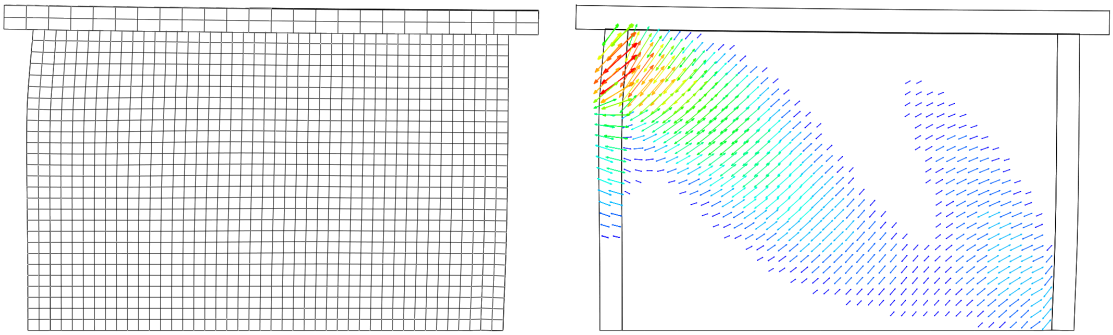
By examining the principal strains (see Figure 5.11) it can be seen that reduction in tangential stiffness around 3mm of displacement is caused by the formation of two crack bands. One at the left side of the wall propagating from the top part of the left flange and another on the right side of the wall propagating from the bottom of the right flange. This behavior can also be observed in the experimental test (see Figure 5.9a) as well.



(a) (b)

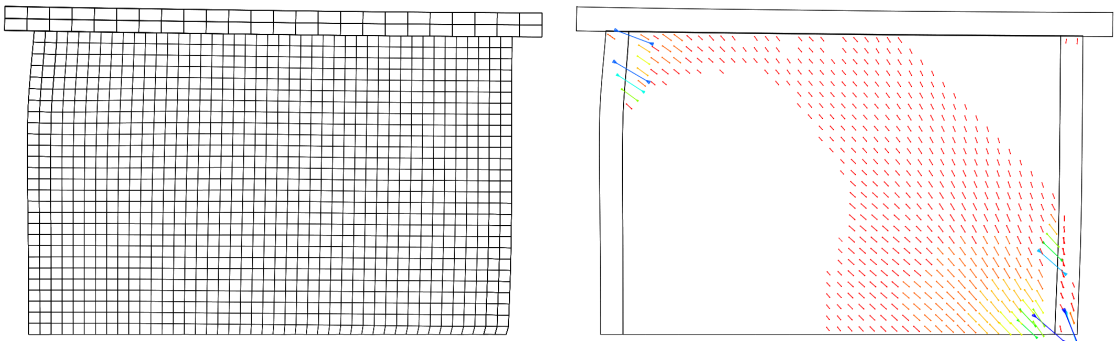
Figure 5.11: Wall W2. Results of the analysis at the displacement of 3mm: (a) deformed mesh (scale 5; (b) max. principle strain directions

Just before the failure of the wall (see Figure 5.12) tensile crack concentration around the initial left crack band can be seen. Furthermore, at the top left flange shear failure between the wall and the flange forms. At the collapse stage (see Figure 5.13) the crushing of the bottom right row of the elements is evident. The crushing forced the stable time increment to reduce and the analysis stopped. One way to avoid such behavior is to enable element deletion. Therefore, the crushed elements would be deleted and the analysis could continue. In this specific test this measure was not required but in other cases e.g. in an analysis of full buildings, element deletion might be required as the analysis should continue even though some elements are crushed in the structure.



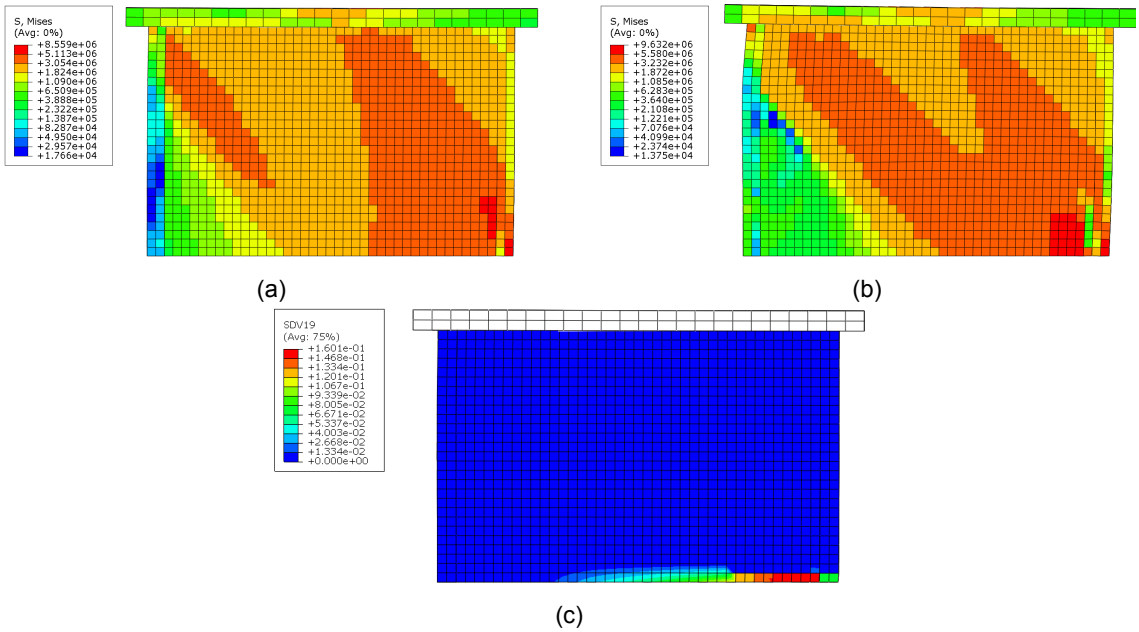
(a) (b)

Figure 5.12: Wall W2. Results of the analysis at the displacement of 8mm just befor the colapse: (a) deformed mesh (scale 5); (b) max. principle stress directions



(a) (b)

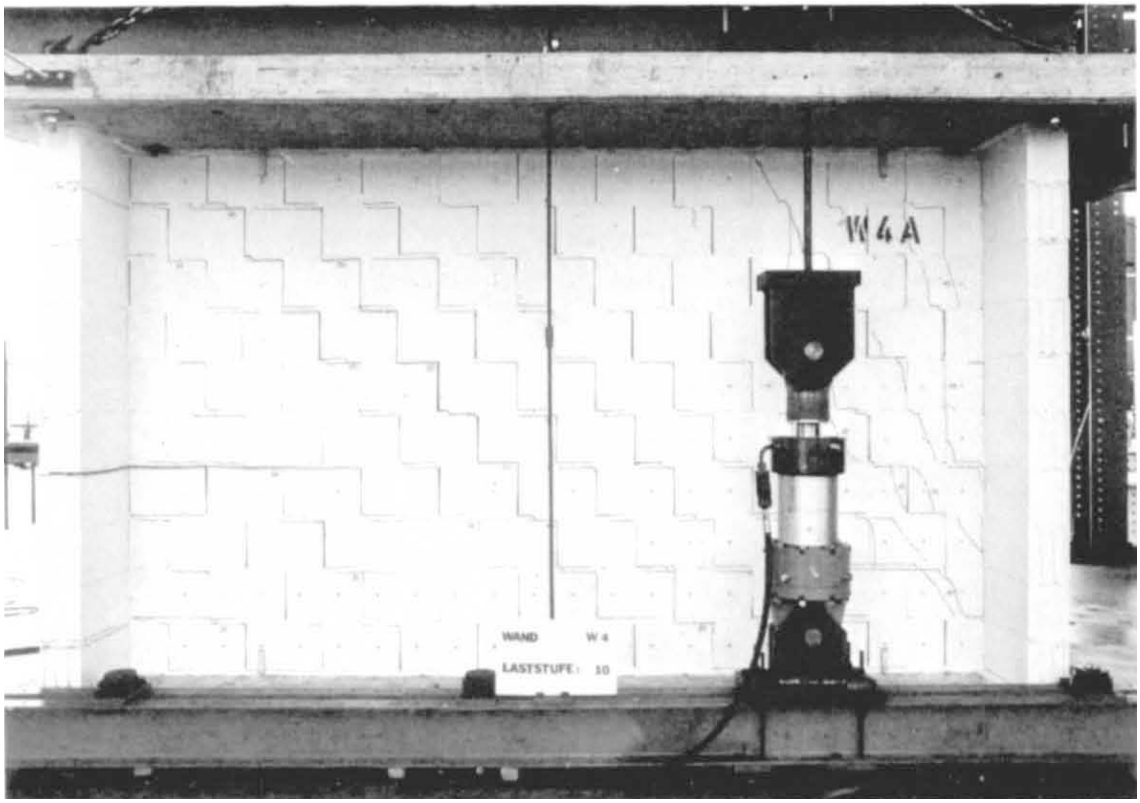
Figure 5.13: Wall W2. Results of the analysis at the displacement of 8mm at the moment of colapse: (a) deformed mesh (scale 5); (b) min. principle stress directions



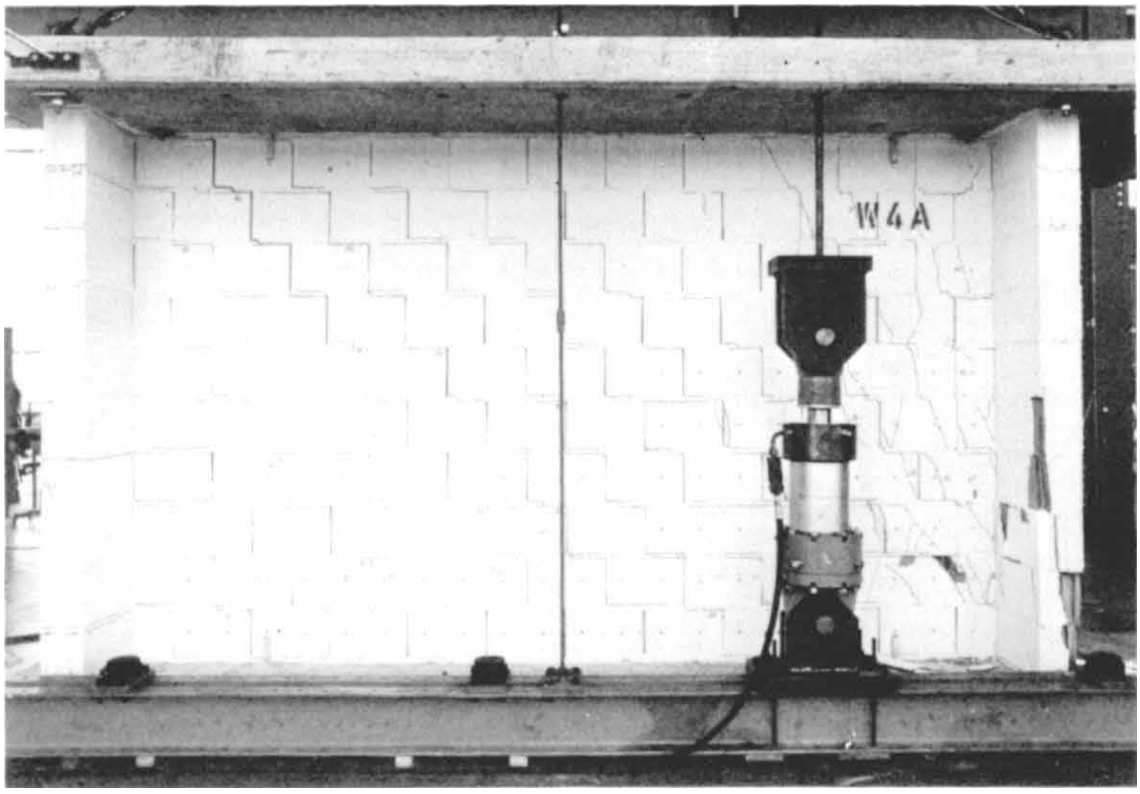
(a) (b)

(c)

Figure 5.14: Wall W2. (a-b) Minimal principal stresses at the displacement of 3 mm and 8 mm (before failure); (c) equivalent plastic strain at 8 mm (after failure).



(a)



(b)

Figure 5.15: Wall $W4$. Experiment failure patterns: (a) at peak load; (b) at end stage

Further, wall $W4$ is analyzed, the wall has a concentrated force $P = 423\ [kPa]$ applied at the distance of $e = 840\ [mm]$ from the center of the wall and distributed through a width of $500\ [mm]$. Moreover, this wall had wider flanges $900\ [mm]$ instead of $600\ [mm]$ like in previous walls. The results from the analysis are again closely related to the experiment (see Figure 5.16). The force-displacement curve follows the experimental results closely, with some minors fluctuations around 3 mm of displacement. The fluctuations are caused due to cracking at the top of the left flange.

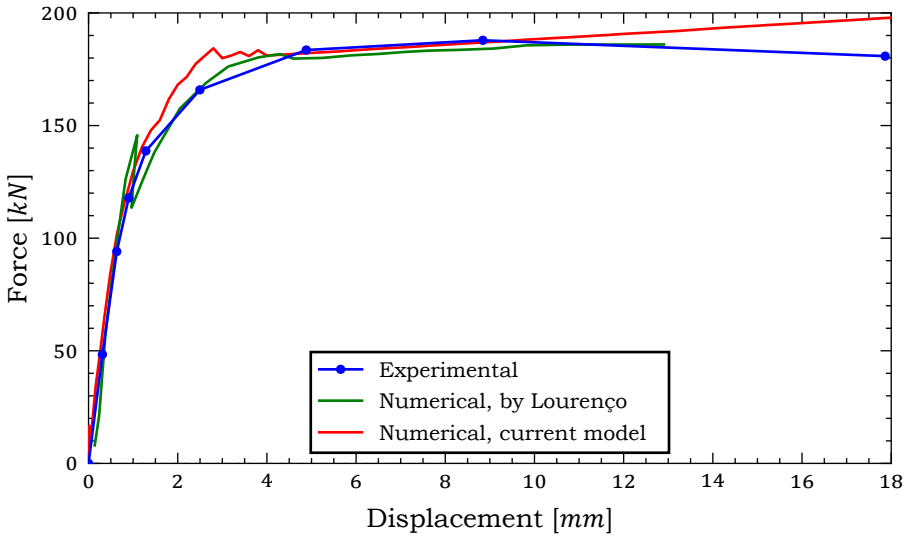
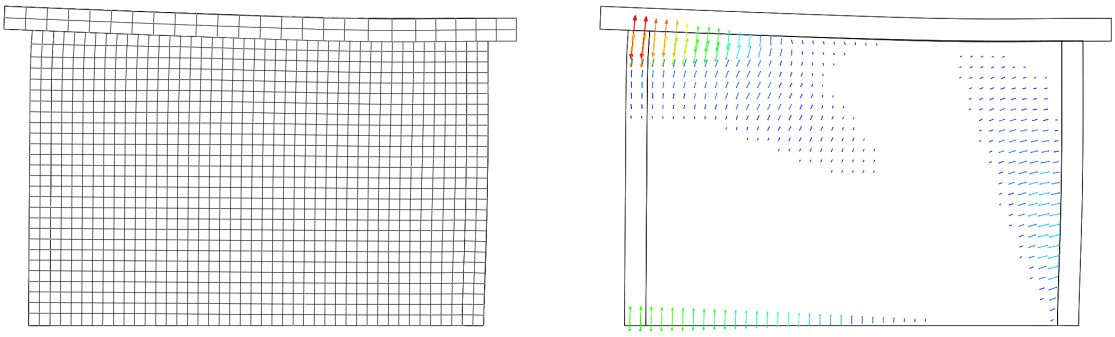


Figure 5.16: Wall $W4$. Horizontal force – displacement diagrams.

Initially, three cracking bands can be identified (see Figure 5.17). The most major one is at the top left, flange and the concrete pad separation, then bottom left toe and bottom pad separation and almost vertical cracking originating from the bottom right corner of the wall and following upwards along the right flange. The crack band right of the wall and the crack and top left of the wall can be also identified in the experiment (see Figure 5.15a). However, none of the cracks at the bottom right toe of the wall are visible.

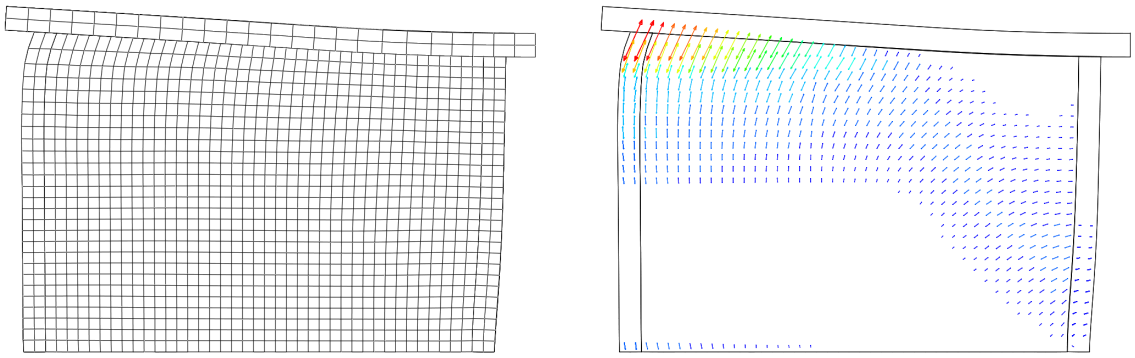
At later stages of the analysis (see Figure 5.18) the cracking at the top left part of the wall expands further and merges with the cracking on the right side of the wall, while the crack at the bottom left toe of the wall closes. These patterns can be also observed in the experiment (see Figure 5.15b). There it appears as the horizontal and vertical cracks between the bricks.

By analyzing minimum principle stresses in the wall at later stages of the analysis (see Figure 5.19b). It can be observed that the crushing starts to appear at the bottom right part of the wall, just nearby



(a) (b)

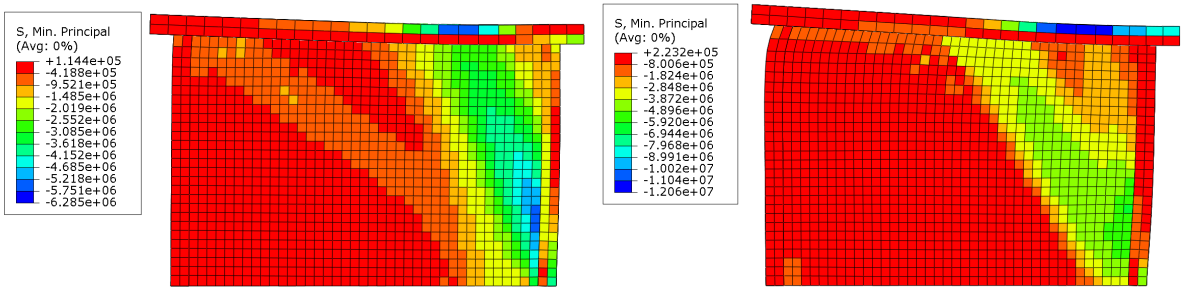
Figure 5.17: Wall $W4$. Results of the analysis at the displacement of 3mm: (a) deformed mesh (scale 10); (b) max. principal strain directions



(a) (b)

Figure 5.18: Wall $W4$. Results of the analysis at the displacement of 17mm: (a) deformed mesh (scale 5); (b) max. principal strain directions

the right flange. Additionally, some crushing appears at the bottom right part of the right flange. In the experiment, the crushing crack patterns are also observed at the right side of the wall and at the bottom of the right flange.

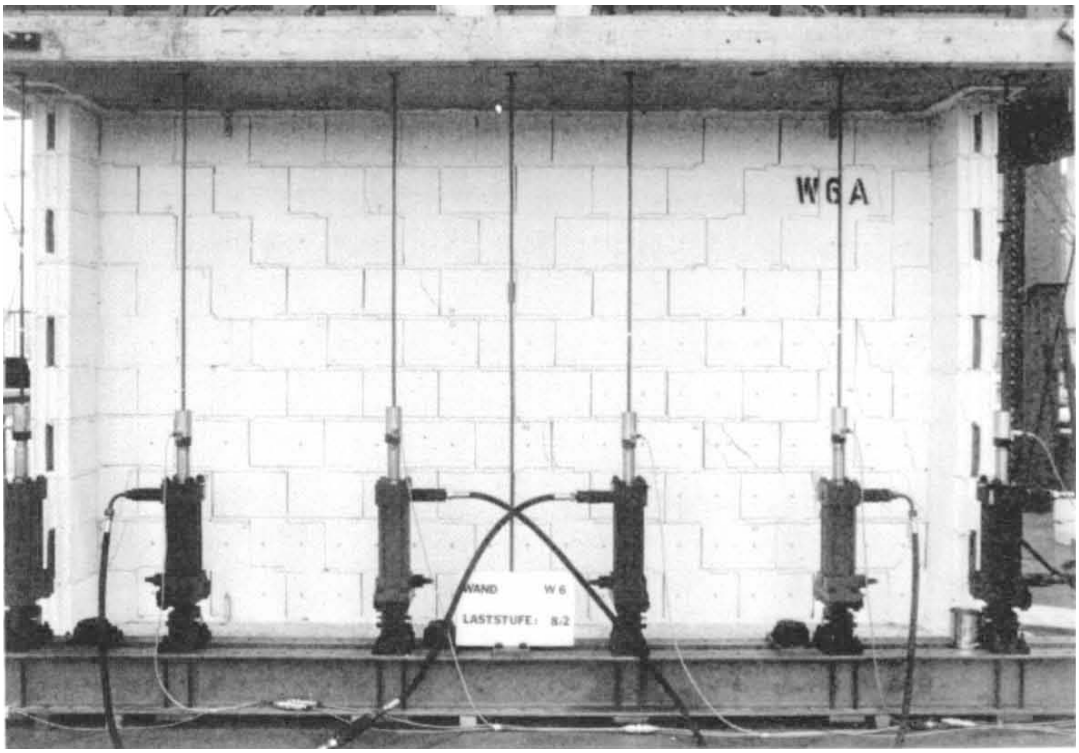


(a) (b)

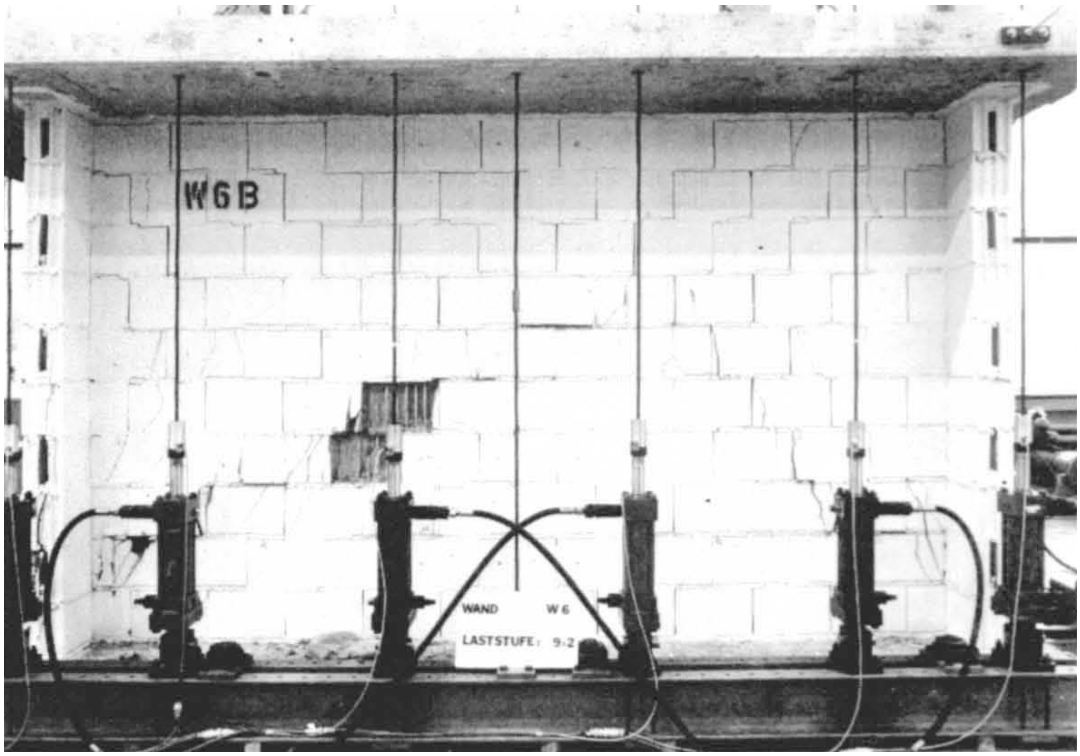
Figure 5.19: Wall $W4$. Minimal principal stresses at the displacement of (a) 3 mm and (b) 17 mm.

Finally, the cyclic analysis of wall $W6$ is executed. The wall has an applied distributed load resulting in $P = 418\ [kN]$. Using the previous analysis settings resulted in a horizontal shear localization at the top part of the wall. The localization has high effects on crack patterns and reaction forces, therefore, for this analysis, the dilation angle was not set to zero and was kept constant. Furthermore, the experiment was carried out trough a long period of time and it had 9 phases (denoted $LS1$ to $LS9$) where each phase consisted of 10 cycles. Such analysis is infeasible with explicit integration method as it would take indefinitely long to complete the simulation. Therefore, a simplified signal was constructed. The new signal consists of 6 phases (see Figure 5.21) starting from a phase $LS4$ as the lower phases act only in the elastic domain of the wall. Each of the phases contains 3 full cycles and lasts 2s.

Each of the phase used in numerical and experimental analysis had an imposed maximal horizontal displacement at the top of the wall of $0.4, 0.8, 1.5, 3, 6, 9\ [mm]$ respectively starting from phase $LS4$. It is import to note that this analysis is heavily simplified, not only by the shorter duration of the test but also by fewer cycles. Furthermore, in the experiment each cycle in a phase had a longer period than in a previous phase, wherein the numerical analysis all of the periods are the same. Therefore, it is extremely difficult to compare the results of the experimental test and the numerical analysis.



(a)



(b)

Figure 5.20: Wall W6. Experiment failure patterns: (a) at peak load; (b) at end stage

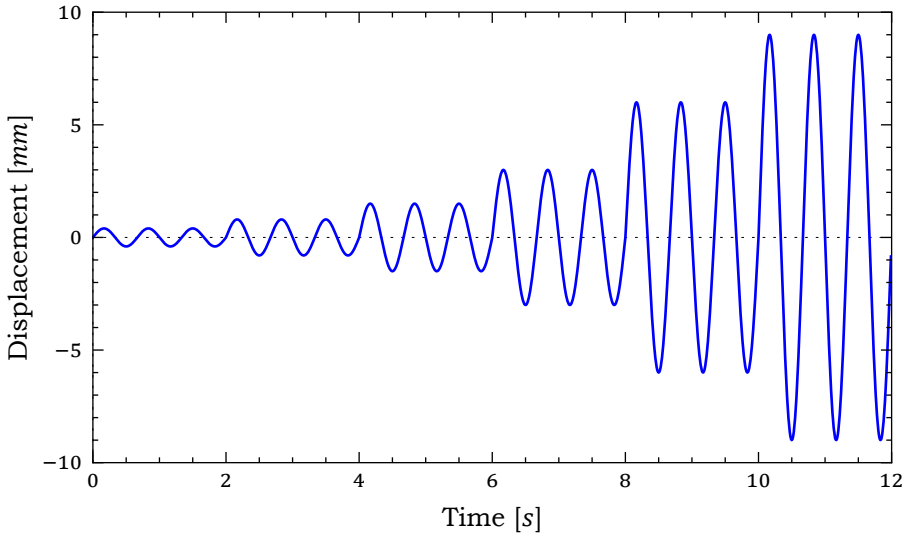


Figure 5.21: Wall $W6$. Input horizontal displacement over time.

Despite the carried out simplifications, by analyzing force-displacement graph (see Figure 5.22), good agreement with the strengths envelope is found. At the last phase, in the experiment, some strength degradation is observed, while in the numerical analysis, this strength degradation is not seen. The main reason might be that the phases in the analysis only had 3 cycles instead of 10. Although at the end of the last cycle, there is a peak of lower strengths, this suggests that if the analysis were to continue, strength degradation would be present.

However, the dissipated energy in the numerical simulation is much higher than in experimental analysis. This issue could be explained by incorrectly modeled boundary conditions of the wall, as it is seen in the experimental results, the wall experiences rocking behavior, however, in the analysis, some degree of rocking was only observed at later cycles. Further examining the pictures provided in the experiment it can be seen that the wall has cracked at the connection of the wall and concrete slab, both, at the bottom and at the top.

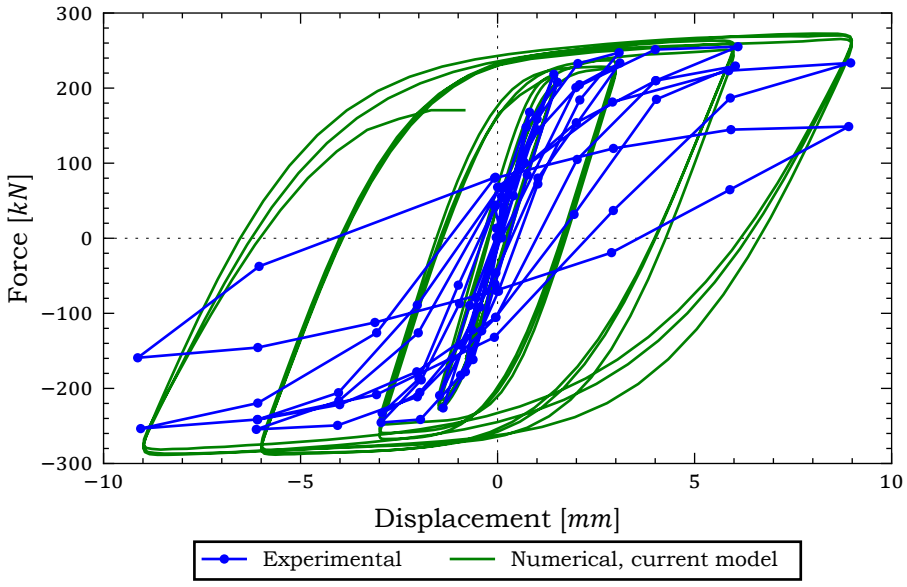
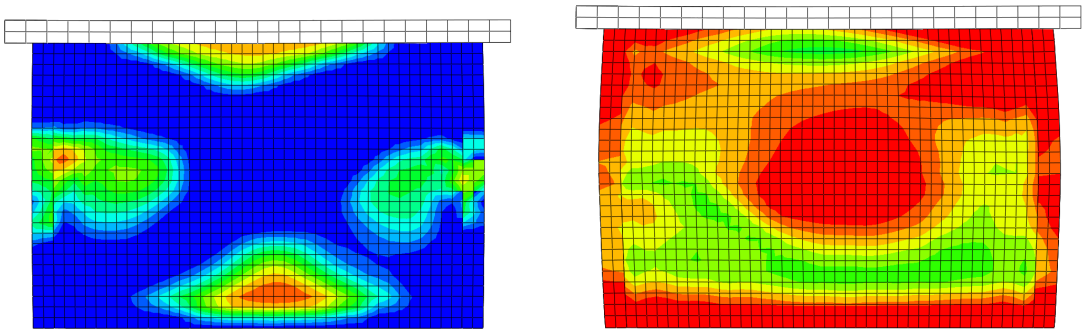


Figure 5.22: Wall $W6$. Horizontal force – displacement diagrams.

To compare the cracking patterns is a difficult task as both in the experiment and in the numerical analysis, the whole wall was cracked. However, in an experiment, a prominent “X” pattern on the wall is seen. By analyzing the results of the analysis it can be observed that the “X” crack pattern can be first seen to form at the analysis time of $7.25\ [s]$ (see Figure 5.23a). Unfortunately, at the later stages of the analysis, the crack pattern is lost in the noise of cracks as the whole wall is cracked. Nevertheless, the crack concentrations can be examined by looking at the equivalent plastic strain generated by the yielding on the Rankine type surface (see Figure 5.23b). There, a higher concentration of equivalent plastic strain can be seen at the mid-wall, flanges, and diagonals spanning from the wall center to the top corners of the wall. Even though, in the experiment a clear Mode-II crack can be seen at the center of the wall, it had more prominent cracks spanning from the center to the bottom corners of the wall.

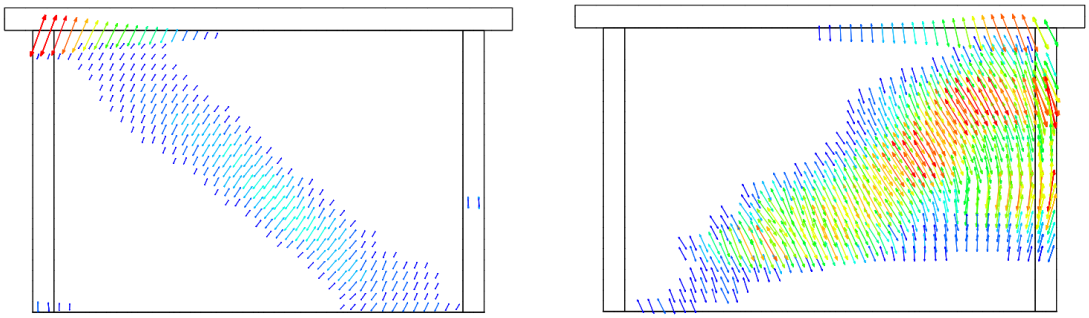


(a) (b)

Figure 5.23: Wall $W6$. Results of the numerical analysis: (a) Tensile strengths degradation at the time of $7.25[s]$ (blue – fully degraded, 1% remaining strength, red – not degraded, 100% remaining strengths); (b) equivalent plastic strain generated by Rankine like yield surface at time of $12[s]$ (red – more strain, green – less strain); in both figures, deformations are to scale.

The extra noise added to the late stage crack patterns can be attributed to the fact that the plastic strain is recorded every time the Rankine type surface is yielding, however, the model also exhibits crack closure in tension. Therefore, in some regions, the generated and displayed plastic strain is fictitious and does not affect the behavior of the wall. As an example, such regions are the flanges of the wall. While on tension, if they are cracked, the cracks would open up and generate fictitious plastic strain, however later when tension is removed and compression is applied, the cracks would close due to the damage in stiffness but the generated plastic strain would remain recorded. Additionally, it should be noted that due to the fact the dilation angle was kept constant throughout the analysis, the wall dilated substantially, furthermore, because of the flanges providing the horizontal constraint, higher stresses was to be expected than in the experiment.

Despite that, a strain increment output could be generated, it would show regions where the strain rate is highest at any given increment. These regions would approximately (the main crack path can change throughout analysis) show the crack pattern in the structure. This kid of output is not available by default in Abaqus. Therefore, a Python script has to be written in order to obtain it (See Appendix B). By examining such output (see 5.24) a reasonable agreement with experimental results is found. Diagonal cracks can be observed starting from the top left and top right parts of the wall. Also, a wall and a concrete slab separation at the top can be observed in both peaks of the displacement. In the experiment, more prominent cracking can be seen in the lower left bottom area of the wall while in the analysis – upper right. But, then again, these differences can be influenced due to the imperfections in the experimental specimens or due to the numerical model simplification for the analysis (i.e. the way the displacement is applied).



(a) (b)

Figure 5.24: Wall $W6$. Results of the numerical analysis: (a) strain increment at peak displacement to the left 11.6 $[s]$; (b) strain increment at peak displacement to the right 11.9 $[s]$; in both figures, deformations are to scale.

5.4. Summary

Four shear walls were analyzed, three subjected to monotonic in-plane horizontal displacement and one to dynamic cyclic horizontal displacement. By examining the obtained results, good agreement with the experimental data was found. The crack patterns and F-U curves go hand in hand with the results of the experiments. As for cyclic analysis, similar behavior in therms of strength envelope was observed, however, the analysis yielded higher energy dissipation than it was present in the experiments. It was concluded that in the experiment rocking behavior was observed and in the analysis it was lacking due to strict boundary conditions that did not allow the separation of the wall base and the bottom concrete pad. Furthermore, it was difficult to define the compression softening curve as the experiment was lacking such data. A curve was assumed, but it did not yield the most favorable results ($W2$ and $W6$) as wanted strength degradation was not observed. Better fitting the compression curve would have yielded better results.

6

Conclusions and recommendations

6.1. Conclusions

In this thesis the following objectives were achieved:

- Through a discussion of basic theoretical considerations it was found that the explicit algorithm is more favorable than the implicit.
- Through a comprehensive literature review the information on the existing knowledge about Continuum Damage Mechanics models were assembled. Two main distinctive works were identified, namely the work Lourenco [56] and the work of Pela [75]. After investigation of strengths and drawbacks of both models, it was decided to use Lourenco [56] model as the basis of the thesis.
- A stable, accurate and robust orthotropic composite masonry model for explicit solver has been developed. The model exhibits 3D elasticity and 2D plane plasticity. Therefore, the modeling and application of boundary conditions are simplistic. Furthermore, it is assumed that two general failure mechanisms are present. One associated with tensile and shear brittle fracture represented by Rankine type yield surface and another with distributed crushing of a material represented by Hill type yield surface. The model exhibits uncoupled damage evolution in tension regime and coupled in compression. The shear brittle fracture and distributed crushing are controlled by $\varepsilon_{pl,t}$ and $\varepsilon_{pl,c}$ respectively. Additionally, the model supports tensile crack closure, while in compression it accumulates the plastic deformations. The model is formulated in such a way that most of the properties in material directions are independent of one another.
- The intersections of the yield surfaces and the apex of the Rankine-like surface cone cause a problematic stress return in the subdifferential zone. The underlying robust algorithms used to mitigate these issues were derived that yielded stable and accurate results.
- Additional measures were taken to ensure stable and smooth analysis, stiffness proportional term of Rayleigh damping was incorporated together with stable time estimation based on the effects of the added damping. To even further increase the stability of an explicit analysis an option to remove crushed element is considered.
- The developed model was tested by examining its behavior in one or few element tests and by comparing it with experimental results. For experimental comparison, four shear walls were modeled, three subjected to monotonic loading and one to cyclic. The analyses closely agree to experimental results even when using raw test data. The model is stable due to explicit approach and it is flexible enough to be used and provide qualitative results in various types of analyses, static or cyclic.

However, during the testing of the model, several flaws emerged:

- The developed nonlinearly decreasing dilation angle appeared to be unstable when elements are subjected to tension regime, by setting of the dilation to zero did not help much as it forced the

shear to localize in horizontal bands and just by having constant dilation influenced the deformations of the cyclic tests as too extensive lift of the wall was observed.

- The way that equivalent plastic strain is being tract appears to be flowed as it does not take in to account the closure of the cracks. So the plastic strain output is not what one would expect. However, this issue could be eliminated with the future iterations of the code. Or temporary mitigated by analyzing strain increment output data.

In addition to the main objectives, two other programs were developed to aid the analysis of masonry:

- A test data fitter in order to obtain the material properties from raw experimental results. It can also be used to check the quality of the fit (see Appendix A).
- An output database formatter, that can be used to rename and vectorize custom output variables and also create a strain increment output (see Appendix B).

6.2. Recommendations for analyses and future development

6.2.1. Further development on non-associated flow

Current model incorporates a smooth flow function for Rankine yield surface. However, this function has drawbacks. It is highly recommended to describe a more stable flow algorithm that does not loose stability when the dilation angle is reduced. Such algorithm would be especially useful in cyclic analyses in which the dilation of the structure could be controlled and excessive lift could possibly be avoided.

6.2.2. Localized damage

Shear damage appears to be localizing in horizontal bands (mesh orientation dependent) when the dilation angle is set to zero and partially localizing (wide bands of cracks) if the dilation angle is kept constant. To this, there could be several reasons. First, the single integration elements are subjected to hour-glassing. Elements affected by hour-glassing will have lower strengths than they are supposed to, therefore, when the dilation angle is set to zero the path of the least resistance for shear propagation are in the orthogonal directions. However applying some means of hour-glassing control exerts unrealistic forces on to the structure (see chapter 27.1.4 Section controls in [14]) and that may influence the results of the analysis in a negative way. Second, the brick elements may be unsuitable for analyses where diagonal localizations are expected as the localization tends to propagate in orthogonal directions. Third, the time step even in explicit analyzes is too big to obtain expected localization. The stress increment per one time increment encompasses several elements, therefore, damaging bigger bands of a mesh.

If the first issue is governing the solution would be to develop a custom element that would have at least four integration points. This way the hour-glassing would be avoided and elements would have appropriate stiffness and strengths when subjected to hour-glassing modes. If the second issue is governing, it is possible to use different type of element already provided by Abaqus such as solid tetrahedral elements. However, they should be used with caution as many types of them are either overly stiff and extremely fine mesh is required for accurate analysis or they exhibit “volumetric locking” at large plastic deformations (see Section 28.1.1 Solid (Continuum) elements in [14]). And finally, if the last case is governing, the solutions would be either to lower the stable time increment,which could lead to a drastic increase of the analysis time until the desired result is obtained or to develop a custom element that would only let the damage propagate in the directions perpendicular to the direction of principal strain. Such algorithm was proposed and discussed by Clemente [12].

6.2.3. Improve compression behavior

The compression behavior in this material model was developed using a rather simplistic approach. No stiffness damage was applied during the crushing of the material, while in experimental tests [22], such damage can be observed to some extent. Even though, the masonry analysis relevant for Groningen region the compressive crushing is not governing, it would increase the field of application and make the model more universal. Additionally, the localization for the compression crushing appears to be mesh and stress rate dependent, therefore often the failure due to compression may be unpredictable

and volatile. This issue could be minimized using higher values for distortion control or for quadratic bulk viscosity (see chapter 27.1.4 Section controls and chapter 6.3.3 Explicit dynamic analysis in [14]).

6.2.4. Further testing

Even though the material model is tested and the results are compared to 4 experimental tests. Further testing still has to be done. Such additional tests could include shear walls with openings, walls subjected to out of plane two-way bending, and full-scale buildings in order to fully verify the material model.

Bibliography

[1] Quakeshield. URL http://www.quake-shield.com/over-quakeshield/.

[2] Argiris Alexandris, Eleni Protopapa, and Ioannis Psycharis. Collapse mechanisms of masonry buildings derived by the distinct element method. In *Proceedings of the 13th world conference on earthquake engineering*, pages 1–6, 2004.

[3] A Anthoine. In-plane behaviour of masonry(a literature review). *EUR(Luxembourg)*, 1992.

[4] RH Atkinson, BP Amadei, S Saeb, and S Sture. Response of masonry bed joints in direct shear. *Journal of Structural Engineering*, 115(9):2276–2296, 1989.

[5] HP Backes. Tensile strength of masonry. *Proc. 7 IBMAC, Melbourne*, pages 779–789, 1985.

[6] Klaus-Jürgen Bathe. *Finite element procedures*. Klaus-Jurgen Bathe, 2006.

[7] Klaus-Jürgen Bathe and Edward L Wilson. Numerical methods in finite element analysis. 1976.

[8] Davide Bigoni and Andrea Piccolroaz. Yield criteria for quasibrittle and frictional materials. *International journal of solids and structures*, 41(11):2855–2878, 2004.

[9] Boris Bresler and Karl S Pister. Strength of concrete under combined stresses. In *Journal Proceedings*, volume 55, pages 321–345, 1958.

[10] Encyclopædia Britannica. Richter scale, 2016. URL https://www.britannica.com/science/Richter-scale.

[11] Robert M Caddell, Ram S Raghava, and Anthony G Atkins. A yield criterion for anisotropic and pressure dependent solids such as oriented polymers. *Journal of Materials Science*, 8(11):1641–1646, 1973.

[12] R Clemente. *Análisis estructural de edificios históricos mediante modelos localizados de fisuración*. Centro Internacional de Métodos Numéricos en Ingeniería, 2007.

[13] Charles Augustin Coulomb. *Essai sur une application des règles de maximis & minimis à quelques problèmes de statique, relatifs à l'architecture*. De l'Imprimerie Royale, 1776.

[14] Dassault Systèmes. *Abaqus Analysis User's Guide*, 2015.

[15] Dassault Systèmes. *Abaqus User Subroutines Reference Guide*, 2015.

[16] D D'ayala and E Speranza. An integrated procedure for the assessment of seismic vulnerability of historic buildings. *disp*, 3(1):3–3, 2002.

[17] Eduardo A de Souza Neto, Djordje Peric, and David Roger Jones Owen. *Computational methods for plasticity: theory and applications*. John Wiley & Sons, 2011.

[18] Vikram S Deshpande, Norman A Fleck, and Michael F Ashby. Effective properties of the octet-truss lattice material. *Journal of the Mechanics and Physics of Solids*, 49(8):1747–1769, 2001.

[19] M Dhanasekar, AW Page, and PW Kleeman. The failure of brick masonry under biaxial stresses. *Proceedings of the Institution of Civil Engineers*, 79(2):295–313, 1985.

[20] Daniel Charles Drucker and William Prager. Soil mechanics and plastic analysis or limit design. *Quarterly of applied mathematics*, 10(2):157–165, 1952.

[21] Rita Esposito, Francesco Messali, and Jan Rots. Tests for the characterization of replicated masonry and wall ties. physical testing and modelling – masonry structures., 2016.

[22] Graziotti F., Tomassetti U., Rossi A., Kallioras S., Mandirola M., Cenja E., Penna A., and Magenes G. Experimental campaign on cavity walls systems representative of the groningen building stock. techreport, EUCENTRE, 2015.

[23] Rui Faria, Javier Oliver, and Miguel Cervera. Modeling material failure in concrete structures under cyclic actions. *Journal of Structural Engineering*, 130(12):1997–2005, 2004. doi: 10.1061/(ASCE)0733-9445(2004)130:12(1997).

[24] GU Fonseka and D Krajcinovic. The continuous damage theory of brittle materials, part 2: uniaxial and plane response modes. *Journal of Applied Mechanics*, 48(4):816–824, 1981.

[25] L. Gambarotta and S. Lagomarsino. Damage models for the seismic response of brick masonry shear walls. part i: The mortar joint model and its applications. 26(4):423–439, April 1997. doi: 10.1002/(sici)1096-9845(199704)26:4<423::aid-eqe650>3.0.co;2-.

[26] L. Gambarotta and S. Lagomarsino. Damage models for the seismic response of brick masonry shear walls. part II: The continuum model and its applications. *Earthquake Engng. Struct. Dyn.*, 26(4):441–462, apr 1997. doi: 10.1002/(sici)1096-9845(199704)26:4<441::aid-eqe651>3.0.co;2-0.

[27] Hans Rudolf Ganz and Bruno Thürlimann. *Versuche über die Festigkeit von zweiachsig beanspruchtem Mauerwerk*. Birkhäuser, 1982.

[28] Hansruedi Ganz and Bruno Thürlimann. *Versuche an Mauerwerksscheiben unter Normalkraft und Querkraft*. Springer, 1984.

[29] M. Gilbert, C. Casapulla, and H.M. Ahmed. Limit analysis of masonry block structures with non-associative frictional joints using linear programming. *Computers & Structures*, 84(13–14):873 – 887, 2006. ISSN 0045-7949. doi: http://dx.doi.org/10.1016/j.compstruc.2006.02.005. URL http://www.sciencedirect.com/science/article/pii/S0045794906000356.

[30] Matthew Gilbert. Limit analysis applied to masonry arch bridges: state-of-the-art and recent developments. In *Proceedings of 5th International Conference on Arch Bridges (ARCH'07), PB Lourenço, DB Oliveira, and A. Portela, eds., Funchal, Madeira, Portugal*, pages 13–28, 2007.

[31] Melbourne Gilbert and C Melbourne. Rigid-block analysis of masonry structures. *Structural engineer*, 72(21), 1994.

[32] A Giuffrè. Vulnerability of historical cities in seismic areas and conservation criteria. In *Congress "Terremoti e civiltà abitative", Annali di Geofisica, Bologna*, 1995.

[33] Antonino Giuffrè. Letture sulla meccanica delle murature storiche. 1990.

[34] Roland Guggisberg and Bruno Thürlimann. *Versuche zur Festlegung der Rechenwerte von Mauerwerksfestigkeiten*. Birkhäuser, 1987.

[35] Arthur L Gurson. Continuum theory of ductile rupture by void nucleation and growth: Part i—yield criteria and flow rules for porous ductile media. *Journal of engineering materials and technology*, 99(1):2–15, 1977.

[36] Arnold W Hendry. *Structural masonry*. Scholium International, 1990.

[37] R. Hill. A theory of the yielding and plastic flow of anisotropic metals. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 193(1033):281–297, may 1948. doi: 10.1098/rspa.1948.0045.

[38] Hubert K Hilsdorf. Investigation into the failure mechanism of brick masonry loaded in axial compression. *Designing, engineering and constructing with masonry products*, pages 34–41, 1969.

[39] P Hofmann and S Stockl. Tests on the shear-bond behaviour in the bed-joints of masonry. *MASONRY INT. Masonry Int.*, (9):1, 1986.

[40] Thomas J.R. Hughes, Karl S. Pister, and Robert L. Taylor. Implicit-explicit finite elements in nonlinear transient analysis. *Computer Methods in Applied Mechanics and Engineering*, 17:159 – 182, 1979. ISSN 0045-7825. doi: http://dx.doi.org/10.1016/0045-7825(79)90086-0. URL http://www.sciencedirect.com/science/article/pii/0045782579900860.

[41] TJR Hughes and WK Liu. Implicit-explicit finite elements in transient analysis: stability theory. *Journal of applied Mechanics*, 45(2):371–374, 1978.

[42] J Janson and J Hult. Fracture mechanics and damage mechanics- a combined approach. In *(International Congress of Theoretical and Applied Mechanics, 14 th, Delft, Netherlands, Aug. 30-Sept. 4, 1976.) Journal de Mecanique Appliquee,*, volume 1, pages 69–84, 1977.

[43] LM Kachanov. Time of the rupture process under creep conditions. *Isv. Akad. Nauk. SSR. Otd Tekh. Nauk*, 8:26–31, 1958.

[44] Royal Netherlands Meteorological Institute (KNMI). Aardbevingen door gaswinning. URL https://www.knmi.nl/kennis-en-datacentrum/uitleg/aardbevingen-door-gaswinning.

[45] Kristian Krabbenhøft. Basic computational plasticity. *Lecture Notes*, 2002.

[46] D Krajcinovic and GU Fonseka. The continuous damage theory of brittle materials, part 1: general theory. *Journal of applied Mechanics*, 48(4):809–815, 1981.

[47] Dusan Krajcinovic. Continuous damage mechanics revisited: basic concepts and definitions. *Journal of Applied Mechanics*, 52(4):829–834, 1985.

[48] SG Lekhnitskii. *Of an anisotropic elastic body*, volume 525. San Francisco: Holden-Day, 1963.

[49] J Lemaitre and Jean-louis Chaboche. Aspect phénoménologique de la rupture par endommagement. *J Méc Appl*, 2(3), 1978.

[50] Jean Lemaitre and Jean-Louis Chaboche. *Mechanics of solid materials*. Cambridge university press, 1994.

[51] Jean Lemaitre and Rodrigue Desmorat. *Engineering damage mechanics: ductile, creep, fatigue and brittle failures*. Springer Science & Business Media, 2005.

[52] JV Lemos. Discrete element modelling of the seismic behaviour of stone masonry arches. *Computer methods in structural masonry*, 4:220–227, 1998.

[53] J. Lopez, S. Oller, E. Oñate, and J. Lubliner. A homogeneous constitutive model for masonry. *International Journal for Numerical Methods in Engineering*, 46(10):1651–1671, 1999. ISSN 1097-0207. doi: 10.1002/(SICI)1097-0207(19991210)46:10<1651::AID-NME718>3.0.CO;2-2. URL http://dx.doi.org/10.1002/(SICI)1097-0207(19991210)46:10<1651::AID-NME718>3.0.CO;2-2.

[54] Hamid R. Lotfi and P. Benson Shing. Interface model applied to fracture of masonry structures. *Journal of Structural Engineering*, 120(1):63–80, jan 1994. doi: 10.1061/(asce)0733-9445(1994)120:1(63). URL http://dx.doi.org/10.1061/(ASCE)0733-9445(1994)120:1(63).

[55] Paulo B Lourenço. The elastoplastic implementation of homogenisation techniques. Technical report, Delft University of Technology, 1995.

[56] Paulo B Lourenço. *Computational strategies for masonry structures*. PhD thesis, Delft University of Technology, 1996.

[57] Paulo B. Lourenço. Experimental and numerical issues in the modelling of the mechanical behaviour of masonry, 1998.

[58] Paulo B. Lourenço and Jan G. Rots. Multisurface interface model for analysis of masonry structures. *J. Eng. Mech.*, 123(7):660–668, jul 1997. doi: 10.1061/(asce)0733-9399(1997)123:7(660). URL http://dx.doi.org/10.1061/(ASCE)0733-9399(1997)123:7(660).

[59] Paulo B Lourenço, René De Borst, and Jan G. Rots. A plane stress softening plasticity model for orthotropic materials. *International Journal for Numerical Methods in Engineering*, 40(21):4033–4057, nov 1997. doi: 10.1002/(sici)1097-0207(19971115)40:21<4033::aid-nme248>3.0.co;2-0.

[60] F Lurati, H Graf, and B Thürlimann. Experimental determination of the strength parameters of concrete masonry. *Rep. No. 8401*, 2, 1990.

[61] G Macchi. Diagnosis of the facade of st. peter's basilica in rome. *Proceedings of SAHC*, pages 309–18, 1995.

[62] C Melboure and Matthew Gilbert. The application of limit analysis techniques to masonry arch bridges. *DEVELOPMENTS IN CIVIL ENGINEERING*, 45:193–193, 1994.

[63] G. Milani, P.B. Lourenço, and A. Tralli. Homogenised limit analysis of masonry walls, part i: Failure surfaces. *Computers & Structures*, 84(3-4):166–180, jan 2006. doi: 10.1016/j.compstruc.2005.09.005.

[64] G. Milani, P.B. Lourenço, and A. Tralli. Homogenised limit analysis of masonry walls, part II: Structural examples. *Computers & Structures*, 84(3-4):181–195, jan 2006. doi: 10.1016/j.compstruc.2005.09.004.

[65] Gabriele Milani. Simple homogenization model for the non-linear analysis of in-plane loaded masonry walls. *Computers & Structures*, 89(17-18):1586–1601, sep 2011. doi: 10.1016/j.compstruc.2011.05.004.

[66] Gabriele Milani, Paulo Lourenço, and Antonio Tralli. 3d homogenized limit analysis of masonry buildings under horizontal loads. *Engineering Structures*, 29(11):3134–3148, 2007.

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

[68] C Molins. Characterization of the mechanical behaviour of masonry. *Structural analysis of historical constructions I*, pages 86–122, 1997.

[69] Zi Mroz. On the description of anisotropic workhardening. *Journal of the Mechanics and Physics of Solids*, 15(3):163–175, 1967.

[70] FKG Odqvist and J Hult. Creep strength of metallic materials, 1962.

[71] Sergio Oller, Eduardo Car, and Jacob Lubliner. Definition of a general implicit orthotropic yield criterion. *Computer methods in applied mechanics and engineering*, 192(7):895–912, 2003.

[72] A.W. Page. The biaxial compressive strength of brick masonry. *Proceedings of the Institution of Civil Engineers*, 71(3):893–906, sep 1981. doi: 10.1680/iicep.1981.1825. URL http://dx.doi.org/10.1680/iicep.1981.1825.

[73] AW Page. The strength of brick masonry under biaxial tension-compression. *Int. J. Masonry Constr*, 3(1):26–31, 1983.

[74] E. Papa. A unilateral damage model for masonry based on a homogenisation procedure. *Mechanics of Cohesive-frictional Materials*, 1(4):349–366, 1996. ISSN 1099-1484. doi: 10.1002/(SICI)1099-1484(199610)1:4<349::AID-CFM18>3.0.CO;2-M. URL http://dx.doi.org/10.1002/(SICI)1099-1484(199610)1:4<349::AID-CFM18>3.0.CO;2-M.

[75] Luca Pelà. *Continuum damage model for nonlinear analysis of masonry structures*. PhD thesis, Universitat Politècnica de Catalunya, 2006.

[76] Luca Pelà, Miguel Cervera, and Pere Roca. An orthotropic damage model for the analysis of masonry structures, apr 2013.

[77] Jean-Victor Poncelet. *Mémoire sur la stabilité de revêtements et de leurs fondations*. Bachelier, 1840.

[78] A.M. Prior. Applications of implicit and explicit finite element techniques to metal forming. *Journal of Materials Processing Technology*, 45(1):649 – 656, 1994. ISSN 0924-0136. doi: http://dx.doi.org/10.1016/0924-0136(94)90413-8. URL http://www.sciencedirect.com/science/article/pii/0924013694904138.

[79] Yu N Rabotnov. On the equations of state for creep. *Progress in Applied Mechanics*, 12:307–315, 1963.

[80] Luís F Ramos. *Análise experimental e numérica de estruturas históricas de alvenaria*. PhD thesis, 2002.

[81] William John Macquorn Rankine. *On the general law of the transformation of energy*. 1853.

[82] N Rebelo, JC Nagtegaal, LM Taylor, and R Passman. Comparison of implicit and explicit finite element methods in the simulation of metal forming processes. In *ABAQUS Users Conf., Newport, RI*, 1992.

[83] Pere Roca, Miguel Cervera, Giuseppe Gariup, and Luca Pela'. Structural analysis of masonry historical constructions. classical and advanced approaches. *Arch Computat Methods Eng*, 17 (3):299–325, jul 2010. doi: 10.1007/s11831-010-9046-1.

[84] P Benson Shing, JL Noland, E Klamerus, and H Spaeh. Inelastic behavior of concrete masonry shear walls. *Journal of Structural Engineering*, 115(9):2204–2225, 1989.

[85] Jacek Skrzypek. Plasticity and creep: theory, examples and problems. 1993.

[86] J.S. Sun, K.H. Lee, and H.P. Lee. Comparison of implicit and explicit finite element methods for dynamic problems. *Journal of Materials Processing Technology*, 105(1–2):110 – 118, 2000. ISSN 0924-0136. doi: http://dx.doi.org/10.1016/S0924-0136(00)00580-X. URL http://www.sciencedirect.com/science/article/pii/S092401360000580X.

[87] D.J. Sutcliffe, H.S. Yu, and A.W. Page. Lower bound limit analysis of unreinforced masonry shear walls. *Computers & Structures*, 79(14):1295–1312, jun 2001. doi: 10.1016/s0045-7949(01)00024-4. URL http://dx.doi.org/10.1016/S0045-7949(01)00024-4.

[88] CA Syrmakezis and PG Asteris. Masonry failure criterion under biaxial stress state. *Journal of Materials in Civil Engineering*, 13(1):58–64, 2001.

[89] H Tresca. Memoir on the flow of solid bodies under strong pressure. *Comptes-rendus de l'académie des sciences*, 59:754–758, 1864.

[90] Ö Türkmen, A Vermeltfoort, and D Martens. Seismic retrofit system for single leaf masonry buildings in groningen. pages 2479–2487, jun 2016. doi: 10.1201/b21889-324.

[91] Stephen W Tsai and Edward M Wu. A general theory of strength for anisotropic materials. *Journal of composite materials*, 5(1):58–80, 1971.

[92] Rob Van der Pluijm. Shear behaviour of bed joints. 1993.

[93] Gideon PAG van Zijl. Modeling masonry shear-compression: Role of dilatancy highlighted. *Journal of engineering mechanics*, 130(11):1289–1296, 2004.

[94] Felix Y Yokel and S George Fattal. Failure hypothesis for masonry shear walls. *Journal of the Structural Division*, 102(3):515–532, 1976.

[95] Mao-hong Yu. Advances in strength theories for materials under complex stress state in the 20th century. *Applied Mechanics Reviews*, 55(3):169–218, 2002.

A

Determining the material properties for wall tests

In order to obtain the material properties for wall tests one of the ways to do it is to apply a least square fit method on the experimental data. This can be easily achieved with mathematical packages like Maple or Wolfram Mathematica. However, these software packages are proprietary and expensive. Therefore, it is more convenient to develop a fitting solution whose usage would not require buying expensive software that would not be used for any other purpose.

To this, a program composed of MS Excel spreadsheet and python, a high-level general-purpose programming language, was developed. The program functions so that the user has to input principle stress data obtained from biaxial tension/compression tests made on square wall specimens. Then the program rotates the stresses to Cauchy stresses. These stresses then are sorted depending on the regime (specified by user) and the data is transfered to python code where the fit is executed. After obtaining the fit parameters, the data is transfered back to the excel spreadsheet where the the accuracy of the fit is estimated and the results are displayed for the end user.

A.1. Input

For the input of the program a material data points are requested. They are provided by entering the data point name, the principle stresses σ_x and σ_y specified in MPa, and the rotation of the bed-joint θ specified in degrees. Each data point should be attributed either to be fit on Hill-like yield surface or Rankine-like. The Data points can be enabled or disabled on demand specifying either yes or no in the column titled “Included”. The example of the input is provided in figure A.1.

When the input is entered, the principle stresses are rotated to the material directions so that the new stresses are either perpendicular or parallel to the bed-joint. For the rotation the following formulas are used:

$$\sigma_x = \sigma_1 \cdot \cos^2\left(\frac{\theta}{180^\circ}\pi\right) + \sigma_2 \cdot \sin^2\left(\frac{\theta}{180^\circ}\pi\right) \tag{A.1}$$

$$\sigma_y = \sigma_1 \cdot \sin^2\left(\frac{\theta}{180^\circ}\pi\right) + \sigma_2 \cdot \cos^2\left(\frac{\theta}{180^\circ}\pi\right) \tag{A.2}$$

$$\tau_{xy} = (\sigma_1 - \sigma_2) \cdot \cos^2\left(\frac{\theta}{180^\circ}\pi\right) \cdot \sin^2\left(\frac{\theta}{180^\circ}\pi\right) \tag{A.3}$$

When the input is all entered, “Execute Fit Procedure” should be clicked. Then the excel will sort all of the input in two tables, for Rankine-like and Hill-like surfaces and pass the data to Python code for fitting.

Input									
Experimental data					Rotated stresses				Done!
#	Surface	σ_x [Mpa]	σ_y [Mpa]	θ [deg]	σ_x [Mpa]	σ_y [Mpa]	τ_{xy} [Mpa]	Included	Execute Fit Procedure
K1	Rankine	0.095311	-1.0389	22.5	-0.071	-0.85	0.401	Yes	
K2	Rankine	0.066566	-1.3313	22.5	-0.138	-1.111	0.494	Yes	
K3	Hill	0	-7.6351	0	0	-7.635	0	Yes	Material properties
K4	Hill	0	-1.601	90	-1.601	0	0	Yes	*Rankine*
K5	Hill	1	1	45	1	1.207	0	No	ftx=
K6	Rankine	0	-0.6346	45	-0.317	-0.317	0.317	Yes	fty=
K7	Rankine	0	-2.3996	22.5	-0.351	-2.048	0.848	Yes	alpha=
K8	Rankine	0	-0.254	67.5	-0.217	-0.037	0.09	Yes	*Hill*
K9	Rankine	0.250521	-2.5052	22.5	-0.153	-2.042	0.974	No	fcx=
K10	Hill	-2.01521	-6.4487	0	-2.015	-6.449	0	Yes	fcy=
K11	Hill	-1.57764	-4.8907	22.5	-2.063	-4.778	1.171	Yes	beta=
K12	Hill	-0.97047	-3.1055	45	-2.038	-2.239	1.068	Yes	gamma=
									S.dev =
									Legend
									13 - User input
									0.123 - Calculation

Figure A.1: Interface for user input.

A.2. Fitting

The fitting process is semi-automatic. Even though the fit itself is automatic, the outliers are not excluded automatically. After the fit is performed the results have to be inspected and if a larger error is present the outliers have to be disabled in the spreadsheet and the fit has to be run again.

The data fit itself is accomplished using `least_squares` function from `scipy.optimize` package.

A.3. Results

The display of the results is provided in two mediums. First the data points are plotted against the yield surfaces as depicted in figure A.2 using `mayavi` visualization package, then the fit data is exported back to excel where the material parameters and the quality of the fit is then displayed (see Figure A.3).

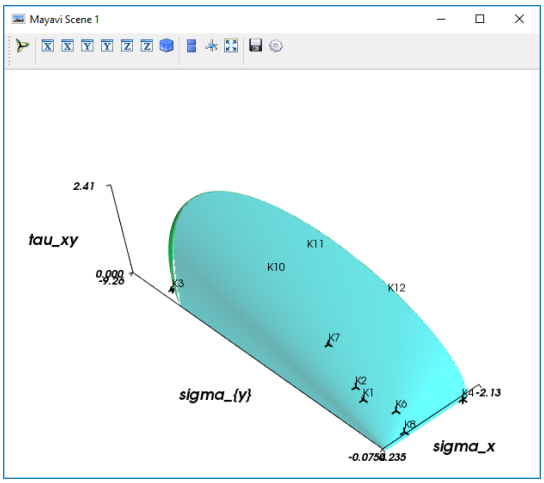


Figure A.2: Visualization of a fit.

Done!		Data for hill surface								Data for rankine surface							
		Fit stresses			Experimental stresses					Fit stresses			Experimental stresses				
Execute Fit Procedure		#	σx [Mpa]	σy [Mpa]	τxy [Mpa]	σx [Mpa]	σy [Mpa]	τxy [Mpa]	Ratio	#	σx [Mpa]	σy [Mpa]	τxy [Mpa]	σx [Mpa]	σy [Mpa]	τxy [Mpa]	Ratio
		K3	0.000	-7.505	0.000	0.000	-7.635	0.000	1.017	K1	-0.069	-0.825	0.389	-0.071	-0.850	0.401	1.030
		K4	-1.730	0.000	0.000	-1.601	0.000	0.000	0.926	K2	-0.142	-1.145	0.509	-0.138	-1.111	0.494	0.970
Material properties		K10	-2.097	-6.711	0.000	-2.015	-6.449	0.000	0.961	K6	-0.344	-0.344	0.344	-0.317	-0.317	0.317	0.922
Rankine		K11	-2.019	-4.675	1.146	-2.063	-4.778	1.171	1.022	K7	-0.344	-2.005	0.830	-0.351	-2.048	0.848	1.021
ftx=	0.24622	K12	-1.897	-2.085	0.994	-2.038	-2.239	1.068	1.074	K8	-0.337	-0.057	0.140	-0.217	-0.037	0.090	0.644
fty=	0.00000																
alpha=	1.71576																
Hill																	
fcx=	1.7297																
fcy=	7.5054																
beta=	-1.1707																
gamma=	1.0000																
S.dev =	± 12.08%																
Legend																	
13	- User input																
0.123	- Calculation																

Figure A.3: Output after running the fit procedure.

The results should be taken with caution as the fit can be easily be usable due to the wide scatter of the experimental results. in that case the parameters can be forced in to the logical boundaries using settings of the program (see Figure A.4).

Fit Bounds					Plot Bounds				Description
Name	Min	Max	Init	Description	Name	Lower	Upper	Description	Here are the settings specifying the plotting and fitting parameters.
ftx	0	inf	1	Tensile strengths in x direction	x	2	1.5	Plot baundaries in x (Ratio of ftx and fcx)	
fty	0	inf	1	Tensile strengths in y direction	y	2	1.5	Plot baundaries in y (Ratio of fty and fcy)	
alpha	1	inf	1	Alpha factor for Rankine surface	z	0	0.5	Plot baundaries in z (Ratio of max(fcx,fcy))	***Fit Bounds*** specify the range for the fitting parameters. Init is the initial guess, helps to speed up fitting for large data sets also helps to avoid local minima. Write *inf* for infinity. Note the units to match your input!
fcx	0	10	1	Compression strengths in x dir.	Plot Options				
fcy	0	10	1	Compression strengths in y dir.	Mesh density in x:		200		
beta	-inf	0	-1	Beta factor for Hill surface	y:		200	Mesh density of the plot in x,y,z	
gamma	1	inf	1	Gamma factor for Hill surface	z:		200		
Configuration									***Plot bound***s set the implicit scalar field boundaries. It is specified as a ratio between material strengths. Higher values might result to the coarser plot. Lower into trimming of the surface
Python	C:\Users\aukselis\App			Full path to python.exe					
Script	C:\Users\aukselis\Dro			Full path to backend python script					***Plot Options*** sets the density of the plotted mesh, set higher values if plot mesh is too coarse, set lower values if mesh generation takes to long, or plot viewer crashes
Options									
OP1	No			Keep temporary files					The ***configuration*** specifies the environment and files.
OP2	no			Command line window					***Options*** set misc. parameters and settings

Figure A.4: Settings for better fit and visualization control.

A.4. Code

A.4.1. Visual Basic for excel

```
Sub Fit()
    Dim rng As Range, cell As Range, i As Integer, j As Integer
    Dim ActSheet As Worksheet
    Dim SelRange As Range
    On Error GoTo errHandler
    Set ActSheet = ActiveSheet
    Set SelRange = Selection
    If Range("A3").Value = "#" Then
        Range("J2").Value = "Running..."
        Application.Wait (Now + TimeValue("0:00:01"))
        Application.ScreenUpdating = False
        Call SortSurf(Range("B4:B100"), i, j)
        Call ExportCsv(Range("M4:S" & 3 + i), "temp_Hill")
        Call ExportCsv(Range("U4:AA" & 3 + j), "temp_Rankine")
        Call ExportCsv(ThisWorkbook.Sheets("Settings").Range("B3:D9"),
          ↪ "temp_Bounds")
        Call ExportCsv(ThisWorkbook.Sheets("Settings").Range("G3:H5"),
          ↪ "temp_Plot_Bounds")
```

```
17          Call ExportCsv(ThisWorkbook.Sheets("Settings").Range("H7:H9"),
              ↪ "temp_Plot_density")
18          If ThisWorkbook.Sheets("Settings").Range("B15") = "Yes" Then
19              python_loc = ThisWorkbook.Sheets("Settings").Range("B11")
20          Else
21              python_loc =
                  ↪ Replace(ThisWorkbook.Sheets("Settings").Range("B11"),
                  ↪ "python.exe", "pythonw.exe")
22          End If
23
24          script_loc = ThisWorkbook.Sheets("Settings").Range("B12")
25          Location = Application.ActiveWorkbook.Path
26
27          RetVal = Shell(python_loc & " " & """" & script_loc & """" & " " &
              ↪ """" & Location & """", vbNormalFocus)
28          Range("J20") = RetVal
29
30          If IsItDone() Then
31              Call ReadOutput
32          End If
33          Call DeleteFile
34          Columns("M").AutoFit
35          Columns("U").AutoFit
36          Application.CutCopyMode = False
37          Range("J2").Value = "Done!"
38      Else
39          MsgBox "Code is being executed on the wrong sheet!"
40          Exit Sub
41      End If
42      Application.ScreenUpdating = True
43      ActSheet.Select
44      SelRange.Select
45  errHandler:
46      Application.ScreenUpdating = True
47  End Sub
48  Sub ExportCsv(mycels As Range, name As String)
49      filename = name & ".csv"
50
51      Open Application.ActiveWorkbook.Path & "\" & filename For Output As #1
52
53      Set myrng = mycels
54
55      For i = 1 To myrng.Rows.Count
56          For j = 1 To myrng.Columns.Count
57              lineText = IIf(j = 1, "", lineText & ",") & myrng.Cells(i, j)
58          Next j
59          Print #1, lineText
60      Next i
61      Close #1
62  End Sub
63
64  Sub SortSurf(rng As Range, i As Integer, j As Integer)
65      i = 0
66      j = 0
67      Range("M4:S999").ClearContents
68      Range("U4:AB999").ClearContents
```

```
    For Each cell In rng
        If cell.Value = "" Then
            Exit For
        ElseIf cell.Value = "Hill" Then
            If Range("I" & cell.Row).Value = "Yes" Then
            Range("C" & cell.Row & ":H" & cell.Row).Copy
            Range("N" & i + 4).PasteSpecial xlPasteValues
            Range("M" & i + 4) = Range("A" & cell.Row)
            i = i + 1
            End If
        Else
            If Range("I" & cell.Row).Value = "Yes" Then
            Range("C" & cell.Row & ":H" & cell.Row).Copy
            Range("V" & j + 4).PasteSpecial xlPasteValues
            Range("U" & j + 4) = Range("A" & cell.Row)
            j = j + 1
            End If
        End If
    Next cell
End Sub
Sub ReadOutput()
    PathF = Application.ActiveWorkbook.Path & "\temp_output.csv"
    Open PathF For Input As #1
    r_no = 0
    l_no = 0
    inc = 0
    Do Until EOF(1)
        Line Input #1, LineFromFile
        LineItems = Split(LineFromFile, ",")
        Range("J20") = inc
        inc = inc + 1
        If LineFromFile = "" Then
            r_no = r_no + 1
            l_no = 0
        Else
            If r_no = 0 Then
                Range("K8") = LineItems(0)
                Range("K9") = LineItems(1)
                Range("K10") = LineItems(2)
                Range("K12") = LineItems(3)
                Range("K13") = LineItems(4)
                Range("K14") = LineItems(5)
                Range("K15") = LineItems(6)
            ElseIf r_no = 2 Then
                Range("N" & 4 + l_no) = LineItems(0)
                Range("O" & 4 + l_no) = LineItems(1)
                Range("P" & 4 + l_no) = LineItems(2)
                Range("T" & 4 + l_no) = 1 / LineItems(3)
                l_no = l_no + 1
            ElseIf r_no = 1 Then
                Range("V" & 4 + l_no) = LineItems(0)
                Range("W" & 4 + l_no) = LineItems(1)
                Range("X" & 4 + l_no) = LineItems(2)
                Range("AB" & 4 + l_no) = 1 / LineItems(3)
                l_no = l_no + 1
            Else
```

```
125                 Exit Do
126             End If
127         End If
128     Loop
129     Close #1
130 End Sub
131
132 Function IsItDone() As Boolean
133     Count = 0
134     Do While True
135         If Application.Wait(Now + TimeValue("0:00:02")) Then
136             If Not Dir(Application.ActiveWorkbook.Path & "\" & "done",
                  ↪ vbDirectory) = vbNullString Then
137                 IsItDone = True
138                 Exit Do
139             Else
140                 IsItDone = False
141             End If
142         End If
143
144         Count = Count + 1
145         If Count > 5 Then
146             MsgBox "Did not receive an ansfer from python! Did the Python
                  ↪ script got stuck?"
147             Exit Do
148         End If
149     Loop
150 End Function
151
152 Sub DeleteFile()
153     Dim LRandomNumber As String
154     Dim Loc As String
155     Loc = Application.ActiveWorkbook.Path & "\"
156
157     If ThisWorkbook.Sheets("Settings").Range("B14") = "Yes" Then
158         LRandomNumber = Format(Now(), "yyyyMMdd_hhmmss")
159         FileCopy Loc & "temp_rankine.csv", Loc & LRandomNumber &
              ↪ "_rankine.csv"
160         FileCopy Loc & "temp_hill.csv", Loc & LRandomNumber & "_hill.csv"
161         FileCopy Loc & "temp_output.csv", Loc & LRandomNumber &
              ↪ "_output.csv"
162         FileCopy Loc & "temp_Bounds.csv", Loc & LRandomNumber &
              ↪ "_bounds.csv"
163         FileCopy Loc & "temp_Plot_Bounds.csv", Loc & LRandomNumber &
              ↪ "_Plot_Bounds.csv"
164         FileCopy Loc & "temp_Plot_density.csv", Loc & LRandomNumber &
              ↪ "_Plot_density.csv"
165     End If
166     Kill Loc & "temp_rankine.csv"
167     Kill Loc & "temp_hill.csv"
168     Kill Loc & "temp_output.csv"
169     Kill Loc & "temp_Bounds.csv"
170     Kill Loc & "temp_Plot_Bounds.csv"
171     Kill Loc & "temp_Plot_density.csv"
172     Kill Loc & "Done"
173 End Sub
```

A.4.2. Python code

```
from scipy import optimize
import numpy as np
from mayavi import mlab
import sys
import csv
import time

def process_csv(file_loc):
    try:
        with open(file_loc + '\\temp_rankine.csv', 'rb') as fl:
            reader = csv.reader(fl)
            data1 = list(reader)
        with open(file_loc + '\\temp_hill.csv', 'rb') as fl:
            reader = csv.reader(fl)
            data2 = list(reader)
        with open(file_loc + '\\temp_Bounds.csv', 'rb') as fl:
            reader = csv.reader(fl)
            data3 = list(reader)
        with open(file_loc + '\\temp_Plot_Bounds.csv', 'rb') as fl:
            reader = csv.reader(fl)
            data4 = list(reader)
        with open(file_loc + '\\temp_Plot_density.csv', 'rb') as fl:
            reader = csv.reader(fl)
            data5 = list(reader)
    except:
        print "ERROR! Could not find the files in:"
        print file_loc
    names1 = []
    names2 = []

    for row in data1:
        names1.append(row[0])
        del row[0:4]
    for row in data2:
        names2.append(row[0])
        del row[0:4]
    data1 = np.array(data1,dtype=float)
    data2 = np.array(data2,dtype=float)
    data3 = np.array(data3,dtype=float)
    data4 = np.array(data4, dtype=float)
    data5 = np.array(data5, dtype=float)
    return [names1,names2,data1,data2,data3,data4,data5]

def rankine_func(A, B, C, x, y, z):
    return (x - A + y - B) / 2 + np.sqrt(((x - A - y + B) / 2) ** 2 + C * z
      ↪ ** 2)

def hill_func(D, E, F, G, x, y, z):
    return np.sqrt(x*((F*y)/2 + (E*x)/D) + y*((F*x)/2 + (D*y)/E) + G*z**2)
      ↪ - np.sqrt(D*E)

def residuals_r(coeff, data):
```

```
    # Function that returns the squared loss.
    # We want the function to choose A, B, C such that all values are close
      ↪ to zero
    A, B, C = coeff
    x, y, z = data.T
    # The function we care about
    objective = rankine_func(A, B, C, x, y, z)
    losses =   (objective - 0)
    return losses

def residuals_h(coeff, data):
    # Function that returns the squared loss.
    # We want the function to choose A, B, C such that all values are close
      ↪ to zero
    A, B, C, D = coeff
    x, y, z = data.T
    # The function we care about
    objective = hill_func(A, B, C, D,x, y, z)
    losses =   (objective - 0)
    return losses

def Surf1 (x,y,z):
    F1 = rankine_func(A, B, C,x,y,z)
    F2 = hill_func(D, E, F, G, x, y, z)
    F1[F2>0] = None
    return F1

def Surf2 (x,y,z):
    F1 = rankine_func(A, B, C,x,y,z)
    F2 = hill_func(D, E, F, G, x, y, z)
    F2[F1>0] = None
    return F2

def write_csv (file_loc, data):
    with open(file_loc + '\\temp_output.csv', 'ab') as fp:
        a = csv.writer(fp, delimiter=',')
        try:
            a.writerows(data)
        except:
            a.writerow(data)
        a.writerow('')




def intersection_r (A,B,C,x0,y0,z0):
    t = (-A * y0 - B * x0 + np.sqrt(A ** 2 * y0 ** 2 + (4 * C * z0 ** 2 - 2
      ↪ * x0 * y0) * B * A + B ** 2 * x0 ** 2)) / (
    2 * C * z0 ** 2 - 2 * x0 * y0)
    return [x0*t,y0*t,z0*t]

def intersection_h (D, E, F, G, x0,y0,z0):
    t = D*E/np.sqrt(D*E*F*x0*y0+D*E*G*z0**2+D**2*y0**2+E**2*x0**2)
    return [x0*t,y0*t,z0*t]

if sys.argv[-1]==sys.argv[0]:
```

```
104     location = r"C:\Users\aukselis\Desktop\Test
          ↪ worksheat".replace("\\","//")
105 else:
106     location = sys.argv[-1]
107
108 data = process_csv(location)
109
110
111 names_r = data[0]
112 names_h = data[1]
113 print "------------INPUT------------"
114 print "Defined names"
115 print "Rankine:",
116 print names_r
117 print "Hill:",
118 print names_h
119
120 print ""
121 data_r = data[2]
122 data_h = data[3]
123 print "Defined data:"
124 print "Rankine:",
125 print data_r
126 print "Hill:",
127 print data_h
128 print ""
129 bounds = data[4]
130 print "Defined bounds:"
131 print bounds
132
133 plot_i = [data[5],data[6]]
134
135 coeff_r0 = bounds.T[2][0:3]
136 coeff_r = coeff_r0
137 coeff_h0 = bounds.T[2][3:]
138 coeff_h = coeff_h0
139
140
141
142
143 print "\n\n------------OUTPUT------------"
144 rankine = optimize.least_squares(residuals_r, coeff_r0, args=(data_r,),
145                                                  bounds=([bounds[0][0],
                                                    ↪ bounds[1][0],
                                                    ↪ bounds[2][0]],
146                                                         [bounds[0][1],
                                                          ↪ bounds[1][1],
                                                          ↪ bounds[2][1]]))
147 ftx,fty,alpha = rankine.x
148 print('Rankine Surface\n -- ftx = {0} \n -- fty = {1} \n -- alpha = {2}
      ↪ '.format(ftx,fty,alpha))
149
150 hill = optimize.least_squares(residuals_h, coeff_h0, args=(data_h,),
151                                    bounds=([bounds[3][0], bounds[4][0],
                                        ↪ bounds[5][0],bounds[6][0]],
```

```
                                                [bounds[3][1], bounds[4][1],
                                                 bounds[5][1],bounds[6][1]]))
fcx,fcy,beta,gamma = hill.x
print('Hill Surface\n -- fcx = {0} \n -- fcy = {1} \n -- beta = {2}\n --
    gamma = {3} '.format(fcx,fcy,beta,gamma))


A, B, C = rankine.x
D, E, F, G = hill.x

write_csv(location,[A,B,C,D,E,F,G])

x0,y0,z0 = data_r.T
xyz1 = np.array(intersection_r(A,B,C,x0,y0,z0))
ratio =
    np.array([np.sqrt(xyz1[0]**2+xyz1[1]**2+xyz1[2]**2)/np.sqrt(x0**2+y0**2+z0**2)])
write_csv(location,np.append(xyz1,ratio,axis=0).T)

x0,y0,z0 = data_h.T
xyz1 = np.array(intersection_h(D, E, F, G, x0,y0,z0))
ratio =
    np.array([np.sqrt(xyz1[0]**2+xyz1[1]**2+xyz1[2]**2)/np.sqrt(x0**2+y0**2+z0**2)])
write_csv(location,np.append(xyz1,ratio,axis=0).T)

open(file_loc + 'done', 'a').close()

minx = -D * plot_i[0][0][0]
maxx = A * plot_i[0][0][1]
spacex = int(plot_i[1][0][0])*1j

miny = -E * plot_i[0][1][0]
maxy = B * plot_i[0][1][1]
spacey =int(plot_i[1][1][0])*1j

minz = max(D,E) * plot_i[0][2][0]
maxz = max(D,E) * plot_i[0][2][1]
spacez =int(plot_i[1][2][0])*1j


x, y, z = np.mgrid[minx:maxx:spacex, miny:maxy:spacey, minz:maxz:spacez]

app = np.append(data_r,data_h,axis=0)

x1,y1,z1 = app.T

names = names_r+names_h


f= mlab.figure(fgcolor=(0., 0., 0.), bgcolor=(1, 1, 1))

mlab.contour3d(x,y,z,Surf1, contours = [0],transparent=True)
mlab.contour3d(x,y,z,Surf2, contours = [0], transparent=True)
mlab.axes(xlabel=r'sigma_x', ylabel=r'sigma_{y}', zlabel=r'tau_xy')

mlab.points3d(x1,y1,z1,mode='axes',scale_factor=0.1)

```

```
204 text=[]
205 for i in range(len(x1)):
206     text.append(mlab.text(x1[i],y1[i],names[i],z=z1[i]))
207     text[i].actor.text_scale_mode = 'none'
208
209 mlab.show()
```

B

Formating the output database

In order to obtain more readable output and compute variables that are not available as output request in Abaqus, The output database has to be formated using Python scripting interface. For this specific case a Python script was written in order to replace the *SDV#* to a more legible output to custom named variables and vectors that can be later be plotted as a vector plots. Furthermore, the strain increment variable was also added, as it was necessary for the dynamic cyclic test to see the crack patterns.

```
from abaqusConstants import *
from odbAccess import *

# ********************************************************************
odbPath = "C:\\Some\\Path\\to\\an\\Output_database.odb" # path to output database
# ********************************************************************
# Do you want to copy non-SDV variables to the new step?
copy = False
# List of sdv's (name, tuple(no),description,type,tuple(validInvariants)):
l_sdv = [["Yt",(1,2),"Tensile yield strengths in material directions",VECTOR,(MAGNITUDE,)],
         ["Yc",(4,5),"Compressive yield strengths in material directions",VECTOR,(MAGNITUDE,)],
         ["rYt",(10,11),"Ratio of remaining tensile yield strengths in material directions",VECTOR,(MAGNITUDE,)],
         ["rYc",(12,),"Ratio of remaining compressive yield strengths",SCALAR,()],
         ["dVol",(17,),"Volume change",SCALAR,()],
         ["rEeq",(18,),"Equivalent Rankine plastic strain",SCALAR,()],
         ["hEeq",(19,),"Equivalent Hill plastic strain",SCALAR,()],
         ["tE",(20,21,22),"Total strain in material directions",VECTOR,(MAGNITUDE,)],
         ["tEm",(23,24,25),"Total maximum strain in material directions",VECTOR,(MAGNITUDE,)],
         ["dam",(27,28),"Tension continiuoty parameters in material directions",VECTOR,(MAGNITUDE,)],
         ["ElDel",(29,),"Element deletion paramerer",SCALAR,()],
         ["Sdamp",(30,31,32,33,34,35),"Damping stress increment",TENSOR_3D_FULL,(PRESS,MAX_PRINCIPAL,MID_PRINCIPAL,MIN_PRINCIPAL)]]
# ********************************************************************
# Open Odb
odb = session.openOdb(name=odbPath,readOnly=FALSE)
```

```
25 # Create an object with all steps
26 stepRepository = odb.steps
27 # Get the names of all steps
28 allSteps = stepRepository.keys()
29
30
31 # Iterate all steps
32 for i in range(len(allSteps)):
33     pr.enable()
34     # Assign a step
35     step = stepRepository[allSteps[i]]
36     # Crate a list containing all frame numbers
37     allFrames = range(len(step.frames))
38     # Get the total time of the step
39     StepTime=step.frames[-1].frameValue
40     # Create a new step to store all of the custom field entries
41     newStep = odb.Step(name=allSteps[i]+'_c_out', description='Step for new
         ↪ fields', domain=TIME, timePeriod=StepTime)
42     # Create an object with all frames (This is recommended by abaqus
         ↪ manual to improve the speed of the script)
43     frameRepository = step.frames
44     # Get a list of all of the field variables available in the step
45     fields = frameRepository[1].fieldOutputs.keys()
46
47     #Iterate all frames in the step
48     for j in allFrames:
49         # Try assigning previous frame if it does not work. Previous frame
             ↪ is a new frame. This is used for the calculation of strain
             ↪ increment.
50         try:
51             pframe = frame
52         except:
53             pframe = frameRepository[j]
54         # Assign a frame to a new frame
55         frame = frameRepository[j]
56         # Create a frame for the custom field variables in the new step
             ↪ created before this loop
57         newFrame = newStep.Frame(frameId=j, frameValue=frame.frameValue,
             ↪ description='Increment: '+str(j) + ' Time: ' +
             ↪ str(frame.frameValue) )
58         # Define or clean sdv list
59         SDV=[]
60
61         # Iterate trough the available field variables
62         for field in fields:
63             # Assign field output to an object
64             current = frame.fieldOutputs[field]
65             # If field output is not an SDV
66             if 'SDV' not in field:
67                 # If copying existing variables is allowed
68                 if copy:
69                     # Copy existing variables to a new step
70                     newField = newFrame.FieldOutput(name=current.name,
                         ↪ description=current.description, field=current)
71                 # If the field output variable is strain calculate strain
                     ↪ increment
```

```
72                  if 'LE' in field:
73                      previous = pframe.fieldOutputs[field]
74                      comp = current - previous
75                      newField = newFrame.FieldOutput(name='dLE',
                          ↪ description='Logarithmic strain increment',
                          ↪ field=comp)
76              # Otherwise collect the SDV objects in to a list
77              else:
78                  SDV.append(frame.fieldOutputs[field])
79          # Define or clean field variable list
80          l_field = []
81          # Define the structure of the field variable list
82          for output in l_sdv:
83              l_field.append([output[0],newFrame.FieldOutput(name=output[0],
                  ↪ descrip-
                  ↪ tion=output[2],type=output[3],validInvariants=output[4]),output[1],[[[]]]])
84          # Iterate trough values of sdv to separate different instances and
              ↪ construct a list of elements
85          for k in range(len(SDV[0].values)):
86              # Element numbers
87              labl = SDV[0].values[k].elementLabel
88              # Element belongs to an instance
89              inst = SDV[0].values[k].instance
90             # For each field output in list of fields construct proper tuples
91              for field in l_field:
92                  if len(field[3][0]) == 1:
93                      field[3][0]=[inst,[],[]]
94                  elif inst != field[3][-1][0]:
95                      field[3].append([inst,[],[]])
96
97                  field[3][-1][1].append(labl)
98                  temp = []
99                  for tup in field[2]:
100                     temp.append(SDV[tup-1].values[k].data)
101                 if len(temp) == 2:
102                     temp.append(0)
103                 field[3][-1][2].append(tuple(temp))
104
105         # For each field output in list of fields
106         for field in l_field:
107             # Iterate trough all instances in each SDV
108             for inst in field[3]:
109                 # Add data of an instance to a field
110                 field[1].addData(position=INTEGRATION_POINT,
                      ↪ instance=inst[0], labels=tuple(inst[1]),
                      ↪ data=tuple(inst[2]))
111         # Print progress to screen
112         print >> sys.__stdout__, '', allSteps[i], ' ', allFrames[j]
113         pr.disable()
114
115 # Save the odb
116 odb.save()
117 # Close the odb
118 odb.close()
```

C

Definition of input

```
*Depvar
      28,
*User Material, constants=YY
     state,       Ex,       Ey,       Ez,     Nuxy,    bdamp,     tpsi,      Gxy
       Gyz,      Gzx,      Ftx,      Fty,    alpha,      Fcx,      Fcy,     beta
     gamma,      tss,     dmax,     Gfxt,     Gfyt,     Fucx,     Fucy,    ecplu
        XX,    ncomp,                {comp_epl},                {comp_stress_r}
```

State	–	Definition of analysis state: 1 – Orthotropic elastic 2 – Orthotropic damaged plasticity
Ex	–	Stiffness modulus in material direction X
Ey	–	Stiffness modulus in material direction Y
Ez	–	Stiffness modulus in material direction Z
Nuxy	–	Poisson's ratio ν_{xy}
bdamp	–	stiffness proportional Rayleigh damping coefficient
tpsi	–	Tangent of the dilation angle
Gxy	–	Shear stiffness modulus in material plane xy
Gyz	–	Shear stiffness modulus in material plane yz
Gzx	–	Shear stiffness modulus in material plane zx
Ftx	–	Tensile yield strengths in direction X
Fty	–	Tensile yield strengths in direction Y
Alpha	–	Yield shape parameter [common from 1.0 to 2.0]
Fcx	–	Compressive yield strengths in direction X
Fcy	–	Compressive yield strengths in direction Y
Beta	–	Yield shape parameter [common from -1.1 to -0.9]
Gamma	–	Yield shape parameter [common from 1.0 to 10.0]
tss	–	Type of shear softening: 0 – weakest direction, 1 – isotropic
dmax	–	Maximal volume change for the element in the analysis
Gfxt	–	Fracture energy in X direction
Gfyt	–	Fracture energy in Y direction
Fuc	–	Ultimate yield strengths for compression, must be consistent with {comp_stress_r}
ecplu	–	Plastic strain at $f_{u,c}$, must be consistent with {comp_epl}
Ncomp	–	Size of input softening table describing compressive behavior
{comp_epl}	–	Plastic strain part of compression table

{comp_stress_r} – Yield strength part of compression table
XX – Unused material property position. Just set 0.0
YY – Number of material properties (26+ncomp*2)

D

Definition of output

SDV01 – Tensile yield strengths X direction
SDV02 – Tensile yield strengths Y direction
SDV03 – Compressive yield strengths X direction
SDV04 – Compressive yield strengths Y direction
SDV05 – Ratio of remaining tensile material strengths in x direction
SDV06 – Ratio of remaining tensile material strengths in y direction
SDV07 – Ratio of remaining compressive strengths in the material
SDV08 – Compressive plastic strain in X direction
SDV09 – Compressive plastic strain in X direction
SDV10 – Volumetric change in an element
SDV11 – Equivalent Rankine plastic strain
SDV12 – Equivalent Hills plastic strain
SDV13 – Total strain in X direction
SDV14 – Total strain in Y direction
SDV15 – Total strain in Z direction
SDV16 – Max strain in tension in X direction
SDV17 – Max strain in tension in Y direction
SDV18 – Max strain in tension in Z direction
SDV19 – Damage parameter $\left[\sqrt{d_x^2 + d_y^2}\right]$
SDV20 – Damage parameter for X direction
SDV21 – Damage parameter for Y direction
SDV22 – Element deletion parameter
0 – Not deleted
1 – Deleted
SDV23 – Storage for damping stress – xx
SDV24 – Storage for damping stress – yy
SDV25 – Storage for damping stress – zz
SDV26 – Storage for damping stress – xy
SDV27 – Storage for damping stress – yz
SDV28 – Storage for damping stress – zx

E

Fortran code

The modules `def_par_math`, `def_par_state`, `def_par_dir` containing parameters that define memory positions of variables in arrays and mathematical constants were removed due to request from *Zonneveld Ingenieurs B.V.*

```
      module utility_routines
      public :: ut_elastic, ut_int_energy, ut_asgn_fstate,
     n    ut_asgn_props, ut_assem_K_mtx, ut_asgn_tstate,
     n    ut_vuhard
      contains

      pure subroutine ut_elastic(Sn, So, de, D)
      use def_par_dir
      include 'vaba_param.inc'
      Real(8), intent(out)::Sn(:)
      Real(8), intent(in)::So(:), de(:), D(:)

      Sn(xx)= So(xx) + D(K11)*de(xx) + D(K12)*de(yy) + D(K13)*de(zz)
      Sn(yy)= So(yy) + D(K21)*de(xx) + D(K22)*de(yy) + D(K23)*de(zz)
      Sn(zz)= So(zz) + D(K31)*de(xx) + D(K32)*de(yy) + D(K33)*de(zz)
      Sn(xy)= So(xy) + D(K44)*de(xy)
      Sn(yz)= So(yz) + D(K55)*de(yz)
      Sn(zx)= So(zx) + D(K66)*de(zx)

      return
      end

      pure subroutine ut_int_energy(eIntNew, eIntOld, Sn, So, de, rho)
      use def_par_dir
      use def_par_math
      include 'vaba_param.inc'

      Real(8), intent(out):: eIntNew
      Real(8), intent(in):: rho, eIntOld
      Real(8), intent(in):: Sn(:), So(:), de(:)

      STRESS_POWER = HALF * (
           n(So(xx) + Sn(xx))*de(xx) + (So(yy) + Sn(yy))*de(yy)
     n    +      (So(zz) + Sn(zz))*de(zz))+ (So(xy) + Sn(xy))*de(xy)
     n    +      (So(yz) + Sn(yz))*de(yz) + (So(zx) + Sn(zx))*de(zx)
```

```
36       eIntNew = eIntOld  + STRESS_POWER/rho
37
38       return
39       end
40
41       subroutine ut_ine_energy(eIneNew, eIneOld, Sn, So, depl,depl1,rho)
42       use def_par_dir
43       use def_par_math
44       include 'vaba_param.inc'
45
46       dimension Sn(:), So(:), depl(:),plastic_work(size(So))
47       real*8    rho, eIntNew, eIntOld,depl1(:)
48
49       plastic_work = (Sn+So)*half*(depl+depl1)
50       eIneNew = eIneOld  + sum(plastic_work)/rho
51
52       return
53       end
54
55       pure subroutine ut_asgn_fstate(yield, epl, te, tem, props, state,
56      n    dmg,eeqpl,ryield,dV)
57       use def_par_dir
58       use def_par_props
59       use def_par_state
60       use def_par_math
61       include 'vaba_param.inc'
62
63       real(8), intent(out):: yield(:), epl(:), te(:), tem(:), dmg(:),
64      n    eeqpl, ryield, dV
65       real(8), intent(in)::  props(:), state(:)
66
67
68       if (state(sYxt).eq.zero) then
69           yield(xxt:yyt) = props(pYxt:pYyt)
70           yield(xxc:yyc) = props(pYxc:pYyc)
71           dmg(xxt:yyt)   = one
72           dV = zero
73       else
74           yield(xxt:yyt) = State(sYxt:sYyt)
75           yield(xxc:yyc) = State(sYxc:sYyc)
76
77           ryield         = State(sPc)
78
79           epl(xxt:yyt)   = State(sepxt:sepyt)
80           epl(xxc:yyc)   = State(sepxc:sepyc)
81
82           te(xx:zz)      = State(stex:stez)
83           dV             = State(dvol)
84           tem(xxt:zzt)   = State(semxt:semzt)
85           dmg(xxt:yyt)   = State(dmgxt:dmgyt)
86           eeqpl          = State(seeqp)
87       end if
88       return
89       end
90
91
```

```
92       subroutine ut_asgn_props(E, xNu, G, Gf, props,ctable)
93       use def_par_dir
94       use def_par_props
95       include 'vaba_param.inc'
96
97       dimension E(:), xNu(xy:), G(xy:), Gf(:), props(:), ctable(:,:)
98
99       E(xx:zz)    = props(Ex:Ez)
100      xNu(xy:zx)  = props(Nuxy:Nuzx)
101      G(xy:zx)    = props(Gxy:Gzx)
102
103      Gf(xxt:yyt) = Props(Gfxt:Gfyt)
104
105      ctable(1,:) = Props(ncomp+1:Props(ncomp)+ncomp)
106      ctable(2,:) = Props(Props(ncomp)+ncomp+1:Props(ncomp)+ncomp*2)
107
108      return
109      end
110
111
112      subroutine ut_assem_K_mtx(D, E, xNu, G)
113      use def_par_dir
114      use def_par_math
115      include 'vaba_param.inc'
116
117      dimension D(:), E(:), xNu(xy:), G(xy:)
118
119      xNu(yx) = xNu(xy)*E(yy)/E(xx)
120      Delta = one/(1-xNu(yx)*xNu(xy))
121
122      D(K11) = E(xx)*Delta
123      D(K22) = E(yy)*Delta
124      D(K33) = E(zz)
125
126      D(K12) = E(xx)*Delta*xNu(yx)
127      D(K13) = zero
128      D(K32) = zero
129
130      D(K44:K66) = two * G(xy:zx)
131
132      return
133      end
134
135
136      subroutine ut_asgn_tstate(state, yield, epl, te, tem, dmg, eeqpl,
137     n     ryield,reeqpl,props,dV)
138      use def_par_dir
139      use def_par_props
140      use def_par_state
141      include 'vaba_param.inc'
142
143      dimension yield(:), epl(:), te(:), tem(:), state(:), dmg(:),
144     n     props(:)
145
146      State(sYxt:sYyt)    = yield(xxt:yyt)
147      State(sYxc:sYyc)    = yield(xxc:yyc)
```

```
148       State(sYxy)         = yield(xy)
149
150       State(sepxt:sepyt)  = epl(xxt:yyt)
151       State(sepxc:sepyc)  = epl(xxc:yyc)
152       State(seeqp)        = eeqpl
153       State(stex:stey)    = te(xxt:yyt)
154
155       State(seqr)         = reeqpl
156
157       State(sPtx:sPty)    = yield(xxt:yyt)/props(pYxt:pYyt)
158       State(sPc)          = ryield
159       State(dvol)         = dV
160       State(semxt:semzt)  = tem(xxt:zzt)
161       State(sdmg)  = (dmg(xx)*dmg(yy))**(1.0/2.0)
162       State(dmgxt:dmgyt)  = dmg(xxt:yyt)
163       return
164       END
165
166
167       subroutine sumepl(So, yield, epl, depl, eeqpl,reeqpl, syield_r,
168      n     syield_h,props)
169       use def_par_dir
170       use def_par_math
171       use def_par_props
172       include 'vaba_param.inc'
173       dimension epl(:), depl(:),tdepl(3),So(:),yield(:),props(:)
174
175       if (syield_r.ge.zero) then
176           if (So(xx).lt.zero.and.So(yy).lt.zero) then
177               if (props(pSsc).eq.one) then
178                   epl(xxt) = epl(xxt) +  reeqpl
179                   epl(yyt) = epl(yyt) +  reeqpl
180               else
181                   if (yield(xxt).lt.yield(yyt)) then
182                       epl(xxt) = epl(xxt) + reeqpl
183                   else
184                       epl(yyt) = epl(yyt) + reeqpl
185                   end if
186               end if
187           else
188               if (props(pSsc).eq.one) then
189                   epl(xxt) = epl(xxt) + abs(depl(xy))
190                   epl(yyt) = epl(yyt) + abs(depl(xy))
191               else
192                   if (yield(xxt).lt.yield(yyt)) then
193                       epl(xxt) = epl(xxt) + abs(depl(xy))
194                   else
195                       epl(yyt) = epl(yyt) + abs(depl(xy))
196                   end if
197               end if
198           end if
199       end if
200       return
201       end
202
203       subroutine generate_dump(array)
```

```
204       include 'vaba_param.inc'
205       dimension array(:)
206       Logical OK
207       INQUIRE( UNIT=1, OPENED=OK )
208       IF ( OK ) Then
209           write(1,"(9999(G12.5,:,','))") array
210       else
211           open(unit = 1, file = "D:\\Temp\\Dump1.txt")
212       end if
213       return
214       end
215       subroutine sume(te,tem,de,So,syield_h,depl)
216       use def_par_dir
217       use def_par_math
218       include 'vaba_param.inc'
219       dimension te(:), de(:), tem(:),So(:),depl(:)
220
221       where (So(xx:yy)>zero)
222           te(xx:yy)= te(xx:yy) + de(xx:yy)
223       endwhere
224
225       if (syield_h .ge. zero) then
226           where (So(xx:yy)>zero)
227               te(xx:yy) = te(xx:yy) - de(xx:yy)
228           endwhere
229       end if
230       tem = max(te,tem)
231       return
232       end
233
234       subroutine ut_dev_stress(So, Sp, dS)
235       use def_par_dir
236       use def_par_math
237       include 'vaba_param.inc'
238       dimension So(:), Sp(:), dS(:)
239       dS = Sp - So
240       return
241       end
242
243       subroutine ut_vuhard(syield, hard, eqplas, table, nvalue)
244       include 'vaba_param.inc'
245       dimension table(2, nvalue)
246       parameter(zero=0.d0)
247
248       syield=table(2, nvalue)
249       hard=zero
250
251       if(nvalue.gt.1) then
252           do k1=1, nvalue-1
253               eqpl1=table(1,k1+1)
254               if(eqplas.lt.eqpl1) then
255                   eqpl0=table(1, k1)
256                   deqpl=eqpl1-eqpl0
257                   syiel0=table(2, k1)
258                   syiel1=table(2, k1+1)
259                   dsyiel=syiel1-syiel0
```

```
260                   hard=dsyiel/deqpl
261                   syield=syiel0+(eqplas-eqpl0)*hard
262                   goto 10
263                end if
264             end do
265 10          continue
266         end if
267         return
268         end
269
270
271         pure subroutine stable_time(D, beta, factor, cL, rho)
272         use def_par_dir
273         use def_par_math
274         include 'vaba_param.inc'
275         Real(8), intent(out):: factor
276         Real(8), intent(in) :: beta, cL, rho
277         Real(8), intent(in) :: D(:)
278
279         dimension omega(3)
280
281         omega(1) = Sqrt((D(K11)+D(K22)+sqrt(D(K11)**2-2*D(K11)*D(K22)
282        n          + D(K12)**2*4+D(K22)**2))/(rho*cL**2))
283         omega(2) = Sqrt((D(K11)+D(K22)-sqrt(D(K11)**2-2*D(K11)*D(K22)
284        n          + D(K12)**2*4+D(K22)**2))/(rho*cL**2))
285         omega(3) = Sqrt(2*D(K33)/(rho*cL**2))
286
287         omega_max = max(omega(1),omega(2),omega(3))
288
289         factor = sqrt(one+((half)*beta*omega_max)**2)-(half)*beta
290        n       * omega_max
291         factor = (one/factor)**2
292         return
293         end
294         endmodule
295
296               module sb_other
297         public damage,assem_dmg_K,damping
298         contains
299
300         subroutine assem_dmg_K(dmg, Dd, D, So, tem, E,yield,epl,props)
301         use def_par_dir
302         use def_par_math
303         include 'vaba_param.inc'
304
305         integer*2 status
306         dimension dmg(:), Dd(:), D(:), So(:), tem(:), E(:), yield(:)
307         dimension status(size(dmg)), epl(:), props(:)
308         status(xx:zz) = zero
309         if (So(xx) .gt. zero) status(xx) = 1
310         if (So(yy) .gt. zero) status(yy) = 1
311
312         Dd(K11:K22) = dmg(xx:yy)**status(xx:yy)*D(K11:K22)
313         Dd(K33) = D(K33)
314
315         Dd(K12) =floor(dmg(yy)**status(yy)*dmg(xx)**status(xx))*D(K12)
```

```
316       Dd(K13) =D(K13)
317       Dd(K32) =D(K32)
318
319       Dd(K44) = sqrt(dmg(yy)**status(yy)*dmg(xx)**status(xx))*D(K44)
320       Dd(K55) = D(K55)
321       Dd(K66) = D(K66)
322
323       E(xx:yy) = dmg(xx:yy)**status(xx:yy) * E(xx:yy)
324       return
325       end
326
327       subroutine damping(So, Sn, dt, Sdamo, beta)
328       use def_par_dir
329       use def_par_props
330       use def_par_math
331       include 'vaba_param.inc'
332       dimension So(:), Sn(:), Sdamo(:), Sdamn(size(Sdamo))
333
334       b = beta/dt
335
336       Sdamn = b * (Sn-So)
337       Sn = Sn + Sdamn
338       Sdamo = Sdamn
339
340       return
341       end
342
343       subroutine damage(dmg, Dd, D, Gf, E,yield,depl,props,cL,
344      nte,tem)
345       use def_par_dir
346       use def_par_math
347       use def_par_props
348       include 'vaba_param.inc'
349
350       parameter (ratio_t=1.d0/100.d0, ratio_c=1.d0/100.d0)
351       integer*2 status
352       dimension dmg(:), Dd(:), D(:), Gf(:), E(:), yield(:),te(:),tem(:)
353       dimension status(size(dmg)), depl(:), props(:)
354       status = 0
355       protect = zero
356
357       dmg(xxt:yyt) = (yield(xxt:yyt)/max(tiny(zero),tem(xxt:yyt)))
358      n             / D(K11:K22)
359
360       dmg(xxt:yyt) = min(one,max(tiny(one),dmg(xxt:yyt)))
361
362       return
363       end
364       endmodule
365
366       module sb_yield_surfaces
367       public Yield_Rankine, Yield_Hills, hardening
368       contains
369
370       subroutine Yield_Rankine(syield, depl, dSpl, Sp,yield, Dd,
371      n                         cL, props, Gf, E, de, dirc,D,reeqpl,
```

```
372      n                                                    sepr,Stn,dV)
373       use def_par_math
374       use def_par_dir
375       use def_par_props
376       use utility_routines
377       include 'vaba_param.inc'
378       integer   dirc
379       logical  stat
380       dimension depl(:), Sp(:), yield(:), Dd(:), props(:), Gf(:),
381      nE(:), dSpl(:), de(:), D(:),Stn(:)
382       dimension a(size(Sp)),hard(size(Gf)),b(size(Sp))
383       parameter (power = 10)
384
385       alpha = props(pYat)
386
387       phi   = props(pYpt)
388       zmax  = props(mdil)
389
390       temp = Sp(xy)
391       if (phi.le.zero) then
392           factor = zero
393           alpha1  = one
394       else
395
396           if (zmax.eq.0) then
397               factor = one
398               alpha1 = one/(phi**2)
399           elseif (dV.ge.zmax) then
400                   factor = zero
401                   alpha1  = one
402           else
403               factor = one
404               alpha1 = one/(phi**2)
405           end if
406       end if
407
408       Stn(3) = alpha1
409       Stn(6) = factor
410
411       if ((Sp(xx).ge.yield(xxt).OR.Sp(yy).ge.yield(yyt))
412      n   .AND.factor.eq.zero) then
413           Sp(xy) = zero
414           stat = .true.
415       end if
416
417       syield1=sqrt((half*(Sp(xx)-yield(xxt)-Sp(yy)+yield(yyt)))**2
418      n+alpha*Sp(xy)**2 )
419       syield = half*(Sp(xx)-yield(xxt)+Sp(yy)-yield(yyt))+syield1
420
421       if (syield.gt.zero) then
422
423           if (syield1.ne.zero) then
424           a(xx) = half+half**2*(Sp(xx)-yield(xxt)-Sp(yy)+yield(yyt))
425      n    / syield1
426           a(yy) = half+half**2*(-Sp(xx)+yield(xxt)+Sp(yy)-yield(yyt))
427      n    / syield1
```

```
428           a(xy) = two * alpha*Sp(xy)/(two*syield1)
429
430           syield2=sqrt((half*(factor*(Sp(xx)-Sp(yy))-yield(xxt)
431       n    +yield(yyt)))**2+alpha1*Sp(xy)**2)
432
433           if (stat) then
434               b = a
435
436           else
437
438          b(xx) = half*factor+(half**2*(factor*(Sp(xx)-Sp(yy))-yield(xxt)
439       n    + yield(yyt))/ syield2)*factor
440         b(yy) = half*factor+(half**2*(factor*(-Sp(xx)+Sp(yy))+yield(xxt)
441       n    - yield(yyt))/ syield2)*factor
442           b(xy) = two * alpha1*Sp(xy)/(two*syield2)
443        end if
444
445           dlambda1 = (Dd(K11)*a(xx)+Dd(K21)*a(yy))*b(xx)
446       n    + (Dd(K12)*a(xx)+Dd(K22)*a(yy))*b(yy)
447       n    + a(xy)*Dd(K44)*b(xy)       /two
448
449           dlambda = syield/dlambda1
450           depl = zero
451           depl(xx:yy) =  dlambda * b(xx:yy)
452           depl(xy) = dlambda*b(xy)
453
454           reeqpl = dlambda
455           call ut_elastic(dSpl, (/(0.0d0,I=1,6)/), depl, Dd)
456           dSpl(xy)=dSpl(xy)/two
457
458           else
459               dSpl(xx:yy) = Sp(xx:yy) - yield(xxt:yyt)
460               depl(xx) =  -(Dd(K12) * dSpl(yy) - Dd(K22) * dSpl(xx))
461       n                   /  (Dd(K11) * Dd(K22) - Dd(K12) ** 2)
462               depl(yy) =  (Dd(K11) * dSpl(yy) - Dd(K12) * dSpl(xx))
463       n                   /  (Dd(K11) * Dd(K22) - Dd(K12) ** 2)
464               reeqpl = sqrt(depl(xx)**2+depl(yy)**2)
465           end if
466
467           if (stat) then
468               dSpl(xy) = temp
469               depl(xy) = two*temp/Dd(K44)
470               reeqpl = reeqpl + abs(depl(xy))
471           else
472               dSpl(xy)=dSpl(xy)/two
473           end if
474           Sp(xy) = temp
475           end if
476
477        return
478        end
479
480
481
482        subroutine return_to_inter(Sint,dSpl,a,Sr1,Sr2,Sh1,Sp)
483        use def_par_math
```

```
      use def_par_dir
      include 'vaba_param.inc'
      dimension a(:),Sr1(:), Sr2(:), Sh1(:), Sp(:), dSpl(:), Sint(:)
      t1 = (Sh1(xx) - Sr1(xx)) * a(xx) + (Sh1(yy) - Sr1(yy))
     n* a(yy) + a(xy) * (Sh1(xy) - Sr1(xy))
      t2 = Sr2(xx) - Sr1(xx)
      t3 = Sr2(yy) - Sr1(yy)
      t4 = Sr2(xy) - Sr1(xy)
      t5 = one / (a(xx) * t2 + a(yy) * t3 + a(xy) * t4)
      Sint = Sp
      Sint(xx) = t1 * t2 * t5 + Sr1(xx)
      Sint(yy) = t1 * t3 * t5 + Sr1(yy)
      Sint(xy) = t4 * t1 * t5 + Sr1(xy)
      dSpl = Sp-Sint
      return
      end



      subroutine Yield_Hill (syield, depl, dSpl, Sp, yield, Dd,
     n           cL, props, Gf, E, de, D, eeqpl, u)
      use def_par_math
      use def_par_dir
      use def_par_props
      use utility_routines
      include 'vaba_param.inc'
      dimension depl(:), Sp(:), yield(:), Dd(:), props(:), Gf(:),
     n           E(:), dSpl(:), de(:), D(:)
      dimension a(size(Sp)),hard(size(Gf)),
     n           fs(size(yield)), dummy(2)
      real(8), optional :: u(size(Sp))
      dimension array1(Size(Sp)+Size(yield)+Size(a) +4)

      beta = props(pYbc)
      gamma = props(pYgc)


      syield1 = sqrt((Sp(xx) * Yield(yyc) / yield(xxc) + half
     n        * Sp(yy) * beta)* Sp(xx) + (half*Sp(xx) * beta +Sp(yy)
     n        * yield(xxc) / yield(yyc)) * Sp(yy) + Sp(xy)**2
     n        * gamma)
      syield  = syield1 - sqrt(yield(xxc)*yield(yyc))


      if (syield.gt.zero) then

          a(xx) = (Sp(xx) * yield(yyc) / yield(xxc) + half * Sp(yy)
     n    * beta) / syield1
          a(yy) = (Sp(yy) * yield(xxc) / yield(yyc) + half * Sp(xx)
     n    * beta) / syield1

          a(xy)=Sp(xy)*gamma/syield1

          dlambda1 = (Dd(K11)*a(xx)+Dd(K21)*a(yy))*a(xx)
     n    + (Dd(K12)*a(xx)+Dd(K22)*a(yy))*a(yy)
     n    + a(xy)*Dd(K44)*a(xy) /two
```

```
540
541           if (dlambda1 .ge. 0) then
542               dlambda1 = max(tiny(dlambda1),dlambda1)
543           end if
544
545           dlambda = syield/dlambda1
546           depl = zero
547           depl(xx:yy) = dlambda * a(xx:yy)
548           depl(xy) = dlambda* a(xy)
549           eeqpl =  dlambda
550           call ut_elastic(dSpl, (/(0.0d0,I=1,6)/), depl, Dd)
551           dSpl(xy)=dSpl(xy)/two
552           if(present(u))then
553               u = a
554           end if
555
556           return
557       end if
558
559       return
560       end
561
562
563       subroutine hardening(yield,bhard,epl,props,cL,Gf,E,D,tem,
564      n                     nstate,dmg,eeqpl,ctable,ryield)
565       use def_par_math
566       use def_par_dir
567       use def_par_props
568       use utility_routines
569
570       include 'vaba_param.inc'
571       dimension yield(:),Gf(:),epl(:),props(:),bhard(:),tem(:),E(:)
572       dimension ein(size(E)),ctable(:,:), tyield(size(yield))
573       dimension dmg(:), D(:)
574       parameter (ratio_t=1.d0/100.d0, ratio_c=1.d0/100.d0)
575
576
577       call ut_vuhard(ryield, hard, eeqpl, ctable,
578      n               int(props(ncomp)))
579       yield(xxc:yyc) = max(props(pYxc:pYyc)*ratio_c,props(Fucx:Fucy)
580      n               * ryield)
581       if (eeqpl.gt.props(ecpl)) then
582           ryield= yield(yyc)/(props(Fucy))
583       else
584           ryield = one
585       end if
586
587       tyield(xxt) = min(props(pYxt),max(props(pYxt) * ratio_t,
588      n              (-props(pYxt)
589      n            * (D(K11) * props(pYxt) * cL * epl(xxt)
590      n            - props(pYxt)**two * cL - two * D(K11)
591      n            * Gf(xxt)) / (props(pYxt)**two * cL + two
592      n            * D(K11) * Gf(xxt)))))
593
594       yield(xxt) = min(props(pYxt),max(props(pYxt)*ratio_t,
595      n             props(pYxt) - props(pYxt)**2 * cL
```

```
596       n           * (tem(xxt)-props(pYxt)/D(K11))
597       n           / (two*Gf(xxt))))
598        yield(xxt) = min(props(pYxt),max(props(pYxt)*ratio_t,ryield
599       n           * (yield(xxt)-(props(pYxt)-tyield(xxt)))))
600
601        tyield(yyt) = min(props(pYyt),max(props(pYyt) * ratio_t,
602       n              (-props(pYyt)
603       n            * (D(K22) * props(pYyt) * cL * epl(yyt)
604       n            - props(pYyt)**two * cL - two * D(K22)
605       n            * Gf(yyt)) / (props(pYyt)**two * cL + two
606       n            * D(K22) * Gf(yyt)))))
607
608        yield(yyt) = min(props(pYyt),max(props(pYyt)*ratio_t,
609       n             props(pYyt) - props(pYyt)**2 * cL
610       n           * (tem(yyt)-props(pYyt)/D(K22))
611       n           / (two*Gf(yyt))))
612        yield(yyt) = min(props(pYyt),max(props(pYyt)*ratio_t,ryield
613       n           * (yield(yyt) - (props(pYyt)-tyield(yyt)))))
614
615        return
616        end
617        endmodule
618
619
620
621        subroutine vumat(
622       1 nblock, ndir, nshr, nstatev, nfieldv, nprops, lanneal,
623       2 stepTime, totalTime, dt, cmname, coordMp, charLength,
624       3 props, density, strainInc, relSpinInc,
625       4 tempOld, stretchOld, defgradOld, fieldOld,
626       5 stressOld, stateOld, enerInternOld, enerInelasOld,
627       6 tempNew, stretchNew, defgradNew, fieldNew,
628       7 stressNew, stateNew, enerInternNew, enerInelasNew )
629
630        use def_par_dir
631        use def_par_props
632        use def_par_math
633        use def_par_state
634
635        use utility_routines
636        use sb_yield_surfaces
637        use sb_other
638
639        include 'vaba_param.inc'
640
641        dimension props(nprops), density(nblock),
642       1 coordMp(nblock,*), charLength(nblock),
643       2 strainInc(nblock,ndir+nshr), relSpinInc(nblock,nshr),
644       3 tempOld(nblock), stretchOld(nblock,ndir+nshr),
645       4 defgradOld(nblock,ndir+nshr+nshr),
646       5 fieldOld(nblock,nfieldv), stressOld(nblock,ndir+nshr),
647       6 stateOld(nblock,nstatev), enerInternOld(nblock),
648       7 enerInelasOld(nblock), tempNew(nblock),
649       8 stretchNew(nblock,ndir+nshr),
650       9 defgradNew(nblock,ndir+nshr+nshr),
651       1 fieldNew(nblock,nfieldv),
```

```
     2 stressNew(nblock,ndir+nshr), stateNew(nblock,nstatev),
     3 enerInternNew(nblock), enerInelasNew(nblock)

      character*80 cmname
      logical idam
      parameter (idam = .TRUE.)
      integer dirc
      LOGICAL OK

      dimension Sp(ndir+nshr), yield(2*ndir+nshr),
     nepl(ndir*2+nshr), Gf(ndir*2+nshr), dmg(ndir), tote(ndir),
     ntotemax(ndir), E(ndir), xNu(xy:xy+ndir*2-1), G(xy:xy+nshr-1),
     ndepl(ndir+nshr),t_depl(4,ndir+nshr), t_dSpl(4,ndir+nshr),
     nD(ndir*2+nshr), Dd(ndir*2+nshr), dummy(2), dS(ndir+nshr),
     ndSpl(ndir+nshr),ctable(2,props(ncomp)), syield(4),
     na(size(Sp)),eeq(4), depl1(ndir+nshr),Sr1(ndir+nshr),
     nSr2(ndir+nshr),Sh1(ndir+nshr),Sint(ndir+nshr)

      Sp      = zero
      yield   = zero
      epl     = zero
      Gf      = zero
      dmg     = zero
      tote    = zero
      totemax = zero
      E       = zero
      xNu     = zero
      G       = zero
      depl    = zero
      t_depl  = zero
      t_dSpl  = zero
      D       = zero
      Dd      = zero
      dummy   = zero
      dS      = zero
      dSpl    = zero
      ctable  = zero
      syield  = zero
      a       = zero
      eeq     = zero
      depl1   = zero
      Sr1     = zero
      Sr2     = zero
      Sh1     = zero
      Sint    = zero
      t_dS    = zero

      eeqpl   = zero
      factor  = zero

      dV      = zero
      ryield  = zero
      syield  = zero

      call ut_asgn_props(E, xNu, G, Gf, props, ctable)
```

```
708       call ut_assem_K_mtx(D, E, xNu, G)
709
710       if(steptime.eq.zero) then
711           do CONCURRENT (n = 1:nblock)
712                   call ut_elastic(StressNew(n,:),StressOld(n,:),
713       n           StrainInc(n,:),D)
714                   call stable_time(D, props(bdamp), factor,
715       n           charLength(n), density(n))
716                   StressNew(n,:) = StressNew(n,:)*factor
717                   call ut_int_energy(enerInternNew(n),enerInternOld(n),
718       n           StressNew(n,:),StressOld(n,:),
719       n           StrainInc(n,:),density(n))
720               end do
721           else
722               do CONCURRENT (n = 1:nblock)
723                   syield_h= -one
724                   syield_r= -one
725
726                   STRESSOLD(n,:)=STRESSOLD(n,:)-stateOld(n,dampx:dampzx)
727
728                   call ut_asgn_fstate(yield, epl, tote, totemax, props,
729       n           stateOld(n,:),dmg,eeqpl,ryield,dV)
730
731                   dV = dV + Sum(StrainINC(n,1:2))
732
733                   if (eeqpl.ge.props(ncomp+props(ncomp))) then
734                       if (SUM(StrainInc(n,: )).ne.zero) then
735                           stateNew(n,delp) = zero
736                       else
737                           StressNew(n,:)=StressOld(n,:)
738                           goto 1212
739                       end if
740                   else
741                       stateNew(n,delp) = one
742                   end if
743
744
745                   if (idam) then
746                       call assem_dmg_K(dmg, Dd, D, STRESSOLD(n,:),
747       n                    totemax, E, yield, epl, props)
748                   else
749                       Dd = D
750                   end if
751
752           call ut_elastic (Sp, STRESSOLD(n,:), StrainINC(n,:), Dd)
753
754           call ut_dev_stress (StressOld(n,:), Sp, dS)
755
756
757           call Yield_Rankine (syield(1),t_depl(1,:),t_dSpl(1,:),
758       n   Sp, yield,Dd, charLength(n), props, Gf, E,
759       n   StrainInc(n,:), dirc,D,eeq(1), Stateold(n,seqr),
760       n   stateNew(n,:),dV)
761
762
763
```

```
764           call Yield_Hill (syield(2), t_depl(2,:), t_dSpl(2,:),
765       n     Sp, yield, Dd, charLength(n), props, Gf,
766       n     E, StrainInc(n,:), D, eeq(2), a)
767
768
769           if (syield(1).ge.0.AND.syield(2).ge.0.) then
770
771                   call Yield_Rankine (syield(3),t_depl(3,:),
772       n           t_dSpl(3,:), Sp-t_dSpl(2,:), yield,Dd, charLength(n),
773       n           props, Gf, E, StrainInc(n,:), dirc, D, eeq(3),
774       n           Stateold(n,seqr), stateNew(n,:),dV)
775                   call Yield_Hill (syield(4), t_depl(4,:),
776       n           t_dSpl(4,:), Sp-t_dSpl(1,:), yield, Dd, charLength(n),
777       n           props, Gf, E, StrainInc(n,:), D, eeq(4))
778
779                   if (syield(3).ge.0.AND.syield(4).ge.0.) then
780                           Sr1  = Sp - t_dSpl(1,:)
781                           Sh1  = Sp - t_dSpl(2,:)
782                           Sr2  = Sh1 - t_dSpl(3,:)
783                       call return_to_inter(Sint,dSpl,a,Sr1,Sr2,Sh1,Sp)
784                           syield_h = syield(2)
785                           syield_r = syield(1)
786                           reeqpl = eeq(1)
787                           eeqpl = eeqpl + eeq(2)
788                           depl1 = t_depl(2,:)
789                           depl = t_depl(1,:)
790                       elseif(syield(3).lt.0.AND.syield(4).ge.0.)then
791                           dSpl = t_dSpl(2,:)
792                           depl = t_depl(2,:)
793                           syield_h = syield(2)
794                           syield_r = -1.0d0
795                           reeqpl = zero
796                           eeqpl = eeqpl+eeq(2)
797                           depl1 = zero
798                       else
799                           dSpl = t_dSpl(1,:)
800                           depl = t_depl(1,:)
801                           syield_h = -1.0d0
802                           syield_r = syield(1)
803                           reeqpl = eeq(1)
804                           eeqpl = eeqpl
805                           depl1 = zero
806                       end if
807                   elseif (syield(1).lt.0.AND.syield(2).ge.0.) then
808                       dSpl = t_dSpl(2,:)
809                       depl = t_depl(2,:)
810                       syield_h = syield(2)
811                       syield_r = -1.d0
812                       reeqpl = zero
813                       eeqpl = eeqpl+eeq(2)
814                       depl1 = zero
815                   else
816                       dSpl = t_dSpl(1,:)
817                       depl = t_depl(1,:)
818                       syield_h = -1.d0
819                       syield_r = syield(1)
```

```
820                     reeqpl = eeq(1)
821                     eeqpl = eeqpl
822                     depl1 = zero
823                   end if
824 7745  continue
825
826               call sume(tote,totemax,STRAININC(n,:),StressOld(n,:),
827      n        syield_h,depl)
828
829               if (syield_r .ge. zero .or. syield_h .ge. zero) then
830
831                   call sumepl(StressOld(n,:), yield, epl, depl,
832      n            eeqpl, reeqpl, syield_r, syield_h,
833      n            props)
834
835                   call hardening(yield,dummy,epl,props,charLength(n),
836      n            Gf, E, D, totemax, 1, dmg, eeqpl,
837      n            ctable,ryield)
838
839                   call damage(dmg, Dd, D, Gf, E, yield, depl, props,
840      n            charLength(n),tote,totemax)
841
842                   STRESSNEW(n,:) = Sp - dSpl
843               else
844
845                   STRESSNEW(n,:) = Sp
846               end if
847
848           call damping(STRESSOLD(n,:), STRESSNEW(n,:),dt,
849      n    statenew(n,dampx:dampzx), props(bdamp))
850
851           call ut_asgn_tstate(stateNew(n,:),yield,epl, tote,
852      n    totemax, dmg, eeqpl, ryield,
853      n    Stateold(n,seqr)+reeqpl,props,dV)
854
855           E = props(Ex:Ez)
856
857           call ut_int_energy(enerInternNew(n),enerInternOld(n),
858      n    StressNew(n,:),StressOld(n,:),
859      n    StrainInc(n,:),density(n))
860
861           call ut_ine_energy(enerInelasNew(n),enerInelasOld(n),
862      n    StressNew(n,:),StressOld(n,:),
863      n    depl,depl1,density(n))
864 1212      continue
865           end do
866       end if
867       return
868       end
```