Department of Construction Sciences
Solid Mechanics

ISRN LUTFD2/TFHF-14/5186-SE(1-84)

XFEM - Analysis and Implementation

Master's Dissertation by

Simon Arnesson

Supervisors:
Matti Ristinmaa, Division of Solid Mechanics

Examiner:
Mathias Wallin , Division of Solid Mechanics

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

Abstract

The finite element method is a popular numerical approach for solving solid mechanical and other types of engineering problems. This work addresses a development of the finite element method called the *eXtended Finite Element Method*, or XFEM for short. This method is designed in order to simulate problems where some kind of discontinuity is present in the system's governing equations. Something that is not possible in the standard FE-method. This paper intends to outline the method's theoretical foundation, discuss the merits of the method for solving a selection of typical cases as well as evaluate a MATLAB implementation of the method that was written in conjunction with this work. Discussion points for the theoretical overview are how one or several discontinuities' geometrical properties can be efficiently stored and evaluated for different kinds of problems with the aid of a *level set function*, how a discontinuity's physical behavior can be described by a *enrichment function* and how the connection between the level set and enrichment functions and the standard FE-method works. The MATLAB program is evaluated in a series of test cases each posing a specific challenge in adapting the XFEM approach to that kind of problem.

Sammanfattning

Finita element metoden är ett populärt verktyg för numerisk simulering av hållfasthets- och andra ingenjörsproblem. Detta verk behandlar en utveckling av finita element metoden kallad *eXtended Finite Element Method* eller förkortat "XFEM"'. Denna metod är utvecklad i syfte att kunna simulera fall där någon form av diskontinuitet finns i de ekvationer som styr det fysikaliska beteende som önskas beskrivas, något som inte är möjligt i den vanliga finita element metoden. Denna studie är gjord med avsikt att ge en övergripande bild av XFE-metodens bakomliggande teori, diskutera möjliga applikationsområden samt att implementera metoden i ett MATLAB-program och visa upp detta programs funktionalitet. Här avhandlas ett sätt att beskriva en diskontinuitets geometri i form av en *level set - funktion* samt hur denna funktion anpassas till olika problem. Likaså skildras hur diskontinuiteters fysikaliska uppförande hanteras i metoden via vad som kallas *berikningsfunktion* samt hur denna kopplas till standard versionen av finita element metoden. Konkreta exempel där berikningsfunktionen implementeras till olika typer av diskontinuiteter ges. MATLAB- programmet utvärderades i en serie exempelfall där varje exempel skildrar anpassning av en eller flera diskontinuiteters geometri och/eller fysikaliskt uppförande för en specifik typ av problem.

Contents

List of Figures

1 Introduction

1.1 Purpose

This thesis is submitted for partial fulfilment of requirements for a Masters degree in mechanical engineering at the engineering faculty (LTH) at Lunds University.

1.2 Background

The finite element method is a numerical method used to find approximate solutions to differential equations. The relevant domain is divided into smaller elements connected by node points. By making some well founded assumption of the equations' behaviour on the element level and applying known boundary condition, nodal values for the equation can be calculated. The idea is that intra element deviation from the true solution will be insignificant in the grand scheme of things. If element scale is chosen with care nodal values will approximate the correct ones well. This method is useful for many engineering applications where some field equation is simulated over some body. The method is however incapable of handling discontinuous fields without carefully designing the element layout. This is often a consuming affair in both time and effort. This is something that the eXtended Finite Method aims to resolve. To circumvent this problem XFEM elements close to the discontinuity have additional nodal values, values originating in a function describing the desired discontinuous behaviour on the element level. For approximating the solution the additional nodal values are superimposed on top of the primary ones.

1.3 Scope

This work will concentrate on XFEM applications for solid mechanics problems. The theoretical framework for the method will be laid out as well as the discretization of the stress-strain function over an arbitrary body. The types of discontinuities this paper will explore are:

- Inclusions - closed, with different material than the rest of the body.
- Voids - holes.
- Bi-material bodies - sudden transition to another material, not closed.
- Cracks - fracture that has not (yet) split the body in two.

For these types of discontinuities a way of describing the discontinuous interfaces' geometry independent of the element grid, in the form of what is called a level set function, will be actualized. Furthermore the enrichment functions describing the physical behaviour of the different types of discontinuities, used for the superimposed secondary nodal values, will be formulated. In some cases a choice of enrichment function has to be made and in these cases

the practical foresight used to make this choice are discussed and explained. In parallel to the theoretical work the discussed items are implemented into the numerical computing environment MATLAB. The ambition level of this code is restricted to two dimensions, small strains and linear elasticity. This program is designed to work in tandem with the finite element library CALFEM, developed by the department of solid- and structural-mechanics at LTH. While the code is intended to be used as a complete program where the user only edits a set up file, individual files can be used as part of a library. This paper will evaluate the code in a series of test cases each corresponding to one or several topics in the theoretical section. The evaluation involves reviewing performance but first and foremost is the achievement of a sufficient solution. In some cases a comparison with the standard FE-method with a specially designed grid can be made, elsewhere existing software is used in the evaluation.

2 XFEM formulation and element integration

The purpose of this section is to introduce the enrichment associated with XFEM such that the structure of the finite element equation set is revealed. Later on details related to the enrichment will be discussed.

2.1 Strong and weak form

The general formulation of the strong form for a body in equilibrium is as follows

$$\tilde{\nabla}\boldsymbol{\sigma} + \mathbf{b} = \mathbf{0} \tag{1}$$

Where $\boldsymbol{\sigma}$ is the stress tensor and $\mathbf{b}$ is the forces in the body, i.e. gravity. The next step is to consider a weak form, or a virtual energy form of the strong formulation of the body's behaviour. This can be done by multiplying with a arbitrary weight function, $\mathbf{v}$, and integrating over the whole body. The equation then takes the form

$$\int_V (\tilde{\nabla}\mathbf{v})^T \boldsymbol{\sigma}\, dV = \int_S \mathbf{v}^T \mathbf{t}\, dS + \int_V \mathbf{v}^T \mathbf{b}\, dV \tag{2}$$

where $\mathbf{t}$ is the surface loading and $\tilde{\nabla}$ is the gradient operator acting on a vector in matrix format.

2.2 XFEM formulation including enrichment.

The next step is the choice of the weight function, $\mathbf{v}$. This is done with the so called Galerkin's method which is discussed in some detail in [1]. With this the weight function, with the enrichment considered, may be written as

$$\mathbf{v} = \mathbf{N}\mathbf{c} + \mathbf{N}_{enr}\mathbf{q} \tag{3}$$

$$\tilde{\nabla}\mathbf{v} = \mathbf{B}\mathbf{c} + \mathbf{B}_{enr}\mathbf{q} \tag{4}$$

where $\mathbf{c}$ is the usual nodal quantities and $\mathbf{q}$ the part related to the enrichment term, these will be discussed in detail later on. In addition, $\mathbf{N}$ and $\mathbf{B}$ are the usual shape function matrix and the strain displacement matrix respectively. In the same manner $\mathbf{N}_{enr}$ and $\mathbf{B}_{enr}$ can be labelled but now related to the enrichment. Moreover, the differential operator was included in eq.(4). Then all the terms in eq.(2) will be put to the left hand side, so we get an equality with zero. Insert eq.s(3) and (4), which describes the weight function, into (2). This will produce

$$\begin{aligned} &\mathbf{c}^T[\int_V \mathbf{B}^T\boldsymbol{\sigma}\, dV \int_S \mathbf{N}^T\mathbf{t}\, dS \int_V \mathbf{N}^T\mathbf{b}\, dV] \\ &+ \mathbf{q}^T[\int_V \mathbf{B}_{enr}^T\boldsymbol{\sigma}\, dV \int_S \mathbf{N}_{enr}^T\mathbf{t}\, dS \int_V \mathbf{N}_{enr}^T\mathbf{b}\, dV] = 0 \end{aligned} \tag{5}$$

Since both $\mathbf{c}$ and $\mathbf{q}$ are arbitrary, eq.(5) can be divided into two parts as

$$\int_V \mathbf{B}^T \boldsymbol{\sigma}\, dV = \int_S \mathbf{N}^T \mathbf{t}\, dS \int_V \mathbf{N}^T \mathbf{b}\, dV \tag{6}$$

$$\int_V \mathbf{B}_{enr}^T \boldsymbol{\sigma}\, dV = \int_S \mathbf{N}_{enr}^T \mathbf{t}\, dS \int_V \mathbf{N}_{enr}^T \mathbf{b}\, dV \tag{7}$$

We now have two equations which must both be fulfilled; one with the standard FEM equations and one which involve terms that comes from the enrichment. The next step is to find a constitute relation between stress and strain and to define and approximation for the displacement, i.e.

$$\boldsymbol{\sigma} = \mathbf{D}\boldsymbol{\epsilon} \tag{8}$$

where $\boldsymbol{\epsilon}$ is the strain matrix defined as $\boldsymbol{\epsilon} = \tilde{\nabla}\mathbf{u}$, and

$$\mathbf{u} = \mathbf{N}\mathbf{u}_{std} + \mathbf{N}_{enr}\mathbf{u}_{xtra} \tag{9}$$

as previously, $\mathbf{u}_{std}$ are the standard nodal displacements and $\mathbf{u}_{xtra}$ are related to the enrichment part. Inserted into eq.s (6) and (7) gives

$$\int_V \mathbf{B}^T\mathbf{D}\mathbf{B}\, dV\, \mathbf{u}_{std} + \int_V \mathbf{B}^T\mathbf{D}\mathbf{B}_{enr}\, dV\, \mathbf{u}_{xtra} = \int_S \mathbf{N}^T\mathbf{t}\, dS \int_V \mathbf{N}^T\mathbf{b}\, dV \tag{10}$$

$$\int_V \mathbf{B}_{enr}^T\mathbf{D}\mathbf{B}\, dV\, \mathbf{u}_{std} + \int_V \mathbf{B}_{enr}^T\mathbf{D}\mathbf{B}_{enr}\, dV\, \mathbf{u}_{xtra} = \int_S \mathbf{N}_{enr}^T\mathbf{t}\, dS \int_V \mathbf{N}_{enr}^T\mathbf{b}\, dV \tag{11}$$

2.3 Defining the XFEM and FEM matrices

From eq.s (10) and (11) we can see that there will be four different types of stiffness matrices; one that is the normal stiffness matrix for FEM, combinations using both the standard and enriched B-matrix, and one with only enrichment. The matrices involving a combination of normal and enriched B-matrices are normally referred to as "blended" stiffness, this will be discussed later on. However it is noted that the equation system can be written as

$$\begin{bmatrix} \mathbf{K}_{std} & \mathbf{K}_{blend} \\ \mathbf{K}_{blend}^T & \mathbf{K}_{xtra} \end{bmatrix} \begin{bmatrix} \mathbf{u}_{std} \\ \mathbf{u}_{xtra} \end{bmatrix} = \mathbf{F} \tag{12}$$

where

$$\mathbf{K}_{std} = \int \mathbf{B}^T\mathbf{D}\mathbf{B}\, dV \tag{13}$$

$$\mathbf{K}_{blend} = \int \mathbf{B}^T\mathbf{D}\mathbf{B}_{enr}\, dV \tag{14}$$

$$\mathbf{K}_{xtra} = \int \mathbf{B}_{enr}^T\mathbf{D}\mathbf{B}_{enr}\, dV \tag{15}$$

For simplicity, in the discussion and examples it will be assumed that a 2-dimensional geometry is concerned. In addition, in the numerical parts it will be assumed that the reference mesh is defined by 4-node isoparametric elements.

3 Level Sets

3.1 Definition

As the idea of the XFEM is to capture discontinuities over some boundary without mesh adjustment it is vital to be able to keep track of this interface. The most common way to do this is with a level set function. To visualize this let us imagine a domain Ω which is divided into two separate, non zero sub-domains Ω_a and Ω_b. The boundary between Ω_a and Ω_b is denoted Γ and represent the interface of interest, see Fig. 1. The level set $\phi(\mathbf{X})$ function has the property of:

$$\phi(\mathbf{X}) < 0 \; for \; \mathbf{X} \in \Omega_b \tag{16}$$

$$\phi(\mathbf{X}) > 0 \; for \; \mathbf{X} \in \Omega_a \tag{17}$$

$$\phi(\mathbf{X}) = 0 \; for \; \mathbf{X} \in \Gamma \tag{18}$$

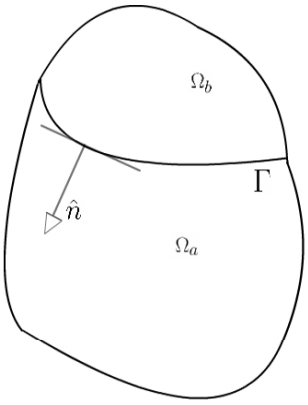


Figure 1: Domain separated into sub-domains by an interface.

The sign of the level set function can now be used to reveal what part of the domain certain coordinates $\mathbf{X}$ belong to, a zero value means the point of interest is located on the interface itself. A helpful definition but an explicit function, chosen to fit some a priori known circumstances, is needed for implementation.

3.2 Moving Interface

Obviously the level set function has a time dependence when the interface of interest is moving [1]. If one assumes knowledge of the initial conditions $\phi(\mathbf{X}, t = 0)$, the interface

evolution function is given by the material time derivative

$$\frac{D\phi}{Dt} = 0 \tag{19}$$

$$\frac{\partial\phi}{\partial t} + F\left\|\nabla\phi\right\| = 0 \tag{20}$$

Where $F(\mathbf{X}, t)$ is the interface's normal outwards velocity. As the level set function is zero at the interface the time derivative must be zero as well. The evolution function can be rewritten with the velocity $\mathbf{v}$

$$\frac{\partial\phi}{\partial t} + \mathbf{v}\cdot\nabla\phi = 0 \tag{21}$$

In an practical situation, where the velocity field is given, an expression for updating the level set function can be derived with the time scheme of choice. A first order explicit time discretization with an arbitrary spatial discretization is given as

$$\frac{\phi^{n+1} - \phi^n}{\Delta t} = -\phi^n_{,i} v^n_i \tag{22}$$

$$\phi^{n+1} = \phi^n - \Delta t \phi^n_{,i} v^n_i \tag{23}$$

The Courant-Friedrichs-Lewy stability condition [2] applies

$$\Delta t \sum_{i=1}^{N} \frac{v_i}{\Delta x_i} \leq C_{max} \tag{24}$$

$$C_{max} = 1 \ (typically) \tag{25}$$

This work will put little emphasis on this subject. It could however be noted that this is a popular area of research, in particular for dynamic crack growth. The problem of crack growth direction have several possible solutions, for example finding a direction such as modus II stress intensity factor is zero [10] maximizing the energy release rate locally. Other more general mixed mode methods exist as well [11]. These methods seem to match experimental results but are not perfectly understood and assumes ideal materials [12]. This is somewhat problematic for real world applications as it ignores potentially crucial factors such as small scale material imperfections.

3.3 Cracks and Other Curve Type Discontinuities

If the interface consist of some kind of curve a convenient choice of level set function is the signed distance function. Let us define a vector $\mathbf{d}$ as the minimum distance from a point of interest $\mathbf{X}$ to the boundary $\mathbf{X}_\Gamma$. Using the outwards normal $\hat{n}$ to the interface the signed distance function is defined as

$$\phi(\mathbf{X}) = \mathbf{d}\cdot\hat{n} \tag{26}$$

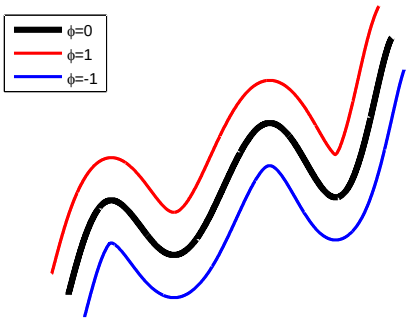


Figure 2: Level set visualisation for a curve discontinuity.

As Fig. 2 shows this ensures that the domain will have a positive and a negative side with respect to the interface and points on the interface will have a zero level set value. As will be discussed later, the impact of the level set functions value is not really important far away from the interface. The main use of the function is to keep track of elements cut by the discontinuous interface and applying the nodal enrichment, characteristic to the XFEM, to these elements.

3.3.1 Crack Specific Treatment

As a crack can not split a body into two well defined, separate regions on its own it is not enough to define the discontinuity with the signed distance function normal to the interface. This is preferably solved by adding an extra level set function φ, this time represented as the signed distance tangential to the closest crack tip from the query point. Assuming the crack has two tips (analogous for edge cracks but with only one help function) φ can be defined with two help functions, one for each tip. Thus boxing in the crack sufficiently

$$\varphi(\mathbf{X}) = \mathbf{max}(\varphi_1, \varphi_2) \tag{27}$$

$$\varphi_i = (\mathbf{X} - \mathbf{X}_{cTip\,i}) \cdot \mathbf{t}_i \tag{28}$$

Where $\mathbf{t}_i$ is the tangent to the ith crack tip (as an imagined extension of the crack). Identifying the crack as all points containing the level sets $\{\phi = 0, \varphi \leq 0\}$. Fig. 3 helps to visualize this.

3.4 Closed Discontinuities

3.4.1 Circular Interfaces

For circular discontinuities the signed distance function translates to the following

$$\phi(\mathbf{X}) = \|\mathbf{X} - \mathbf{X}_c\| - r \tag{29}$$

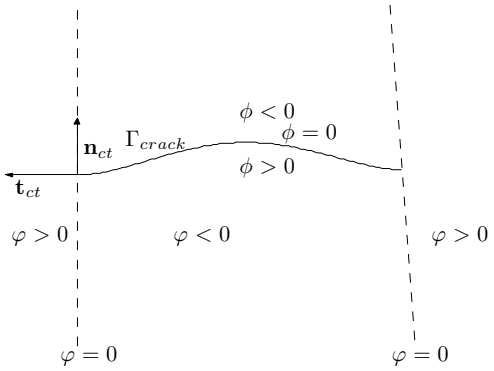


Figure 3: The two types of level set functions for cracks.

Where $\mathbf{X}_c$ is the coordinates for the circles centre and r is the circle radius. The level set value will be negative inside the circle, zero on the interface and positive outside.

3.4.2 Elliptical Interfaces

Similarly to the circular level set function the elliptical function comes from the definition of an ellipsis

$$\phi(\mathbf{X}) = \left\| \left(\frac{x}{a}\right)^2 + \left(\frac{y}{b}\right)^2 \right\| - 1 \tag{30}$$

In the case that the semi major and semi minor lengths a and b do not coincide with the chosen coordinate axes some transformation is needed

$$\mathbf{X}' = \mathbf{R}(\mathbf{X} - \mathbf{X}_c) \tag{31}$$

$$\phi(\mathbf{X}) = \left\| \left(\frac{x'}{a}\right)^2 + \left(\frac{y'}{b}\right)^2 \right\| - 1 \tag{32}$$

Where $\mathbf{R}$ is a matrix rotating the coordinate vector to fit the ellipses orientation. The elliptical level set function does not follow the pattern above of representing a signed distance, it does however serve the exact same purpose. Like the circular level set function all points inside the ellipse will have a negative level set value, positive outside and zero on the interface. As earlier stated it is the sign of the function and not the actual number that is of interests.

3.4.3 Polygonal Interface

A practical way of defining a level set function for a closed polygonal interface is to consider the N line segments forming a polygon, and use the signed distance from the point of

interest to the closest segment as the level set value [3]

$$\phi(\mathbf{X}) = \|\mathbf{X} - \mathbf{X}_{\Gamma\, min}\| \, \mathbf{sign}\Big((\mathbf{X} - \mathbf{X}_{\Gamma\, min}) \cdot \mathbf{n}_i\Big) \tag{33}$$

$$1 \leq i \leq N \tag{34}$$

Where $\mathbf{X}_{\Gamma\, min}$ are the coordinates on the ith line segment one finds if one follows the normal belonging to this segment from the coordinates $\mathbf{X}$ to said line segment when this distance is the shortest out of the N possible distances. There are cases when no unique normal can be inserted into the above function, i.e. when the point $\mathbf{X}_{\Gamma\, min}$ belong to the connected endpoints of two neighbouring line segments. In these cases the sign of the level set function is considered to be positive if $\|\mathbf{X} - \mathbf{X}_{\Gamma\, min}\|$ belong to the cone of normals existing on this part of the interface and negative otherwise. Fig. 4 can aid in visualizing this. Assuming the segments are numbered counter clockwise and the segments in question are numbered i and i+1

$$\arctan\left(\frac{\hat{n}_{i\,x}}{\hat{n}_{i\,y}}\right) \leq \arctan\left(\frac{(\mathbf{X} - \mathbf{X}_{\Gamma\, min})_x}{(\mathbf{X} - \mathbf{X}_{\Gamma\, min})_y}\right) \leq \arctan\left(\frac{\hat{n}_{i+1\,x}}{\hat{n}_{i+1\,y}}\right) \tag{35}$$

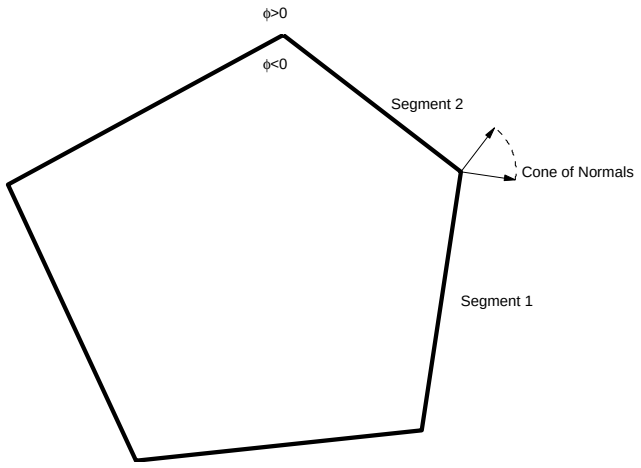


Figure 4: Construction of level set for polygon.

4 Enrichment

4.1 Enrichment Function

The level set function provides knowledge of the location of the geometry. The remaining part is to define the enrichment function in the XFEM approximation. The idea is to add new degrees of freedom to the system and superimpose these on top of the standard FEM DOFs with some weight function. The task of the enrichment function is to supply this weight in a fashion that captures the behaviour of the discontinuity. This assumes knowledge about the type of discontinuity, but it is hard to imagine a meaningful practical situation where this is not the case. As the behaviour of the discontinuity strongly relates to the shape of the interface it is common to choose an enrichment function formulated with the level set function.

4.1.1 Partition of Unity

As the discontinuities are an uniquely local event some restriction must be applied. A partition of unity is a set of functions which sum is one over a specified domain Ω^{PU}

$$\sum_i f_i(\mathbf{X}) = 1 \tag{36}$$

$$\forall\, \mathbf{X} \in \Omega^{PU} \tag{37}$$

The partition of unity method allows for the introduction of an arbitrary function [4], in our case enrichment function $\psi(\mathbf{X})$, in the approximation. In FEM, usually, the shape functions gives this partition of unity and it is common practise to let shape functions play the same role in the enriched part of the XFEM. This work will not deviate from this norm but it is noted that because the local approximation of the standard and enriched part of the XFEM formulation does not share a common origin it is not necessary to use the same shape functions. A FEM approximation using a partition of unity function f, over a domain consisting of M nodes, could look like this

$$u^{aprox}(\mathbf{X}) = \sum_{I=1}^{M} f_I u_I^{std} \tag{38}$$

Adding the XFEM part to this, the domain has L enriched nodes starting with node number L_1

$$u^{aprox}(\mathbf{X}) = \sum_{I=1}^{M} f_I u_I^{std} + \sum_{I=1}^{M} f_I \sum_{J=L_1}^{L} \psi(\mathbf{X}) u_J^{xtra} \tag{39}$$

When introducing the shape functions, i.e. $f_I = N_I$ this approximation becomes exclusively local

$$u^{aprox}(\mathbf{X}_I) = u_I^{std} + \psi(\mathbf{X}_I) u_I^{xtra} \tag{40}$$

4.1.2 Strong vs. Weak Discontinuities

The single most important criteria for choosing enrichment function is weather the discontinuity is strong or weak. A strong discontinuity has a jump in some field variable. Examples of this type of discontinuities are holes and cracks. Depending on the specific type of discontinuity the chosen enrichment function are often some sort of binary on/off type of function. Examples of this are the Heaviside and step- functions, which are to be explored in more detail later on. Fig. 5 gives an example of what shape the enrichment function can take for a strong discontinuity.

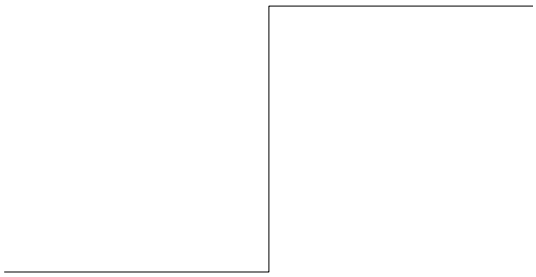


Figure 5: Step type enrichment function.

Weak discontinuities are for example an inclusion of another material within a body. This type of problem requires a kink, see Fig. 6, rather then a jump in the displacement field. In these cases the displacement is actually continuous and the discontinuity only appears in the derivative (strain) of the primary field. The enrichment function chosen to capture this behaviour are some form of ramp function.

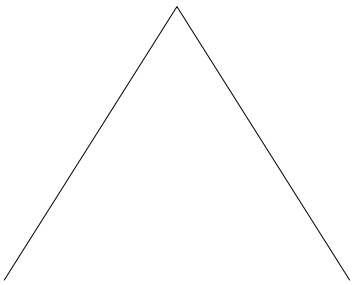


Figure 6: Ramp type of enrichment function.

4.1.3 Voids

A common choice of enrichment function for voids is the Heaviside function [3],

$$H(\mathbf{X}) = \begin{cases} 1 \; if \; \phi(\mathbf{X}) > 0 \\ 0 \; if \; \phi(\mathbf{X}) < 0 \end{cases} \tag{41}$$

A binary distinction between material and no material is achieved, as negative level set value indicates that the coordinates in question are located inside the void. In practise it is common to remove all degrees of freedom associated with elements located completely inside the void and only consider contributions from elements only containing material and the elements whose nodal support is cut of by the void.

4.1.4 Inclusions

Inclusions fall under the category weak discontinuities. As mentioned earlier it is not the primary field that is discontinuous but its derivative and the choice of enrichment function needs to fit this criterion. An obvious contender is the following type of ramp function

$$\psi(\mathbf{X}) = \mid \phi(\mathbf{X}) \mid = \left| \sum_I N_I \phi_I \right| \tag{42}$$

A continuous shape is acquired with a kink and undefined derivative where ϕ equals zero. This choice is sufficient for most purposes but might introduce convergence problems in blended elements [3]. This problem can be addressed in a number of ways, one possibility is to only let the above definition apply to the intersected elements and construct some other fitting value for the blended elements. However, a choice [5] that has several other inherited advantages is the following

$$\psi(\mathbf{X}) = \sum_I |\phi_I| \, N_I \; - \; \left| \sum_I \phi_I N_I \right| \tag{43}$$

Besides having the desired shape this function has the advantage of being zero at all points outside the intersected elements. The computational advantage gained by this is that the blended elements will not contribute to the extended parts of the stiffness matrix. Furthermore this enrichment function assures zero values at all nodes in the intersected elements as well, giving the computational benefit of not having the extended inclusion DOFs contribute to the displacement field.

4.1.5 Crack Enrichment

Besides having a discontinuous primary field (strong discontinuity) on opposite sides of the crack, special consideration must be given to the crack tip. Because elements containing a crack tip can not be fully divided by the crack, a step function can not be used as enrichment function here, for this purpose the extra level set function comes in handy.

First lets use the level set functions ϕ and φ to define some help variables to make the crack tip enrichment a little more straightforward

$$\theta = \arctan \frac{\varphi}{\phi} \tag{44}$$

$$r = \sqrt{\phi^2 + \varphi^2} \tag{45}$$

Without diving too deep into the realm of fracture mechanics it can be stated that for linear elastic fracture mechanics the displacement field near the crack tip [6], can be written as

$$\begin{aligned} u_x = &\frac{K_I}{2\mu}\sqrt{\frac{r}{2\pi}}\cos\frac{\theta}{2}\left(\kappa - 1 + 2\sin^2\frac{\theta}{2}\right) \\ &+\frac{K_{II}}{2\mu}\sqrt{\frac{r}{2\pi}}\sin\frac{\theta}{2}\left(\kappa + 1 + 2\cos^2\frac{\theta}{2}\right) \end{aligned} \tag{46}$$

$$\begin{aligned} u_y = &\frac{K_I}{2\mu}\sqrt{\frac{r}{2\pi}}\sin\frac{\theta}{2}\left(\kappa + 1 - 2\cos^2\frac{\theta}{2}\right) \\ &-\frac{K_{II}}{2\mu}\sqrt{\frac{r}{2\pi}}\cos\frac{\theta}{2}\left(\kappa - 1 - 2\sin^2\frac{\theta}{2}\right) \end{aligned} \tag{47}$$

Where K_I and K_{II} are the mode 1 and 2 stress intensity factors. The Koslov constant is defined as

$$\kappa = 3 - 4\nu \qquad (plane\,stress) \tag{48}$$

$$\kappa = \frac{3-\nu}{1+\nu} \qquad (plane\,strain) \tag{49}$$

It can be shown that the crack tip displacement field is contained by four functions [7],

$$\gamma(r,\theta) = \begin{bmatrix} \sqrt{r}\cos\frac{\theta}{2} \\ \sqrt{r}\sin\frac{\theta}{2} \\ \sqrt{r}\sin\frac{\theta}{2}\sin\theta \\ \sqrt{r}\cos\frac{\theta}{2}\sin\theta \end{bmatrix} \tag{50}$$

Furthermore, it is possible to combine these functions for the crack tip enrichment[6]. Unfortunately this enrichment requires that one new degree of freedom per node is introduced for each of these four functions. On the other hand the problem of crack tip enrichment is solved and only two elements in a domain can contain a crack tip (one for edge cracks). The functions used are not discontinuous on their own, only when combined with the help variables defined with the level set functions the desired properties show up. If we consider a domain where the part of the domain containing crack tip enrichment is denoted Ω_{ct} and

the part of with normal crack enrichment Ω_H, the XFEM displacement approximation for such a case would read as

$$u^{aprox}(\mathbf{X}_I) = u_I^{std} + H(\mathbf{X}_I)u_I^{xtra} \quad (51)$$
$$for I \in \Omega_H$$
$$u^{aprox}(\mathbf{X}_I) = u_I^{std} + \gamma_1(\mathbf{X}_I)u_I^{xtra1} + \gamma_2(\mathbf{X}_I)u_I^{xtra2} + \gamma_3(\mathbf{X}_I)u_I^{xtra3} + \gamma_4(\mathbf{X}_I)u_I^{xtra4} \quad (52)$$
$$for I \in \Omega_{ct}$$

Note that no node can have with both heaviside and crack tip enrichment.

5 Test Case 1: Circular inclusion

5.1 Preprocessing and Set Up

5.1.1 Geometry

A relative simple start problem was chosen, in part to simplify validation and troubleshooting but mainly for the author to explore the most essential aspects of XFEM. The problem was decided to be a 2D quadratic plate with a circular inclusion of another material. This was implemented for a plane stress case with the boundary conditions that the bottom edge of the plate was locked in place in vertical direction resting on rollers, and the left most corner of this edge was also locked in place in the horizontal direction. The force was applied on the plates top edge and equally distributed along this edge.

5.1.2 Level Set

The implemented level set function was the following:

$$\phi(\mathbf{X}) = |\mathbf{X} - \mathbf{X_c}| - r \tag{53}$$

Where $\phi(\mathbf{X})$ is the level set value at coordinates $\mathbf{X}$. $\mathbf{X_c}$ is the centre coordinates for the circular inclusion and r is the radius of the inclusion. These values were stored at all nodes for later use. To extract the level set value at arbitrary coordinates the element shape functions are used, since the used shape functions are for first order interpolation they will not be able to fully capture the circle. Instead the extracted interface will have the shape of a polygon. This has no impact on elements that are not cut by the interface. Elements containing the interface will be cut up in polygons anyway for integrating purposes.

5.2 Main Calculation

5.2.1 Enrichment Function

The implemented enrichment function is:

$$\psi(\mathbf{X}) = \sum_{I=1}^{4} |\phi_I| N(\mathbf{X})_I - |\sum_{I=1}^{4} \phi_I N(\mathbf{X})_I| \tag{54}$$

The enrichment function ψ at coordinates $\mathbf{X}$ is evaluated as a combination of nodal level set values and the shape functions. Compared to other choices for enrichment functions this has two major advantages. As the enrichment function has a zero value outside cut elements as well as in cut element nodes there will be no contribution to the stiffness matrix in an element that are not cut by the interface and the displacements evaluated at the nodes will not have any extra contribution either.

5.2.2 Stiffness Matrix

Three different kinds of element stiffness matrices are present , these are for the standard, the blended and the cut elements cf. eq. 12,

$$\mathbf{K}_{enr} = \begin{bmatrix} \mathbf{K}_{std} & \mathbf{K}_{blend} \\ \mathbf{K}_{blend}^T & \mathbf{K}_{xtra} \end{bmatrix} \tag{55}$$

Where $\mathbf{K}_{std}$ represent the elements normal degrees of freedom, $\mathbf{K}_{blend}$ is the overlapping between standard degrees of freedom and the extra DOFs and $\mathbf{K}_{xtra}$ are purely for the extra DOFs. For the standard elements, i.e. elements with no enriched nodes, the element stiffness is the usual one obtained in FEM calculations:

$$\mathbf{K}_{elStd} = \int_{el} \mathbf{B^T D B} \mathrm{d}A \tag{56}$$

where

$$\mathbf{B}_I = \begin{bmatrix} N_{I,x} & 0 \\ 0 & N_{I,y} \\ N_{I,y} & N_{I,x} \end{bmatrix} \tag{57}$$

As the standard part of the element stiffness is identical with standard FEM no further description is needed. The other two part are calculated as follows

$$\mathbf{K}_{blend} = \int_{el} \mathbf{B}_{enr}^T \mathbf{D B}_{std} \mathrm{d}A \tag{58}$$

$$\mathbf{K}_{xtra} = \int_{el} \mathbf{B}_{enr}^T \mathbf{D B}_{enr} \mathrm{d}A \tag{59}$$

where

$$\mathbf{B}_{enr\,I} = \begin{bmatrix} (\psi(\mathbf{X})N_I), x & 0 \\ 0 & (\psi(\mathbf{X})N_I), y \\ (\psi(\mathbf{X})N_I), y & (\psi(\mathbf{X})N_I), x \end{bmatrix} \tag{60}$$

The comma convention reads

$$(\psi(\mathbf{X})N_I), x = \psi(\mathbf{X})\frac{\partial N_I}{\partial x} + \frac{\partial \psi(\mathbf{X})}{\partial x} N_I \tag{61}$$

and

$$\frac{\partial \psi(\mathbf{X})}{\partial x} = \sum |\phi_I| \frac{\partial N_I}{\partial x} - \frac{|\sum \phi_I N_I|}{\sum \phi_I N_I} \sum \phi_I \frac{\partial N_I}{\partial x} \tag{62}$$

For blended elements, i.e. elements neighbouring to the elements cut by the interface and thus inherits enriched nodes, the chosen enrichment function conveniently makes sure all extra parts have a zero value. For fully enriched (cut) elements two different material properties are present. Therefore the code divides the element into triangles trying to capture the circular interface as true as possible where each triangle have only one constitutive matrix **D** The full integral is calculated as the sum of all partial integrals over each individual triangle, c.f. Appendix A for numerical integration.

5.2.3 Displacement Field

The equation system to be solved:

$$\begin{bmatrix} \mathbf{K}_{std} & \mathbf{K}_{blend} \\ \mathbf{K}_{blend}^T & \mathbf{K}_{xtra} \end{bmatrix} \begin{bmatrix} \mathbf{u}_{std} \\ \mathbf{u}_{xtra} \end{bmatrix} = \mathbf{F}_{external} \tag{63}$$

The displacement of a node is the sum of the standard displacement and the extra displacement weighted with the enrichment function. However the chosen enrichment function is once again found to be extremely practical as it is zero at the nodes, $\mathbf{u}_{std}$ gives the correct displacements directly.

5.3 Post Processing

5.3.1 Stress Calculation

The stress is calculated at all nodes according to:

$$\boldsymbol{\sigma} = \mathbf{D}\boldsymbol{\epsilon} \tag{64}$$

$$\boldsymbol{\sigma} = \mathbf{D}\tilde{\boldsymbol{\nabla}}\mathbf{u} \tag{65}$$

With the element interpolation approximation applied this translates to:

$$\boldsymbol{\sigma} = \mathbf{D}\mathbf{B}_{std}\mathbf{u}_{std} \tag{66}$$

for non enriched elements and

$$\boldsymbol{\sigma} = \mathbf{D} \begin{bmatrix} \mathbf{B}_{std} & \mathbf{B}_{enrch} \end{bmatrix} \begin{bmatrix} \mathbf{u}_{std} \\ \mathbf{u}_{xtra} \end{bmatrix} \tag{67}$$

for enriched elements. For non enriched elements the code will use the element coordinates ($[\xi_1\ \eta_1] = [-1\ -1]$, coordinates in the isoparametric domain) for a corner and calculate the stresses at the nodes with the appropriate **D** matrix and store these.

A program developed by Matthew Pais at University of Florida[8] was used for comparison in which enriched elements are treated the same way as non enriched ones. In this implementation elements cut by the level set interface are once again divided into triangles creating pseudo elements where some of the new pseudo nodes does not coincide

with the original elements nodes. The calculation then proceeds using the parent element coordinates to extract the values at the pseudo nodes. This somewhat cumbersome procedure was implemented in hopes of capturing the stresses at the interface in a more precise manner. As the new pseudo elements are considered to be either the positive or negative side of the interface average nodal stresses can be calculated for the positive and negative side separately. When displaying the stresses the code makes sure no interpolation takes place between elements on opposite sides of the interface before plotting in a contour plot.

5.3.2 Validation

The implemented code was run on an identical set up as Pais code with the following properties

E-modulus positive side	69×10^9 GPa
E-modulus negative side	205×10^9 GPa
Poissons ratio pos. side	0.33
Poissons ratio neg. side	0.3
Inclusion radius	0.45 m
Applied force magnitude	1 MN
Inclusion placement	centre of plate
Plate dimensions	2×2 m
Thickness	1m

A plate of a steel like material have an inclusion of an aluminium like material. Pais program, Fig. 8, and the implemented code produce very similar results. When both this code and Pais code are set up to use 80×80 elements the maximal difference is in the order of 10^{-6} %. As the result gives reasonable values and for all intents and purposes assumes an expected symmetric stress distribution the author will consider these results adequate. To examine whether the result is grid independent is more complicated. As the splitting into triangles technique used will effectively approximate the inclusion as a polygon, the approximation will become better and better as the number of elements increase and the inclusion will change shape with this increase. The simulation depicted in Fig. 7 is run on 3025 elements. No practical difference was found when running with 6400 elements. As there are no practical application intended for this simulation this result will be considered sufficient. It should be mentioned that the stress concentration found along the boundary in Figs. 7 and 8 are a result of numerical issues and is not consistent with what would be expected from a theoretical point of view. The code can not perfectly account for the discontinuity when calculating stress in nodes on the boundary. The more lopsided and/or extreme angled an element is cut by the interface the worse the approximation used to find intra element level set values works. In extreme cases it has been found that error in this approximation has been in the order of 10% of the element length.

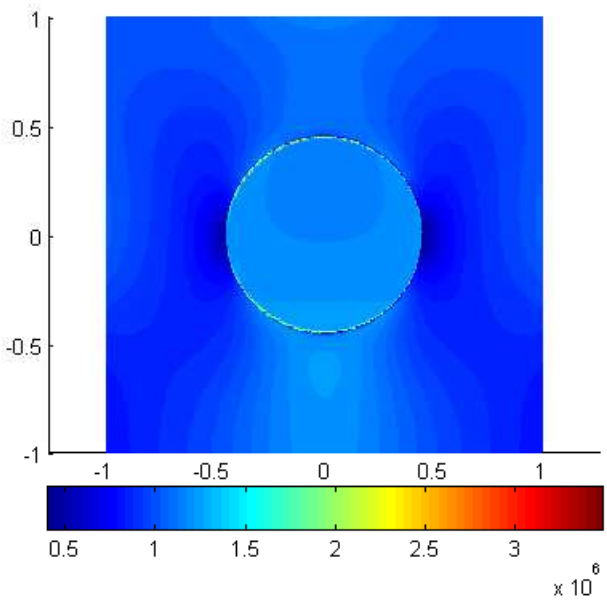


Figure 7: Implemented codes resulting von Mises stress [Pa].

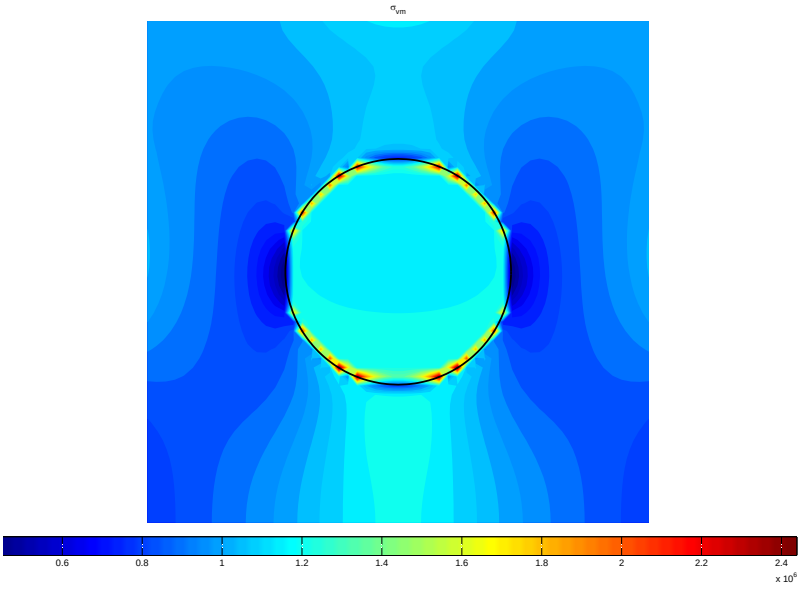


Figure 8: Resulting von Mises stress in Pais code [Pa].

6 Test Case 2: Circular Hole

6.1 Preprocessing and Set Up

6.1.1 Geometry

Very similar to test case 1, square domain with circular hole instead of inclusion. Plane stress thickness was 0.2m. Size of square domain was 2×2 m and radius of the centrally located hole was 0.45 m. A variety of grid sizes were tested, with the properties

E-modulus positive side	69×10^9 GPa
Poissons ratio pos. side	0.33
Hole radius	0.45 m
Applied force magnitude	1 MN
Inclusion placement	centre of plate
Plate dimensions	2×2 m
Plate thickness	0.2 m

The boundary conditions were the same as for test case 1 i.e. bottom edge locked in the vertical direction and bottom left is locked in horizontally as well. The force was placed along the top edge, evenly distributed pulling the plate.

6.1.2 Level Set

The used level set function is identical to the one used in test case 1, i.e.

$$\phi(\mathbf{X}) = |\mathbf{X} - \mathbf{X_c}| - r \tag{68}$$

6.1.3 Enrichment Function

Again identical to test case 1, the enrichment is taken as

$$\psi(\mathbf{X}) = \sum_{I=1}^{4} |\phi_I| N(\mathbf{X})_I - \sum_{I=1}^{4} |\phi_I N(\mathbf{X})_I| \tag{69}$$

The main difference between test case one and two are what constitutive matrices are used.

6.1.4 Stiffness Matrix

As in test case one, when integrating over an element cut by the discontinuity a different constitutive matrix is used depending on which side of the interface the current gauss point is located. In this case the negative level set side has a zero-matrix as constitutive matrix.

6.1.5 Displacement Field

As in test case one, treatment is identical to 5.2.3

6.2 Post Processing

6.2.1 Stress Calculation

The treatment is analogous with test case 1 with the exception that the constitutive matrix is a zero matrix in the void region.

6.2.2 Validation

Again a comparison with Mathew Pais code[8] is made. A basic visual grid independence check was made, at 55×55 elements this was considered satisfactory.

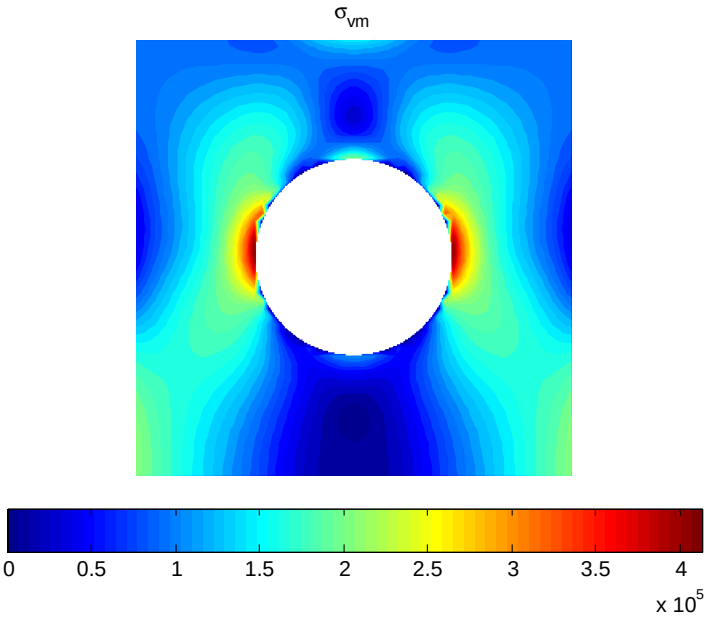


Figure 9: Resulting von Mises stress in circular hole [Pa], Pais code.

Some minor differences in peak stress are obtained, explained by the difference in stress calculation methods. Otherwise the stress distributions are close to identical. The match in results comes as no surprise as the circular hole is more or less a special case of the circular inclusion. To further validate the results an comparison with the standard FE-method can be made. With some effort a specialised grid is constructed, Fig. 11, afterwards filling in the stress calculation Fig. 12 are rather straight forward. Comparing Fig. 10 with Fig. 9 and Fig. 12 one can conclude that differences are insignificant.

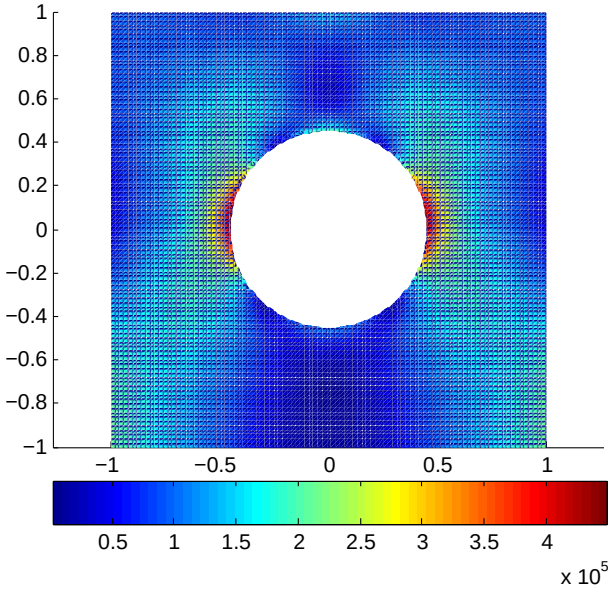


Figure 10: Resulting von Mises stress in circular hole [Pa].

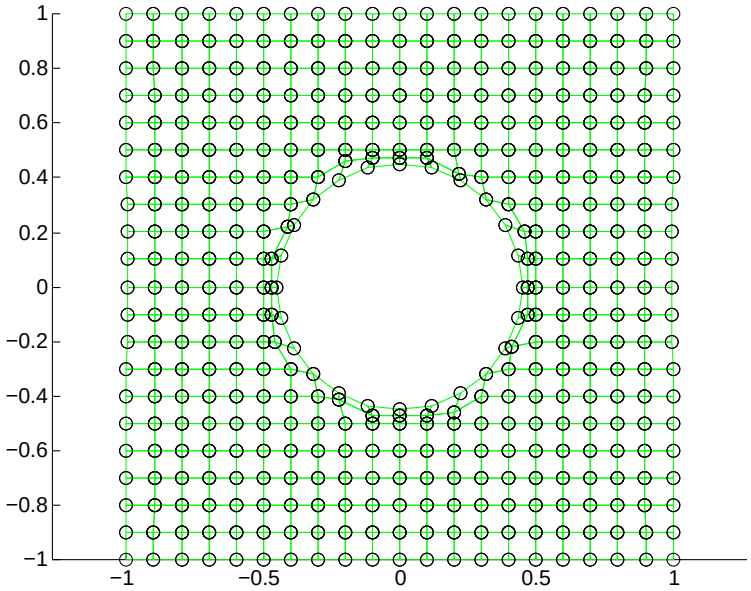


Figure 11: Traditional FEM grid

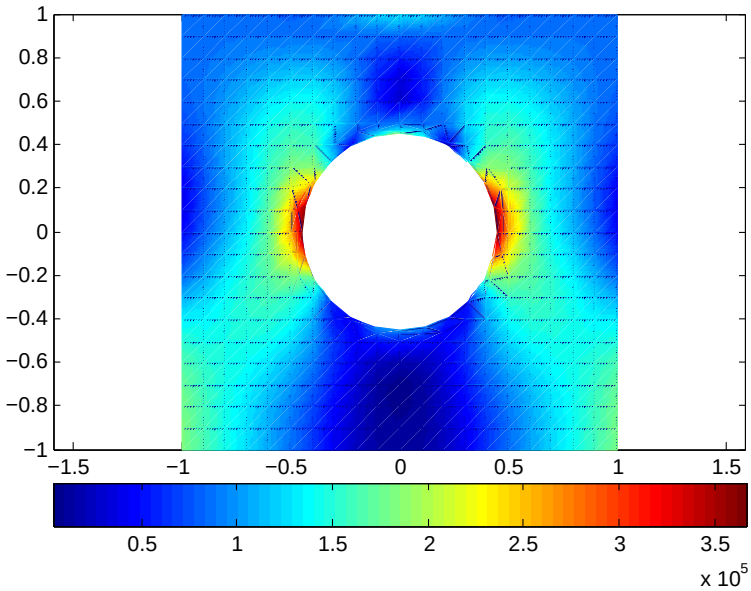


Figure 12: Traditional FEM von Mises stress [Pa]

7 Test Case 3: Polygonal Level Set Boundary

7.1 Preprocessing and Set Up

7.1.1 Geometry

The idea behind implementing options for a polygonal interface is to make it easier to create odd boundary shapes. This expanded functionality is demonstrated by Fig. 13 and Fig. 14.

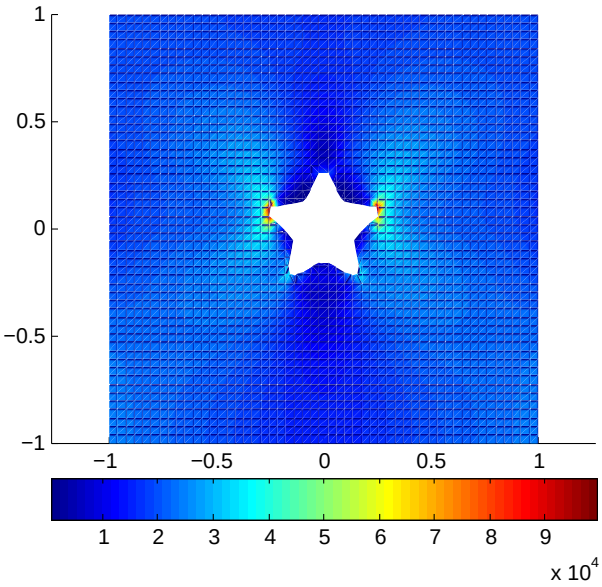


Figure 13: Star shaped inclusion boundary.

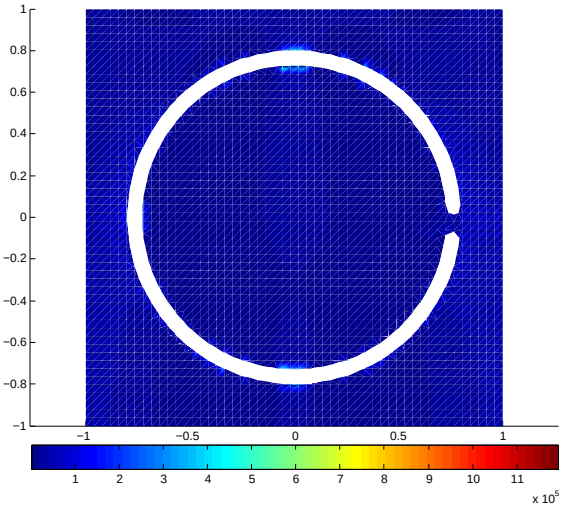


Figure 14: Ring shaped geometry.

However for validation purposes this test case will try to duplicate the results from test

case 1 and 2, circular inclusion and hole. The circle will be represented by a series of line segments where endpoints always meet on the circle circumference. Other then the interface set up geometry and load settings will match those of the other two test cases.

7.1.2 Level set

The implemented level set function was a signed distance from the coordinates of interest to the interface defined in section 3.4.3. A quite computational heavy technique for calculating this was implemented. From each node and every line segment, the intersecting point between a line along the line segments outwards normal and the node and the line segment itself was calculated. If no such point could be found the closest line segment end point was chosen to account for cone of normals cases, see Fig. 4.

7.2 Main Calculation

7.2.1 Enrichment Function

The implemented enrichment function is the same as in examples 1 and 2, i.e.

$$\psi(\mathbf{X}) = \sum_{I=1}^{4} |\phi_I| N(\mathbf{X})_I - \sum_{I=1}^{4} |\phi_I N(\mathbf{X})_I| \tag{70}$$

This to ensure that the desired result should be comparable with the previous test cases.

7.2.2 Stiffness Matrix

Follows what was outlined in example 1.

7.2.3 Displacement Field

Same as for example 1.

7.3 Post Processing

7.3.1 Stress Calculation

There are no differences in the stress calculation method used in this test case compared to the two previous cases. It is worth pointing out that even though it is possible to create much more complex geometries with the polygonal level set it is significantly more computationally expensive. Element level computations, like stress calculation, relies on approximating level set boundaries as straight. As the polygonal level set method already uses straight lines there is less to gain when decreasing element sizes.

7.3.2 Validation

The implemented code was run on an identical set up as test case 1 and 2.

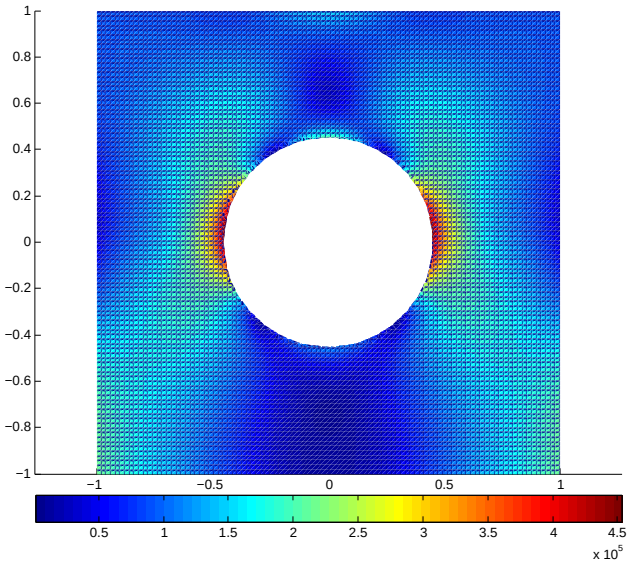


Figure 15: Implemented codes resulting von Mises stress [Pa] polygonal hole.

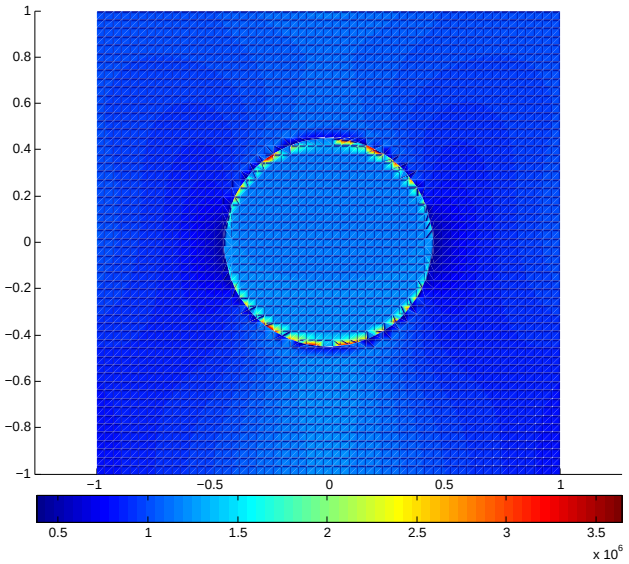


Figure 16: Resulting von Mises stress in circular inclusion [Pa], polygonal set up.

The results are presented in Fig. 15 and Fig. 16. With the exception of some minor differences in stress around the inclusion boundary the polygonal boundary method provides practically the same results as test cases 1 and 2, cf. Figs. 7 and 10.

8 Test Case 4: Straight Bimaterial Boundary

8.1 Preprocessing and Set up

An other type of discontinuous interface is a non enclosed material boundary. The example chosen to illustrate this is depicted in Fig. 17 and was set up as follows:

E-modulus top side	205×10^9 GPa
E-modulus bottom side	69×10^9 GPa
Poissons ratio top side	0.3
Poissons ratio bottom side	0.33
interface start	[-1 -0.8]
interface stop	[1 0.8]
Applied force magnitude	1 MN
Plate thickness	0.08 m
Plate dimensions	2×2 m

Plane stress is assumed to prevail. Boundary conditions and force application are identical to test case 1.

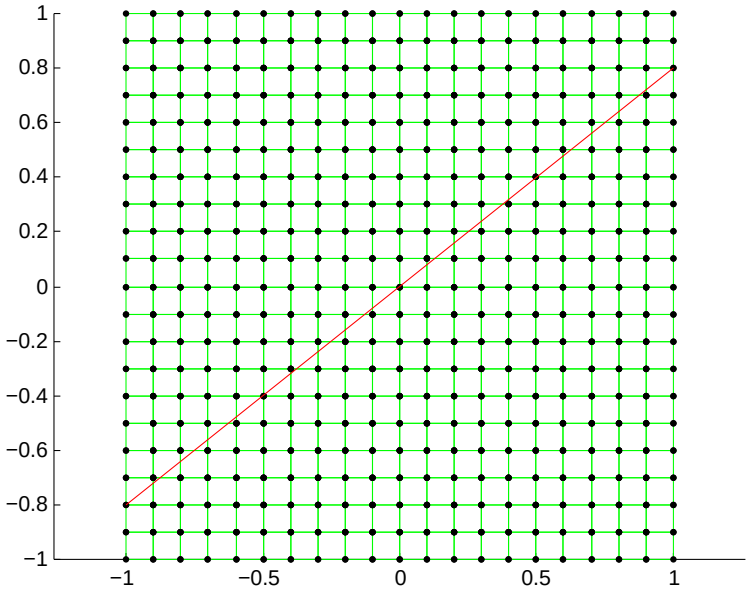


Figure 17: Visualization of geometry

Admittedly a case of limited practical use. The author speculates about potential applications for certain types of welding joints and other attachment methods involving material contact where joint durability is either irrelevant for the applied forces or can be incorporated into the model. The chosen example case is a continuous square material matrix with a well defined transition boundary from one material to another. This case was cho-

sen to be similar to other test cases and to demonstrate a different type of implementation method.

8.1.1 Geometry

As for the other examples the domain is chosen as a 2×2m plate, this time with a thickness of 0.08m and a diagonal bi-material interface. A symmetrical element grid is placed over the domain with no consideration to interface placement. The test case was implemented for both 20×20 and 201×201 elements to study grid dependency issues.

8.1.2 Level Set

For a non enclosed boundary the level set description found in 3.3.1 is used, however as the interface spans across the entire grid there is no need to keep track of endpoints and position normal to the boundary. The level set function used is the signed tangential distance, of the lowest value, from the coordinates of interest to the interface. In this case the interface is a straight line thus all points on the interface share a single normal $\mathbf{n}$. To calculate the level set ϕ for a given point $\mathbf{P}$ given two points on the interface $\mathbf{A}$ and $\mathbf{B}$ the following equation system is solved

$$\mathbf{A} + \alpha(\mathbf{B} - \mathbf{A}) = \mathbf{P} + \phi\mathbf{n}. \tag{71}$$

Where α is a factor of no interest to the calculation and can be eliminated. Assuming $(Bx - Ax)\,(B_y - A_y) \neq 0$, otherwise special consideration must be taken to avoid division by zero

$$\begin{pmatrix} \frac{P_x - A_x}{B_x - A_x} - \frac{P_y - A_y}{B_y - A_y} \\ \frac{B_x - A_x}{B_y - A_y} + \frac{B_y - A_y}{B_x - A_x} \end{pmatrix} |\mathbf{AB}\,| = \phi \tag{72}$$

8.2 Main Calculation

No different from earlier test cases.

8.2.1 Enrichment Function

The level set function is the same as for the previous cases the enrichment is

$$\psi(\mathbf{X}) = \sum_{I=1}^{4} |\phi_I| N(\mathbf{X})_I - \sum_{I=1}^{4} |\phi_I N(\mathbf{X})_I| \tag{73}$$

8.2.2 Stiffness Matrix

As the enrichment function is the same as previously description follows from test case 1.

8.2.3 Displacement field

As before the system of equations provides the standard and enriched displacements after which enriched displacements needs to be weighted and super positioned on top of the standard ones for the desired results.

8.3 Post Processing

Follows what was outlined in test case 1.

8.3.1 Validation

Fig. 18 shows the resulting stresses for the domain using more than 40000 elements. Grid independence seems solid, scaling up to a 150% finer grid results in about 1% change in peak stresses. Comparing the results produced by Pais code shown in Fig. 19 it is obvious the current code produce higher peak stresses around the interface. While displacements have greater degree of concordance, the difference between the stresses are due to different methods used for the stress calculation, around the interface. Similar to the stress concentration oddities discussed in section 5.3.2 Fig.s 18 and 19 show some unnatural stress behaviour close to the discontinuity.

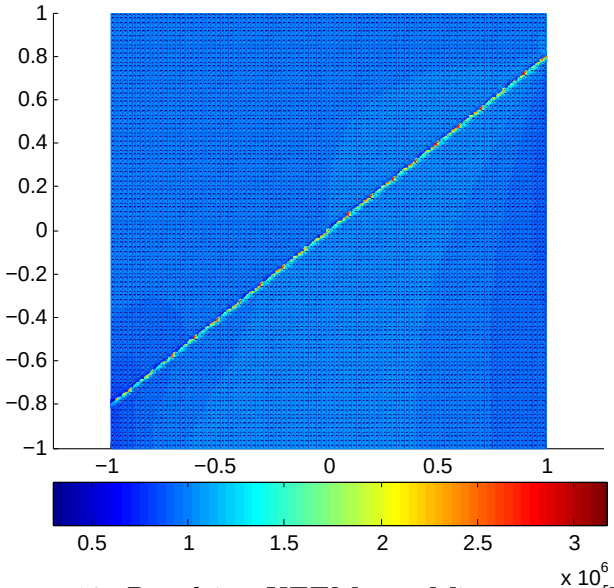


Figure 18: Resulting XFEM von Mises stress [Pa]

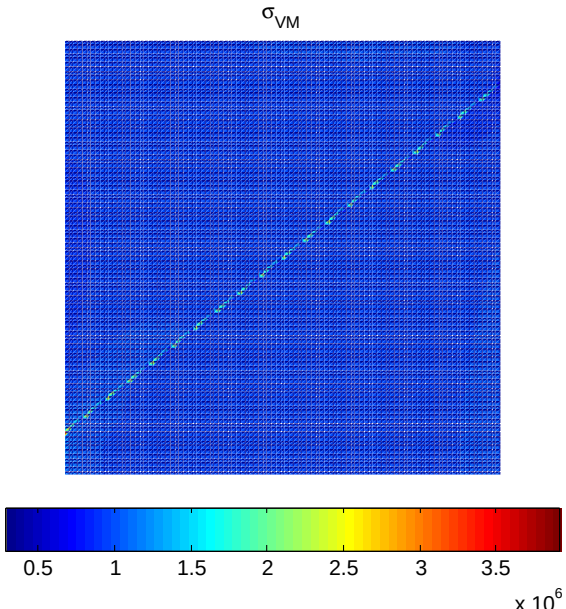


Figure 19: Pais resulting XFEM von Mises stress [Pa] for test case 4

9 Test Case 5: Straight Crack

9.1 Preprocessing and Set up

9.1.1 Geometry

Again a square domain, 2×2 m, was chosen because it made the set up easy. A fairly common application could be a crack in a thin plate with a load trying to force the crack open orthogonally, to the crack direction. Plain stress was assumed. To avoid symmetry the crack was slightly tilted. If the centre of the plate is considered to be origin for a standard Cartesian coordinate system with the y-direction pointing directly upwards the both crack tips was placed at the points [-0.8 -0.08] and [0.8 0.08]. Fig. 20 shows how the crack was placed as well as potential enriched nodes for a 29×29 grid, cracktip enrichment are red and Heaviside enrichment are blue. The test case is summarized by:

E-modulus	69×10^9 GPa
Poissons ratio	0.33
Left crack tip coordinates	x=-0.8, y=-0.08
Right crack tip coordinates	x=0.8, y=0.08
Plate dimensions	2×2 m
Plane stress thickness	0.08 m
Force magnitude	1MN

Boundary conditions and force distribution are identical to those of test case one.

9.1.2 Level Set

As described in subsection 3.3.1 when dealing with a crack it is necessary to have two level set function. One to describe a points location perpendicular to the crack (above/below) and one level set function to describe the points spatial relation to the crack in the direction parallel to the crack (inside/outside a crack tip). These both functions are named the ϕ and φ function, respectively. To calculate these values the code first calculates a normal for the crack and two outwards pointing tangents, one for each crack tip. Then the code loops over all grid points and traces a line parallel to the normal until it intersects the extended line created by the crack. The ϕ level set value is the length of this line, the sign is determined by checking if the traced line follows the normal or the negative normal. The code then follows the same principle for the both crack tip tangents and chooses the one which results in the shortest vector as the φ level set value

$$\mathbf{n} = \frac{1}{|\mathbf{X_{ct2}} - \mathbf{X_{ct1}}|} \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} (\mathbf{X_{ct2}} - \mathbf{X_{ct1}}) \tag{74}$$

$$\mathbf{t_{ct1}} = \frac{\mathbf{X_{ct2}} - \mathbf{X_{ct1}}}{|\mathbf{X_{ct2}} - \mathbf{X_{ct1}}|} \tag{75}$$

$$\mathbf{t_{ct2}} = \frac{\mathbf{X_{ct1}} - \mathbf{X_{ct2}}}{|\mathbf{X_{ct1}} - \mathbf{X_{ct2}}|} \tag{76}$$

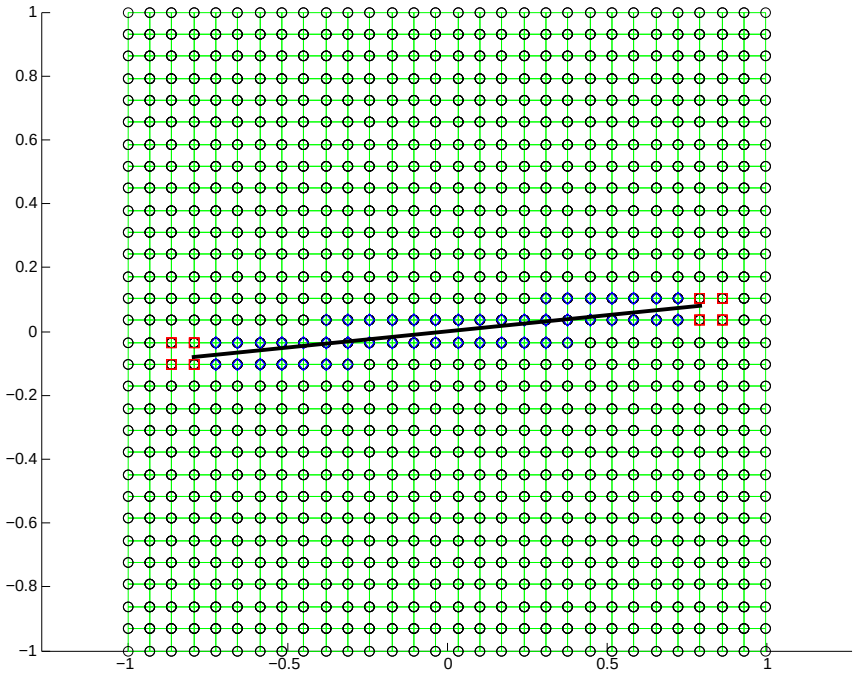


Figure 20: Crack placement.

To extract ϕ and φ level set values at coordinates $\mathbf{X}$ the following equation system was used

$$\mathbf{X} + \phi\mathbf{n} = \mathbf{X_{ct}} + \varphi\mathbf{t_{ct}} \tag{77}$$

which simplifies to:

$$\begin{cases} \phi = \frac{x_{ct} - x - (y_{ct} - y)\frac{t_{ctx}}{t_{cty}}}{n_x - n_y \frac{t_{ctx}}{t_{cty}}} \\ \\ \varphi = \frac{y - y_{ct} + \phi n_y}{t_{cty}} \end{cases} \tag{78}$$

Alternatively if $t_{cty} = 0$

$$\begin{cases} \phi = \frac{y_{ct} - y}{n_y} \\ \varphi = \frac{x - x_{ct} + \phi n_x}{t_{ctx}} \end{cases} \tag{79}$$

The system are solved for both crack tips and the solution with the lowest value for $|t_{ct}|$ is chosen.

9.2 Main Calculation

9.2.1 Enrichment Function

Two types of enrichments are needed Heaviside and cracktip. Heaviside enrichment accounts for the loss of connection between nodes in an element cut by the crack, cracktip enrichment is used for capturing special crack tip behaviour. The Heaviside enrichment needs to be slightly modified compared to a hole set up. In a hole case the enrichment indiscriminatingly gives all coordinates with negative level set value a zero value enrichment function. In the crack case the role of this enrichment is to break connection between the two sides of the crack. Either side still contains material, however there is no contribution to the stiffness to the other side. In practise it is handled as follows, for an arbitrary Gauss point dealing with element node i:

$$H(\mathbf{X_i}) = \frac{1 + sign(\phi(\mathbf{X}_{gp}))sign(\phi(\mathbf{X}_i))}{2} \tag{80}$$

Resulting in $\phi = 1$ (normal contribution) for connected nodes and $\phi = 0$ (no contribution) for unconnected nodes.

Cracktip enrichment needs to be expressed in crack tip coordinates θ and r. However due to the fact that the outwards pointing cracktip tangents used to calculate the level set functions are pointing in opposite directions some difficulties, relating to coordinate system orientation, appears when using arctan function in the code. To bypass this an alternative calculation method is used. The entire order of operations are as follows

1. Calculate crack angle $\omega_1 = \arctan\left(\frac{y_{ct2}-y_{ct1}}{x_{ct2}-x_{ct1}}\right)$
2. Rotate coordinate system $\mathbf{X}_{roti} = \begin{bmatrix} cos(\omega_i) & sin(\omega_i) \\ -sin(\omega_i) & cos(\omega_i) \end{bmatrix} (\mathbf{X} - \mathbf{X_{cti}})$
3. Calculate help variables r and θ
   (a) $r_i = \sqrt{x_{roti}^2 + y_{roti}^2}$
   $r = min(r_1, r_2)$
   (b) $\theta = atan2(y_{rot}, x_{rot})$
4. use help variables to extract the four crack tip enrichment functions.
   (a) $f_1 = \sqrt{r}\ \cos(\frac{\theta}{2})$
   (b) $f_2 = \sqrt{r}\ \sin(\frac{\theta}{2})$
   (c) $f_3 = \sqrt{r}\ \sin(\theta)\ \sin(\frac{\theta}{2})$
   (d) $f_4 = \sqrt{r}\ \sin(\theta)\ \cos(\frac{\theta}{2})$

9.2.2 Stiffness Matrix

First the code determines for each element if it is enriched or not and what kind of enrichment should be used. This is illustrated by Fig. 21

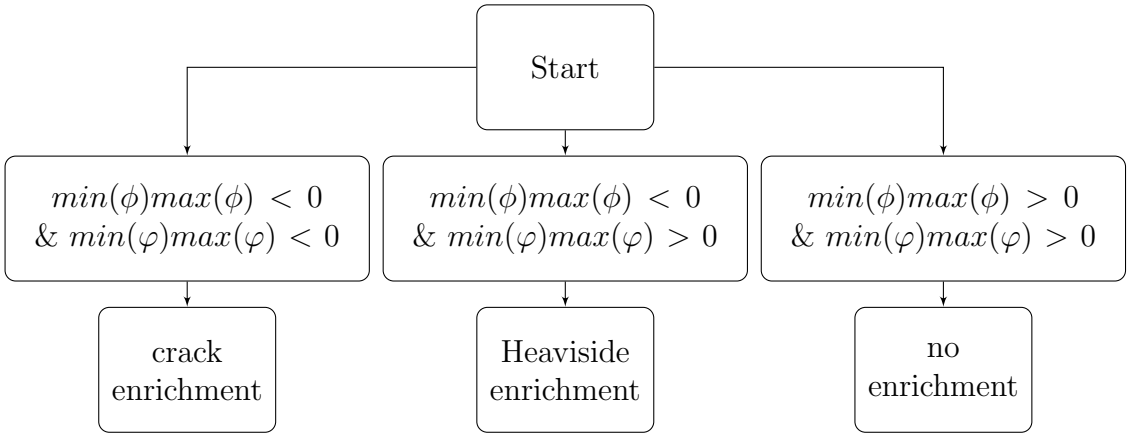


Figure 21: Illustration of calculation procedure.

All nodes within an element found to be enriched receives the appropriate enrichment, however a node can not have both heaviside and crack tip enrichment. If two neighbouring elements sharing one or more nodes are found to have different types of enrichment crack tip enrichment takes precedents over heaviside enrichment. Obviously any type of enrichment is chosen before no enrichment. It is important to remember that crack tip enrichment in practise is four types of enrichment and will add eight new DOFs to the node (four in the x-direction and four in the y direction). Once it is established how many DOFs each element have been extended with the actual element stiffness calculation can be made. This code numbers the new DOFs in such a way that Heaviside enriched DOFs are numbered lower that crack tip enriched DOFs, which in turn have a lower number than crack tip DOFs of type two etc. The code cycles through the new DOFs and latch on the appropriate matrices to the B-matrix

$$B_{tot} \;=\; [B_{std}\; B_H\; B_{f1}\; B_{f2}\; B_{f3}\; B_{f4}] \tag{81}$$

If node i have been found to have Heaviside enrichment, the B-matrix are extended according to:

$$B_{Hi} \;=\; H_{shifti} \begin{bmatrix} \frac{\partial N_i}{\partial x} & 0 \\ 0 & \frac{\partial N_i}{\partial y} \\ \frac{\partial N_i}{\partial y} & \frac{\partial N_i}{\partial x} \end{bmatrix} \tag{82}$$

Note that $\frac{\partial H}{\partial x} = 0$. Similarly in case node j needs crack tip extra DOFs the code extends the B-matrix as follows:

$$B_{f1j} = \begin{bmatrix} \frac{\partial N_j}{\partial x} f_{1j} + \frac{\partial f_1 j}{\partial x} N_j & 0 \\ 0 & \frac{\partial N_j}{\partial y} f_{1j} + \frac{\partial f_{1j}}{\partial y} N_j \\ \frac{\partial N_j}{\partial y} f_{1j} + \frac{\partial f_1 j}{\partial y} N_j & \frac{\partial N_j}{\partial x} f_{1j} + \frac{\partial f_1 j}{\partial x} N_j \end{bmatrix} \tag{83}$$

The other three variants follow exactly the same pattern. There is however a nest of chain rule derivatives hidden within $\frac{\partial f_1 j}{\partial x}$ and $\frac{\partial f_1 j}{\partial y}$ that could use some clarification. The B-matrix requires the derivative of the enrichment function with respect to the global coordinates. The enrichment function is given in crack tip polar coordinates

$$\frac{\partial f_1}{\partial x} = \frac{\partial f_1}{\partial r}\frac{\partial r}{\partial x} + \frac{\partial f_1}{\partial \theta}\frac{\partial \theta}{\partial x} \tag{84}$$

Crack tip polar coordinates are a function of Cartesian crack coordinates which in turn are a rotation of the global coordinates with the crack angle ω. First it is noted that

$$\frac{\partial r}{\partial x} = \frac{\partial r}{\partial x_{rot}}\frac{\partial x_{rot}}{\partial x} + \frac{\partial r}{\partial y_{rot}}\frac{\partial y_{rot}}{\partial x} \tag{85}$$

$$\frac{\partial \theta}{\partial x} = \frac{\partial \theta}{\partial x_{rot}}\frac{\partial x_{rot}}{\partial x} + \frac{\partial \theta}{\partial y_{rot}}\frac{\partial y_{rot}}{\partial x} \tag{86}$$

Recalling some useful relationships between the polar/Cartesian and rotated coordinate systems:

$$x_{rot} = r\cos(\theta) \tag{87}$$

$$y_{rot} = r\sin(\theta) \tag{88}$$

$$x_{rot} = x\cos(\omega) + y\sin(\omega) \tag{89}$$

$$y_{rot} = -x\sin(\omega) + y\cos(\omega) \tag{90}$$

The final pieces of the puzzle are then produced by:

$$\frac{\partial r}{\partial x_{rot}} = \cos(\theta) \tag{91}$$

$$\frac{\partial r}{\partial y_{rot}} = \sin(\theta) \tag{92}$$

$$\frac{\partial \theta}{\partial x_{rot}} = \frac{-\sin(\theta)}{r} \tag{93}$$

$$\frac{\partial \theta}{\partial y_{rot}} = \frac{\cos(\theta)}{r} \tag{94}$$

$$\frac{\partial x_{rot}}{\partial x} = \cos(\omega) \tag{95}$$

$$\frac{\partial x_{rot}}{\partial y} = \sin(\omega) \tag{96}$$

$$\frac{\partial y_{rot}}{\partial x} = -\sin(\omega) \tag{97}$$

$$\frac{\partial y_{rot}}{\partial y} = \cos(\omega) \tag{98}$$

Finally, a comprehensive list of all crack tip enrichment derivatives with respect to global coordinates expressed in available crack tip coordinates can be obtained

$$\frac{\partial f_1}{\partial x} = \frac{1}{2\sqrt{r}}\left(\cos\left(\frac{\theta}{2}\right)\cos\left(\omega\right) - \sin\left(\frac{\theta}{2}\right)\sin\left(\omega\right)\right) \tag{99}$$

$$\frac{\partial f_2}{\partial x} = \frac{1}{2\sqrt{r}}\left(-\sin\left(\frac{\theta}{2}\right)\cos\left(\omega\right) - \cos\left(\frac{\theta}{2}\right)\sin\left(\omega\right)\right) \tag{100}$$

$$\frac{\partial f_3}{\partial x} = \frac{1}{2\sqrt{r}}\left(-\sin\left(\frac{3\theta}{2}\right)\sin(\theta)\cos(\omega) - \left(\sin\left(\frac{\theta}{2}\right) + \sin\left(\frac{3\theta}{2}\right)\cos(\theta)\right)\sin(\omega)\right) \tag{101}$$

$$\frac{\partial f_4}{\partial x} = \frac{1}{2\sqrt{r}}\left(-\cos\left(\frac{3\theta}{2}\right)\sin(\theta)\cos(\omega) - \left(\cos\left(\frac{\theta}{2}\right) + \cos\left(\frac{3\theta}{2}\right)\cos(\theta)\right)\sin(\omega)\right) \tag{102}$$

$$\frac{\partial f_1}{\partial y} = \frac{1}{2\sqrt{r}}\left(\cos\left(\frac{\theta}{2}\right)\sin(\omega) + \sin\left(\frac{\theta}{2}\right)\cos(\omega)\right) \tag{103}$$

$$\frac{\partial f_2}{\partial y} = \frac{1}{2\sqrt{r}}\left(-\sin\left(\frac{\theta}{2}\right)\sin(\omega) + \cos\left(\frac{\theta}{2}\right)\cos(\omega)\right) \tag{104}$$

$$\frac{\partial f_3}{\partial y} = \frac{1}{2\sqrt{r}}\left(-\sin\left(\frac{3\theta}{2}\right)\sin(\theta)\sin(\omega) + \left(\sin\left(\frac{\theta}{2}\right) + \sin\left(\frac{3\theta}{2}\right)\cos(\theta)\right)\cos(\omega)\right) \tag{105}$$

$$\frac{\partial f_4}{\partial y} = \frac{1}{2\sqrt{r}}\left(-\cos\left(\frac{3\theta}{2}\right)\sin(\theta)\sin(\omega) + \left(\cos\left(\frac{\theta}{2}\right) + \cos\left(\frac{3\theta}{2}\right)\cos(\theta)\right)\cos(\omega)\right) \tag{106}$$

Once the extended B-matrix is sorted out the element stiffness matrix is calculated in the same manner as a previously, i.e.

$$\mathbf{K_{elm}} = \mathbf{B_{ext}}^T\mathbf{D}\mathbf{B_{ext}} \tag{107}$$

9.2.3 Displacement field

The system of equations are solved as normal. Unlike the enrichment function used in test case one and two the enrichment functions used for a crack are not designed in such a way that node values are guaranteed to be zero. Therefore a bit of extra computational effort needs to be put in to calculate nodal enrichment values for extra DOFs weighting.

9.3 Post Processing

This code continues to use the method of splitting elements up into triangular pseudo elements, if they are cut by the interface, and stresses are calculated over these rather then the proper elements. The advantage of this is that behaviour right on the interface is captured better. This advantage shrinks as the number of elements grow, it is also dubious if this is a good method to capture crack tip behaviour. As stresses approaches ∞ as polar crack tip coordinate r approaches zero, inevitably some practical concession must be made. Even though the crack tip itself is not used for integrating purposes a fail safe is in place which limits r to some small value, even with this added level of uncertainty the stress calculation method is kept for convenience.

9.3.1 Stress Calculation

The lengthy discussion in section 9.2.2 applies here as well. Split elements into triangles and use the extended B-matrix to calculate the stresses.

9.3.2 Validation

Consider first the displacement field in Fig. 22. This behaviour is inline with what is expected. At about 300 times magnification it is easy to see the crack elegantly opens. Comparing this with Pais code in Fig. 23, the results are nearly identical with the largest displacements differing $\approx \frac{1}{10}$ ‰. As stress should approach infinity at the crack tips some scaling has been made for increased visibility, in this case stress figures shows the logarithm of the von Mises principal stress. Fig. 24 shows this scaled von Mises stress distribution which at a glance look very reasonable. Even when comparing to the resulting stress in Pais code in Fig. 25. Both code suffers from the fact that stresses approach infinity at the crack tip. This will mean that finer grid size result in higher maximum stress until the artificial cap is reached, as grid points move in closer to the tip.

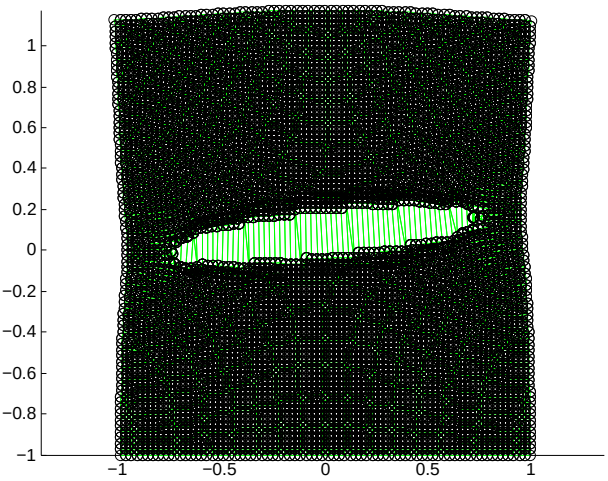


Figure 22: Displacement field in the crack case.

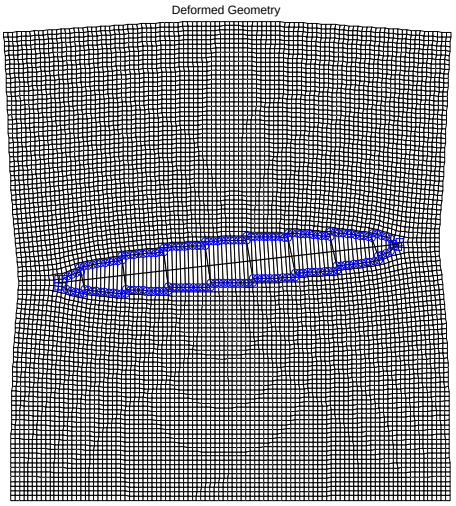


Figure 23: Displacement field for a crack Pais code.

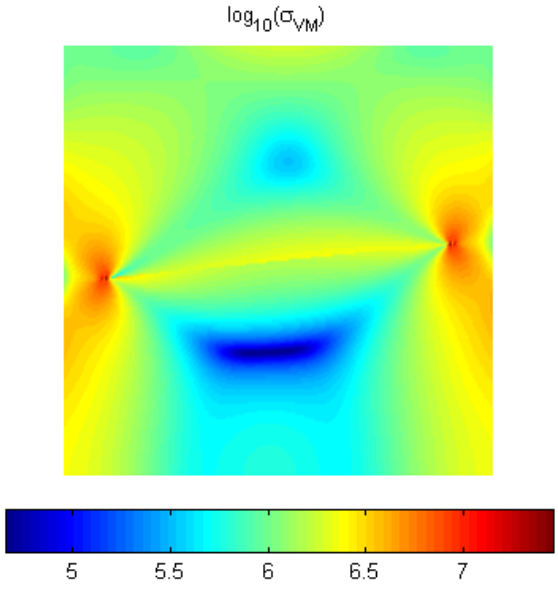


Figure 24: The codes von Mises stress distribution [Pa].

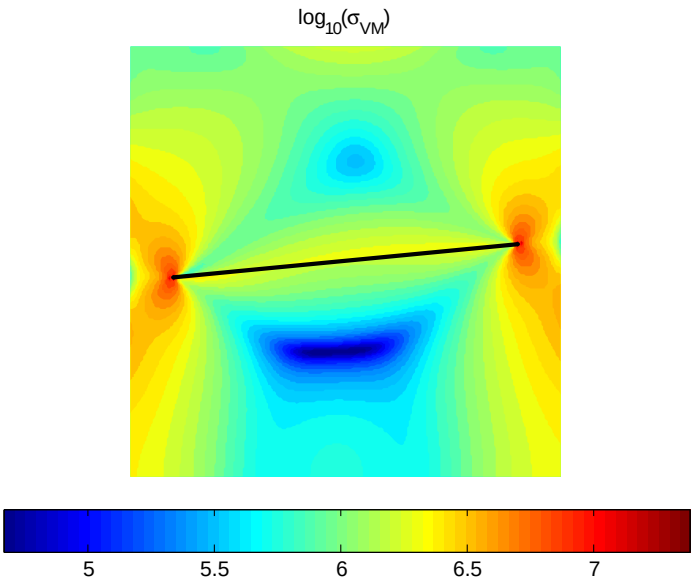


Figure 25: von Mises stresses for a crack, Pais code [Pa].

10 Test Case 6: Multiple Types of Discontinuities

10.1 Preprocessing and Set up

The final test case is meant to make a comprehensive demonstration of multiple discontinuities in the same geometry. To add a little realism the chosen geometry is that of a standardised fracture mechanics stress test specimen. The chosen specimen is the 'compact tension specimen' [9] commonly used in fracture mechanics testing, which is a suitable problem for XFEM applications. For convenience reasons the geometry will only be an approximation and the load will be 1MN evenly distributed force along the top edge. The specimen will be of a steel like material. Plane stress is assumed and boundary conditions are again identical to those used in test case 1.

10.1.1 Geometry

The compact tension test specimen geometry is approximated as Fig. 26. In reality there is an open slot from the edge to the crack tip extending inwards 0.45W, this open slot is not modelled. The geometry parameters are described by the width W and thickness T, and crack extension a. The load is denoted by P. For this demonstration case approximating the slot with a crack was deemed good enough. The geometry and other parameters are summarized by:

E-modulus	205×10^9 GPa
Poissons ratio	0.3
W	0.1 m
T	0.02 m
P	1 MN
crack length,a	0.625W

10.1.2 Level Set

The holes uses the level set function and was defined in test case 5. To conclude, this combines test case two and five. Meaning that each point in the geometry has three level set values assigned to it.

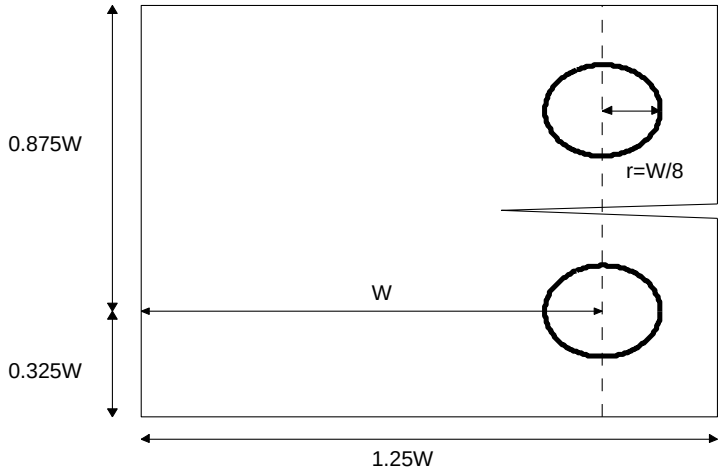


Figure 26: Multiple discontinuities geometry.

10.1.3 Enrichment Function

The crack uses the enrichment function described in subsection 9.2.1. For elements cut by the crack:

$$H(\mathbf{X_i}) = \frac{1 + sign(\phi(\mathbf{X}_{gp}))sign(\phi(\mathbf{X}_i))}{2} \tag{108}$$

For the crack tip, four enrichment functions:

1. $f_1 = \sqrt{r}\ \cos(\frac{\theta}{2})$
2. $f_2 = \sqrt{r}\ \sin(\frac{\theta}{2})$
3. $f_3 = \sqrt{r}\ \sin(\theta)\ \sin(\frac{\theta}{2})$
4. $f_4 = \sqrt{r}\ \sin(\theta)\ \cos(\frac{\theta}{2})$

are used. Where θ is the crack angle and r is the distance to the crack tip. The voids uses the enrichment function discussed in 6.1.3

$$\psi(\mathbf{X}) = \sum_{I=1}^{4} |\phi_I| N(\mathbf{X})_I - |\sum_{I=1}^{4} \phi_I N(\mathbf{X})_I| \tag{109}$$

10.1.4 Stiffness Matrix

The stiffness matrix is extended like the other test cases but this time using all the (six) applied enrichment types void, crack and four types of crack tip enrichment. Lets number the possible B-matrices 1-7. $\mathbf{B}_1$ is the standard B-matrix. Enriched B-matrices (2-7) are calculated according to:

$$\mathbf{B}_{enrch_J\, I} = \begin{bmatrix} (\psi_J(\mathbf{X})N_I), x & 0 \\ 0 & (\psi_J(\mathbf{X})N_I), y \\ (\psi_J(\mathbf{X})N_I), y & (\psi_J(\mathbf{X})N_I), x \end{bmatrix} \tag{110}$$

Where each B matrix uses components from all four shape functions (I=1-4) and its appropriate enrichment function ψ_J (J=2-7). Resulting in a total extended stiffness matrix

$$\begin{bmatrix} K_{11} & K_{12} & K_{13} & K_{14} & K_{15} & K_{16} & K_{17} \\ K_{21} & K_{22} & K_{23} & K_{24} & K_{25} & K_{26} & K_{27} \\ K_{31} & K_{32} & K_{33} & K_{34} & K_{35} & K_{36} & K_{37} \\ K_{41} & K_{42} & K_{43} & K_{44} & K_{45} & K_{46} & K_{47} \\ K_{51} & K_{52} & K_{53} & K_{54} & K_{55} & K_{56} & K_{57} \\ K_{61} & K_{62} & K_{63} & K_{64} & K_{65} & K_{66} & K_{67} \\ K_{71} & K_{72} & K_{73} & K_{74} & K_{75} & K_{76} & K_{77} \end{bmatrix} \tag{111}$$

10.1.5 Displacement field

As all other test cases, the only difference is that the stiffness matrix is larger, i.e. symbolically the solution is given as

$$\begin{bmatrix} K_{11} & K_{12} & K_{13} & K_{14} & K_{15} & K_{16} & K_{17} \\ K_{21} & K_{22} & K_{23} & K_{24} & K_{25} & K_{26} & K_{27} \\ K_{31} & K_{32} & K_{33} & K_{34} & K_{35} & K_{36} & K_{37} \\ K_{41} & K_{42} & K_{43} & K_{44} & K_{45} & K_{46} & K_{47} \\ K_{51} & K_{52} & K_{53} & K_{54} & K_{55} & K_{56} & K_{57} \\ K_{61} & K_{62} & K_{63} & K_{64} & K_{65} & K_{66} & K_{67} \\ K_{71} & K_{72} & K_{73} & K_{74} & K_{75} & K_{76} & K_{77} \end{bmatrix}^{-1} \mathbf{F} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \end{bmatrix} \tag{112}$$

10.2 Post Processing

Again as before, the enriched degrees of freedom are superimposed on top of the standard dofs which are used to calculate stresses.

10.2.1 Stress Calculation

Using all appropriate enrichments, the stress is calculated according to:

$$\boldsymbol{\sigma} = \mathbf{D} \left[\begin{array}{ccc} \mathbf{B}_1 & \ldots & \mathbf{B}_7 \end{array} \right] \left[\begin{array}{c} \mathbf{u}_1 \\ \ldots \\ \mathbf{u}_7 \end{array} \right] \tag{113}$$

10.2.2 Validation

The code yields the von Mises stresses shown in Fig. 27. At first glance the result seems

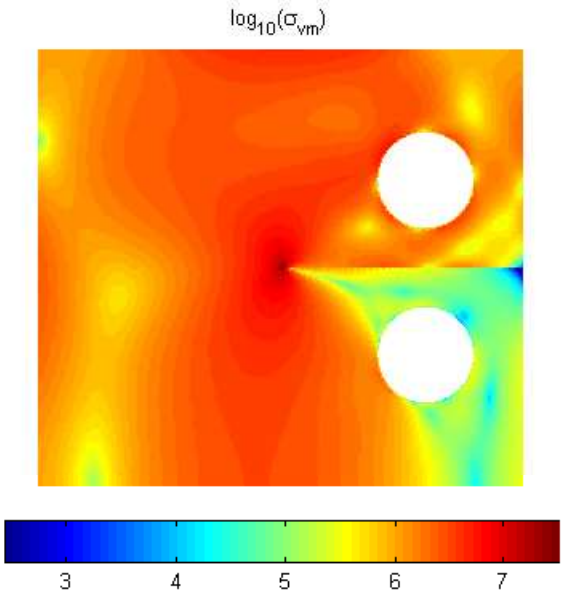


Figure 27: Multiple discontinuities von Mises stress [MPa].

reasonable. Because of the codes automated set up process elements are required to be square. This makes the choice of grid and thus testing for grid independence somewhat cumbersome as the domain is rectangular. The code was first run on a 190×171 element grid and then scaled up to 250×225. No significant changes were observed. Again a comparison was made with Pais code to draw conclusions about the correctness of the results. All stress distribution figures are scaled logarithmically for better visibility. The Pais code was set up in the same manner and the resulting von Mises stresses are shown in Fig. 28

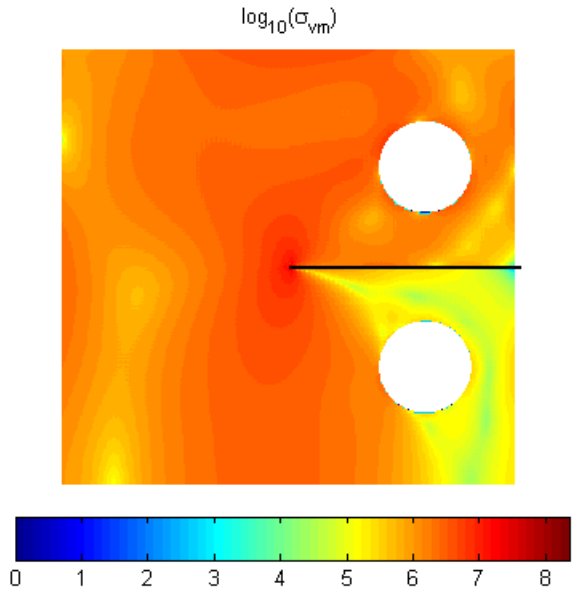


Figure 28: Comparative code, von Mises stress [MPa].

11 Conclusion and Summary

11.1 Level Sets

The level set method proved to be a convenient way to keep track of the discontinuity's placement. As long as the interface is not set by a large number of vertices, which the method described in section 3.4.3 would allow, very little computational effort needs to be spent to obtain a mesh independent description of the geometrical shape of the discontinuity. Even though enrichment can be quite different, no particular difference between open and closed discontinuities are found. All level sets discussed in this work are some form of a signed distance function which is both computationally inexpensive and easy to implement.

In test case three (section 7) it was shown that an arbitrary discontinuity interface could be created by a set of vertices. The test case dealt with closed interfaces but there are no particular obstacles to overcome when using this method on open interfaces, something that could be useful when simulating for example crack growth in directions not aligned with the original crack. This method involves some approximation as connections between vertices are considered as straight lines. This assumption does however, in most cases, not create larger errors than the element partition used when calculating element stiffness using first order shape functions. The method does however severely increase the computational effort needed as level set data needs to be calculated for each pair of neighbouring vertices before selecting the best fit rather than inserting coordinates into a fixed function.

When introducing moving boundaries some extra computational work will appear. Each iteration step must have a sub step where relevant physical properties are calculated and compared to some movement criteria. This should be mitigated by the mesh independent nature of the level set method meaning no remeshing will be needed, though a judgement call for each type of problem is advised.

11.2 Enrichments

The enrichment functions addressed in this work were all relatively easy to implement while providing a solid method of obtaining the desired physical behaviour. All while being more or less directly linked to the level set function(s). For example in the crack tip enrichment case, due to the duality of the tangential level set function, level set orientation is obscured on an element level and help variables for distance to crack tip (r) and crack angle (θ) must be calculated. The enrichment function described in section 4.1.4 was proven very useful. Having the enrichment equal zero in element nodes saves a lot of computational effort and allows for some short cuts to be made. For crack tip enrichment the enrichment functions were a bit more involved but as this type of enrichment can only be present in a low number of elements per simulation this extra expense is hardly noticeable.

11.3 Application

Some effort needs to be spent in post processing to display the results graphically. In the program accompanying this paper the element partitioning technique, used in element stiffness matrix calculation, is revisited for elements cut by the interface and each side of the interface have its stresses averaged and graphed individually. The gain in computational effort XFEM provides is most obvious when simulations normally would require updating the mesh. An obvious example is crack growth simulation. Other potential favourable implementations could be in a design phase where several simulations over a single geometry are needed. Perhaps for finding optimum hole or inclusion placement, a single mesh could be used for all simulations. Another potential candidate is numerical calculations on composite materials. The different properties of such a material's layers could be described by the XFE-method. The program developed in conjunction to this work have several weaknesses. It is not very efficient, it was consistently outperformed by the more optimized comparative program. Another weakness is the incapability of enriching an element with multiple types of enrichment. As the code determine a single type of enrichment and calls a function for that type of element stiffness matrix. A better way would be to cycle trough all types of possible enrichment and ask "is this present?" and assemble an element B-matrix before calculating the element stiffness matrix.

11.4 Future Work

The most pressing feature to expand the code with is probably to introduce some crack growth possibility. A good start would be to introduce quasi static crack growth. Fracture mechanics parameters must be calculated and some criteria for crack growth and path are needed. This has the potential to be further expanded to a dynamic simulation with cyclic load for use when fatigue fracture is of interest. The code could use some optimisation as it is not particularly efficient. The desired "plug and play" feature of the 'SetUp' and 'XFEMmain' files are possibly of limited use as it does not allow for much variation in geometry, though the rest of the individual files could find use as extending an existing library. Another improvement of the code could use is allowing for multiple types of enrichment in the stiffness matrix assembly process.

Appendices

A Numerical Integration

Consider an arbitrary triangular geometry for which a function f(x,y) determines some property at coordinates [x; y]. Suppose an other property, that is evaluated by integration of f(x,y) over the geometry is of interest. An analytic calculation of the integrand is either impossible or very expensive. It is then convenient to use *Gauss' quadrature formula* in which the integrand is approximated as a weighted sum of the function values at a number of predetermined points (exact for n points for a function of order m < n). For portability reasons these points are tabled in a transformed coordinate system according to the Fig. 31. To integrate

$$I = \int_{\triangle} f(x, y) dx dy \tag{114}$$

The domain needs to be changed. The transformation is performed with a *Jacobian matrix*, **J**, and half of the Jacobians' determinant will scale back the area of the triangle to the original coordinate system. This transformation is defined by

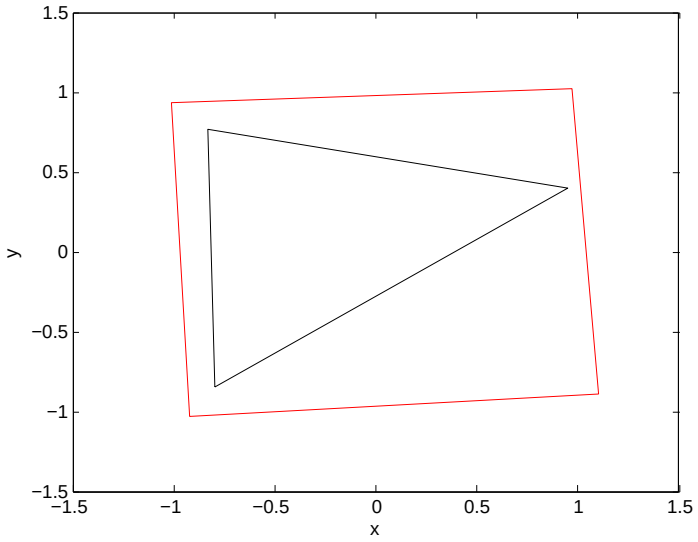


Figure 29: Original configuration of triangle inside element

$$\begin{bmatrix} \mathrm{d}x \\ \mathrm{d}y \end{bmatrix} = \mathbf{J} \begin{bmatrix} \mathrm{d}\xi \\ \mathrm{d}\eta \end{bmatrix} \tag{115}$$

Where

$$\mathbf{J} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial x}{\partial \eta} \\ \frac{\partial y}{\partial \xi} & \frac{\partial y}{\partial \eta} \end{bmatrix} \tag{116}$$

It then follows that

$$I = \frac{1}{2}\int_0^1 \int_0^{1-\eta} f(\xi,\eta) \begin{vmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial x}{\partial \eta} \\ \frac{\partial y}{\partial \xi} & \frac{\partial y}{\partial \eta} \end{vmatrix} \mathrm{d}\xi\, \mathrm{d}\eta \tag{117}$$

By numerical integration the quantity can be evaluated

$$I \simeq \frac{1}{2}\sum_1^i w_i f(\xi_i,\eta_i) \begin{vmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial x}{\partial \eta} \\ \frac{\partial y}{\partial \xi} & \frac{\partial y}{\partial \eta} \end{vmatrix}_i \tag{118}$$

This will be useful in the XFEM code when element stiffness matrices for elements with enriched nodes are evaluated.

The level set curve will cut through an element which then will be subdivided into triangles trying to capture the level set curve as true as possible. Each sub-triangle will thereafter be considered to be completely positioned on one side of the level set curve and the integral for the stiffness matrix calculated as the sum of integrals each calculated over one of the triangles. This will allow for the integration of a discontinuous function. Fig. 29 shows an example triangle inside an element in the untransformed x-y coordinate system. The stiffness matrix will have a shape as follows:

$$\mathbf{K}_{ij}^e = \iint_{elm} \mathbf{B}_i^T \mathbf{D} \mathbf{B}_j t \mathrm{d}x \mathrm{d}y \tag{119}$$

$$\mathbf{B}_i = \begin{cases} \mathbf{B}_{STD} \text{ if i=1} \\ \mathbf{B}_{ENR} \text{ if i=2} \end{cases} \tag{120}$$

For a four node isoparametric element the shape functions are defined in the isoperimetrical domain. Because the $\mathbf{B}$ matrices are functions of the shape functions an extra transformation is required. Lets name the different coordinate systems. The x-y system is the original, chosen for the geometry being analysed. When specific element is of interest the x-y coordinates are transformed into the ξ η coordinates. Fig. 30 shows the element and triangle from Fig. 29 transformed into the isoparametric domain. These range from -1 to 1 transforming the element into a perfect square. In the isoperimetric coordinates the element shape functions are defined as follows

$$\mathbf{N}_{el} = \frac{1}{4}\begin{bmatrix} (1-\xi)(1-\eta) \\ (1+\xi)(1-\eta) \\ (1+\xi)(1+\eta) \\ (1-\xi)(1+\eta) \end{bmatrix} \tag{121}$$

A mapping to x-y coordinates can be made, using the elements nodal coordinates, according to

$$x(\xi,\eta) = \mathbf{N}_{el}^T \mathbf{x}_{el} \tag{122}$$

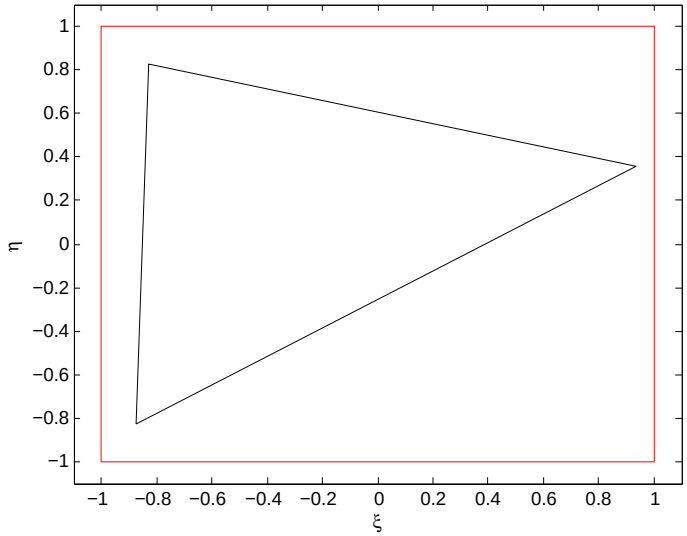


Figure 30: element and triangle transformed into coordinate system appropriate for B-matrix calculation

One additional transformation is needed to allow for a numerical integration over a triangle inside this element. This coordinate system will be denoted ξ'-η'. When a transformation into these coordinates are made the triangles corners are mapped upon:

$$[\xi'_1\,\eta'_1] = [0\,0];\ [\xi'_2\,\eta'_2] = [1\,0];\ [\xi'_3\,\eta'_3] = [0\,1] \tag{123}$$

Fig. 31 shows how this final transformation can look as well as the Gauss points, for a three point integration, which are the purpose for this final transformation. To be able to extract coordinate depending properties in one coordinate system and transfer them into another shape functions are needed. If the triangle is treated as a triangular element these will become:

$$\mathbf{N}_{\triangle} = \begin{bmatrix} 1-\xi'-\eta' \\ \xi' \\ \eta' \end{bmatrix} \tag{124}$$

With this it is possible to express the coordinates x-y as

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x(\xi',\eta') \\ y(\xi',\eta') \end{bmatrix} \tag{125}$$

Two transformations requires two Jacobians, an example of how to calculate the components of these are illustrated by

$$\mathbf{J}_{el} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial x}{\partial \eta} \\ \frac{\partial y}{\partial \xi} & \frac{\partial y}{\partial \eta} \end{bmatrix} \tag{126}$$

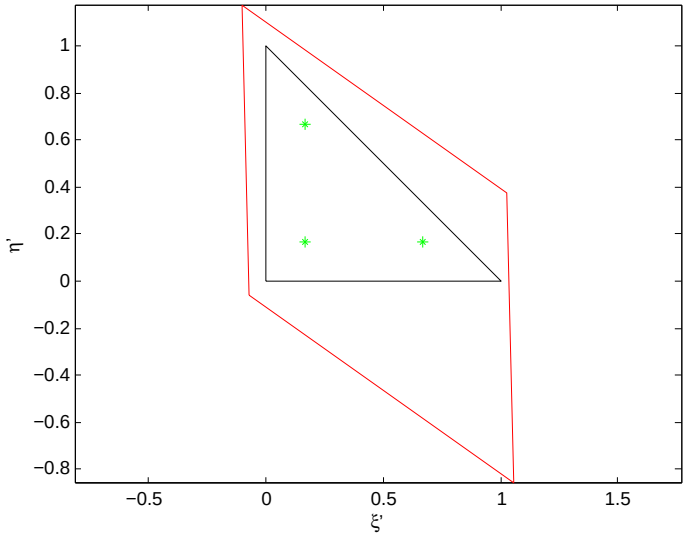


Figure 31: element and triangle transformed into coordinate system compatible with given Gauss-points

$$\frac{\partial x}{\partial \xi} = \frac{\partial \mathbf{N}_{el}}{\partial \xi}\mathbf{x}_{el} \tag{127}$$

$$\mathbf{J}_{\triangle} = \begin{bmatrix} \frac{\partial \xi}{\partial \xi'} & \frac{\partial \xi}{\partial \eta'} \\ \frac{\partial \eta}{\partial \xi'} & \frac{\partial \eta}{\partial \eta'} \end{bmatrix} \tag{128}$$

$$\frac{\partial \xi}{\partial \xi'} = \frac{\partial \mathbf{N}_{\triangle}}{\partial \xi'}\boldsymbol{\xi}_{\triangle} \tag{129}$$

Now all the tools needed for the numerical integration are available.

The stiffness matrix contribution from the triangle can be calculated as

$$\iint_{\triangle} \mathbf{B}_i^T(x,y)\mathbf{D}\,\mathbf{B}_j(x,y)\,t\mathrm{d}x\mathrm{d}y = \frac{1}{2}\sum_{k=1}^{n} \mathbf{B}_i(x,y)^T\mathbf{D}\mathbf{B}_j(x,y)tw_k|\mathbf{J}_{\triangle}|\,|\mathbf{J}_{el}| \tag{130}$$

Where $\mathbf{B}_i$ and $\mathbf{B}_j$ are either $\mathbf{B}_{std}$ or $\mathbf{B}_{enr}$, depending on what part of the stiffness matrix are currently being calculated. Examples using an arbitrary enrichment function ψ follows

$$\mathbf{B}_{std} = \begin{bmatrix} N_{1,x} & 0 & \dots & N_{4,x} & 0 \\ 0 & N_{1,y} & \dots & 0 & N_{4,y} \\ N_{1,y} & N_{1,x} & \dots & N_{4,y} & N_{4,x} \end{bmatrix} \tag{131}$$

$$\mathbf{B}_{enr} = \begin{bmatrix} \frac{\partial(\psi N_1)}{\partial x} & 0 & \dots & \frac{\partial(\psi N_4)}{\partial x} & 0 \\ 0 & \frac{\partial(\psi N_1)}{\partial y} & \dots & 0 & \frac{\partial(\psi N_4)}{\partial y} \\ \frac{\partial(\psi N_1)}{\partial y} & \frac{\partial(\psi N_1)}{\partial x} & \dots & \frac{\partial(\psi N_4)}{\partial y} & \frac{\partial(\psi N_4)}{\partial x} \end{bmatrix} \tag{132}$$

This is the target expression

$$N_{1,x} = \frac{\partial \xi}{\partial x}\frac{\partial N_1}{\partial \xi} + \frac{\partial \eta}{\partial x}\frac{\partial N_1}{\partial \eta} \tag{133}$$

For the enriched B-matrix

$$\frac{\partial(\psi N_1)}{\partial x} = N_1\frac{\partial \psi}{\partial x} + \frac{\partial N_1}{\partial x}\psi \tag{134}$$

The enrichment function can be chosen to make the derivative trivial. The shape function is given in isoperimetrical coordinates

$$\frac{\partial N_1}{\partial \xi} = \eta - 1 \tag{135}$$

The Jacobian can be used to access the derivatives of the shape functions w.r.t. x-y coordinates

$$\begin{bmatrix} \frac{\partial N_1}{\partial \xi} \\ \frac{\partial N_1}{\partial \eta} \end{bmatrix} = \mathbf{J}^T \begin{bmatrix} \frac{\partial N_1}{\partial x} \\ \frac{\partial N_1}{\partial y} \end{bmatrix} \tag{136}$$

Plug in the current gauss-point to extract the desired quantities

$$\begin{bmatrix} \frac{\partial N_1}{\partial x} \\ \frac{\partial N_1}{\partial y} \end{bmatrix} = \mathbf{J}(\xi_{gp}, \eta_{gp})^{T^{-1}} \begin{bmatrix} \eta_{gp} - 1 \\ \xi_{gp} - 1 \end{bmatrix} \tag{137}$$

B Program Manual

B.1 Introduction

This collection of files are meant to be used for eXtended Finite Element (XFEM) calculations. They are written in a syntax compatible with Matlab. Some of the files are required to run in tandem with files from the finite element package CALFEM, developed by the division for solid mechanics at LTH Lunds University. The XFEM files can be used individually or together as a pre-built program according to the flowchart found at the end of this document.

B.2 crackLvlSet

B.2.1 Purpose

Calculates ψ and ϕ element level set values.

B.2.2 Syntax

$$[lvlsCrack] = crackLvlSet(ex,\ ey,\ start,\ stop)$$

B.2.3 Description

lvlscrack = $[\psi_1\ \psi_2\ \psi_3\ \psi_4\ \phi_1\ \phi_2\ \phi_3\ \phi_4]$

ex and ey are element coordinates for the entire set of N elements

$$ex = \begin{bmatrix} x_1^1 & x_2^1 & x_3^1 & x_4^1 \\ & & \vdots & \\ x_1^N & x_2^N & x_3^N & x_4^N \end{bmatrix}$$

$$ey = \begin{bmatrix} y_1^1 & y_2^1 & y_3^1 & y_4^1 \\ & & \vdots & \\ y_1^N & y_2^N & y_3^N & y_4^N \end{bmatrix}$$

start and *stop* are coordinates for the first and second cracktip on the form $[x_{ct}\ y_{ct}]$

B.3 crackStiff

B.3.1 Purpose

Calculates element stiffness matrix with crack enrichments.

B.3.2 Syntax

$$[K_{crack}] = crackStiff(ex,\ ey,\ lvlset,\ cracktipXY,\ edof,\ D,\ thickness)$$

B.3.3 Description

Identifies type of enrichment from edof, partitions the element defined by ex and ey according to the lvlset vector, and calculates the element stiffness matrix;

$$ex = [x_1\ x_2\ x_3\ x_4]$$

$$ey = [y_1\ y_2\ y_3\ y_4] \tag{138}$$

$$\begin{aligned} lvlset(1:12) &= [not\ crack\ related] \\ lvlset(13:16) &= [\psi_1\ \psi_2\ \psi_3\ \psi_4] \\ lvlset(17:20) &= [\phi_1\ \phi_2\ \phi_3\ \phi_4] \end{aligned}$$

Corner level set values orthogonally/perpendicular (ψ/ϕ)to the crack edof(1:32) is reserved for other degrees of freedom.
Crack enrichment:

$$edof(34:41) = [Cdof_1\ Cdof_2\ ...\ Cdof_8]$$

Crack tip enrichment (4 types):

$$
\begin{aligned}
edof(42:49) &= [CTdof_1^1\ CTdof_2^1\ ...\ CTdof_8^1] \\
edof(50:57) &= [Ctdof_1^2\ ...\ CTdof_8^2] \\
edof(58:65) &= [CTdof_1^3\ ...\ CTdofs_8^3] \\
edof(66:73) &= [CTdofs_1^4\ ...\ CTdofs_8^4]
\end{aligned}
$$

D is the constitutive matrix *thickness* is plate thickness $cracktipXY$ is coordinates for both cracktips in the format

$$
\begin{bmatrix} x_{ct1} & y_{ct1} \\ x_{ct2} & y_{ct2} \end{bmatrix}
$$

B.4 crackStress

B.4.1 Purpose

Calculates element stresses from the displacements for an crack enriched element.

B.4.2 Syntax

$$[esx\ esy\ esxy] = crackStress(aEl,\ ex,\ ey,\ edof,\ lvlset,\ D,\ cracktipXY)$$

B.4.3 Description

Given element displacement (aEl), including those from enriched dofs, will calculate the diagonal values in the element stress matrix using the element corners as integration points. ex and ey are element coordinate vectors. edof is a list of dofs for the element, information relevant for cracks are kept in edof(34:72). lvlset are element corner level set values, lvlset(13:16)=[ψ_1 ... ψ_4], lvlset(17:20)=[ϕ_1 ... ϕ_4]. D is the constitutive matrix and cracktipXY=[CT_{1x} CT_{1y}; CT_{2x} CT_2y].

B.5 cutK

B.5.1 Purpose

Calculating the element stiffness matrix for an element on an interface.

B.5.2 Syntax

$$[Kuu\ Kua\ Kaa] = cutK(elx,\ ely,\ trix,\ triy,\ npnts,\ Dplus,\ Dminus,\ nodlvl)$$

B.5.3 Description

elx and ely are the element coordinates. trix and triy are coordinates for triangles, inside the element where each triangle has two points with the same sign on its level set value, x and y respectively one row for each triangle. npnts are the number of gauss points desired and nodlvl is the element level set values. Dplus and Dminus are the constitutive matrices for the positive and negative valued sides of boundary with respect to the level set value.

B.6 definelvlSet

B.6.1 Purpose

To calculate element level set data.

B.6.2 Syntax

$$[lvlSet\ edof] = definelvlSet(discs,\ ex,\ ey,\ edof)$$

B.6.3 Description

Input variable discs is a set up variable describing the number and types of discontinuities. One row for each discontinuity. First number in the row is the type of discontinuity.

type=1 - inclusion
if circular:
discs=[1 centre-xcoord centre-ycoord radius incl.E-modulus incl.pois-ratio]
if polygonal (user defined vertices)
discs=[1 vert1-x vert1-y vert2-x vert2-y ... vertn-x vertn-y incl.E-mod incl.pois-rat]
type=2 - void
if circular:
discs=[2 centre-x centre-y radius]
polygonal:
discs=[2 vert1-x vert1-y ... vertn-x vertn-y]
type=3 - bimaterial domain, line shaped interface
you need to supply two points on this line (POL1 POL2), endpoints <u>not</u> required. Supplied E-modulus and poissons ratio are for the side of the interface with negative level set values.
discs=[3 POL1x POL1y POL2x POL2y E-mod poissons-ratio]
discs=4 - crack
Two points in the domain that are the crack tips (CT1 CT2)
discs=[4 CT1-x CT1-y CT2-x CT2-y]

ex and ey are element coordinate matrices and edof is element dofs for a standard FEM case. Return variable lvlset is the element level set values distributed as follows:
lvlset(n,:)=level set values for element n.
lvlset(n,1:4)=ζ level set values for inclusions.
lvlset(n,5:8)=ζ level set values for voids.

lvlset(n,9:12)=ψ level set values for a bimaterial interface.
lvlset(n,13:16)=ψ level set values for cracks.
lvlset(n,17:20)=ϕ level set values for cracks

Returning edof is the standard edof expanded with the appropriate XFEM dofs. edof is expanded like so:
edof(n,10:17) - inclusion dofs for element n
edof(n,18:25) - void dofs
edof(n,26:33) - bimaterial dof
edof(n,34:41) - crack dofs
edof(n,42:49) - cracktip enrichment type 1 dofs
edof(n,50:57) - cracktip enrichment 2 dofs
edof(n,58:65) - cracktip enrichment 3 dofs
edof(n,66:73) - cracktip enrichment 4 dofs

B.7 findXDOF

B.7.1 Purpose

Calculate extra dofs for extension of a element degrees of freedom list for a closed boundary. (used inside defineLvlSet to extend edof)

B.7.2 Syntax

$$[edofx] = findXDOF(edof,\ nrdof,\ nodlvl)$$

B.7.3 Description

edof is the standard edof matrix suitable for CALFEM.
nrdof is the highest numbered degree of freedom in the standard edof.
nodlvl is the element level set values.

B.8 polyLvlSet

B.8.1 Purpose

Calculates ζ level set values for a closed boundary defined by a series of vertices. (This function is called within *definelvlSet*).

B.8.2 Syntax

$$[polyLvlSet] = polyLvlset(verts,\ ex,\ ey)$$

B.8.3 Description

polyLvlSet(n,:)=[ζ_1 ζ_2 ζ_3 ζ_4], for element # n
The code approximates the boundary as the straight line between two neighbouring vertices. Therefore tighter placement of vertices gives better results.

B.9 remap

B.9.1 Purpose

Equation solver for backwards element mapping, global coordinates -> isoparametric element coordinates. Convenient when doing numeric integration for an element split up into triangles and element coordinates for triangle corners (which are given in global coordinates) are needed.

B.9.2 Syntax

$$[xi\ eta] = remap(ex,\ ey,\ x,\ y)$$

B.9.3 Description

ey and ex are the element coordinates (y and x respectively).
x and y are the user wants translated into element coordinates.
The returned xi and eta are the element coordinates ξ & η.

B.10 trisplit

B.10.1 Purpose

Splits an element into triangles with uniform level set values.

B.10.2 Syntax

$$[tx\ ty] = trisplit(ex,\ ey,\ lvlSet);$$

B.10.3 Description

If the element is cut by the interface boundary described by the element level set values supplied by lvlset this function will split the element, defined by the element coordinate arrays ex and ey, into triangles where every point on each triangle have the same sign on its level set value.

B.11 weightedCrackDispl

B.11.1 Purpose

Superimposes XFEM crack dofs displacement with correct weight factor on corresponding standard FE dofs.

B.11.2 Syntax

$$[atot] = weightedCrackDispl(a, edof, lvlset, ex, ey, CT)$$

B.11.3 Description

a are entire XFEM system resulting displacements, no weights applied.
edof are the list of element degrees of freedom.
ex and ey are lists of element coordinates, x and y respectivley.
CT are cracktip coordinates, [CT1x CT1y; CT2x CT2y].

B.12 XFEMcutstress

B.12.1 Purpose

Calculates element stresses for a element split by the discontinuity, as well as a new set of smaller elements derived from the original each with a consistent level set sign sometimes useful for post processing.

B.12.2 Syntax

$[exnew\ eynew\ es] = XFEMcutstress(tx,\ ty,\ displ,\ ex,\ ey,\ nodlvl,\ edof,\ Dplus,\ Dminus)$

B.12.3 Description

tx and ty are triangles combining to make up the element. This new element division is needed for the numeric integration.
displ is the element displacements, including enriched dofs.
ex and ey are the element coordinates.
nodlvl and edof are the element- level sets / dofs respectively, consistent with **definelvlSet**.
D_plus and D_minus are the constitutive matrices for the positive and negative level set sides of the discontinuity interface.
exnew and eynew are new elements created for post processing purposes, no new elements are cut by the discontinuity interface.
es is the element stress matrix. Each new element gets one row for each type of calculated stress (for example y direction), the 4 columns represent the four corner on the elements.
Example:

$$es = \begin{bmatrix} \sigma_x^{el_1} & & & \\ \sigma_y^{el_1} & \dots & \dots & \dots \\ \sigma_{xy}^{el_1} & & & \\ & & \vdots & \\ & \sigma_x^{el_n^{corner\ 2}} & & \\ \dots & \dots & \sigma_y^{el_n^{corner\ 3}} & \dots \\ & & & \sigma_{xy}^{el_n^{corner\ 4}} \end{bmatrix}$$

B.13 XFEMelstress

B.13.1 Purpose

Calculates element stresses for a enriched element, not cut by the ζ set = 0 interface.

B.13.2 Syntax

$$[es] = XFEMelstress(displ,\ ex,\ ey,\ nodlvl,\ edof,\ Dplus,\ Dminus)$$

B.13.3 Description

ex and ey are the element coordinate vectors.
edof is the element dofs vector, including enriched ones following the convention from **definelevelSet**.
displ are the relevant displacements, including those from enriched nodes.
nodlvl are the element level set values, also following **definelvlSet** conventions.
D_{plus} and D_{minus} are the constitutive matrices for the positive and negative level set side respectively.
Returns element corner stresses es, one column per element corner one row per type of stress:
es(:,1) - stresses for corner 1 (southwest). etc.

B.14 XFEMgeom

B.14.1 Purpose

To calculate element coordinate and degrees of freedom matrices.

B.14.2 Syntax

$$[ex\ ey\ edof] = XFEMgeom(domain,\ elSize)$$

B.14.3 Description

Given the set up parameters domain (number of elements in x and y direction respectively) and elSize (length of a square elements side) this function will number all elements in the domain and return list of all elements corner x-coordinates, y-coordinates and what standard FEM degrees of freedom each element contain. The corners are ordered like so: [Southwest Southeast Northeast Northwest]. Dofs follow the same system with the x-direction being counted before the y-direction. All returnedmatrices will have one row for each element.

$$ex = \begin{bmatrix} \vdots & & & \\ x_1 & x_2 & x_3 & x_4 \\ \vdots & & & \end{bmatrix}$$

$$ey = \begin{bmatrix} \vdots & & & \\ y_1 & y_2 & y_3 & y_4 \\ \vdots & & & \end{bmatrix}$$

$$edof = \begin{bmatrix} \vdots & & & & & & & & \\ elnumr & dof_1 & dof_2 & dof_3 & dof_4 & dof_5 & dof_6 & dof_7 & dof_8 \\ \vdots & & & & & & & & \end{bmatrix}$$

B.15 XFEMload

B.15.1 Purpose

Creates the load and boundary condition vectors.

B.15.2 Syntax

$$[f\ bc] = XFEMload(F,\ BC,\ edof,\ domain,\ lvlSet)$$

B.15.3 Description

F(1) is the location of the load on the rectangular domain
1 - west
2 - south
3 - east
4 - north
F(2) is the total magnitude of the load.
BC is the location where the domain is locked in place, follows the same convention as F(1)
edof is the element degrees of freedom and lvlset is the element level set values, used in case enriched elements also are loaded/have locked dofs.
domain(1) - number of elements in vertical direction.
domain(2) - number of elements in horizontal direction.

B.16 XFEMplot

B.16.1 Purpose

Automated post processing.

B.16.2 Syntax

$$[ex2\ ey2\ es] = XFEMplot(a,\ ex,\ ey,\ edof,\ lvlSet,\ plots,\ Discs,\ parameters)$$

B.16.3 Description

a- total system displacements, including those from enriched dofs.
ex and ey - element coordinates.
edof - element dofs, including enriched dofs as described in *definelvlSet*.
lvlset - element level set values.
plots - series of integers each symbolising a type of figure the user wants ploted:
1 - undeformed geometry, including boundary illustration.
2 - deformed geometry, auto scaled for visibility.
3 - Stresses in x direction.
4 - Stresses in y direction.
5 - Shear Stresses
6 - Von Mises stresses.
Discs and parameters are discontinuity and domain data as used in *SetUp*.
This function calls an appropriate sub-function to calculate stresses before ploting, returns these as es for further processing.
ex2 and ey2 are new element coordinates. The function creates these, when appropriate, to ensure elements with uniform level set value signs. Sometimes useful for calculating stresses.

B.17 XFEMstiffness

B.17.1 Purpose

Returns a combined stiffness matrix, appropriate to what types of discontinuities are present.

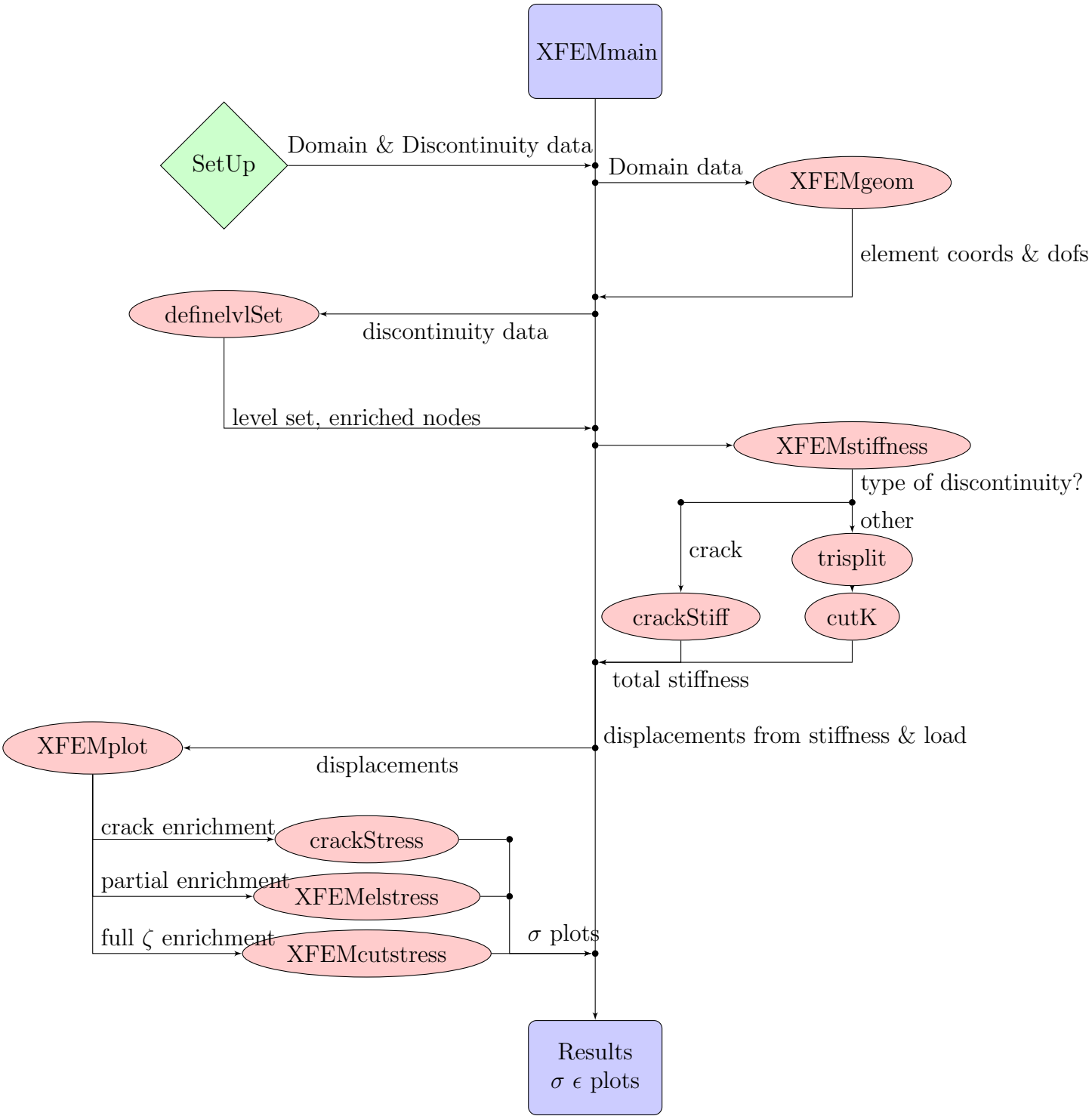
B.17.2 Syntax

$$K = XFEMstiffness(ex,\ ey,\ Discs,\ lvlSet,\ edof,\ parameters)$$

B.17.3 Description

Using the level set values in lvlset to determine what type of discontinuity (or lack thereof) an element have. Calls the appropriate subfunction and using the element degrees of freedom list edof to insert the subfunction supplied element stiffness matrix in the correct place in the system stiffness matrix
ex and ey are element coordinatematrices. Discs and parameters are set up variables discussed in more detail in the description for the file *SetUp*.

B.18 Program Flowchart



References

[1] S. Osher, J. Sethian.
Fronts Propagating with Curvature-Dependent Speed: Algorithms Based on Hamilton-Jacobi Formulations.
Journal of Computational Physics vol 79, 1988.

[2] S. Bordas, A. Legay.
XFEM Mini-Course.
Ecole Polytechnique Féderale de Lausanne, 2005.

[3] N. Sukumar, D.L. Chopp, N Moës, T. Belytschenko.
Modeling Holes and Inclusions by Level Sets in the Extended Finite-Element Method.
Computer Methods in Applied Mechanics and Engineering vol 190, 2001.

[4] J. Melenk, I. Babuska.
The Partition of Unity Finite Element Method: Basic Theory and Application.
Computer Methods in Applied Mechanics and Engineering vol 139, 1996.

[5] N. Moës, M. Cloirec, P. Cartraud, J.F. Remacle.
A Computational Approach to Handle Complex Microstructure Geometries.
Computer Methods in Applied Mechanics and Engineering vol 192, 2003.

[6] T. Belytschko, T. Black.
Elastic Crack Growth in Finite Elements with Minimal Remeshing.
International Journal for Numerical Methods in Engineering vol 45, 1999

[7] M. Fleming, Y.A. Chu, B. Moran, T. Belytschko.
Enriched Element-free Galerkin Methods for Crack Tip Fields.
International Journal for Numerical Methods in Engineering vol 40, 1997

[8] M. Pais.
MATLAB Extended Finite Element (MXFEM) Code v1.3.
www.matthewpais.com, 2011

[9] ISO 7539-6 Corrosion of metals and alloys - Stress corrosion testing - Part 6: Preparation and use of pre-cracked specimens for tests under constant load or constant displacement. 2nd Ed. 2003

[10] B.Cotterell, J.R. Rice.
Slightly Curved or Kinked Cracks.
International Jurnal of Fracture vol 40, 1980

[11] M. Patricio, R. Mattheij.
Crack Propagation Analysis.
CASA report 07-03, 2007.

[12] A. Carpinteri, L.P. Pook, L. Susmel, S. Vantadori.
Foreward: Fatigue Crack Paths 2012 (CP 2012)
International Journal of Fatigue, vol 58, 2014