CRACK PROPAGATION MODELING IN PLANE STRUCTURES USING TWO-SCALE GENERALIZED/EXTENDED FINITE ELEMENT METHOD

Mohammad Malekan

FEDERAL UNIVERSITY OF MINAS GERAIS (UFMG)
SCHOOL OF ENGINEERING
GRADUATE PROGRAM IN STRUCTURAL ENGINEERING (PROPEES)

"Crack Propagation Modeling in Plane Structures Using Two-scale Generalized/Extended Finite Element Method"

Mohammad Malekan

Thesis submitted to the Graduate Program in Structural Engineering (PROPEEs) of Federal University of Minas Gerais (UFMG), as a requirements for obtaining the title "Doctor of Philosophy in Structural Engineering".

PhD Committee:

Prof. Dr. Felicio Bruzzi Barros
DEES - UFMG (Adviser)

Prof. Dr. Roque L.S. Pitangueira
DEES - UFMG (Internal Examiner)

Prof. Dr. Gabriel de Oliveira Ribeiro
DEES - UFMG (Internal Examiner)

Prof. Dr. Paulo de Tarso Rocha Mendonca
MEC - UFSC (External Examiner)

Prof. Dr. Eduardo Alberto Fancello
MEC - UFSC (External Examiner)

Belo Horizonte, August 7^{th}, 2017

Malekan, Mohammad.
M245c Crack propagation modeling in plane structures using two-scale generalized/extended finite element method [manuscrito] / Mohammad Malekan. - 2017.
xvi, 118 f., enc.: il.

Orientador: Felicio Bruzzi Barros.

Tese (doutorado) Universidade Federal de Minas Gerais, Escola de Engenharia.

Anexos: f. 95-110.

Bibliografia: f.111-118.

1. Engenharia de estruturas - Teses. 2. Método dos elementos finitos - Teses. 3. Mecânica da fratura - Teses. 4. Programação orientada a objetos (Computação) - Teses. I. Barros Felício Bruzzi. II. Universidade Federal de Minas Gerais. Escola de Engenharia. III. Título.

CDU: 624(043)

Abstract

Finite Element Method (FEM) has been widely used for the numerical modeling of structural/mechanical problems. Use of computer-based FEM programs was greatly facilitated with the development of pre- and post-processors rich interactive graphics capabilities, allowing users with basic knowledge of geometry to easily work with them. However, modeling of discontinuous fields with a standard finite element approximation presents challenges like restrictions on the finite element mesh to align with the discontinuity and the need for remeshing as the discontinuity evolves. The generalized or extended FEM (G/XFEM) was proposed as a numerical method to solve some of these challenges. The G/XFEM method enriches the standard finite element shape functions locally with enrichment functions which are based on the physics associated with the problem.

The goal of this thesis is to fracture modeling in thin-walled structure, specifically *Plate* structures, by extending the available capabilities of the G/XFEM method implemented in INSANE (*INteractive Structural ANalysis Environment*) in-house code, a computational environment developed by the Department of Structural Engineering (DEEs) at the Federal University of Minas Gerais (UFMG), which has been implemented using Object Oriented Programming (OOP). A stable version of G/XFEM is implemented to have a well-conditioning systems of equations. Then, the crack propagation strategy is applied to plane stress/strain and Reissner-Mindlin problems using classical and two-scale G/XFEM. These whole implementations and design are explained in detail and their robustnesses and accuracies are examined by solving various structural problems.

Keywords: *Generalized/Extended Finite Element Method (G/XFEM)*, *Reissner-Mindlin Plate*, *Fracture Mechanics*, *Object-Oriented Programming (OPP)*, *Two-scale Analysis*

Resumo

O Método dos Elementos Finitos (MEF) tem sido amplamente utilizado para a modelagem numérica de problemas estruturais/mecânicos. O uso de programas baseados em MEF foi grandemente facilitado com o desenvolvimento pré e pós-processadores ricos em recursos gráficos interativos, permitindo aos usuários com conhecimento básico de geometria trabalhar facilmente com eles. No entanto, a modelagem de campos descontínuos com uma aproximação de elementos finitos padrão apresenta desafios como restrições na malha de elementos finitos para alinhar com a descontinuidade e a necessidade de remalhar à medida que a descontinuidade evolui. O MEF generalizado ou estendido (MEFG/X) foi proposto como um método numérico para resolver alguns desses desafios. O método MEFG/X enriquece localmente as funções de forma padrão de elementos finitos com funções de enriquecimento que se baseiam na física associada ao problema.

O objetivo desta tese é a modelagem de fratura em estruturas de paredes finas, especificamente estruturas *Placas*, estendendo as capacidades disponíveis do método MEFG/X implementado no ambiente INSANE (*INteractive Structural ANalysis Environment*), uma plataforma computacional desenvolvida no Departamento de Engenharia de Estruturas (DEEs) da Universidade Federal de Minas Gerais (UFMG), totalmente implementada utilizando programação orientada a objetos. Uma versão estável do MEFG/X é implementada para se ter sistemas de equações bem condicionados. Em seguida, a estratégia de propagação de fissuras é aplicada a problemas de tensão/deformacão plana e de placas de Reissner-Mindlin, usando o MEFG/X clássico e na versão global-local. Todas essas implementações são explicados em detalhes e a robustez e precisão são examinadas pela resolução de vários problemas estruturais.

Palavras-Chave: *Método dos Elementos Finitos Generalizados ou Estendido*, *Placa Reissner-Mindlin*, *Mecânica da fratura*, *Programação Orientada a Objetos*, *Estratégia duas escalas*

Acknowledgment

Thank you Almighty Allah, Who is merciful and gracious, for everything I have in my life. A very special thanks to my beloved wife *Maryam*, for her unconditional love and braveness to accompanying me with this long and hard journey in a foreign land. To my parents and siblings for their constant pray, kindness, and encouragement in all journeys of my life. I could have never imagined myself where I am at this moment without their endless support.

It's almost impossible to list all the people who gave their contribution to the current research. Therefore, only a few people who are more directly linked to this thesis are mentioned here and for others, I deeply express my gratitude for their help and contributions. I would like to thank my adviser Prof. Dr. Felicio Bruzzi Barros for giving me the opportunity to work with him as a PhD student and learn too many things with our discussion on different topics. I also appreciate the guidance from my PhD committee members, Prof. Dr. Roque L.S. Pitangueira, Prof. Dr. Gabriel de Oliveira Ribeiro, Prof. Dr. Paulo de Tarso Rocha Mendonca, and Prof. Dr. Eduardo Alberto Fancello. In addition, I want to thank other professors at the the DEEs department for their support and help during these last three and a half years, Prof. Dr. Carlos Alberto Cimini Jr., Prof. Dr. Samuel S. Penna, Prof. Dr. Ramon P. Silva, and Prof. Dr. Estevam B. de Las Casas.

To the members of the INSANE laboratory in DEEs-UFMG, Anderson, Anelize, Debora, Leandro, Lapo, and Marcella. Thank all of you guys for sharing this amazing experience with me. I wish you all the best in your future.

I also gratefully acknowledge the fully and partially financial supports from the Brazilian research agencies CAPES (Coordination for the Improvement of Higher Education Personnel), CNPq (National Council for Scientific and Technological Developments) and FAPEMIG (Research Support Foundation of the State of Minas Gerais).

To my parents and my lovely wife.

Table of Contents

List of Tables

List of Figures

List of Abbreviations and Acronyms

BC	*Boundary Condition*
CNPq	Conselho Nacional de Desenvolvimento Cientfico e Tecnolgico
DEES	Departamento de Engenharia de Estruturas
DOF	*Degree of Freedom*
FAPEMIG	Fundao de Amparo Pesquisa do Estado de Minas Gerais
FEM	*Finite Element Method*
GL	*Global-Local*
GP	*Gauss Point*
G/XFEM	*Generalized/Extended Finite Element Method*
G/XFEM^{gl}	*Generalized/Extended FEM with GL strategy*
INSANE	*INteractive Structural ANalysis Environment*
LEFM	*Linear Elastic Fracture Mechanics*
OOP	*Object-Oriented Programming*
P2	*Polynomial enrichment function of order 2*
PCCG	*Preconditioning Conjugate Gradient*
PU	*Partition of Unity*
PUM	*Partition of Unity Method*
SG/XFEM	*Stable Generalized/Extended Finite Element Method*
SIF	*Stress Intensity Factor*
UFMG	Universidade Federal de Minas Gerais
UML	*Unified Modeling Language*
XML	*eXtensible Markup Language*

List of Symbols

$\boldsymbol{C}$	Stiffness matrix
$\boldsymbol{D}$	Vector of forces
$\boldsymbol{D}_h$	Hook's tensor
h_j	Scale factor (Diameter of the bigger element in the cloud of node j)
L_{ji}	Local approximation function
$\mathcal{N}_j$	FE Shape function
$\mathbb{R}^2$	Bi-dimensional domain
$\tilde{\boldsymbol{u}}(\boldsymbol{x})$	G/XFEM displacement approximation
$\boldsymbol{b}_{ji}$	Nodal parameters associated with G/XFEM
$\boldsymbol{X}$	Vector of degrees of freedom
$\boldsymbol{X}^t_G$, $\boldsymbol{X}^t_L$	Space of FEM or G/XFEM functions in global and local domains
$\boldsymbol{X}^t_{GL}$	Space of FEM or G/XFEM functions in third GL step
η	Coefficient of Penalty method
ϕ_{ji}	G/XFEM shape function
κ	Spring stiffness in Cauchy condition
Ω_G, Ω_L	Global and Local domains
Γ_G, Γ_L	Boundaries of the global and local domains
Γ^σ_G	Global boundaries where the Neumann condition is applied
Γ^u_G	Global boundaries where the Dirichlet condition is applied
$\boldsymbol{u}^t_G$, $\boldsymbol{u}^t_L$	Solution of the initial global and local problems at time step t
θ	Angle between x coordinate and direction of the crack front
r_m	Scalar multiplier for interaction energy integral
h_{elem}	Element characteristic length
Γ_c	Discontinuity boundaries
$\phi(\boldsymbol{x})$	Signed distance function

$H(\boldsymbol{x})$	Heaviside function
E	Young's modulus
ν	Poisson ratio
Δa	Crack increment length
K_I, K_{II}, K_{III}	Mode-I, II and III stress intensity factors
W	Strain energy density
σ_{ij}, ϵ_{ij}	Stress and strain tensors
σ	Cauchy stress tensor
δ	Kronecker delta
I	Interaction integral
J	J-integral
Ω, $\partial\Omega$	Problem domain and boundary
$\partial\Omega_t$, $\partial\Omega_u$	Surface traction and displacement boundaries
$\boldsymbol{b}$	Body force vector
$\bar{\boldsymbol{t}}$	External traction vector
ε	Linear strain tensor
$\hat{\boldsymbol{n}}$	Unit outward normal
q	Weighting function
$\boldsymbol{m}_j, n_j$	Outward unit normal vectors in contour integral
C_+, C_-	Upper and lower crack surfaces
Γ, C_0	Inner and outer J-integral boundaries
M	Bending moment
Q	Shear load
w	Transverse displacement of the plate
ψ	Section rotation of the plate middle plane
ϵ_b, ϵ_s	Bending and shear strains
$\bar{\boldsymbol{u}}$	Prescribed displacement
q_j	A set of linearly independent functions defined at each nodal cloud
μ	Shear modulus
$\varkappa$	Material constant
w_{gp}	Weight of each Gauss point
$detJ$	Determinant of the Jacobian
N_{gp}	Number of Gauss points

Chapter 1

INTRODUCTION

1.1 Literature

Many of natural phenomena can be described by mathematical models based on differential equations. In engineering, particularly the use of this feature allows an understanding of several problems: solid mechanics, fluid mechanics, electro-magnetics, heat flow, and similar problems. The common feature of these problems is using Partial Differential Equations (PDEs) relating field variables within a given domain beside to meet boundary conditions on its boundaries. Numerical solution of PDEs having discontinuities is important, since it is applicable to a wide range of solid mechanics problems: such as modeling the crack propagation, dislocations and material inclusions.

Finite Element Method (FEM) has been widely used for the numerical modeling of structural problems (Hughes, 2000). With the advent and popularization of high-performance computers, the FEM has gained more space over the past two decades, specifically for special technique aiming to reduce the computational costs either with mesh reduction technique or using parallel processing approach. The application of the FEM, in the various branches of engineering, requires constant improvement to treat nonlinear problems, material behavior and geometric changes, or even the spread of micro or macro defects throughout the area. In addition, the use of computer-based FEM programs was greatly facilitated with the development of pre- and post-processors. Rich interactive graphics capabilities, allowing users with basic knowledge of geometry to easily work with them. On the other hand, there are phenomena that FEM can not satisfactorily describe, raising the development of new strategies for this purpose. One of them are the presence of the discontinuities in the domain, such as cracks, micro-cracks, voids, inclusions, etc. The main focus of early implementations of Finite Element (FE) models for discontinuity problems was defining meshes conformed to the discontinuity surface(s) (Ingraffea and Saouma, 1985; Cook et al, 2002). This constraint brings two issues: the first challenging issue is the creating of a mesh compatible with the discontinuity surfaces, and the second one is the inaccurate solutions due to high mesh-dependency (de Borst et al, 1993). Meshless methods, among them the smooth particle hydrodynamics

(Monaghan, 1982), reproducing kernel particle method (W. K. Liu and Zhang, 1995), element free Galerkin method (Belytschko et al, 1993) and *hp*-cloud method (Duarte and Oden, 1996) were proposed aiming to avoid this remeshing by defining approximations built on a set of nodes. The mesh independence, however, is accomplished by establishing non-analytic approximations, that demand a higher computation cost and that is unable to satisfy the Kronecker delta property (necessary to directly impose the boundary conditions).

1.1.1 Partition of Unity-based methods for fracture mechanics

Another approach, so-called continuous/discontinuous FEMs, has been developed (Ortiz et al, 1987; Belytschko and Black, 1999) in oder to overcome the draw-backs related to modeling discontinuity problems using the classical FEM, in which specific functions have been added to the classical FE approximations in order to capture discontinuities. These specific functions are selected in order to represent the discontinuous behavior of the problem studied. This helps to model the discontinuity in any part for the model, without any need of either remeshing or conforming mesh at the discontinuity boundaries.

Two families of enrichment strategies can be distinguished here: first is the so-called embedded FEMs (EFEMs) and second is based on the Partition of Unity (PU) approach. In the EFEM, the enrichment functions are defined on the local enhanced degrees of freedom (DOFs) for each element and these enhanced DOFs are removed by static condensation prior to the global tensor assembly (Ortiz et al, 1987). Therefore, total number of DOFs remains unchanged. However, this approach has some limitations, such as: (1) using the elemental enrichments that are defined on the internal DOFs corresponding to the jump over discontinuity surfaces, which leads to a non-conforming formulation in which the compatibility of strain fields is not satisfied and it is only enforced in a weak sense (due to discontinuous enrichment over element surfaces and its boundaries); (2) lack of the kinematic decoupling due to connectivity of the strains on both sides of the discontinuity surface (in elements that are crossed by the discontinuity surface); and (3) the uniqueness and numerical robustness of solutions, because of some numerical problems due to use some special approaches to eliminate the additional DOFs from the model.

The partition of unity concept uses the nodal enrichment as the so-called special FEM approach (Babuška et al, 1994). In this approach, the discontinuity surfaces are modeled by enriching the polynomial functions with special functions that are associated with additional parameters, called *enriched DOFs*. These enriched DOFs are added globally to the discretized system; therefore, they increase the total number of DOFs with contrast to the EFEM method. Generalized or eXtended Finite Element Method (G/XFEM) (Strouboulis et al, 2000a, 2001; Belytschko and Black, 1999), a partition of unity-based method (PUM), is an example of the method that uses the local enrichments. In these methods, the local enrichments have been used in sub-domains around discontinuities and special numerical integration algorithms have been utilized for cells intersected by discontinuities.

A lot of experiments has been developed, showing the success of the before-mentioned technique in solving complex problems and pointing to future studies in the field of computational parallelization, as discussed in Pereira et al (2011) and Kim et al (2011). Its performance in terms of convergence rate is already well established in the literature, allowing advances in solving problems in elastoplastic media, partly fragile, or metal forming problems for large localized deformations.

Three important advantages can be identified for these mesh-based formulations when they are compared to the meshless methods. Firstly, the strategy of extrinsically enrich the PU Method (PUM) presents a lower computation cost. Following, the interpolation characteristics of the approximate functions produce solutions that satisfy the Kronecker delta property. Finally, the similarity between PUM/GFEM/XFEM and FEM can allow a straightforward migration for these new approaches and also the reuse of the FEM structure. From the computational point of view, both the migration and reusing aspects depend, of course, on how the FEM code is implemented.

On the other hand, the enrichment strategy of G/XFEM can produce a badly conditioned stiffness matrix, which leads to severe loss of accuracy in the numerical solution. In this case, preconditioners should be used such as the ones proposed by Béchet et al (2005) and Menk and Bordas (2011). Babuška and Banerjee (2011, 2012) presented a new approach, named Stable G/XFEM (SG/XFEM), which can overcome such problems and keeps the good properties of conventional G/XFEM. Another advantage is that the SG/XFEM does not use ramp-functions in the transitory elements between regions of different kinds of enriched functions as proposed by Fries (2008). These are the so-called blending-elements, considered by the first time by Chessa et al (2003). In such work it is shown that the presence of blending elements can negatively affect the accuracy and the rate of convergence of the method. Two strategies aiming to overcome this problem are described, the first based on the Hu-Washizu variational principle and the other one through a enhanced strain field. Other works has presented further developments at this subject such as (Fries and Belytschko, 2006; Gracie et al, 2008; Tarancón et al, 2009).

1.1.2 Crack propagation Modeling using G/XFEM

Initial implementation of the G/XFEM for crack propagation problems was introduced by Belytschko and Black (1999) and Moës et al (1999). After that, this method has been extensively used for the simulation of crack propagation problems, such as: three-dimensional crack propagation was proposed by Duarte et al (2001); a quasi-static crack growth was proposed by Sukumar and Prévost (2003) using G/XFEM; three-dimensional modeling of initiation, branching, growth of crack in non-linear solids including statics and dynamics problems was presented by Bordas et al (2008); Rabczuk et al (2010) proposed some crack tracking techniques for three-dimensional problems using partition of unity and meshfree technique; and an enhanced G/XFEM method for modeling of dynamic crack branching presented by Xu et al (2014).

Dolbow et al (2000b) modeled the fracture in Reissner-Mindlin plate with XFEM and computed the mixed-mode stress intensity factor using the domain forms of interaction integral. Subsequently, different researchers have used G/XFEM formulations based on Reissner-Mindlin plate theory to develop fracture modeling in shell and plate structures, see for example (Barcellos et al, 2009; Lasry et al, 2010; Mendonca et al, 2011; Zeng et al, 2016). Areias and Belytschko (2005) established a new formulation for cracked shells with the G/XFEM and an enhanced strain formulation is used to attenuate the locking in thin structures with a Reissner-Mindlin four-node shell element. In these two researches, a 3D '*solid-like*' shell based models were used to model the crack propagation. Then, they proposed a quadrilateral Kirchhoff-Love shell element and the overlapped paired elements method with reduced integration for thin shells in (Areias et al, 2006). The aforementioned works are presented for the static problems. Studies with regard to the dynamic growth of crack in the shells using G/XFEM is limited in literature. Song and Belytschko (2009) developed a phantom nodes method for the a special shell element by modifying the displacement formulation of the standard G/XFEM to model crack propagation in shell structures. An introduction of G/XFEM into the continuum-based shell element and modeling of crack growth in the shells is presented in (Zhuang and Cheng, 2011; Zheng et al, 2014, 2015). Larsson et al (2011) used a cohesive-zone-based concept applied to a kinematically consistent shell model enhanced with an XFEM-based discontinuous kinematical representation in order to model fractures in shell structures. An extension of this method for elastoplastic analysis is presented in (Mostofizadeh et al, 2013). A geometrically nonlinear discontinuous solid-like shell element with phantom node method was proposed by (Ahmed et al, 2012).

1.1.3 Global-Local methodology for large-scale problems

In most structures, the nonlinear damages mainly occurs in some local regions. The correlation between the local damage and the global structural behavior is helpful to predict clearly the failure mechanism. Thus, an efficient numerical method is necessary to simulate the relation of the global behavior and local damage. Among the different analysis methodologies that have high potential for the accurate prediction of detailed stress distribution in structures, are the global-local methodologies. These methods are basically hybrid modeling and/or analysis techniques. Global-local FEM was proposed by Noor (1986) in order to solve non-linear problems. This method was presented after zooming method proposed by Hirai et al (1985). A local problem is defined where a local phenomenon happens. The global-local FEM approach has two steps. The first step is done with a coarse FEM mesh that ignores the effect of the local phenomena. This is followed by the second step which includes an analysis of the local region using refined finite element meshes. The key parameter for the local analysis is the application of field state variables as boundary conditions on the local boundaries. Once the solution of local problem is obtained, the global-local FEM analysis can be finished. In global-local G/XFEM (G/XFEM^{gl}) (Duarte et al, 2007), a variation of the standard G/XFEM, enrichment functions are constructed numerically from the solution

of a local problem. G/XFEM^{gl} approach has three steps, its first and second steps are the same as the global-local FEM. In the third step, the results of the local problem are used to enrich the global problem which improves the approximate solution. In the case of fracture mechanic problems, the stress field around the crack tip presents high gradients and it is discretized using a large number of elements. The great advantage is providing a well-refined description of the local problem. This approach allows to solve the problem using coarse global meshes around crack fronts.

The global-local strategy based on G/XFEM approach is applied to high-cycle fatigue crack growth in 3D bodies by Pereira et al (2011). Their coarse-scale mesh in the G/XFEM^{gl} doesn't need to model the crack surface explicitly. Instead, the cracks are modeled through global-local enrichment functions. A two-scale approach using the G/XFEM^{gl} applied to multi-site cracking problems was presented by Evangelista et al (2013) where realistic boundary conditions are applied and multiple cracks with different geometries in a three-dimensional airfield slab are considered. Plews and Duarte (2014) used an interdependent solution of global and local problems in order to resolve multi-scale effects due to fine-scale heterogeneities under G/XFEM strategy. Although there are many investigations on two-scale/multi-scale analysis of fracture problems using G/XFEM method, all of them were done only for three-dimensional problems, while there is no thorough study for Reissner-Mindlin plate problems.

1.1.4 Object-oriented programming G/XFEM codes

The viability of extending a traditional FEM code by adding the G/XFEM enrichment strategy has already been demonstrated by Strouboulis and co-workers in the remarkable papers (Strouboulis et al, 2000a, 2001). Using an object-oriented strategy, the necessary modifications are outlined from a software engineering perspective and an abstract structure of a G/XFEM framework is presented. Beyond the general ideas about the migration from FEM to other numerical methods such as PUM/GFEM/XFEM, Sukumar and Prévost (2003) describe details of its implementation within a general-purpose finite element code written in `Fortran`.Emphasis is placed on the design of such a code to enable the modeling of discontinuous phenomena by using G/XFEM. Several of the advances about this topic can be found in the overview presented by Fries and Belytschko (2010).

Application of object-oriented programming for FEM has been receiving great attention over the last two decades (Mackie, 2009). The object-oriented codes can be slower than structured language codes, mainly because of their size and the way they manage their data. However, improvements in computational tool performances have quite solved the speed problem. Some main concepts of OOP that makes it easy to use and maintenance are: *objects* and *classes*, *data abstraction and encapsulation*, *inheritance*, and *polymorphism*. Here are some OOP based FEM codes: a FE analysis to solve structural problems using OOP approach within: Object NAP code (Forde et al, 1990), OOFEM (Zimmermann et al, 1992), and a

FE code (Patzák and Bittnar, 2001); a finite element differential equations analysis library (Bangerth et al, 2007); an object-oriented environment to solve multidisciplinary problems (combination of thermal, fluid dynamics, and structural different fields) (Dadvand et al, 2010); implementation of a unified library of nonlinear solution schemes in FE programming scheme (Leon et al, 2011). Beside this, the OOP has been successfully used to represent also different numerical methods, such as the boundary element method (Lage, 1998) and meshfree methods (Barbieri and Meo, 2012). Also, a bunch of G/XFEM codes used object-oriented concept as their implementation strategy: an extension of a FEM code by adding the G/XFEM enrichment strategy (Strouboulis et al, 2000b, 2001); demonstration of an open source architecture for G/XFEM (Bordas et al, 2007); an extension of an original FEM code (Chamrová and Patzák, 2010); implementing a G/XFEM code from scratch (Dunant et al, 2007); automated meshing for integrated experiments project proposed by Dunant (2009); and a generic toolbox for finite element methods in `C++` (GetFEM++ library) (Renard and Pommier, 2017). An available FEM programming environment was expanded to enclose the standard version of G/XFEM in (Alves et al, 2013). This environment, so called INSANE (Interactive Structural Analysis Environment) is an open source software available at `http://www.insane.dees.ufmg.br`, and it was firstly proposed in (Fonseca and Pitangueira, 2007). Afterwards, the G/XFEM method was extended to have numerically-built enrichment function within the so-called global-local G/XFEM (G/XFEM^{gl}) method in (Malekan et al, 2016a, 2017b; Malekan and Barros, 2016). The current G/XFEM implementation in INSANE is able to solve different types of structural problems with material nonlinearities and it can easily extended to incorporate any kind of enrichment function.

1.2 Motivation and Objectives

Engineering structures are designed to withstand their service loads until a crack/flaw start to initiate in the structure. The stress values close to the maximum admissible stress must never be attained under the service loading. Thus, high stress concentrations are avoided and a reasonable safety margin is taken into account. However, as mentioned before, the occurrence of fracture due to propagation of a pre-exist crack/flaw is something that engineers have to take into account in their design. So, it is very important to see how a structure will respond to a pre-crack condition, in other words, what the crack path is look like. On of the demanding area to use this kind of analysis is the aircraft structures. There are three ways to model a thin-walled structure, like aircraft fuselage, under fracture condition: 1) three-dimensional model of the structure which is computationally expensive and cumbersome, and 2) mixed-dimensional model of the structure where can reduce the computational cost, and 3) using the plate/shell element types to reduce the computational costs by their specific features.

The possibilities offered by technological resources for software development are broad research field in the area of numerical and computational methods applied to engineering.

The domain of these resources and to implement them in the progressive improvement of the models require a segmented computing environment, friendly to changes and scalable in complexity, as was proposed in section 1.1.4. The INSANE computational environment is composed by three great applications: pre-processor, processor and post-processor. The pre and post-processor are interactive graphical applications that provide tools to build different discrete representations of a structural problem as well as its results visualization. The processor is the numerical core of the system and it is responsible for obtaining the results from the analysis. The general objective of this work is to add some new features to the INSANE numerical core in order to establish classical and two-scale G/XFEM methodology for crack propagation problems in thin-walled structures.

According to section 1.1.3, there is a lack of using numerically built global-local enrichment in modeling of crack propagation in thin-walled structures (Plates and Shells). Thus, one can use the advantages of global-local approaches to analyze fracture mechanics problems of thin-walled structures like shells and plates.

This PhD thesis is focusing on the crack propagation modeling in two-dimensional and Reissner-Mindlin plates using either classical or two-scale G/XFEM method. The stable G/XFEM will be also adapted to have a well-conditioned system of equations. The whole numerical steps are already implemented within the INSANE computational platform. The final goal of this project is ***to fracture modeling in plate structure by extending the available capabilities of the G/XFEM method implemented in INSANE in-house code***. It is important to mention here that *the crack in the Reissner-Mindlin plate problems is a through-thickness crack and the effect of crack fronts orientation over the thickness is not considered in this PhD thesis. In addition, all numerical modelings are done under linear elastic fracture mechanics assumptions.*

As a conclusion, the main objective of this work can be explained as: to design an efficient, simple and reliable G/XFEM-based global-local framework for crack propagation problems in plate structures implemented in INSANE in-house platform. This framework uses available tools from INSANE and extends them to provide required functionalities to support the Global-Local G/XFEM and SG/XFEM methods. In summary, the following specific goals should be achieved through this framework:

- Development and design of an object-oriented data structure as well as implementation of a geometry engine algorithm based on G/XFEM strategy for crack growth parameterization for two-dimensional and Reissner-Mindlin plate problems.
- Extend available G/XFEM^{gl} approach in INSANE platform to work with the Reissner-Mindlin plates.
- Design and implementation of a classical G/XFEM algorithm for crack growth simulations in plane stress/strain and Reissner-Mindlin plate problems.
- Extend classical G/XFEM implementation for crack propagation to two-scale, i.e.,

G/XFEM^{gl}, in order to work for plane stress/strain and Reissner-Mindlin plate problems.

1.3 Thesis outline

The remainder of the present thesis is organized as follows:

- **Chapter 2:** Presents theoretical background for current project. It includes: the basic formulation of the generalized/extended FEM, stable G/XFEM, and global-local G/XFEM, and an overview of the Reissner-Mindlin plate formulations,
- **Chapter 3:** A general explanation of INSANE system is made in this chapter,
- **Chapter 4:** Implementations and numerical examples for stable G/XFEM methodology for various enrichment types,
- **Chapter 5:** Presenting the crack propagation procedure based on G/XFEM for both plane stress/strain and Reissner-Mindlin plate problems,
- **Chapter 6:** Presenting the crack propagation procedure based on two-scale G/XFEM for both plane stress/strain and Reissner-Mindlin plate problems,
- **Chapter 7:** The PhD thesis concludes with the presentation of a concluding remarks and also present possible future works as an extension of this research.

Chapter 2

THEORETICAL BACKGROUND

2.1 Introduction

This chapter deals with the concepts and formulations of different mathematical methodologies that are used in this PhD thesis, from classical and two-scale G/XFEM to main formulation of the Reissner-Mindlin plate problem.

2.2 Problem description

Consider a domain (Ω) shown in Fig. 2.1. The boundary of domain is partitioned into displacement (Γ_u), traction (Γ_t) and crack surface (Γ_c) boundaries. The strong form of the equilibrium equations and boundary conditions can be written as:

$$\nabla \cdot \boldsymbol{\sigma} + \boldsymbol{b} = 0 \quad in\ \Omega \tag{2.1}$$

$$\boldsymbol{\sigma} \cdot \hat{\boldsymbol{n}} = \bar{\boldsymbol{t}} \quad on\ \Gamma_t :\ \text{external traction} \tag{2.2}$$

$$\boldsymbol{\sigma} \cdot \hat{\boldsymbol{n}} = 0 \quad on\ \Gamma_c :\ \text{traction free crack} \tag{2.3}$$

$$\boldsymbol{u} = \bar{\boldsymbol{u}} \quad on\ \Gamma_u :\ \text{prescribed displacement} \tag{2.4}$$

where $\boldsymbol{\sigma}$ is the Cauchy stress tensor, $\boldsymbol{u}$ is the displacement field vector, $\hat{\boldsymbol{n}}$ is the unit outward normal and $\boldsymbol{b}$ and $\bar{\boldsymbol{t}}$ are the body force and external traction vector. For small strains and displacements, strain-displacement relation can be written as:

$$\boldsymbol{\varepsilon} = \boldsymbol{\varepsilon}(\boldsymbol{u}) = \nabla_{\boldsymbol{s}} \boldsymbol{u} \quad in\ \Omega \tag{2.5}$$

In the above equation $\nabla_{\boldsymbol{s}}$ is the symmetric part of the gradient operator and $\boldsymbol{\varepsilon}$ is the linear strain tensor. The constitutive relation for linear elastic material is given by Hook's law:

$$\boldsymbol{\sigma} = \boldsymbol{D}_h \, \boldsymbol{\varepsilon} \tag{2.6}$$

where $\boldsymbol{D}_h$ is the Hook's tensor. The variational form of the equilibrium equation can be written as:

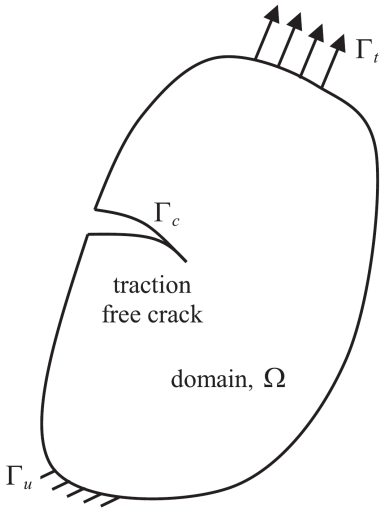


Figure 2.1: The linear elastic fracture mechanics problem in 3D

Find $\boldsymbol{u}$ such that $\forall\, \boldsymbol{v}$:

$$\int_{\Omega} \boldsymbol{\sigma}(\boldsymbol{u}) : \varepsilon(\boldsymbol{v}) d\Omega = \int_{\Omega} \bar{\boldsymbol{b}} \cdot \boldsymbol{v} d\Omega + \int_{\Gamma_t} \bar{\boldsymbol{t}} \cdot \boldsymbol{v} d\Gamma \tag{2.7}$$

After substituting the trial and test functions and using the arbitrariness of nodal variations, the following discrete system of equations are obtained:

$$[\boldsymbol{K}]\{\boldsymbol{d}\} = \{\boldsymbol{f}\} \tag{2.8}$$

where $\boldsymbol{K}$ is the global stiffness matrix, $\boldsymbol{d}$ is the vector of nodal unknowns (both standard and enriched) and $\boldsymbol{f}$ is the external force vector. The goal is to find an approximation to the solution $\boldsymbol{u}$ of the problem defined by Eqns. 2.1 - 2.6 for a thin-walled structure, like plates and shells, by extending the capabilities of the G/XFEM method implemented in the INSANE computational platform.

2.3 Generalized/Extended Finite Element Method

Early implementations of the finite elements were mainly focused on models for discontinuity problems to define meshes that conformed to the discontinuity surfaces (Ingraffea and Saouma, 1985; Cook et al, 2002; Swenson and Ingraffea, 1988). One may faces some issues to model discontinuity within finite element analysis, such as developing the finite element models to have a mesh compatible with discontinuity surfaces is cumbersome in many physical problems, the model may suffer from lack of accuracy and mesh-dependency (Bazănt, 1976; de Borst et al, 1993), and mesh refinement is inevitable to capture the solution of evolving discontinuities (Camacho and Ortiz, 1996). The G/XFEM was introduced to overcome aforementioned problems which is a partition of unity-based method that uses enrichment

functions to represent global/local phenomena of the problem and also to guarantee interelement continuity. This method was developed mainly for modeling structural problems with discontinuities (Melenk and Babuška, 1996; Duarte and Oden, 1995; Oden et al, 1998; Fries and Belytschko, 2010; Alves et al, 2013). The partition of unity functions, PU, are an important building block in the G/XFEM. The PU (Melenk and Babuška, 1996) functions over a domain are defined as the collection of smooth non-negative functions $\mathcal{N}_i$'s which satisfy two key requirements: (1) having $\sum_{i=1}^{n} \mathcal{N}_i(x) = 1$ everywhere in the domain, and (2) every point in the domain has only a finite number of non-zero $\mathcal{N}_i$'s acting on it.

A generic patch of elements or cloud $\omega_j \in \bar{\Omega}$ is obtained by the union of finite elements sharing the vertex node $\boldsymbol{x}_j$. The assemblage of the interpolation functions, built at each element $\mathcal{K}_e \subset \omega_j$ and associated with node $\boldsymbol{x}_j$, composes the function $\mathcal{N}_j(\boldsymbol{x})$ defined over the support cloud ω_j. The set of functions $\{\mathcal{N}_j(\boldsymbol{x})\}_{j=1}^{N}$ in the domain $\bar{\Omega}$ constitutes a PU. The generalized FE shape functions are determined by the enrichment of the PU functions, which is obtained by the product of such functions by each one of the components of the enrichment function at the generic cloud ω_j:

$$\{\phi_{ji}\}_{i=1}^{q_j} = \mathcal{N}_j(\boldsymbol{x}) \times \{L_{ji}(\boldsymbol{x})\}_{i=1}^{q_j} \tag{2.9}$$

where $\{L_{ji}(\boldsymbol{x})\}_{i=1}^{q_j}$ is called local approximations (also called enrichment functions) and defined as a set of q_j linearly independent functions associated with each cloud ω_j. The resulting shape function $\phi_{ji}(\boldsymbol{x})$, inherits characteristics of both functions, i.e., the compact support of the PU and the approximate character of the local function.

As a consequence, the generalized global approximation, denoted by $\tilde{\boldsymbol{u}}(\boldsymbol{x})$, can be described as a linear combination of the shape functions associated with each node:

$$\tilde{\boldsymbol{u}}(\boldsymbol{x}) = \sum_{j=1}^{N} \mathcal{N}_j(\boldsymbol{x}) \left\{ \boldsymbol{u}_j + \sum_{i=2}^{q} L_{ji}^{p}(\boldsymbol{x}) \boldsymbol{b}_{ji} + \sum_{\alpha=2} L_{j\alpha}^{s}(\boldsymbol{x}) \boldsymbol{c}_{j\alpha} \right\} \tag{2.10}$$

where $\boldsymbol{u}_j$ are nodal parameters associated with standard FE shape function, $\mathcal{N}_j(\boldsymbol{x})$, $\boldsymbol{b}_{ji}$ and $\boldsymbol{c}_{j\alpha}$ are nodal parameters associated with G/XFEM shape functions, $\mathcal{N}_j(\boldsymbol{x}) \cdot L_{ji}(\boldsymbol{x})$, p and s are representing polynomial and singular enrichments, respectively. $L_{ji}^{p}(\boldsymbol{x})$ and $L_{j\alpha}^{s}(\boldsymbol{x})$ are polynomial and singular enrichment functions, respectively. An example of the enrichment function in $\Re^2$, L_{ji}^{p}, based on polynomials is:

$$L_{ji}^{p}(\boldsymbol{x}) = \left(\frac{x - x_j}{h_j} \right)^m \times \left(\frac{y - y_j}{h_j} \right)^n \tag{2.11}$$

where (x_j, y_j) are the coordinates of node $\boldsymbol{x}_j$, m and n are degree of polynomials in x and y directions, respectively, and h_j is an scaling factor. Another example for the enrichment function by considering the crack under mode-I and mode-II in linear elastic medium can be defined as (Duarte et al, 2000):

$${}^{x}L_{j\alpha}^{s}(\boldsymbol{x})|_{\alpha=1}^{2} = \left\{ \sqrt{r} \left[\left(\varkappa - \frac{1}{2} \right) \cos\frac{\theta}{2} - \frac{1}{2} \cos\frac{3\theta}{2} \right], \sqrt{r} \left[\left(\varkappa + \frac{3}{2} \right) \sin\frac{\theta}{2} + \frac{1}{2} \sin\frac{3\theta}{2} \right] \right\} \tag{2.12}$$

$$ {}^{y}L^{s}_{j\alpha}(\boldsymbol{x})|^{2}_{\alpha=1} = \left\{ \sqrt{r}\left[\left(\varkappa + \frac{1}{2}\right)\sin\frac{\theta}{2} - \frac{1}{2}\sin\frac{3\theta}{2}\right], \sqrt{r}\left[\left(\varkappa - \frac{3}{2}\right)\cos\frac{\theta}{2} + \frac{1}{2}\cos\frac{3\theta}{2}\right]\right\} \tag{2.13} $$

where r and θ are the polar coordinates centered on the crack-tip, and $\varkappa$ is the material constant $(3-4\nu)$ for plane strain and $\frac{3-\nu}{1+\nu}$ is for plane stress state. The superscripts x and y are referred to x and y directions, respectively. These two enrichment functions are obtained from the two-dimensional elasticity solution displacement field in the vicinity of crack-tip (Szabo and Babuška, 1991; Pereira et al, 2009; Gupta and Duarte, 2016). In both expressions, the first terms are the x and y components related to the first term of mode-I expansion, respectively. Likewise, the second term of ${}^{x}L^{s}_{j\alpha}(\boldsymbol{x})$ and ${}^{y}L^{s}_{j\alpha}(\boldsymbol{x})$ is extracted from the first term of the mode-II expansion.

Another example of the enrichment function, which is called the near-tip enrichment, is defined as (Belytschko and Black, 1999):

$$ [F_i(r,\theta)] = \left[\sqrt{r}\sin\frac{\theta}{2}, \sqrt{r}\cos\frac{\theta}{2}, \sqrt{r}\sin\frac{\theta}{2}\sin\theta, \sqrt{r}\cos\frac{\theta}{2}\sin\theta\right] \tag{2.14} $$

where i is the number of crack-tip functions $F(r,\theta)$ and (r,θ) denotes the local polar coordinate defined at the crack-tip.

2.4 Stable Generalized/Extended FEM

Although the G/XFEM offers advantages over the classical FEM approximation, there are still critical issues that bring some difficulties in practical aspects. Two issues addressed in (Tian, 2013): the extra nodal degrees of freedom (DOFs) and linear dependence between standard FEM and enrichment functions. The extra DOF results in a fast expanding linear system for a high order of local approximation. In the linear dependency, when both the PU and the local function are polynomial, the resulting global stiffness matrix is positive semi-definite due to the linear dependency of the functions used to build the approximate space (Duarte et al, 2000; Tian, 2013). There are remarkable works to solve the first issue, see (Duarte et al, 2000; Oden et al, 1998; Cai et al, 2010). The linear dependence issue is also discussed in the context of G/XFEM (Chessa et al, 2003; Wu and Li, 2015).

In addition, the local enrichments of the G/XFEM approximations leads to arising the blending elements (an element containing both enriched and non-enriched nodes) into the problem domain (Chessa et al, 2003; Gracie et al, 2008; Tarancón et al, 2009; Shibanuma and Utsunomiya, 2009). Figure 2.2 shows the schematic of a local enrichment and hence, presence of the blending elements, i.e., Ω^{blnd} and thus the partition of unity cannot be satisfied over these elements. The presence of these elements results in an arbitrarily ill-conditioned matrix and penalizes the convergence rate of the approximate solution. To address those problems,

Menk and Bordas (2011) proposed a method based on pre-conditioners, Laborde et al (2005); Béchet et al (2005) used enrichment functions at a fixed region, called geometrical enrichment, which lead to optimal convergence rates in the G/XFEM. Shibanuma et al (2014) were able to obtain good results in terms of reproducibility of stress field in two-dimensional linear fracture mechanics problem, by explicitly using the partition of unity idea to blend the approximate functions of the enriched and non-enriched sub-domains on the blending elements.

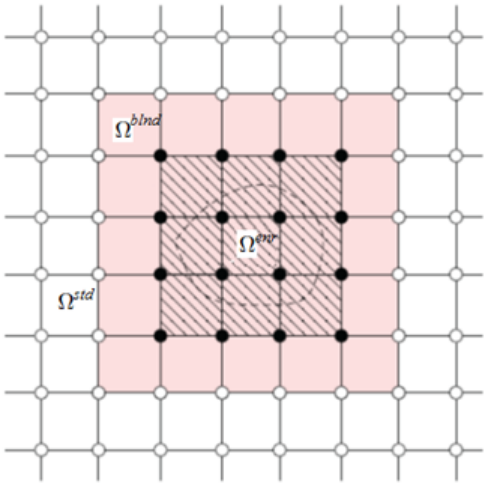


Figure 2.2: Schematic of local enrichment and presence of the blending element.

Recently, Babuška and Banerjee (2011, 2012) presented a new approach for one-dimensional domains, so-called Stable GFEM (SG/XFEM hereafter, considering the equivalence between GFEM and XFEM) which involves simple modification of the enrichment functions in order to create an enrichment space that is near-orthogonal to the finite element approximation space, while preserving all the attractive features of the G/XFEM. The SG/XFEM is aiming to improve the conditioning property of the G/XFEM. Another advantage is that the SG/XFEM does not use ramp-functions in the transitory elements between regions of different kinds of enriched functions as proposed by Fries (2008). Babuška and Banerjee (2011, 2012) show that the SG/XFEM is optimally convergent and it has no issues with the blending elements. Gupta et al. (Gupta et al, 2013, 2015; Gupta, 2014) extended one-dimensional SG/XFEM to two- and three-dimensional fracture mechanics. Following (Babuška and Banerjee, 2011, 2012; Gupta et al, 2013, 2015), the SG/XFEM enrichment functions are constructed based on a local modification of the G/XFEM enrichments functions, as following:

$$L_{ji}^{S}(\boldsymbol{x}) = L_{ji}(\boldsymbol{x}) - I_{\omega_j}(L_{ji})(\boldsymbol{x}) \tag{2.15}$$

with:

$$I_{\omega_j}(L_{ji})(\xi) = \sum_{\alpha \in \mathcal{I}(\tau)} L_{ji}(\boldsymbol{x}_\alpha)\mathcal{N}_\alpha(\xi) \tag{2.16}$$

where $I_{\omega j}(L_{ji})$ is the piecewise bi-linear FE interpolant of the G/XFEM enrichment function L_{ji} and L_{ji}^{S} is the modified SG/XFEM enrichment function, vector $\boldsymbol{x}_\alpha$ has the coordinates

of node α of element τ and $\mathcal{N}_\alpha$ is the piecewise linear FE shape function for node α. Then, similar to G/XFEM formulation (2.9), the shape function of SG/XFEM can be calculated as (see Fig. 2.3):

$$\phi^S_{ji}(\boldsymbol{x}) = \mathcal{N}_j(\boldsymbol{x}) \times L^S_{ji}(\boldsymbol{x}) \tag{2.17}$$

By referring to Eqs. (2.15) and (2.16), the main additional part for the SG/XFEM method is the evaluation of the finite element interpolant, $I_{\omega j}(L_{ji})$, which its computational evaluation is almost straightforward. In other words, we only need to compute the FE shape functions values at integration points. Further details on the numerical aspects of SG/XFEM and its detailed explanations can be found in (Gupta et al, 2013, 2015; Malekan and Barros, 2016).

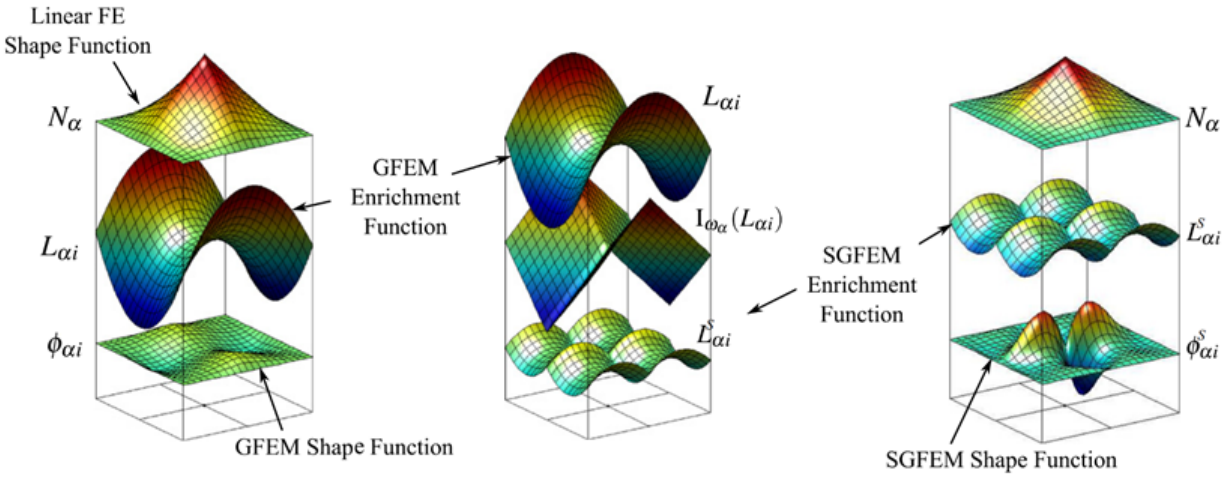


Figure 2.3: Stable G/XFEM enrichment and shape functions calculation procedure (Gupta et al, 2013).

2.5 Two-scale/Global-Local G/XFEM for Crack Propagation

The global-local G/XFEM originally proposed by Duarte and Babuška (2005), combines the standard G/XFEM with the global-local strategy proposed by Noor (1986). G/XFEM$^{\text{gl}}$ is suitable for problems with local phenomena, such as stress field next to the crack tip. The analysis is divided in three steps: *Initial global problem (step 1)* that uses a coarse FEM mesh, *Local (fine-scale) problem (step 2)* which uses a refined mesh in a small part of the initial global problem, and the *Final global problem (step 3)* that some of the nodes from initial global problem are enriched using numerical functions calculated in step 2. Figure 2.4 shows three the global-local steps for quasi-static crack propagation at time step t. The time step, t, here refers to the number of crack propagation steps during the quasi-static analysis.

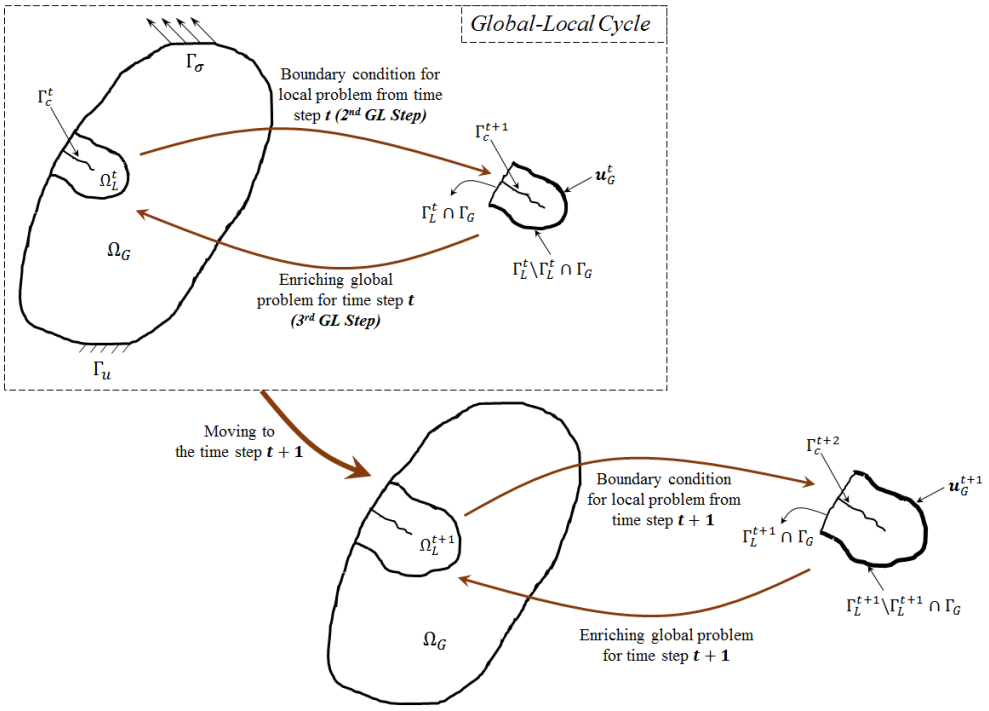


Figure 2.4: Global-local steps for quasi-static crack propagation at time step t. The global solution u_G^t at a crack propagation time step t provides boundary conditions for local problem defined in the domain Ω_L^t of the crack surface Γ_c^{t+1}. Then, the solution of the local problem is used to enrich the global problem at crack propagation time step $t+1$.

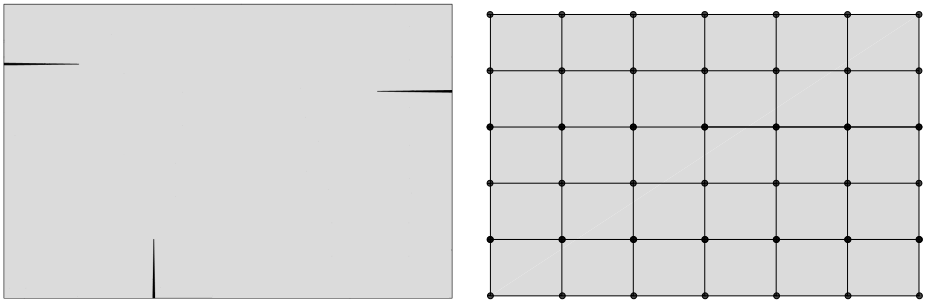
2.5.1 Initial Global Problem (step 1)

A coarse FEM mesh is used through the whole domain. Figure 2.5 exemplifies this step for a problem with several cracks. The position of these cracks can be either between the element edges or inside of the element boundaries, over their areas. Consider a domain $\bar{\Omega}_G = \Omega_G \cup \Gamma_G$ of an elastic problem in R^n. The boundary is decomposed in $\Gamma_G = \Gamma_G^u \cup \Gamma_G^\sigma$ with $\Gamma_G^u \cap \Gamma_G^\sigma = \oslash$, where indices u and σ refer to the Dirichlet and Neumann boundary conditions. $\boldsymbol{u}_G^t \in \boldsymbol{X}_G^t(\Omega_G)$ represents the solution of the approximate space $\boldsymbol{X}_G^t(\Omega_G)$ for the initial global problem in its weak form, shown in:

Find $\boldsymbol{u}_G^t \in \boldsymbol{X}_G^t(\Omega_G)$ such that $\forall\, \boldsymbol{v}_G^t \in \boldsymbol{X}_G^t(\Omega_G)$:

$$\int_{\Omega_G} \boldsymbol{\sigma}(\boldsymbol{u}_G^t) : \boldsymbol{\varepsilon}(\boldsymbol{v}_G^t) d\boldsymbol{x} = \int_{\Gamma_G^\sigma} \bar{\boldsymbol{t}} \cdot \boldsymbol{v}_G^t d\boldsymbol{s} \tag{2.18}$$

where $\boldsymbol{\sigma}$, $\boldsymbol{\varepsilon}$, $\boldsymbol{v}_G^t \in \boldsymbol{X}_G^t(\Omega_G)$, and $\bar{\boldsymbol{t}}$ are stress tensor, strain tensor, test functions, and prescribed traction vector, respectively.



(a) Problem with several cracks (b) Discretization of the problem

Figure 2.5: Typical problem with several local domains

2.5.2 Fine-scale Problem (step 2)

A refined mesh is used in a small part of the initial global problem. Figure 2.6 shows an example where each crack composes a local problem. Ω_L^t is a sub-domain from Ω_G. This sub-domain may contain cracks, holes or other special features. The corresponding local solution $\boldsymbol{u}_L^{t+1} \in \boldsymbol{X}_L^t(\Omega_L^t)$ is obtained from:

Find $\boldsymbol{u}_L^t \boldsymbol{X}_L^t(\Omega_L^t)$ such that $\forall\, \boldsymbol{v}_L^t \boldsymbol{X}_L^t(\Omega_L^t)$:

$$\begin{aligned} &\int_{\Omega_L^t} \boldsymbol{\sigma}(\boldsymbol{u}_L^t) : \boldsymbol{\varepsilon}(\boldsymbol{v}_L^t) d\boldsymbol{x} + \eta \int_{\Gamma_L^t \cap \Gamma_G^u} \boldsymbol{u}_L^t \cdot \boldsymbol{v}_L^t d\boldsymbol{s} + \kappa \int_{\Gamma_L^t \setminus (\Gamma_L^t \cap \Gamma_G^\sigma)} \boldsymbol{u}_L^t \cdot \boldsymbol{v}_L^t d\boldsymbol{s} = \\ &\int_{\Gamma_L^t \cap \Gamma_G^\sigma} \bar{\boldsymbol{t}} \boldsymbol{v}_L^t d\boldsymbol{s} + \eta \int_{\Gamma_L^t \cap \Gamma_G^u} \bar{\boldsymbol{u}} \cdot \boldsymbol{v}_L^t d\boldsymbol{s} + \int_{\Gamma_L^t \setminus (\Gamma_L^t \cap \Gamma_G^\sigma)} (\boldsymbol{t}(\boldsymbol{u}_G^t) + \kappa \boldsymbol{u}_G^t) \cdot \boldsymbol{v}_L^t d\boldsymbol{s} \end{aligned} \tag{2.19}$$

where $\boldsymbol{v}_L^t \in \boldsymbol{X}_L^t(\Omega_G)$ represents the test functions, $\boldsymbol{X}_L^t(\Omega_G)$ is the space generated by FEM or G/XFEM functions, η is the penalty parameter and κ is the stiffness parameter to consider Cauchy boundary condition. $\kappa = 0$, $\kappa = \eta >> 1$, and $0 < \kappa < \eta$, if Eq. (2.19) corresponds to a Neumann (if $\Gamma_L^t \cap \Gamma^u = \emptyset$), Dirichlet and Cauchy problem, respectively (Kim et al, 2010). The spring stiffness, κ, can be selected as (Kim et al, 2012, 2010):

$$\kappa = \frac{E}{\sqrt[\mathrm{nd}]{V_0\, J}} \tag{2.20}$$

where E is the Young's modulus, nd is the number of spatial dimensions of the problem, V_0 is the volume of the master element used and J is the Jacobian of the global element across the local boundary where the spring boundary condition is imposed.

In G/XFEM$^{\mathrm{gl}}$, numerical solution produced by local analysis can be easily introduced into the global problem. The local problem can be strongly refined and it does not affect the computational performance of the problem. Therefore, the problem is solved using less memory and processing machine than standard G/XFEM, as observed by Duarte and Kim (2008) and Kim et al (2010).

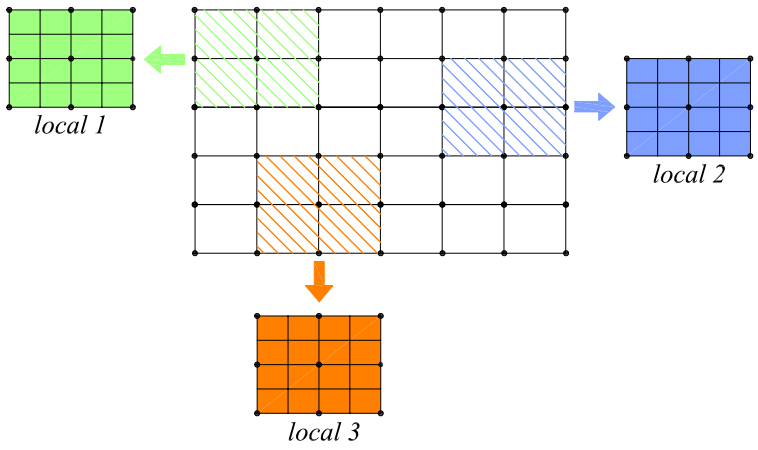


Figure 2.6: Several local problems, adapted from (Alves, 2012)

2.5.3 Enriched Global Problem (step 3)

Some of the nodes from initial global problem are enriched using numerical functions calculated in step 2. The global problem is enriched by solution $\boldsymbol{u}_L^t$ (from step 2). The new solution $\boldsymbol{u}_{GL}^t \in \boldsymbol{X}_{GL}^t(\Omega_G)$ is obtained from:

Find $\boldsymbol{u}_{GL}^t \in \boldsymbol{X}_{GL}^t(\Omega_G)$ such that $\forall\, \boldsymbol{v}_{GL}^t \in \boldsymbol{X}_{GL}^t(\Omega_G)$:

$$\int_{\Omega_G} \boldsymbol{\sigma}(\boldsymbol{u}_{GL}^t) : \boldsymbol{\varepsilon}(\boldsymbol{v}_{GL}^t) d\boldsymbol{x} = \int_{\Gamma_\sigma} \bar{\boldsymbol{t}} \cdot \boldsymbol{v}_{GL}^t d\boldsymbol{s} \tag{2.21}$$

where $\boldsymbol{v}_{GL}^t$ represents the test functions and $\boldsymbol{X}_{GL}^t(\Omega_G)$ is the third step approximation space obtained by increasing the initial space, $\boldsymbol{X}_G^t(\Omega_G)$, by $\boldsymbol{u}_k^{gl,\,t}(\boldsymbol{x})$ from local problem $\boldsymbol{u}_L$:

$$\boldsymbol{X}_{GL}^t(\Omega_G) = \left\{ \tilde{\boldsymbol{u}}(\boldsymbol{x}) = \sum_{j=1}^{N} N_j(\boldsymbol{x}) \hat{\boldsymbol{u}}_j^t(\boldsymbol{x}) + \sum_{k \in \mathcal{I}_{gl,\,t}} N_k(\boldsymbol{x}) \boldsymbol{u}_k^{gl,\,t}(\boldsymbol{x}) \right\} \tag{2.22}$$

and $k \in \mathcal{I}_{gl,\,t}$ represent the set of nodes enriched by the local solution and $\boldsymbol{u}_k^{gl,\,t}(\boldsymbol{x})$ is the function obtained from the local solution $\boldsymbol{u}_L^t$ from Eq. (2.19). For global-local analysis of the static problems, it only needs to neglect the superscript t from whole formulation of this section.

The global-local cycle shown in Fig. 2.4 can have multiple global-local enrichment process during each simulation time step. This means that there is a complete global-local enrichment strategy for each global-local cycle, i.e., solving global problem, then solving local problem, and finally enriching the global problem. For each crack propagation time step, at least one global-local cycle must be considered. However, to have better results from the global-local strategy, one can use multiple global-local cycles at the same time step. An important issue in the global-local analysis is the applied boundary conditions over the local domain boundaries. As it said before, the BCs can be either Dirichlet, Neumann, or Cauchy BC. In addition, the

size of the local domain plays an important role on the effect of the BCs, since for a local problem with a crack, more close the boundaries to the crack line, the bigger perturbations gets the crack from the boundaries. So, it is very important to choose an appropriate type of BCs to be transferred from global to local problem as well as a reasonable local domain size to have a quite enough distance between the local boundary and the crack line(s) to exclude the interference of the boundaries over the crack movement.

2.6 Reissner-Mindlin Plate

A structural element which is thin and flat is called *plate*. The "thin" means that the plate transverse dimension, or thickness, is small compared to the length and width dimensions. A mathematical expression of this idea is:

$$t/L \ll 1 \tag{2.23}$$

where t and L represent the plate thickness and a representative length or width dimension, respectively, as shown in Fig. 2.7. Plates might be classified as very thin if $L/t > 100$, moderately thin if $20 < L/t < 100$, thick if $3 < L/t < 20$, and very thick if $L/t < 3$. The "classical" theory of plates, Like Kirchoff and Reissner-Mindlin plate theories, is applicable to very thin (Kirchoff and Reissner-Mindlin) and moderately thin (Reissner-Mindlin) plates, while higher order theories are useful for thick plates (Steele and Balch, 2009).

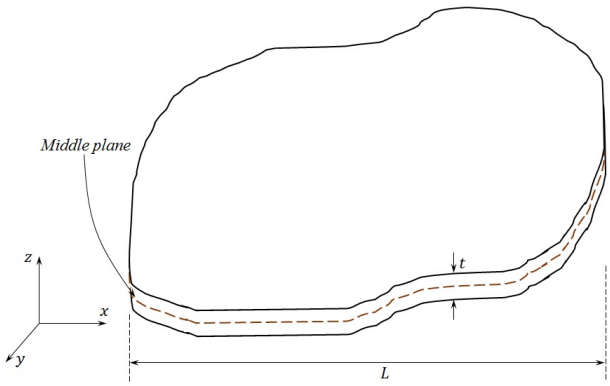


Figure 2.7: An arbitrary plate geometry

Reissner-Mindlin plate theory assumes that the normals to the plate do not remain orthogonal to the mid-plane after deformation, thus allowing for transverse shear deformation effects. This allows us to use C^0 continuous elements. Unfortunately, some difficulties arise when Reissner-Mindlin elements are used for thin plate situations due to the excessive influence of the transverse shear deformation terms like the *shear locking* defect. Elimination of shear locking is possible via reduced integration, linked interpolations or assumed transverse shear strain fields (Onate, 2013).

The assumptions of Reissner-Mindlin plate theory are the following, while it is assumed that the middle plane is equidistant from the upper and lower faces:

(1) In the points belonging to the middle plane ($z = 0$):

$$u = v = 0 \tag{2.24}$$

In other words, the points on the middle plane only move vertically.

(2) The points along a normal to the middle plane have the same vertical displacement (i.e. the thickness does not change during deformation).

(3) The normal stress σ_z is negligible (plane stress assumption).

(4) A straight line normal to the undeformed middle plane remains straight but not necessarily orthogonal to the middle plane after deformation (Fig. 2.8), in contrast to the Kirchoff plate theory that a straight line normal to the undeformed middle plane remains straight and normal to the deformed middle plane (normal orthogonality condition).

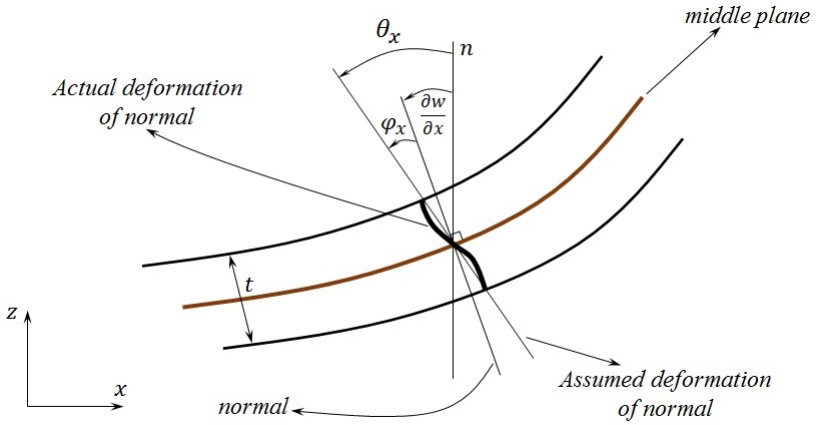


Figure 2.8: Reissner-Mindlin plate theory, rotations of the normal.

Assumptions 1, 2, 4 and Fig. 2.8 leads to:

$$\left.\begin{aligned} u(x,y,z) &= -z\,\theta_x(x,y) \\ v(x,y,z) &= -z\,\theta_y(x,y) \end{aligned}\right\} \text{ assumptions 1 and 4}$$
$$w(x,y,z) = w(x,y) \qquad \text{(assumption 2)} \tag{2.25}$$

where w is the vertical displacement of a point on the middle plane and the rotations θ_x and θ_y are corresponding to the angles followed by the normal vectors contained in the planes xz and yz, respectively. Vector $u = [w, \theta_x, \theta_y]^T$ is the displacement vector of a point on the middle plane of the plate.

From Fig. 2.8, following expressions can be obtained for the rotations angles, as a result of assumption 4:

$$\begin{aligned} \theta_x &= \frac{\partial w}{\partial x} + \varphi_x \\ \theta_y &= \frac{\partial w}{\partial y} + \varphi_y \end{aligned} \tag{2.26}$$

The rotation of the normal in each of the two vertical planes xz and yz is obtained as the sum of two terms: 1) the adequate slope of the plate middle plane, and 2) an additional rotation ϕ resulting from the lack of orthogonality of the normal with the middle plane after deformation (Fig. 2.8). Consequently, the rotations θ_x and θ_y can not be computed in terms of the deflection only and, therefore, are treated as independent variables. This is a substantial difference between Reissner-Mindlin and Kirchhoff plate theories. The assumption that the normals remain straight is only an approximation since in practice the normals are distorted, as shown in Fig. 2.8, and the angles θ_x and θ_y depend on the thickness coordinate. The hypothesis of a straight normal is equivalent to assuming an average uniform rotation for the normal, which obviously simplifies the kinematics.

Following Dolbow et al (2000b), the strain is given by:

$$\frac{1}{2}\left(\nabla \boldsymbol{u} + (\nabla \boldsymbol{u})^t\right) = z\,\boldsymbol{\epsilon}_b(\theta) + \frac{1}{2}\left[\boldsymbol{\epsilon}_s(w,\theta) \otimes z + z \otimes \boldsymbol{\epsilon}_s(w,\theta)\right] \tag{2.27}$$

with the bending and shear definitions as:

$$\boldsymbol{\epsilon}_b(\theta) = \frac{1}{2}\left(\nabla\theta + (\nabla\theta)^t\right) \tag{2.28a}$$

$$\boldsymbol{\epsilon}_s(w,\theta) = \nabla w + \theta \tag{2.28b}$$

The z related components are zero for both $\boldsymbol{\epsilon}_b$ and $\boldsymbol{\epsilon}_s$. The virtual internal work is defined by:

$$\partial W^{int} = \int_\Omega \boldsymbol{\sigma} \,:\, \nabla(\partial u)\, d\Omega \tag{2.29}$$

where $\boldsymbol{\sigma}$ is the symmetric stress tensor, and du is an arbitrary virtual displacement from the current position. After a few manipulations, the following relation can be obtained:

$$\boldsymbol{\sigma} \,:\, \nabla(\partial u) = z\boldsymbol{\sigma}^p : \boldsymbol{\epsilon}_b(\partial\theta) + \boldsymbol{\sigma}^s \cdot \boldsymbol{\epsilon}_s(\partial w, \partial\theta) \tag{2.30}$$

where the p superscript indicates a reduction of the operator to the in plane (x, y) components and $\boldsymbol{\sigma}^s$ is the shear stress vector $\boldsymbol{\sigma}^s = \boldsymbol{\sigma} \cdot z$.

Substituting Eq. (2.30) into Eq. (2.29) and integrating through the thickness gives the work expression:

$$\partial W^{int} = \int_\Omega \left(M : \boldsymbol{\epsilon}_b(\theta) + Q \cdot \boldsymbol{\epsilon}_s(w,\theta)\right) d\Omega \tag{2.31}$$

where the moment $\boldsymbol{M}$ and shear $\boldsymbol{Q}$ are defined by:

$$\boldsymbol{M} = \int_{-t/2}^{t/2} z\boldsymbol{\sigma}^p dz \tag{2.32a}$$

$$\boldsymbol{Q} = \int_{-t/2}^{t/2} \boldsymbol{\sigma}^s dz \tag{2.32b}$$

The virtual external work is composed of the action of the bending and twisting moments gathered in a couple vector $\boldsymbol{C}$, and of the shear traction $\boldsymbol{T}$. The virtual external work is then:

$$\partial W^{ext} = \int_{\Gamma} \boldsymbol{C} \cdot \partial\theta d\Gamma + \int_{\Gamma} \boldsymbol{T} \cdot \partial w d\Gamma \tag{2.33}$$

Equating the internal and external virtual work, and applying the divergence theorem yields the equilibrium equations in Ω:

$$\nabla \cdot \boldsymbol{M} - \boldsymbol{Q} = 0 \tag{2.34a}$$

$$\nabla \cdot \boldsymbol{Q} = 0 \tag{2.34b}$$

and the traction boundary conditions on Γ:

$$\boldsymbol{C} = \boldsymbol{M} \cdot n \tag{2.35a}$$

$$\boldsymbol{T} = Q \cdot n \tag{2.35b}$$

where n is the unit outward normal to the boundary. The constitutive relationships are obtained by energetic equivalence between the plate and three-dimensional model. Assuming the plate is made of an isotropic homogeneous elastic material of Young's modulus E and of Poisson's ratio ν, the constitutive relations are given by:

$$\begin{bmatrix} M_{xx} \\ M_{yy} \\ M_{xy} \end{bmatrix} = \frac{Et^3}{12(1-\nu^2)} \begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & 1-\nu \end{bmatrix} \begin{bmatrix} \epsilon_b^{xx} \\ \epsilon_b^{yy} \\ \epsilon_b^{xy} \end{bmatrix} \tag{2.36a}$$

$$\begin{bmatrix} Q_{xz} \\ Q_{yz} \end{bmatrix} = \mu k t \begin{bmatrix} \epsilon_s^{xz} \\ \epsilon_s^{yz} \end{bmatrix} \tag{2.36b}$$

where μ is the shear modulus. The correction factor k accounts for the parabolic variation of the transverse shear stresses through the thickness of the plate, and is taken to be $k = 5/6$. The above equations can be rewritten in a more compact form using the fourth order bending stiffness tensor $\boldsymbol{D}_b$ and the second order shear stiffness tensor $\boldsymbol{D}_s$:

$$\boldsymbol{M} = \boldsymbol{D}_b \, \epsilon_b \tag{2.37a}$$

$$\boldsymbol{Q} = \boldsymbol{D}_s \, \epsilon_s \tag{2.37b}$$

These equations are valid only for isotropic materials.

2.7 Summary

The focus of this chapter was to present the concept and formulations of G/XFEM, stable G/XFEM and G/XFEM$^{\text{gl}}$ methods that will be used through this whole PhD thesis. In addition, some main formulations of the Reissner-Mindlin plate are presented to have the main idea of this kind of structural problems.

Chapter 3

INSANE COMPUTATIONAL PLATFORM, A GENERAL OVERVIEW

3.1 Introduction

INSANE (INteractive Structural ANalysis Environment) computational platform, (Fonseca and Pitangueira, 2007), is an open source software implemented in *Java*, an Object-Oriented Programming (OOP) language. The choice for an OOP language was made because of the several benefits brought by this programming paradigm, as extensibility, robustness and maintenance. Additionally, Java language provides a simpler object model, high-level facilities and the Java application can run on any computer architecture with a Java Virtual Machine. Similar to all FEM codes, it is composed of three parts: pre-processor, processor (numerical core) and post-processor. The numerical core is composed by the interfaces *Assembler* and *Persistence* and the abstract classes *Solution* and *Model*, aiming to analyze different problems, from linear to nonlinear static and dynamic problems using different numerical approaches (FEM (Fonseca and Pitangueira, 2007; Fonseca, 2008), BEM (Peixoto, 2016; Peixoto et al, 2016), classical, stable, and two-scale G/XFEM (Wolff, 2010; Alves, 2012; Alves et al, 2013; Malekan et al, 2016a, 2017b; Malekan and Barros, 2016; Malekan et al, 2016b), Meshless method (Faria, 2014; Silva, 2012; Hosseini et al, 2017; Gori et al, 2017), and Hp-cloud method (Pereira, 2015; Pereira et al, 2015)). The persistence of data among these three segments of INSANE is performed by structured data files written in eXtensible Markup Language (XML) format.

3.2 INSANE Environment

The INSANE numerical core is composed by the interfaces *Assembler*, *Model* and *Persistence* and the abstract class *Solution*. Figure 3.1 shows the Unified Modeling Language (UML) diagram from INSANE numerical core. Three different colors are used here in UML diagrams in order to facilitate the visualization of the extensions and modifications made: *cream* for

the modified classes, *green* for the new classes that were created for this work, and *white* for the unchanged classes. This chapter mainly describes the INSANE numerical core in a general way. Several of the classes that appear here are discussed in detail in the next chapters together with the implemented strategy.

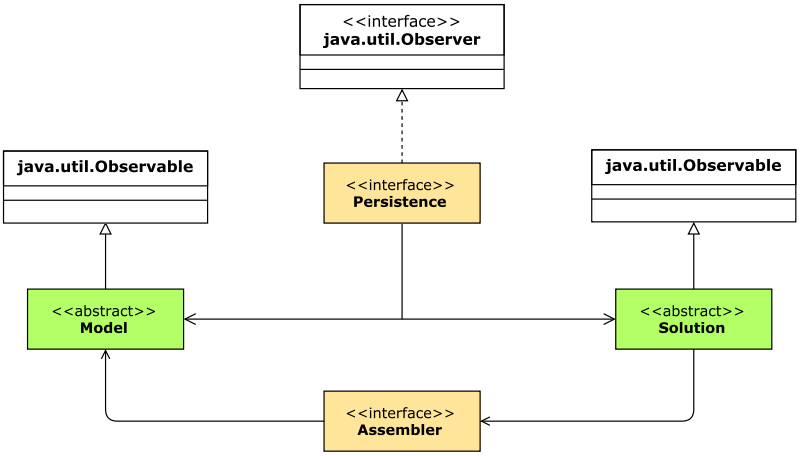


Figure 3.1: Organization of the INSANE numerical core

There is an observation strategy inside INSANE numerical core. This strategy is determined by the *Observer-Observable* design pattern, which is a change propagation mechanism. When an object of type *observer* (which implements the interface *java.util.observer*) is instantiated, it is added to a list of observers of other objects of type *observables* (which extends the class *java.util.observable*). Any modification in the state of an observed object notifies the corresponding observer object that updated itself.

3.2.1 Persistence Interface

Persistence interface treats the input data and persists the output data. For current work, this class is extended to deal with either static or quasi-static fracture analysis with both two-dimensional and Reissner-Mindlin plate problems. Figure 3.2 shows the components of the *persistence* package.

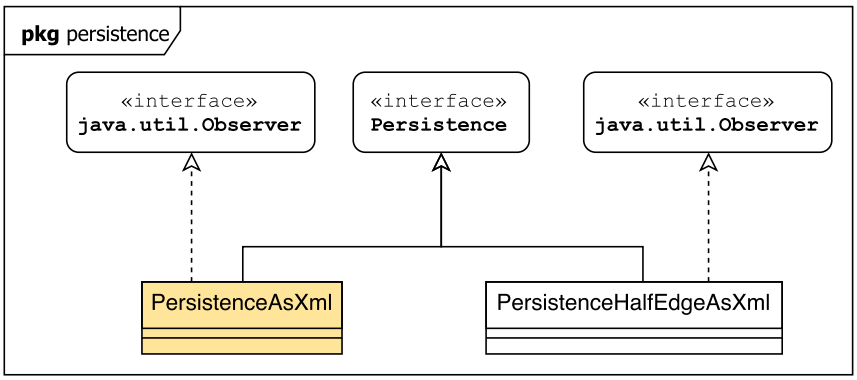


Figure 3.2: UML diagram of the *persistence* package

3.2.2 Assembler Interface

Assembler interface is responsible for assembling the linear equation system provided by the discretization of the boundary or initial value problem. This class is implemented following the generic representation:

$$\boldsymbol{A}\ \ddot{\boldsymbol{X}} + \boldsymbol{B}\ \dot{\boldsymbol{X}} + \boldsymbol{C}\ \boldsymbol{X} = \boldsymbol{D} \tag{3.1}$$

where $\boldsymbol{X}$ is the solution vector; the single dot represent its first time derivative and the double dots its second time derivative; $\boldsymbol{A}$, $\boldsymbol{B}$ and $\boldsymbol{C}$ are matrices with the properties of the problem and $\boldsymbol{D}$ is a vector that represents the system excitation. In static analysis, Eq. (3.1) is simplified by eliminating the two first terms. The resulting matrix system is:

$$\begin{bmatrix} \boldsymbol{C}_{uu} & \boldsymbol{C}_{up} \\ \boldsymbol{C}_{pu} & \boldsymbol{C}_{pp} \end{bmatrix} \begin{Bmatrix} \boldsymbol{X}_u \\ \boldsymbol{X}_p \end{Bmatrix} = \begin{Bmatrix} \boldsymbol{D}_p \\ \boldsymbol{D}_u \end{Bmatrix} \tag{3.2}$$

$$\begin{Bmatrix} \boldsymbol{D}_p \\ \boldsymbol{D}_u \end{Bmatrix} = \begin{Bmatrix} \boldsymbol{N}_p \\ \boldsymbol{N}_u \end{Bmatrix} + \begin{Bmatrix} \boldsymbol{E}_p \\ \boldsymbol{E}_u \end{Bmatrix} \tag{3.3}$$

In these equations, the matrix $\boldsymbol{C}$ is the model stiffness matrix, $\boldsymbol{X}$ is the vector of nodal displacements, $\boldsymbol{D}$ is the vector of forces. The vector $\boldsymbol{D}$ is composed of two parts: the vector $\boldsymbol{N}$ and the vector $\boldsymbol{E}$ which are the forces vector applied directly to the nodes and the forces/displacements prescribed by equivalent nodal vector, respectively. The sub-indices u and p informs if the vector is unknown or prescribed. This class is also extended to deal with either static or quasi-static fracture analysis with both two-dimensional and Reissner-Mindlin plate problems.

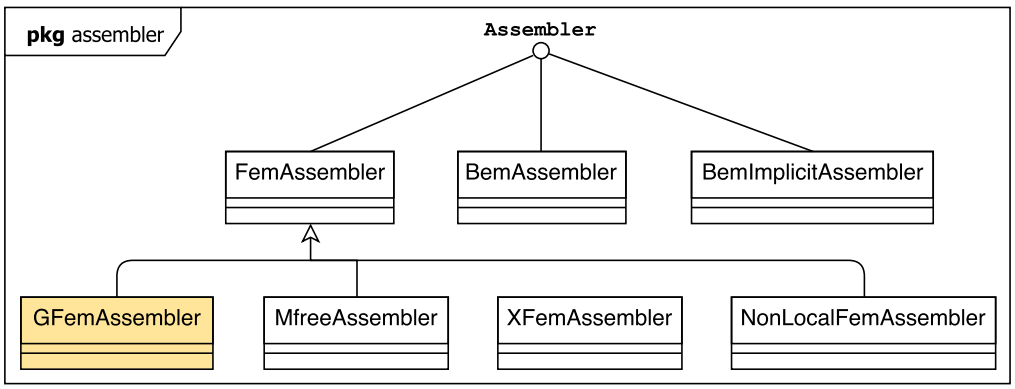


Figure 3.3: UML diagram of the *Assembler* package

3.2.3 Solution Abstract Class

Solution abstract class starts the solution process and has the necessary resources for solving the matrix system of the fracture analysis approach. It contains different classes to handle

either static, dynamic, or modal analysis. Some of this classes are either modified or extended to have new capabilities to handle the shell structures. Figure 3.4 shows component of the *Solution* package. Also, it contains different direct and iterative solvers to solve the system of equations. Also, a *Sparse* matrix (see Appendix C) is incorporated for the linear problems in order to accelerate the solution procedure. This class is extended to deal with either static or quasi-static fracture analysis for both two-dimensional and Reissner-Mindlin plate problems.

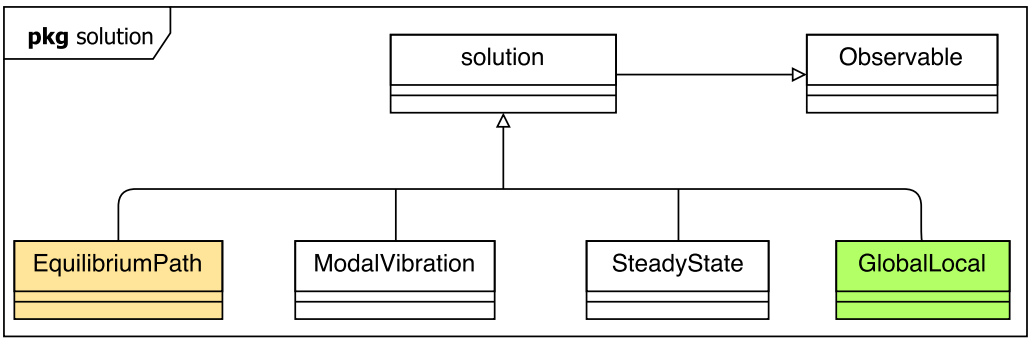


Figure 3.4: The *solution* package

The main class for solving linear equation systems from Eq. (3.1), is the *LinearEquationSystems* class, as its UML diagram is shown in Fig. 3.5. This class contains several solver types to solve the static system of equations, using either direct or iterative solvers. Nowadays, the *SteadyState* and *GlobalLocal* classes use different solvers from the *LinearEquationSystems* class to solve the problem with the classical and two-scale approaches, respectively.

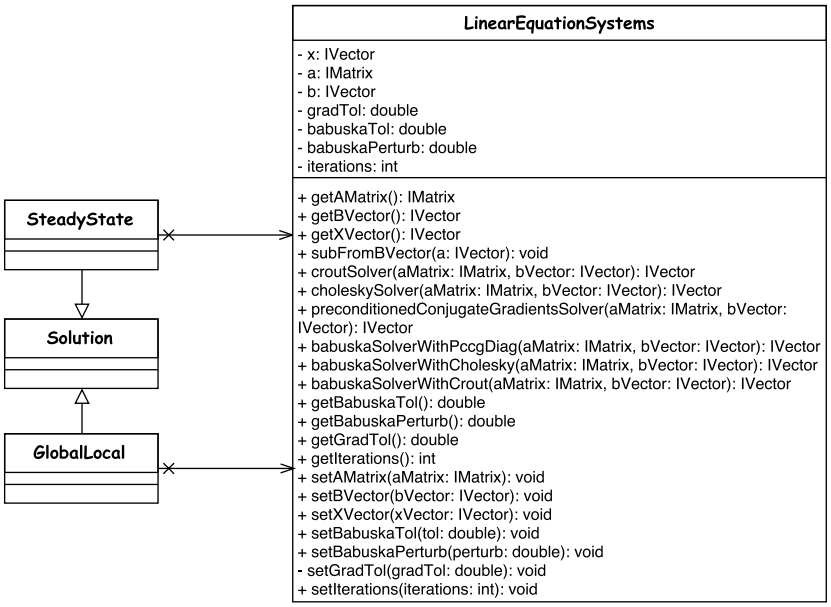


Figure 3.5: UML diagram of the *LinearEquationSystems* class

3.2.4 Model Interface

The *Model* interface contains the data of the discrete model and provides to *Assembler* informations to assemble the final matrix system. Both *Model* and *Solution* communicate with the *Persistence* interface, which treats the input data and persists the output data

to the other applications, whenever it observes a modification of the discrete model state. For current work, several classes are implemented or modified under this interface aiming to provide required information for the process. Some examples of these classes are: different *Enrichment* classes for either classical G/XFEM or stable and two-scale G/XFEM methods, and *ProblemDriver* class as a part of matrix assembly process. This class is extended to deal with either static or quasi-static fracture analysis for both two-dimensional and Reissner-Mindlin plate problems. The *ProblemDriver* abstract class is designed to inform *Assembler* class all the necessary data for assembling the Eq. (3.1). Different finite element formulations originally provided within the INSANE environment are implemented by different classes derived from *SolidMech* abstract class, each one with a specific way to calculate the different parts of the Eq. (3.1). Some of these classes are: parametric (class *Parametric*), Kirchhoff thin plate (class *KirchhoffThinPlate*), and frame (class *Frame*). In addition, there are various models already implemented within INSANE code: space frame, plane stress/strain, plate and solid. When the *Assembler* starts to assemble the system of equations (3.1), it asks the object of *AnalysisModel* for characteristics about the model, such as how the deformation matrix should be calculated. Figure 3.6 presents the *model* package.

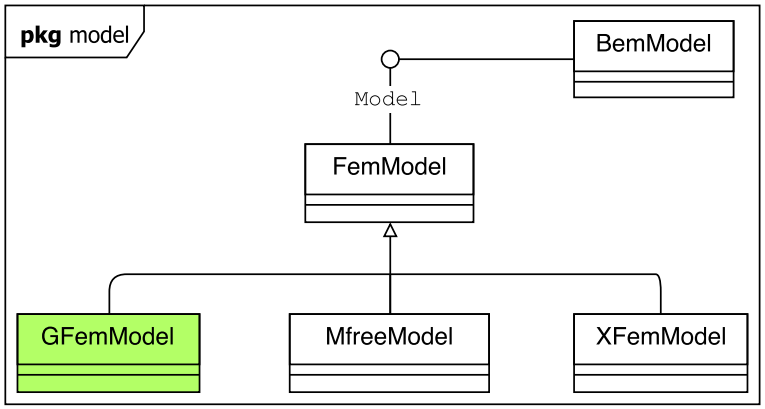


Figure 3.6: UML diagram of the *model* package

3.3 Summary

An overview of the INSANE numerical core with its main classes was presented in this chapter. The focus was to present the general aspects of INSANE environment, emphasizing the requirement for the static and quasi-static fracture analysis using G/XFEM method. Detail implementations along with the numerical examples for each part is presented in the following chapters.

Chapter 4

STABLE GENERALIZED/EXTENDED FINITE ELEMENT METHOD

4.1 Introduction

Using the locally-enriched strategy to enrich a small/local part of the problem by generalized/extended finite element method (G/XFEM) leads to non-optimal convergence rate and ill-conditioning system of equations due to presence of blending elements. The local enrichment can be chosen from polynomial, singular, branch or numerical types. The so-called stable version of G/XFEM method provides a well-conditioning approach when only singular functions are used in the blending elements.

This chapter shows and discusses a generic implementation of the well-conditioning approach toward generalized/extended finite element method. These implementations are performed into an academic computational platform, follows the object-oriented approach presented before for the standard version of G/XFEM in which the shape functions of finite elements are hierarchically enriched by analytical functions, according to the problem behavior. The stable version of G/XFEM is employed here to avoid the bad effects of blending elements on the approximate solution convergence rate. In global-local G/XFEM, the enrichment functions are constructed numerically from the solution of a local problem. Furthermore, several enrichment strategies are adopted along with the global-local enrichment. The results obtained with these enrichments strategies are discussed in detail, considering convergence rate in strain energy, growth rate of condition number, and computational processing. All the explanations and numerical results for this chapter are brought from Malekan et al (2016b); Malekan and Barros (2016).

4.2 OOP Implementations

The *EnrichmentType* class is a part of the *Model* interface which provides required information for the enrichment strategy of the G/XFEM method. The *GlobalLocalEnrichment*, *PolynomailEnrichment*, and *CrackEnrichment* (singularity enrichment) extend *EnrichmentType* class to have G/XFEM enrichment functions for various problems.

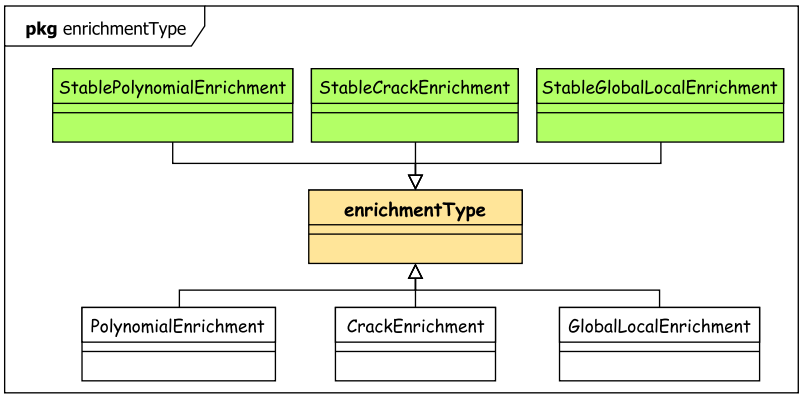


Figure 4.1: UML diagram of the *EnrichmentType* package

The class *PolynomailEnrichment* holds some arrays to represent the monomial functions used in Eqs. (2.9) and (2.11). On the other hand, the class *CrackEnrichment* (Eqs. (2.12) and (2.13)) holds some necessary parameters to simulate functions with high singularity in the derivatives. In addition, *GlobalLocalEnrichment* provides specific methods to build the enriched functions from the solution of the local problem and applied in the third step of the global-local problem, according to procedure described in section 2.5. *EnrichmentType* class has some generic methods which are responsible to calculate enrichment functions and their derivatives in a point. Each node, instance of an object *Node*, can have a list of the objects of class *EnrichmentType*. This relationship allows the existence of multiple types of enrichment functions for the mesh as a whole and also for each node.

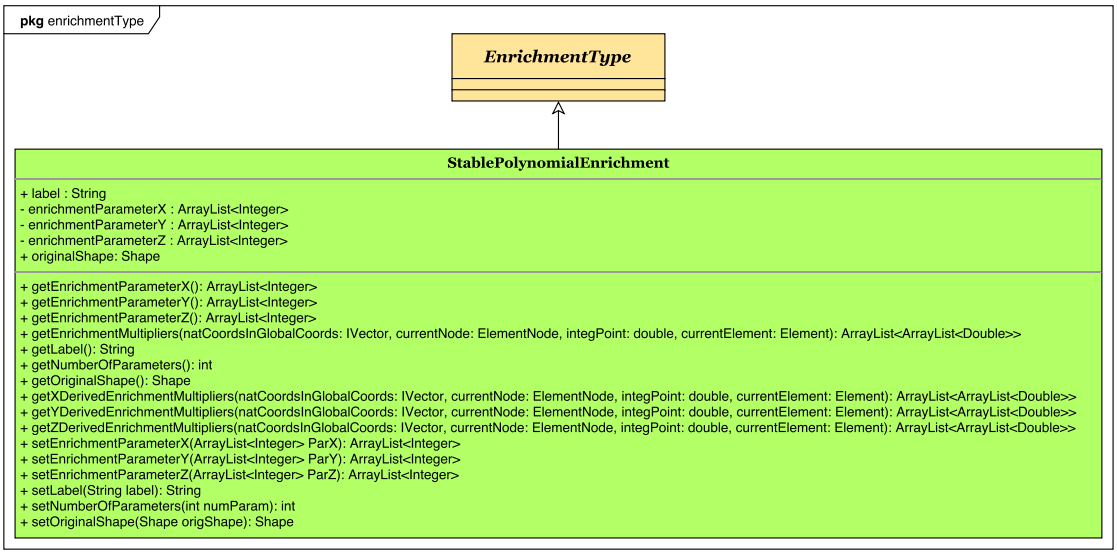


Figure 4.2: UML diagram of the *StablePolynomialEnrichment* class

Furthermore, to implement the SG/XFEM approach, the *StableGlobalLocalEnrichment*, *StablePolynomailEnrichment*, and *StableCrackEnrichment* are added under *EnrichmentType* class. The *StableGlobalLocalEnrichment* provides specific methods to build the SGFEM enriched functions from the solution of the local problem and applied in the third step of the global-local problem. Figure 4.1 shows the *EnrichmentType* class UML diagram. *EnrichmentType* is an abstract class and its methods are abstract. Thus, both G/XFEM and SG/XFEM related enrichment classes contain the same method, as the *EnrichmentType* class. Figures 4.2, 4.3, and 4.4 show the UML diagrams of these new classes. In all UML diagrams, white, cream, and green colors represent unchanged, modified, and new classes, respectively.

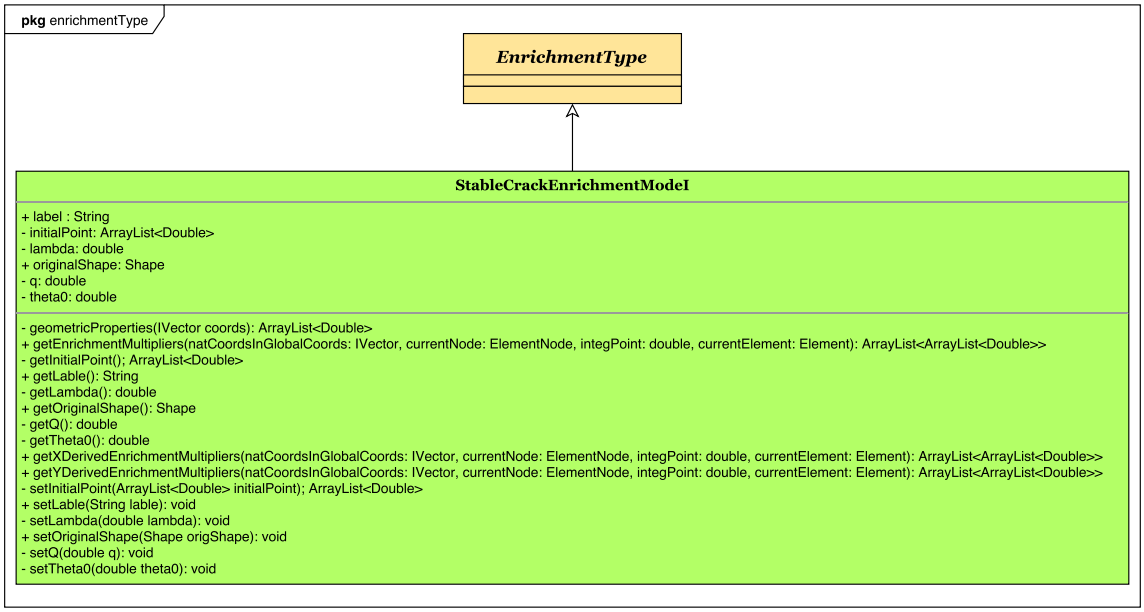


Figure 4.3: UML diagram of the *StableCrackEnrichment* class

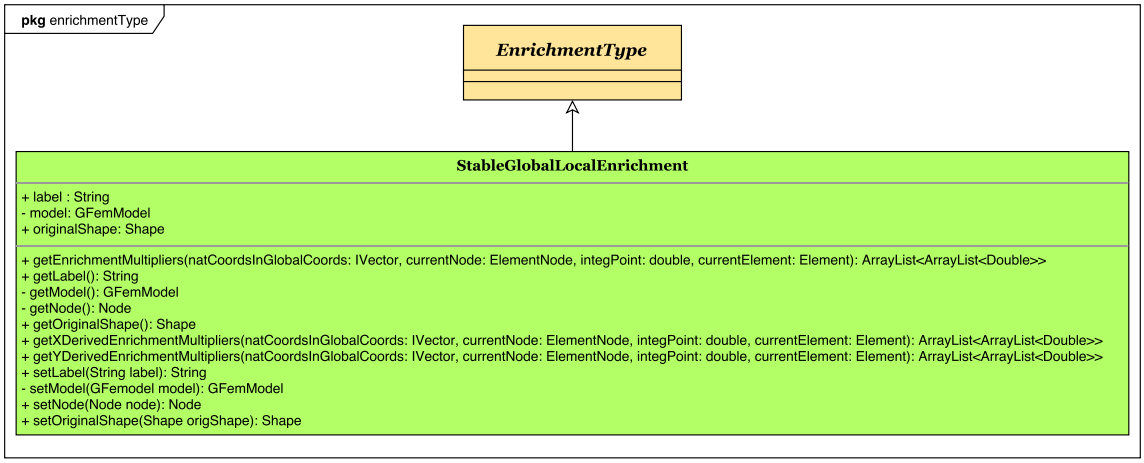


Figure 4.4: UML diagram of the *StableGloblLocalEnrichment* class

4.3 Numerical Example

This section presents three linear-elastic problems in $\Re^2$. Section 4.3.1 presents a double-edge cracked plate, section 4.3.2 presents a plate with an edge crack, and a beam with a crack is presented in section 4.3.3. The geometry and boundary conditions are very simple and the goal of choosing them is to demonstrate the capabilities of the SG/XFEM method as well as the SG/XFEM$^{\text{gl}}$ method. The cracked zone in these problems produces singular stress field near the crack tip.

In examples 4.3.1, 4.3.2 and 4.3.3, the crack surface is geometrically represented in the global and local models using double nodes. This approach is chosen here because, by our experience, if the crack isn't described in the initial global problem, the boundary conditions of the local problem provided by the initial global solution impact the quality of the final solution. Among the three aforementioned boundary conditions, the Dirichlet boundary condition (a limiting case of Cauchy boundary condition) leads to worse results than Cauchy boundary condition (Kim et al, 2010). Thus, the first one will be applied on the local problem boundaries in order to demonstrate the robustness of the methodology in the worst case scenario. Numerical integration for the first and second steps of the global-local analysis is done based on standard Gaussian quadrature procedure. The integration order of G/XFEM$^{\text{gl}}$ and SG/XFEM$^{\text{gl}}$ for both global and local problems is chosen to be enough to reproduce the polynomial approximation and are equal to 6×6, respectively. In the third step, the numerical integration for those global elements that contain local elements is done over the Gauss points of local elements, as proposed by Kim et al (2010). Consider that a global element contains n^{Le} local elements and the number of Gauss points for each local element is equal to GP. Thus, the number of integration points for this global element is obtained by: $\sum_{i=1}^{n^{Le}} GP_i$.

All problems are analyzed under plane stress state, have the following parameters (in consistent units): modulus of elasticity $E = 1.0$, Poisson ratio $\nu = 0.3$, and the shear stress $\tau = 1.0$, if applicable.

4.3.1 Double-edge cracked problem

This example considers a double-edge cracked problem submitted to a shear stress, as shown in Fig. 4.5 and is analyzed under plane stress condition. The cracked zone produces singular stress field near the crack tips. The objective of this example is to illustrate the use of SG/XFEM with singular enrichment (shown in Eqs. (2.12) and (2.13)) for fracture mechanic problems.

The reference solution of this problem is obtained using a mesh of 14531 quadrilateral elements (CPS4, a 4-node bilinear plane stress quadrilateral element) in $ABAQUS^{®}$. The mesh used around the crack tip is shown in Fig. 4.7. The number of integration points for this problem is considered equal to 8×8.

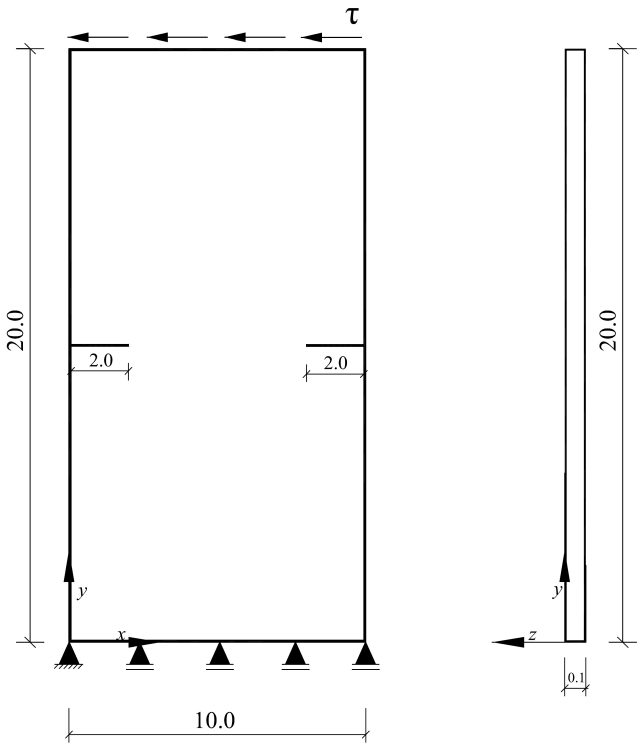
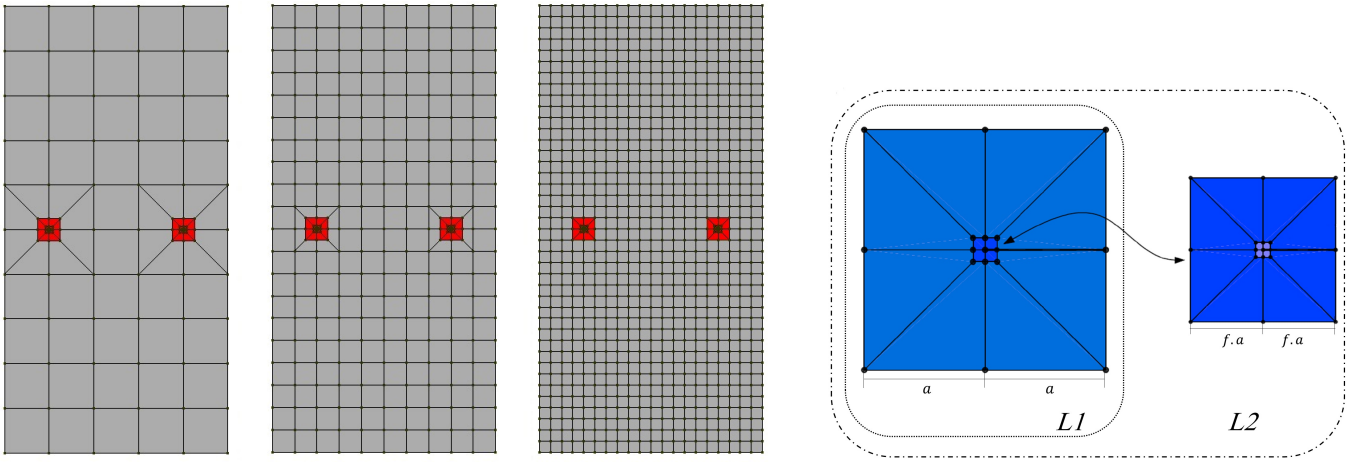


Figure 4.5: *Geometry and loading of the double-edge cracked problem*

Using INSANE, there are three different average element sizes (h) in this study, $h = 2.0$, 1.0, and 0.5. The total number of elements are 98, 248, and 832 elements with a combination of regular and geometrical mesh distribution considering these three element sizes, as shown in Fig. 4.6(a). As it can be seen from this figure, only four elements from the cloud associated with the crack tip is discretized with the geometric mesh. The geometric refining approach in this study is based on four refinement level (L4) with $f = 10\%$ reduction rate, as shown in Fig. 4.6(b), according to Szabo and Babuška (1991). Only nodes **inside** the red area (38 nodes) are enriched with the polynomial and singular enrichment functions. The goal is having an equivalent enrichment zone for all three meshed. Of course, there is no need to geometric mesh when the problem is enriched with singular enrichment, because this enrichment function is able to capture the singularity around the crack tip. We decided to have the same mesh for all enrichment cases. The P2 (quadratic) enrichment function is considered here to enrich the problem with the polynomial enrichment. The formulation of this function for x direction is as follows:

$$\phi_j^T(\boldsymbol{x}) = \begin{bmatrix} \mathcal{N}_j(\boldsymbol{x}) & 0 & \left(\frac{x-x_j}{h_j}\right)^2 \mathcal{N}_j(\boldsymbol{x}) & 0 & \left(\frac{y-y_j}{h_j}\right)^2 \mathcal{N}_j(\boldsymbol{x}) & 0 \\ 0 & \mathcal{N}_j(\boldsymbol{x}) & 0 & \left(\frac{x-x_j}{h_j}\right)^2 \mathcal{N}_j(\boldsymbol{x}) & 0 & \left(\frac{y-y_j}{h_j}\right)^2 \mathcal{N}_j(\boldsymbol{x}) \end{bmatrix} \tag{4.1}$$



(a) Three meshes (b) Geometric mesh with reduction rate of f

Figure 4.6: Double-edge cracked problem meshing strategy.

Similar expression is used for y direction. The enrichment function is cubic and the is polynomials with order one, i.e., linear. The reason behind using only quadratic polynomials is that the linear polynomials in the case of SG/XFEM returns zero for enrichment functions. The polynomial enrichment of order one provides null enrichment in Eq. (2.15), because the FE interpolant I_{ω_j}, used in the stable strategy is also linear. The FE interpolant is not considered in the SG/XFEM for the linear terms of the polynomial enrichment, so the interpolation is performed only for the polynomials of second degree.

Figure 4.8 shows the convergence rate results against the inverse of the element size for different enrichment types. Here are different case used here: GFEM/SGFEM-poly is the problem enriched with polynomials of order 2, GFEM/SGFEM-crackMixed is the problem enriched with mode-I and mode-II singular enrichments, from Eqs. (2.12) and (2.13), and GFEM/SGFEM-poly_crackMixed is the problems enriched with both polynomials and the singular enrichment functions. As it can be seen from this figure, both error values and rate of convergences for SG/XFEM shown an improvement over the G/XFEM and FEM approaches. Specially, for having both polynomial and singular enrichment together (poly_crackMixed), the difference between the convergence rate is meaningful, $0.3 - 0.4$ for both FEM and G/XFEM, while $0.4 - 0.9$ for SG/XFEM. The FEM approach here also uses the same mesh shown in Fig. 4.6(a) and the problem is solved using the INSANE code.

The condition numbers for three methods are illustrated in Fig. 4.9. Rate of growth in condition numbers for all three methods are almost similar. This is because we only enrich a small part of the problem and with limited number of nodes, so the total system of equations between three method cannot change heavily.

Table 4.1 shows the CPU time of the conventional approach use here which is Babuška iterative approach. Only corresponding data to the enriching problem with the singular enrichment are brought in this Table.

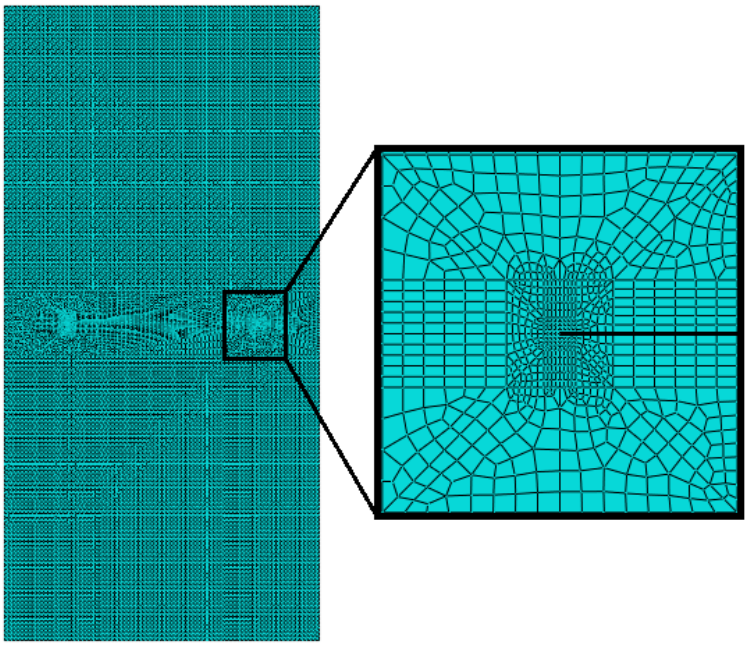


Figure 4.7: Mesh discretization used in ABAQUS for the whole problem and around one the crack tips.

Table 4.1: CPU time for the Babuška solver

	G/XFEM			***SG/XFEM***		
h size	2	1	0.5	2	1	0.5
DOF	389	718	1940	389	718	1940
CPU time (msec) *Babuška iterative method*	145	722	6369	139	667	6081

4.3.2 Problem with an edge crack

A two-dimensional linear elastic fracture problem is solved by different enrichment strategies based on global-local analysis to approximate the problem solution. This example considers a rectangular plate with an edge crack submitted to a shear stress, as shown in Fig. 4.10.

The reference solution of this problem is obtained using a mesh of 89711 quadrilateral elements (CPS4, a 4-node bilinear plane stress quadrilateral element) in $ABAQUS^{®}$. For SG/XFEM^{gl} analysis, however, a smaller number of finite elements as well as of DOFs is used. The reason for using a smaller number of DOFs is explained by the use of global-local enrichment function, which is suitable for high stress concentration. For this problem, the penalty parameter, η, of Dirichlet boundary condition is chosen equal to 1×10^8.

4.3.2.1 Type of the mesh refinement

There are three different element sizes (h) for global mesh in this study, $h = 2.0$, 1.0, and 0.5. Therefore, the total number of elements are 50, 200, and 800 elements with regular

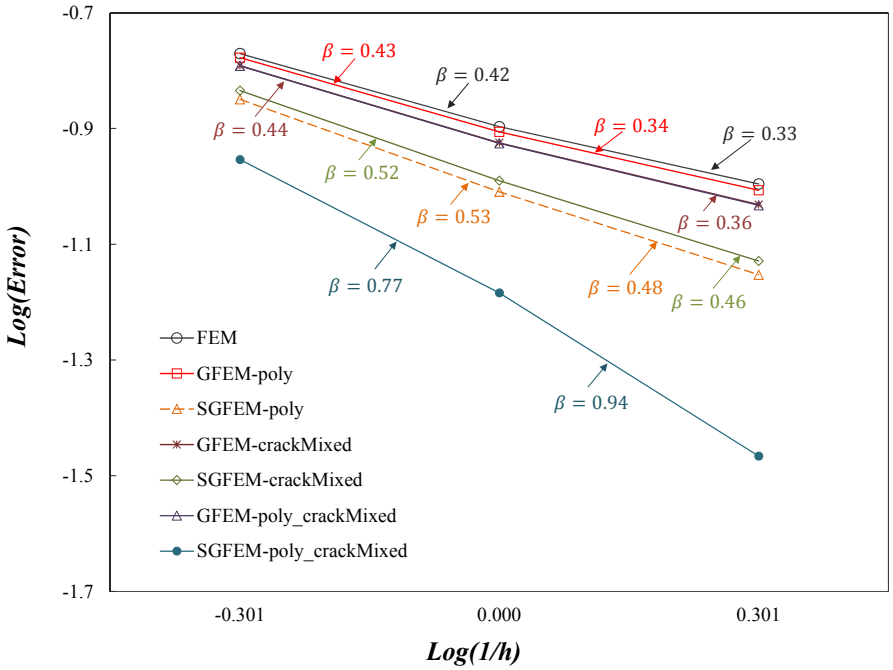


Figure 4.8: Relative error in strain energy against the inverse of average element size $(1/h)$ for FEM, G/XFEM, and SG/XFEM methods with different enrichment strategies. β refers to the convergence rate.

distribution considering these three element sizes, as shown in Fig. 4.11. The final answer for a problem solved using global-local strategy depends on the size of the local domain as well as the local problem mesh, as shown by Duarte and Kim (2008). Large local domains are preferable, since they give a better numerical solution to the global problem. However, large local domains also increase the computational cost in solving problems.

The dimension of local domain considered here is equal to 6×8. This domain is composed by 12, 48, and 192 global element with $h = 2.0, 1.0$, and 0.5, respectively, see colored elements of Fig. 4.11. The local mesh is shown in Fig. 4.12 and it is fixed with 224 elements for all three global meshes. For the three discretization, the local problem has exactly the same description with a combination of regular and geometric mesh (with refinement of level L4), as shown in Fig. 4.13. Only the four elements from the cloud associated with the crack tip in local problem are discretized with geometric mesh. The geometric refining approach in this study is based on four refinement level (L4) with $f = 10\%$ reduction rate, as shown in Fig. 4.13, according to Szabo and Babuška (1991).

Results of strain energy for the three global meshes are presented in Table 4.2. These results are obtained for global problem (first step in global-local strategy) with no enrichment.

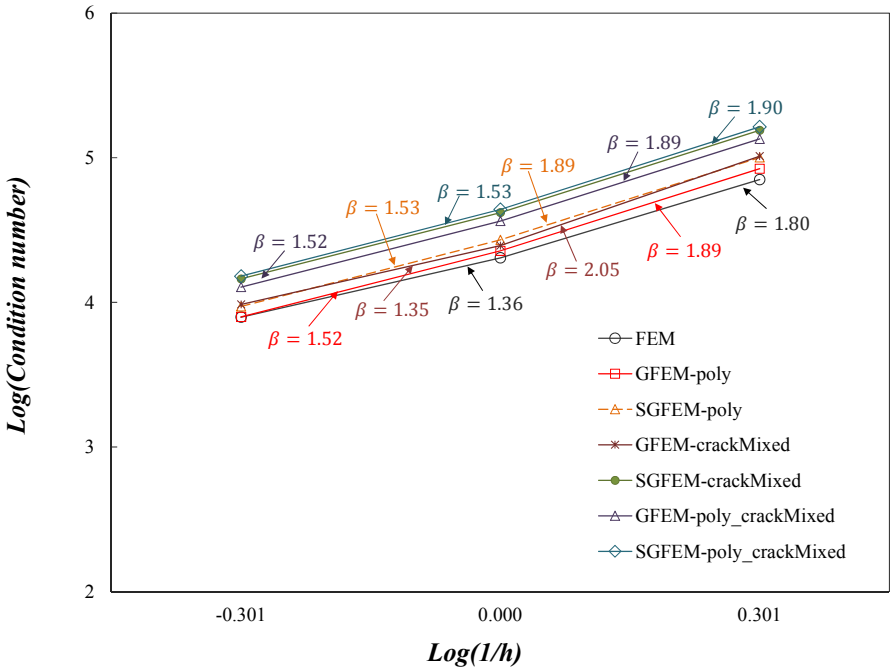


Figure 4.9: Condition number against the inverse of average element size $(1/h)$ for FEM, GFEM, and SGFEM methods with different enrichment strategies. β is the growth rate of the condition number.

Table 4.2: DOFs and strain energy for different regular mesh size in the global problem

Analysis	DOFs	Strain energy	% of error in strain energy
Reference	180328	266.483	—
50 elements	134	209.4957	21.38
200 elements	468	220.8901	17.11
800 elements	1730	229.5691	13.85

4.3.2.2 Enrichment strategy

There are two global-local enrichment strategies used here: enriching only the crack-tip node and enriching several nodes around the crack. Figure 4.14 shows the global-local steps and also the local domain discretization for the case of $h = 2.0$.

Those strategies are used with different combinations of polynomial and numerical enrichments, with or without the stable modification given by the Eq. (2.15). They are represented as $SGFEM^{gl}_{P_G=X,P_L=Y}$ or $GFEM^{gl}_{P_G=X,P_L=Y}$ if the stable enrichment with the numerical local solution is performed or not. 'P_G' and 'P_L' are the polynomial order used for global and local problems, respectively. If P_G or P_L is preceded by the letter S, a stable polynomial enrichment is employed, instead of conventional polynomial enrichment. As it said before, $GFEM^{gl}$ refers to global-local G/XFEM method while $SGFEM^{gl}$ refers to stable global-local G/XFEM method. The resulting strategies are:

– GFEM$^{gl}_{P_G=P_L=0}$ and SGFEM$^{gl}_{P_G=P_L=0}$: No polynomial enrichment for both global

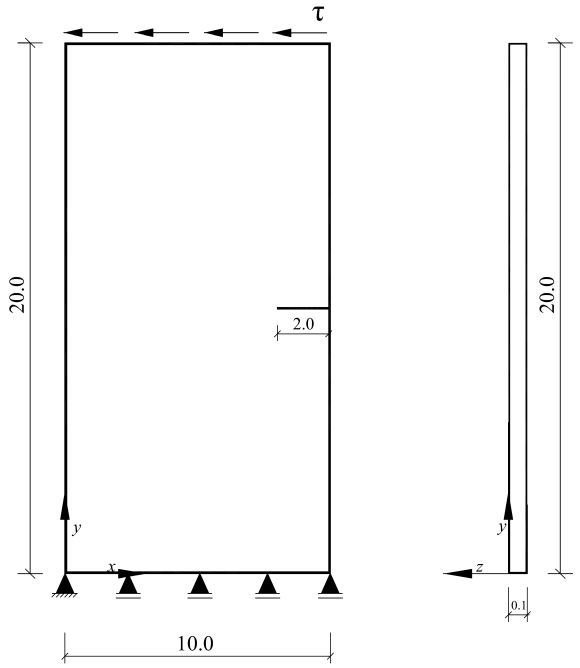


Figure 4.10: *Geometry and loading of a problem with an edge crack under plane stress state submitted to a shear stress. With (in consistent units): Young modulus* $E = 1.0$*, Poisson's ratio* $\nu = 0.3$*, and the shear stress* $\tau = 1.0$*.*

and local problems.

- GFEM$^{gl}_{P_G=0,P_L=2}$ and SGFEM$^{gl}_{P_G=0,P_L=2}$: No polynomial enrichment for global problem and with second order for local problem.
- GFEM$^{gl}_{P_G=0,SP_L=2}$ and SGFEM$^{gl}_{P_G=0,SP_L=2}$: No polynomial enrichment for global problem and stable polynomial enrichment of second order for local problem.
- GFEM$^{gl}_{P_G=2,P_L=0}$ and SGFEM$^{gl}_{P_G=2,P_L=0}$: Polynomial enrichment of second order for global problem and no polynomial enrichment for local problem.
- GFEM$^{gl}_{P_G=P_L=2}$ and SGFEM$^{gl}_{P_G=P_L=2}$: Polynomial enrichment of second order for both global and local problems.
- GFEM$^{gl}_{P_G=SP_L=2}$ and SGFEM$^{gl}_{P_G=SP_L=2}$: Polynomial enrichment of second order for global problem and stable polynomial enrichment of second order for local problem.
- GFEM$^{gl}_{SP_G=2,P_L=0}$ and SGFEM$^{gl}_{SP_G=2,P_L=0}$: Stable polynomial enrichment of second order for global problem and no polynomial enrichment for local problem.
- GFEM$^{gl}_{SP_G=SP_L=2}$ and SGFEM$^{gl}_{SP_G=SP_L=2}$: Stable polynomial enrichment of second order for global and local problems.

The polynomial enrichment of order one has not been used because the FE interpolant I_{ω_j}, used in the stable strategy, is also linear and provides null enrichment in Eq. (2.15).

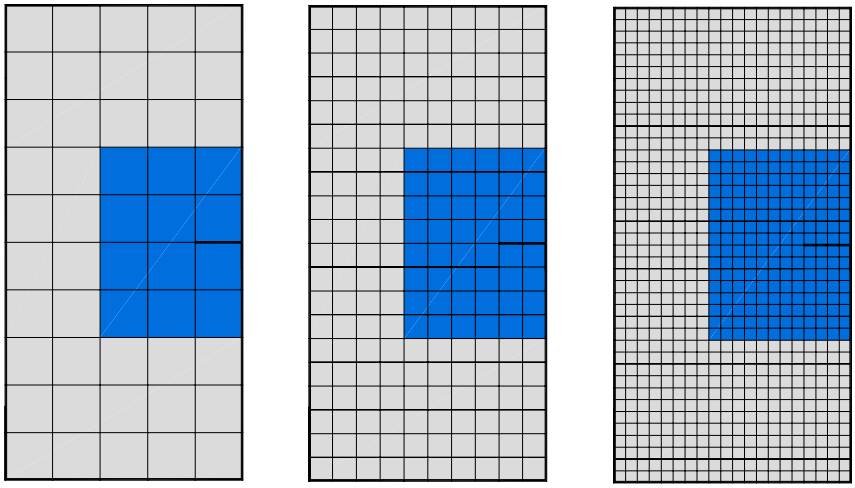


Figure 4.11: Schemes of three global meshes. Blue elements represent the local domain.

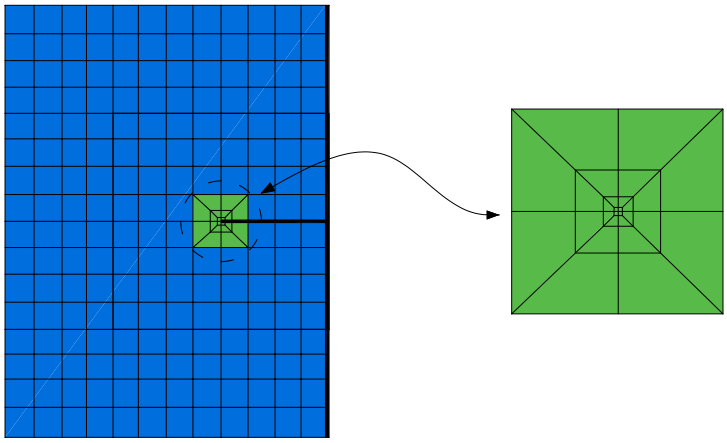


Figure 4.12: Mesh representation of the local problem

4.3.2.3 Enriching crack-tip node

Firstly only the crack tip node is enriched using the global-local strategy. Figure 4.15 shows the relative error in strain energy against the inverse of element size $(1/h)$ considering all enrichment cases described in section 4.3.2.2 for both G/XFEM^{gl} and SG/XFEM^{gl} approaches. Although the SG/XFEM^{gl} error is smaller than G/XFEM^{gl} for the $h = 2.0$, the convergence rate of G/XFEM^{gl} is higher than SG/XFEM^{gl} case. Higher convergence rate of G/XFEM^{gl} is in accordance with results presented in Gupta (2014) for one-cycle-based global-local analysis which is the case here. Although, there are some differences between convergence rate in the strain energy error obtained from current research and those presented in Gupta (2014) (0.1-0.35 from current work and 0.5-1.0). The difference in number of enriched nodes using global-local numerically enrichment function, explain the mis-consistency between the

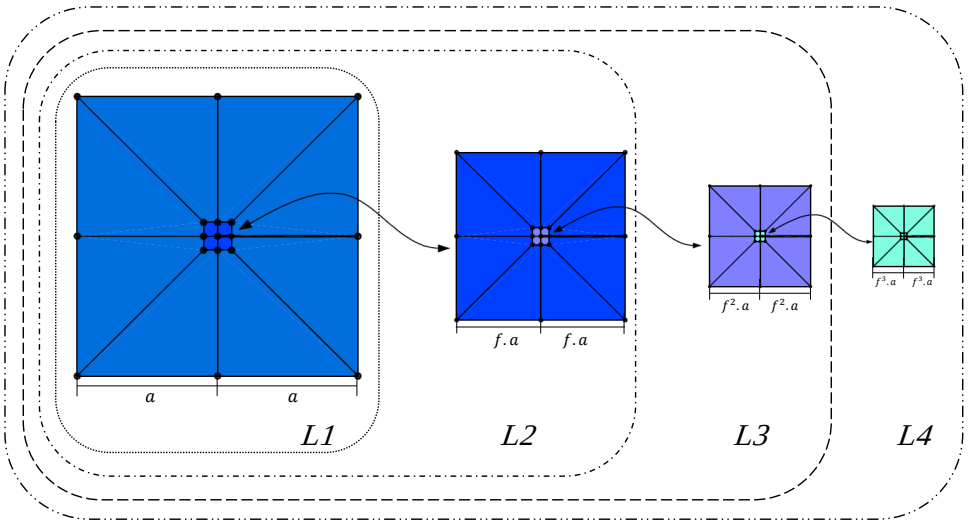


Figure 4.13: Geometric mesh with reduction rate of f

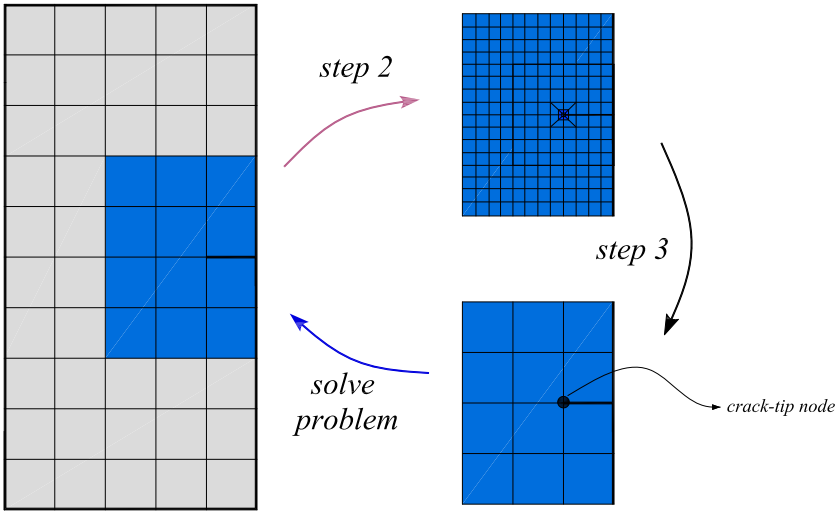
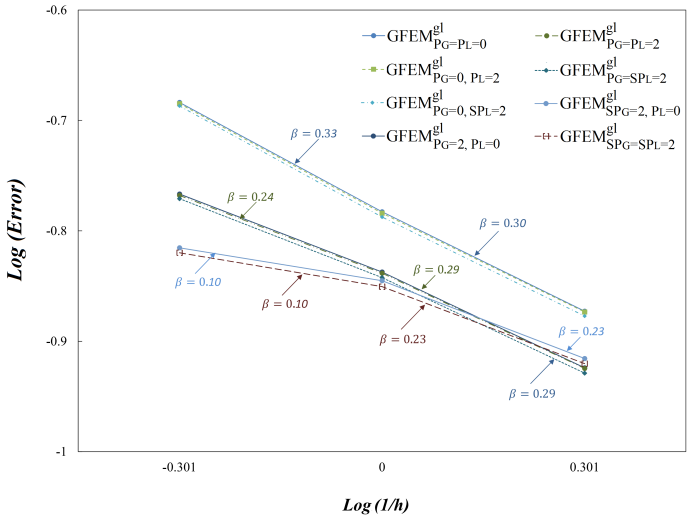


Figure 4.14: Global-local strategy sequences and local domains discretization for global element size of $h = 2.0$.

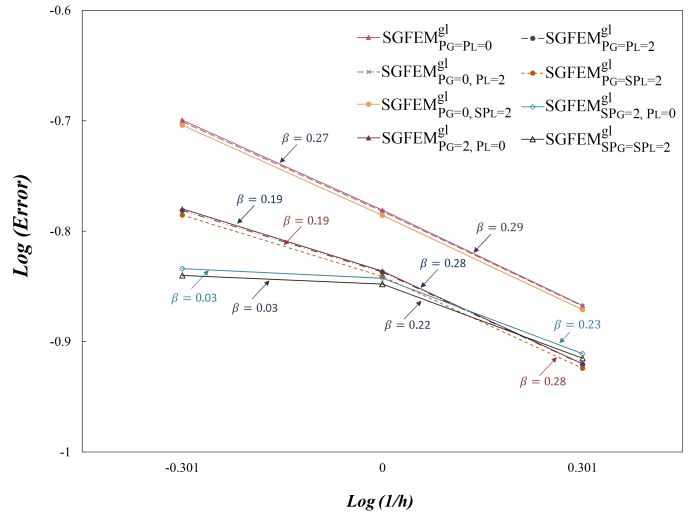
results. In addition, enriching global problem with the polynomial/stable polynomial enrichments along with global-local enrichment leads to lower convergence rate.

According to (Babuška and Banerjee, 2012; Gupta et al, 2013, 2015), one of the important feature of the SG/XFEM is that its conditioning is better than that of the G/XFEM. Figure 4.16 presents the condition number of the scaled stiffness matrix obtained against the inverse of element size, again considering whole enrichment cases described in Section 4.3.2.2 for both G/XFEM^{gl} and SG/XFEM^{gl} approaches.

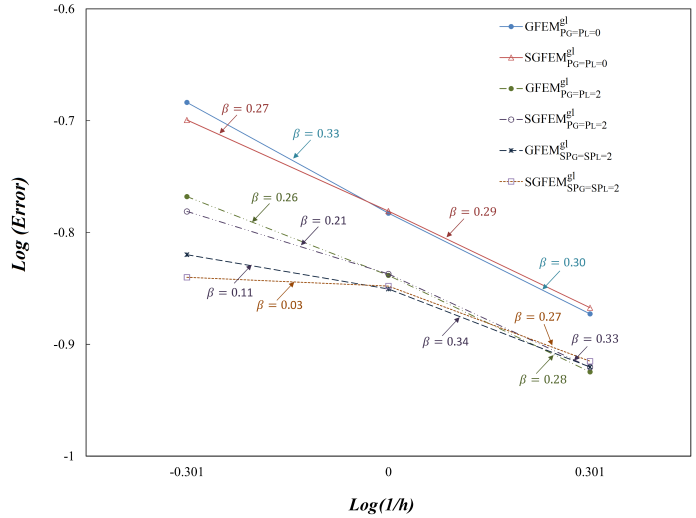
Because the stable GFEM improves the conditioning of the system of equations, computational efficiency can be improved compared with the GFEM method. Figure 4.17 compares the number of iterations for the preconditioning conjugate gradient (PCG) method ran inside of the Babuska procedure (Strouboulis et al, 2000a) between G/XFEM^{gl} and SG/XFEM^{gl}



(a) G/XFEM^{gl}

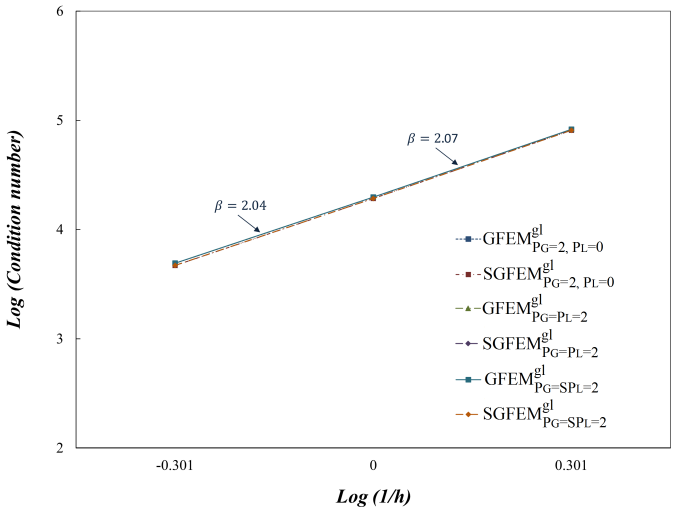


(b) SG/XFEM^{gl}

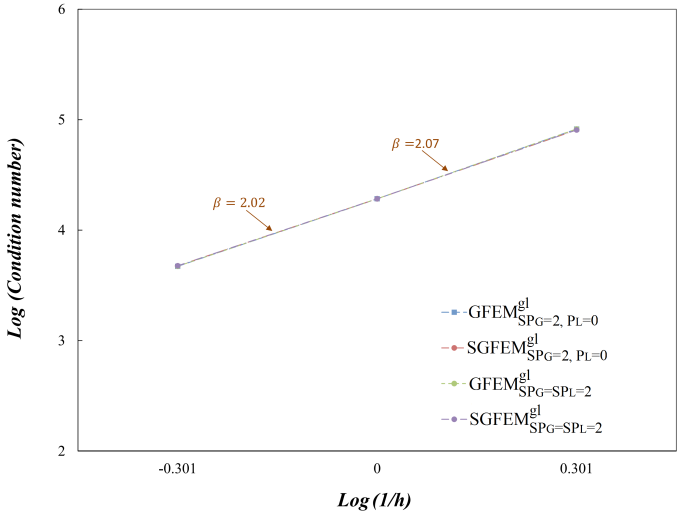


(c) both G/XFEM^{gl} and SG/XFEM^{gl} for $P_G = P_L = 0$, $P_G = P_L = 2$, and $SP_G = SP_L = 2$

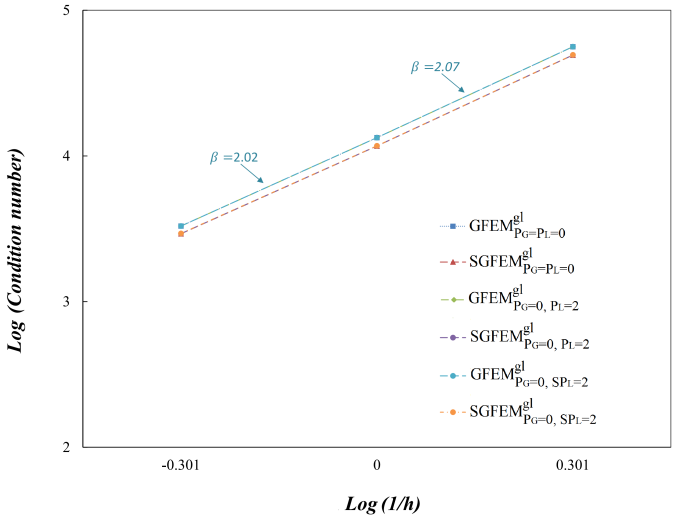
Figure 4.15: Relative error in strain energy against the inverse of element size (1/h) for global problem (third step) for enriching only crack-tip node in the cases of both G/XFEM^{gl} and SG/XFEM^{gl}. β refers to the convergence rate.



(a) G/XFEM^{gl} and SG/XFEM^{gl} for $P_G = 2$



(b) G/XFEM^{gl} and SG/XFEM^{gl} for $SP_G = 2$



(c) G/XFEM^{gl} and SG/XFEM^{gl} for $P_G = 0$

Figure 4.16: Condition number against the inverse of element size (1/h) for global problem (third step) for enriching only crack-tip node in the cases of both G/XFEM^{gl} and SG/XFEM^{gl}. β is the growth rate of the condition number.

with different enrichment strategies, against the inverse of element size. The preconditioner used is the Jacobi (or diagonal) preconditioner and the target error for PCG method was set equal to 1×10^{-12}. The results show that the SG/XFEM^{gl} method needs smaller number of iterations to solve the problem than G/XFEM^{gl} method, i.e. the SG/XFEM^{gl} method is faster than the G/XFEM^{gl} method, especially when h is equal to 0.5.

4.3.2.4 Enriching several nodes around the crack

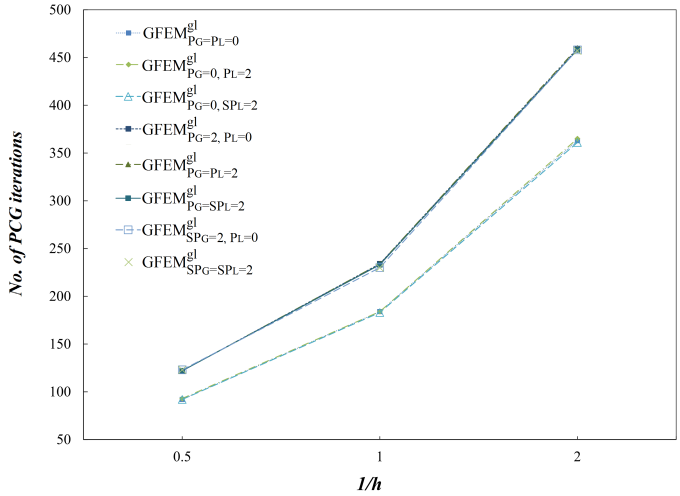
This section deals with the results obtained by enriching several nodes around the crack. Béchet et al (2005); Laborde et al (2005) showed that for a optimal convergence rate, the crack tip enrichment must be fixed within a predefined geometry. This approach with a fixed enrichment domain which is independent of the mesh size is called the *geometrical enrichment*. The geometrical enrichment can be used to achieve the optimal convergence rate by enriching the elements located in an predefined area. Figure 4.18 shows location of enriched nodes for different mesh sizes using geometrical enrichment strategy. The number of enriched nodes for global element size of $h = 2.0$, 1.0, and 0.5 are equal to 10, 27, and 85, respectively.

It was shown in section 4.3.2.3 that having polynomial/stable polynomial enrichment for global problem along with global-local enrichment leads to lower convergence rate. Thus, only the local problem will be enriched with polynomial/stable polynomial functions in this section.

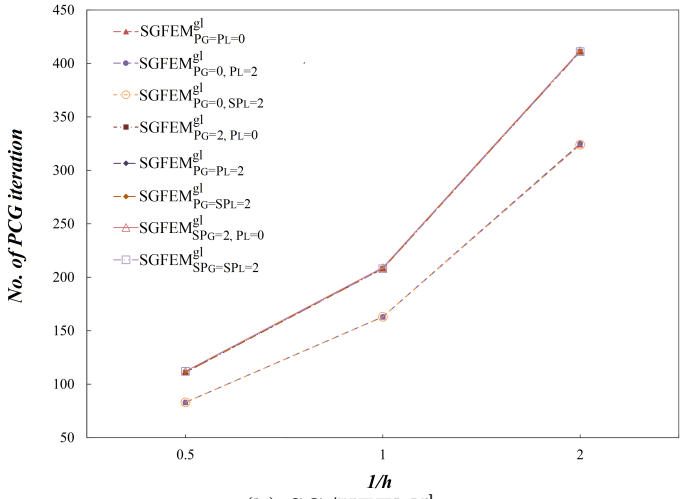
The convergence rate results against the element size shows an improvement for the geometrical enrichment approach compared to those results shown before for crack-tip node enrichment, when the results from Fig. 4.19 are compared with the ones from Fig. 4.15. Specially, for SG/XFEM^{gl}, the difference between the convergence rate is meaningful. In fact, expected convergence rate for a problem with a crack, in terms of strain energy, is $\mathcal{O}(h^{0.5})$ Szabo and Babuška (1991). On the other hand, if the approximation is able to represent the non smooth behavior of the solution, the optimal convergence can be recovered. In the case of a linear finite element problem, this optimal convergence rate is $\mathcal{O}(h^1)$. As it can seen in Fig. 4.19, the convergence rate for G/XFEM^{gl} method, considering all enrichment cases, is about 0.3 while for SG/XFEM^{gl} is between $0.5 - 1.2$.

The condition numbers for global-local GFEM/SGFEM methods for geometrical enrichment are illustrated in Fig. 4.20. When geometrical enrichment is used, condition numbers corresponding to the G/XFEM^{gl} for different enrichment cases presents a growth rate of about $3.5 - 3.8$, while for SG/XFEM^{gl} is about $2.6 - 2.8$. These values are in accordance with the values obtained for the standard G/XFEM and S/XGFEM by Gupta (2014).

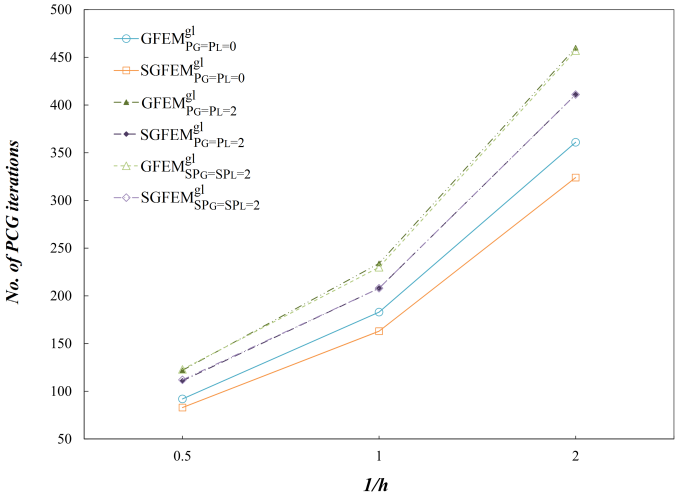
The number of PCG iterations required for both global-local G/XFEM and global-local SG/XFEM to solve the problem based on geometrical enrichment is presented in Fig. 4.21. It can be seen that the number of required PCG iterations increases as the mesh is refined for the current enrichment strategy. Again, the number of iterations for SG/XFEM^{gl} is smaller



(a) G/XFEM^{gl}



(b) SG/XFEM^{gl}



(c) both G/XFEM^{gl} and SG/XFEM^{gl} for $P_G = 0$,$SP_L = 2$, $P_G = P_L = 2$, and $SP_G = SP_L = 2$

Figure 4.17: Number of iterations for Preconditioning conjugate gradient method vs. the inverse of element size (1/h) for global problem (third step) for enriching only crack-tip node in the cases of both G/XFEM^{gl} and SG/XFEM^{gl}.

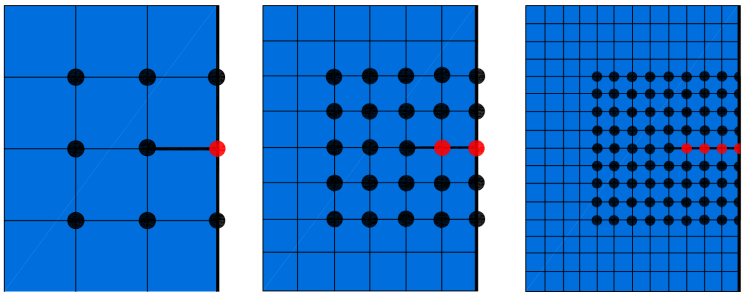


Figure 4.18: Enriched nodes in the local region from global domain using geometrical enrichment strategy for element size of: $h = 2.0$ (left picture), $h = 1.0$ (middle picture), and $h = 0.5$ (right picture). Again, the red marker indicates that there are two overlapped nodes there.

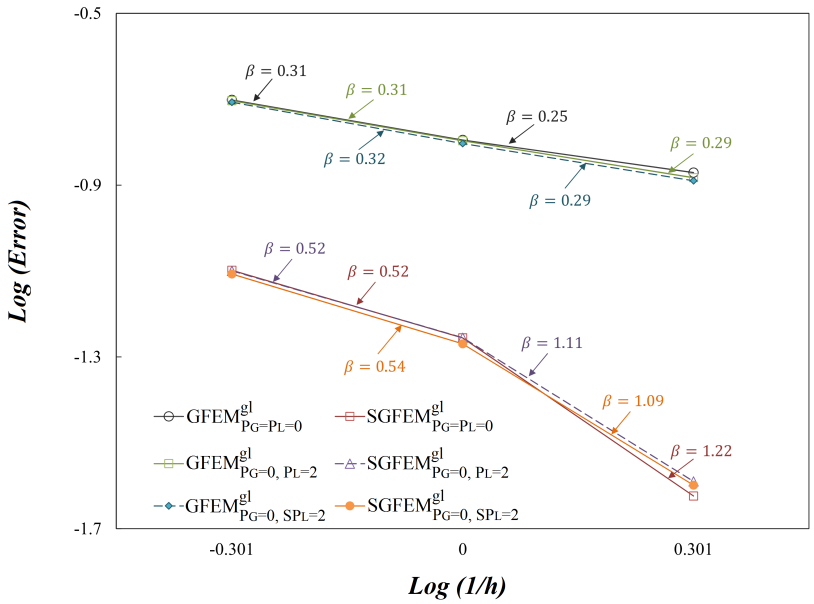


Figure 4.19: Strain energy error vs. the inverse of element size (1/h) for global problem (third step) in the case of geometrical enrichment. β refers to the convergence rate.

than G/XFEM^{gl} method which means that stable strategy is faster than standard G/XFEM strategy.

4.3.3 Beam with a crack

In this section a beam under distributed load is analyzed using both G/XFEM^{gl} and SG/XFEM^{gl} enrichment strategies. Figure 4.22 shows the geometry and related data of the beam. The reference solution of this problem is obtained using a mesh of 21912 quadrilateral elements (CPS4, a 4-node bilinear plane stress quadrilateral element) in $ABAQUS^{®}$. Similar to the previous example, there are three different element sizes (h) for global mesh in this study, h = 20, 10, and 5. Therefore, the total number of elements are 125, 500, and 2000 elements with regular distribution considering these three element sizes, as shown in Fig. 4.23. A local domain with 120×80 is used and it is composed by 24, 96, and 384 elements for global

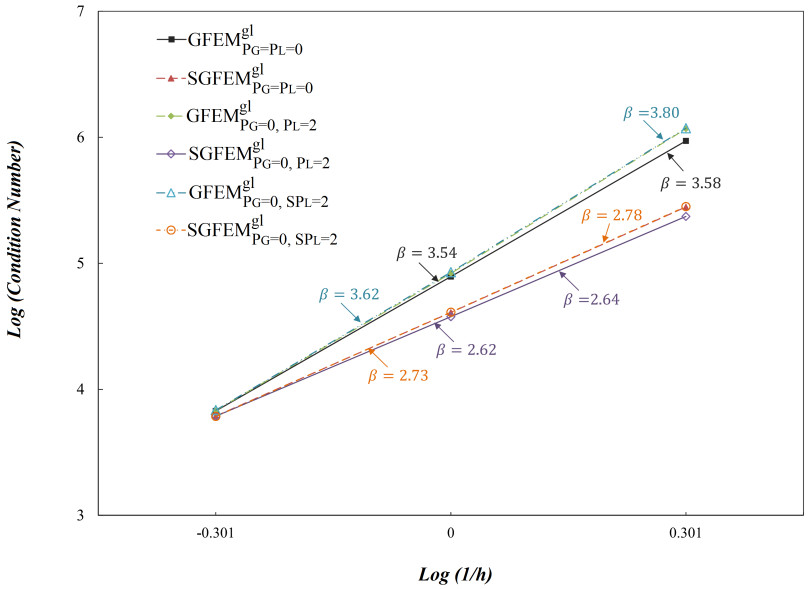


Figure 4.20: Condition number against the inverse of element size (1/h) for global problem (third step) in the case of geometrical enrichment. β is the growth rate of the condition number.

element size of $h = 20, 10$, and 5, respectively, as shown by red elements in Fig. 4.23. In addition, the local mesh is fixed for all three global meshes and it consists of 408 elements. The local mesh construction is exactly the same as the previous example, except the use of L3 refinement (with $f = 30\%$ reduction rate) for the geometric mesh. The penalty parameter is chosen equal to 1×10^{10}.

Also, the nodes to be enriched by global-local strategy are shown in Fig. 4.24, defining a fixed enriched domain, as it is expected in the geometrical enrichment. The number of them are 21, 65, and 225 for $h = 20, 10$, and 5, respectively.

Results of strain energy for three global meshes are presented in Table 4.3. These results are obtained for global problem (first step in global-local strategy) with no enrichment.

Table 4.3: DOFs and strain energy for different mesh size in the global problem of the beam problem

Analysis	DOFs	Strain energy	% of error in strain energy
Reference	44644	0.285375	—
125 elements	310	0.200738	29.66
500 elements	1122	0.219133	23.21
2000 elements	4246	0.234318	17.89

Figure 4.25 shows the results of error in strain energy and the rate of growth in condition number against the inverse of the element size. As it can seen in Fig. 4.25(a), the convergence rate for G/XFEM^{gl} method, considering all enrichment cases, is about between $0.2 - 0.4$ while for SG/XFEM^{gl} is between $0.5 - 0.9$. The condition numbers for global-local

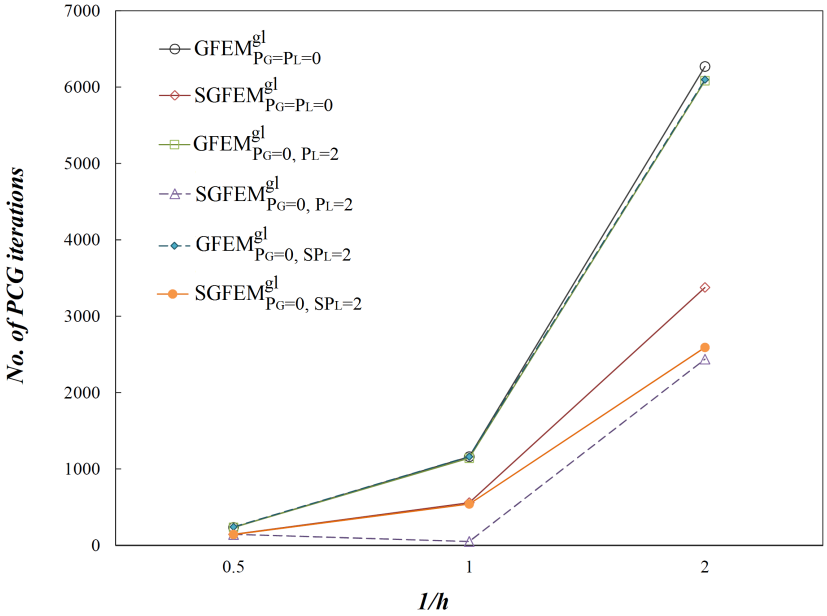


Figure 4.21: Number of PCG iterations against the inverse of element size (1/h) for global problem (third step) for geometrical enrichment.

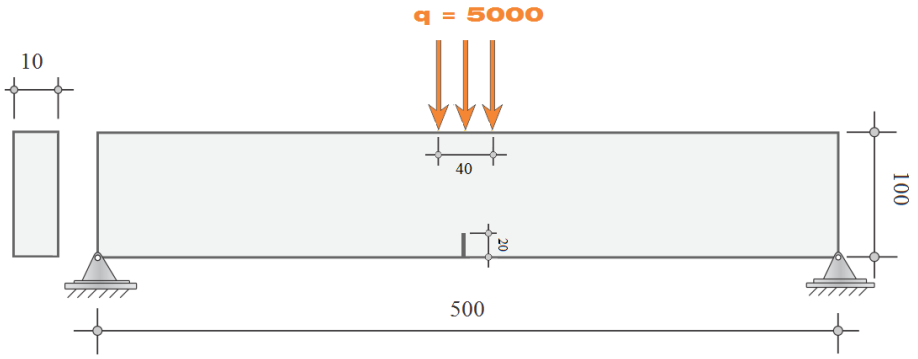


Figure 4.22: *Geometry and loading of a beam with a crack. With (in consistent units): Young modulus* $E = 200 \times 10^9$*, Poisson's ratio* $\nu = 0.3$.

GFEM/SGFEM methods are illustrated in Fig. 4.25(b). The rate of growth in condition number for G/XFEM^{gl} for different enrichment cases is about $4.0-5.7$, while for SG/XFEM^{gl} is about $2.0-2.6$. These values of convergence rates and rate of growth in condition number are close to the values presented for the standard G/XFEM and S/XGFEM in section 4.3.2.4.

4.4 Summary and Conclusion

The aim of this chapter was to present the implementation of stable G/XFEM for the IN-SANE computational framework. The validation of these implementations and some additional conclusions about the SG/XFEM for different enrichment strategies were presented by numerical examples for solid mechanics. The SG/XFEM method is implemented and tested

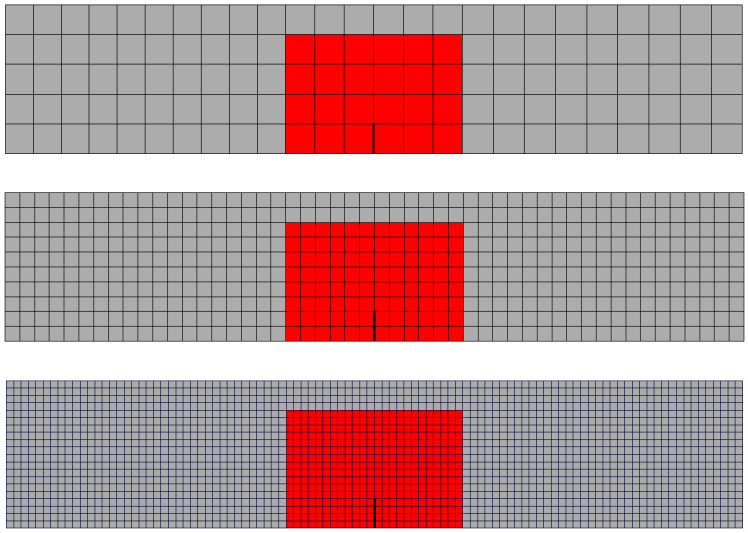


Figure 4.23: Schemes of three global meshes for the beam example. Red elements represent the local domain.

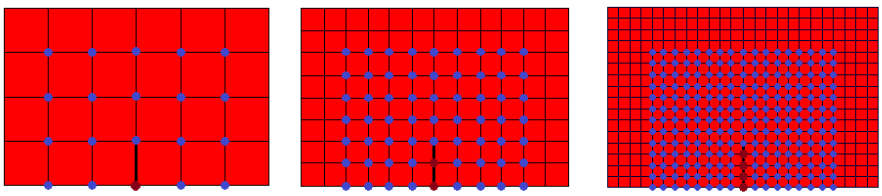


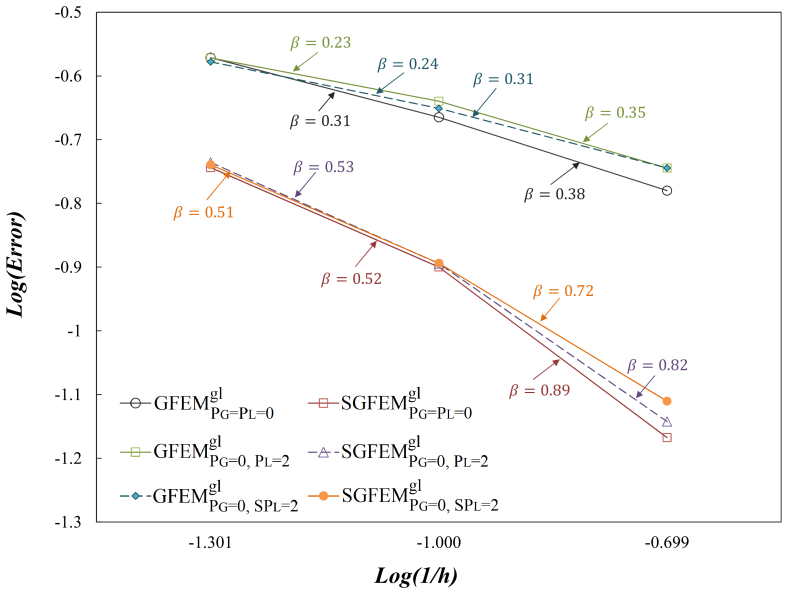
Figure 4.24: Enriched nodes using global-local enrichment for element size of: $h = 20$ (left picture), $h = 10$ (middle picture), and $h = 5$ (right picture). Again, the brown marker indicates that there are two overlapped nodes there.

for both standard and global-local approaches. In all enrichment cases, the SG/XFEM delivers better convergence rates in strain energy. The following are the main conclusions obtained from this chapter:

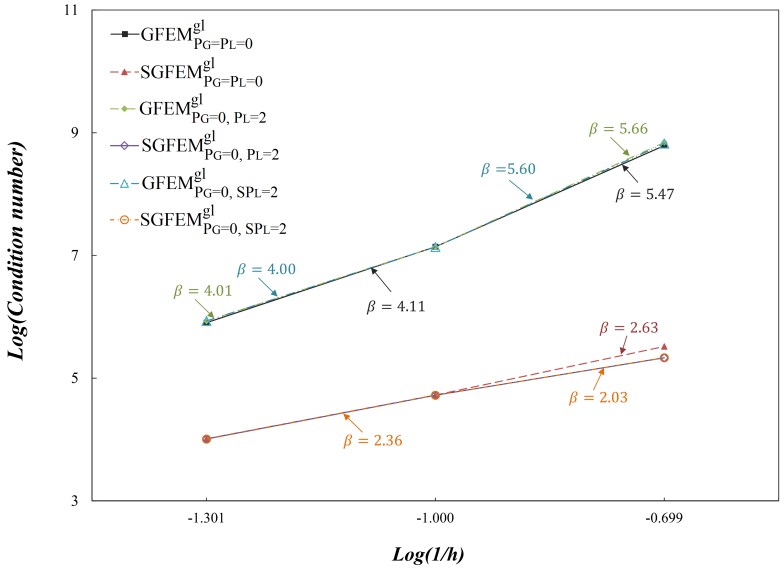
- The stable G/XFEM for both polynomial and singular enrichment shows a better convergence rate. The rate of growth in condition number for these two cases were almost the same, because a small number of nodes were enriched with the G/XFEM and SG/XFEM methods. If more nodes are enriched, a comparable difference would be extracted from these two strategies.

- Two enrichment strategies were adopted here, enriching the crack-tip node and enriching several nodes in a fixed region around the crack (a geometrical enrichment), along with different polynomial enrichment cases. The geometrical enrichment strategy was substantially superior in terms of accuracy, which is consistent with researches presented before in (Laborde et al, 2005; Béchet et al, 2005; Rojas-Diaz et al, 2011). In all cases, geometrical enrichment performs quite better, specially for SG/XFEM^{gl}.

- The conditioning of SG/XFEM^{gl} is near to G/XFEM^{gl} in the case of enriching crack-tip only, but it becomes considerably lower than G/XFEM^{gl} in the case of geometrical enrichment. Thus, it can be inferred that the SG/XFEM^{gl} can deliver a more stable



(a) Relative error in strain energy



(b) Rate of growth in condition number

Figure 4.25: Relative error in strain energy against the inverse of element size (1/h) for global problem (third step) for geometrical enrichment in the cases of both G/XFEM^{gl} and SG/XFEM^{gl}. β refers to the convergence rate.

results than G/XFEM^{gl}, even when different polynomial enrichments are used for both global and local problems.

– Fries proposed a method in (Fries, 2008), called corrected XFEM, which may lead to ill-conditioned system matrices if more than one enrichment function are employed. Here, in addition to global-local enrichment function, various cases of polynomial enrichment function are also considered to assess the effect of combined enriching strategy on the behavior of the numerical analysis. When polynomial enrichments are used in global problem, one can observe the bad convergence rate in the case of enriching only the crack tip node.

Chapter 5

CRACK PROPAGATION MODELING USING CLASSICAL G/XFEM

5.1 Introduction

A numerical implementation of generalized/extended finite element method (G/XFEM) to analyze a fractured structure, under the Linear Elastic Fracture Mechanics (LEFM) approach, is presented in this chapter. A discontinuous function along with the asymptotic crack-tip displacement fields are used to represent the crack without explicitly meshing its boundaries. Generally speaking, the enrichment functions can be continuous, discontinuous or numerically-built (global-local) functions. One of the most important parameters in fracture mechanics is the determination of the crack propagation direction under mixed mode conditions. In the concept of linear elastic fracture mechanics, the stress intensity factor can be used to either determine the crack propagation direction or propagation status, i.e. the crack can start to propagate or not. This chapter presents a fracture modeling using an object-oriented based implementation of two-dimensional problems using G/XFEM method. A domain-based interaction energy integral is used to extract the stress intensity factor for different fracture modes. Also, maximum circumferential stress criterion is selected for calculation of the crack propagation direction. Accuracy and robustness of the implementation are presented in detail by solving different linear elastic fracture mechanics problems for two cases: plane stress and Reissner-Mindlin plate problem. The numerical results are compared with the reference solutions from the analytical, numerical and the literature. All the explanations and numerical results for this chapter are brought from Malekan et al (2017c,d,a).

5.2 Representation of Fixed and Moving Discontinuities

This section presents the procedure of discontinuity modeling in order to analyze a static crack propagation problem. The term crack growth or crack propagation here is referred to *quasi-static crack growth*, in which inertia effects are neglected. In this approach, the problem is assumed in equilibrium at all time steps.

5.2.1 Crack Representation Procedure

The traditional approach to analyze a problem with discontinuity is to generate a mesh conform to the line, or surface, of discontinuity Γ_c in Fig. 5.1. However, in the G/XFEM the discontinuity along Γ_c may be modeled using special enrichment functions that describe the discontinuity as well as the stress field behavior close the the crack tip. In this case, the appropriate enrichment function must be selected and applied to those nodes that are around/close to the discontinuity surfaces. *The signed distance function along with the so-called Heaviside function are used here to represent the discontinuity in a model.* For linear elastic fracture mechanics, the crack-tip singularity can be captured with either singular enrichment (Eqs. (2.12) and (2.13)) or near-tip enrichment function (Eq. (2.14)). Following subsections are presented in detail the procedure of modeling a discontinuity within a problem using the G/XFEM approach and also the corresponding formulation that is used for this research.

5.2.1.1 The Signed Distance Function

The level-set method is a numerical tool for the tracking of the moving interfaces (Sethian, 1999). The signed distance function is one particularity of the level-set method which is used to represent the moving interface and also is one dimension higher than the interface dimension.

Consider a domain Ω divided into two non-overlapping domains Ω_A and Ω_B, sharing an interface, or surface of discontinuity, denoted by Γ_c, as shown in Fig. 5.1. The signed distance function is defined for the representation of the interface position as:

$$\phi(x) = \parallel \boldsymbol{x} - \boldsymbol{x}^* \parallel \text{ sign}\left(\boldsymbol{n}_{\Gamma_c} \cdot (\boldsymbol{x} - \boldsymbol{x}^*)\right) \tag{5.1}$$

where $\boldsymbol{x}^*$ is the closest point projection of $\boldsymbol{x}$ onto the discontinuity Γ_c, and $\boldsymbol{n}_{\Gamma_c}$ is the vector normal to the interface at point $\boldsymbol{x}^*$. In this definition, $\parallel\ \parallel$ denotes the Euclidean norm, where $\parallel \boldsymbol{x} - \boldsymbol{x}^* \parallel$ specifies the distance of point $\boldsymbol{x}$ to the discontinuity Γ_c (Fig. 5.1).

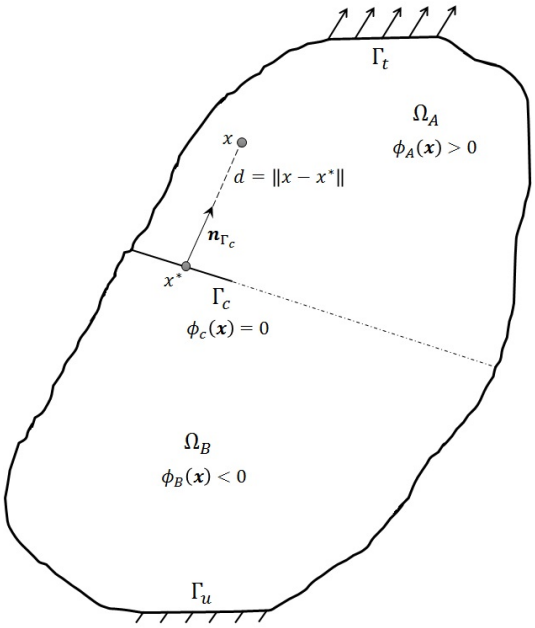


Figure 5.1: Signed distance function $\phi(\boldsymbol{x})$ representing a two-dimensional crack.

5.2.1.2 The Heaviside Function

A strong discontinuity is defined as a jump in the displacement field. The discontinuity in the displacement occurs where the displacement of one side of the crack is completely different from the displacement field of the other side. In such cases, the kinematics of the strong discontinuity can be defined based on the Heaviside function Belytschko et al (2001). This function is one of the most popular functions used to model the crack discontinuity in the G/XFEM formulation and is defined as:

$$H(\boldsymbol{x}) = \begin{cases} 1 & \text{if } \phi(\boldsymbol{x}) > 0 \\ 0 & \text{if } \phi(\boldsymbol{x}) < 0 \end{cases} \tag{5.2}$$

in which $\phi(\boldsymbol{x})$ is the signed distance function, defined in Eq. (5.1), and the discontinuity can be represented using this function as, according to Fig. 5.1:

$$\phi(x) = \begin{cases} > 0 & \text{if } \boldsymbol{x} \in \Omega_A \\ = 0 & \text{if } \boldsymbol{x} \in \Gamma_c \\ < 0 & \text{if } \boldsymbol{x} \in \Omega_B \end{cases} \tag{5.3}$$

In a more general case such as that shown in Fig. 5.2 which is considered in our implementations, the crack tip will not coincide with an element edge, and in this instance the discontinuity cannot be adequately described using only a function such as $H(\boldsymbol{x})$. The jump enrichment of the circled nodes in this case only provides for the modeling of the discontinuity until point P. To seamlessly model the entire discontinuity along the crack, the squared nodes are enriched with the asymptotic crack tip functions with the technique developed in (Belytschko and Black, 1999). The approximation for the case of an arbitrary crack, as shown in Fig. 5.2, takes the form:

$$\tilde{\boldsymbol{u}}(\boldsymbol{x}) = \sum_{j\in J} \mathcal{N}_j(\boldsymbol{x})\boldsymbol{u}_j + \sum_{i\in I} \mathcal{N}_i(\boldsymbol{x}) H(\boldsymbol{x})\boldsymbol{b}_i + \sum_{k\in K_1} \mathcal{N}_k(\boldsymbol{x}) \left(\sum_{l=1}^{n} C_k^{l1} L_l^1 \right) + \sum_{k\in K_2} \mathcal{N}_k(\boldsymbol{x}) \left(\sum_{l=1}^{n} C_k^{l2} L_l^2 \right) \tag{5.4}$$

in which J is the set of all nodes, I is the set of nodes enriched with Heaviside function, K_1 and K_2 are the sets of nodes to be enriched for the first and second crack tip, respectively. The function $L_l^1(\boldsymbol{x})$ and $L_l^2(\boldsymbol{x})$ are the crack tip enrichments that can be either the singular enrichment of Eqs. (2.12) and (2.13) or near tip enrichment function (Eq. (2.14)), and n is number of enrichment functions used.

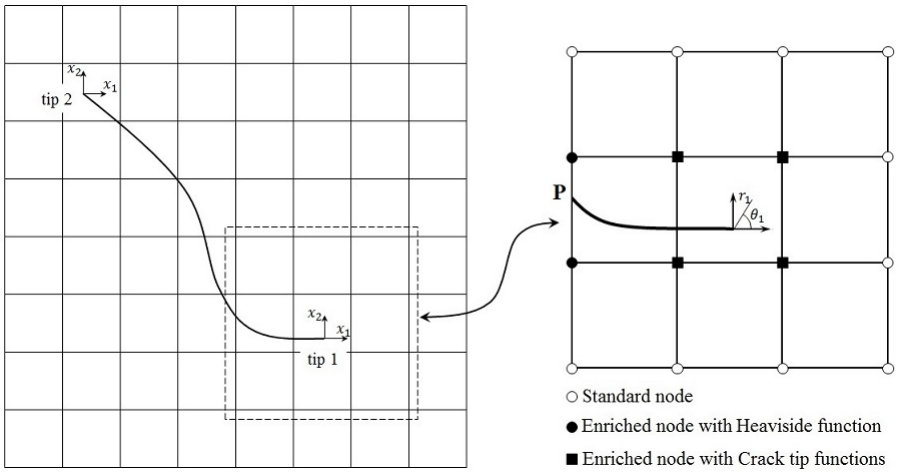


Figure 5.2: Local axes for the polar coordinates at the crack tips for an arbitrary crack shape and types of nodes in a general case.

5.2.1.3 Node Selection for Enrichment Strategy

Heaviside enrichment function enriches the nodes that belongs to those element that are completely cut by the discontinuity. According to (Dolbow, 1999), a direct use of this approach could provide an ill-conditioned stiffness matrix. Consider a crack/discontinuity cutting through some elements, as shown in Fig. 5.3. Since the crack doesn't cross through element E from Fig. 5.3(a), nodes i and j are enriched by Heaviside function whereas nodes k and l are not enriched. In other case, if crack crosses the element E completely, then all nodes must be enriched by the Heaviside function. However, the classical and enriched shape functions at these nodes will only differ in the very thin band of width ε (Fig. 5.3(b)), leading to ill-conditioned system of equations. This happens because the resulting basis functions are almost identical.

In this particular situation, nodes k and l must not be enriched by the Heaviside function. To overcome this situation, a criterion is defined by Dolbow (1999) in which, for a certain node j (see Fig. 5.3(c)), if the values of $A^+/(A^+ + A^-)$ or $A^-/(A^+ + A^-)$ are smaller than the allowable tolerance value of 10^{-4}, the node must not be enriched. The A^+ and A^- are the area of the influence domain of a node above and below the crack, respectively.

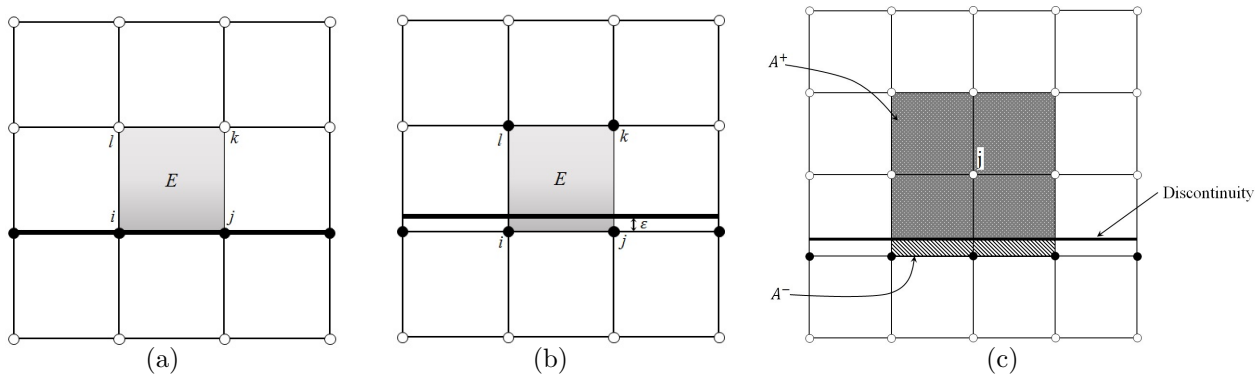


Figure 5.3: Node selection strategy: (a) the crack is aligned with a mesh, (b) the crack is almost aligned with a mesh, and (c) the criterion for enriching node j based on the area of the influence domain of node j.

5.2.2 Criteria for Mixed-Mode Crack Propagation

In crack propagation problems, there are two main requirements at each time step: crack propagation status and propagation direction. The crack propagation criteria may be a function of the stress intensity factors (SIFs), the strain energy release rate, the strain energy density, and so on. The direction of the crack can be determined based on the fracture toughness of brittle material, which is usually measured in a pure mode-I loading conditions by K_{IC}.

For the determination of crack kinking angle, researchers have introduced several criteria. Some of these criteria determine the crack growth direction based on stresses and strains field at the crack tip. These criteria generally give acceptable results for LEFM, such as maximum principal stress, maximum circumferential stress (Erdogan and Sih, 1963), and maximum strain (Maiti and Smith, 1984). However, for non-linear fracture mechanics more complicated methods are needed such as criteria that determine the crack growth direction based on energy distribution on a cracked body. The most commonly used criterion is the maximum strain energy release rate (Hussain et al, 1974). Some other criteria are based on the nature of crack creation such as criteria that use micro-void continuum damage for determination of crack growth direction. In these theories, the crack growth is controlled by the creation and propagation of micro-voids in the vicinity of the crack tip. Therefore, the crack propagates in the direction that most of the voids have been nucleated (Gurson, 1997; Biglari and O'Dowd, 2002). In this work, the maximum circumferential tensile stress criterion is used to determine the crack direction angle. This theory was first presented by Erdogan and Sih (1963), based on the near the crack-tip state of stress. Based on this theory, and representing the stress field in term of polar coordinate (r, θ) with its center at the crack tip, the crack propagates perpendicularly to the direction of maximum tension, when $\sigma_\theta^{\max}$ reaches a critical material-dependent constant. In this case, the hoop stress reaches its maximum value on the plane of zero shear stress. The singular term solutions of stress at the crack tip can be used to determine the crack propagation angle, where the

shear stress becomes zero. Considering the mixed-mode loading conditions, the asymptotic crack tip circumferential stress can be defined in polar coordinate system as (Miranda et al, 2003; Khoei, 2015):

$$\sigma_r = \frac{1}{\sqrt{2\pi r}} \cos\frac{\theta}{2}\Big\{ K_I[1+\sin^2\frac{\theta}{2}] + \frac{3}{2}K_{II}\sin\theta - 2K_{II}\tan\frac{\theta}{2}\Big\}$$

$$\sigma_\theta = \frac{1}{\sqrt{2\pi r}} \cos\frac{\theta}{2}\Big\{ K_I\cos^2\frac{\theta}{2} - \frac{3}{2}K_{II}\sin\theta\Big\} \tag{5.5}$$

$$\tau_{r\theta} = \frac{1}{\sqrt{2\pi r}} \cos\frac{\theta}{2}\big\{ K_I\sin\theta + K_{II}[3\cos\theta - 1]\big\}$$

where K_I and K_{II} are stress intensity factors of mode-I and mode-II fracture, respectively, and r and θ are polar coordinate of a point with respect to the crack tip point, as previously shown in Fig. 5.2. The crack is represented in this work as a set of straight line segments that are connected to each other. It is necessary to compute the critical crack propagation angle, θ_c, and increment length, Δa, for the new propagation step, see Fig. 5.4. The critical angle can be determined by setting the derivative of the shear stress $\tau_{r\theta}$ with respect to θ equal to zero. Thus, the θ_c is as follows:

$$\theta_c = 2\arctan\frac{1}{4}\left[\frac{K_I}{K_{II}} \pm \sqrt{\left(\frac{K_I}{K_{II}}\right)^2 + 8}\right] \tag{5.6}$$

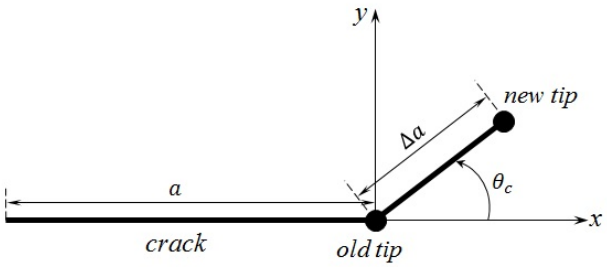


Figure 5.4: Representation of the crack extension and its new orientation, θ_c.

The result that gives the sign as opposite to sign of K_{II} is the correct one. Using the derivative of the shear stress with respect to θ in mode-I loading ($K_{II} = 0$), the crack propagation angle is zero. In mode-II loading, by solving the equation $K_{II}[3\cos\theta - 1] = 0$, the crack propagation angle is $\pm 70.5°$. So the maximum range of the crack propagation angle under linear elastic fracture mechanics approach is limited to an angle range of [$-70.5°$ to $70.5°$]. If $K_{II} > 0$, the crack growth direction $\theta_c < 0$, and if $K_{II} < 0$, the crack growth direction $\theta_c > 0$. An efficient expression of the critical angle of crack propagation can also be given as:

$$\theta_c = 2\arctan\left[\frac{-2K_{II}/K_I}{1+\sqrt{1+8(K_{II}/K_I)^2}}\right] \tag{5.7}$$

In addition, the crack increment length, Δa, should be chosen in such a way to have a reasonable and stable crack propagation procedure. According to (Huang et al, 2003), an

appropriate value must be chosen according to the type of crack propagation, i.e., straight or curved crack, and mesh size to have a reliable crack propagation path. Small values could help to obtain a better accuracy, however, if Δa is too small with respect to the element size, multiple changes in the direction of the crack path may occur which leads to a time consuming element partitioning for numerical integration.

5.3 Object-oriented Implementations

Following subsection present in detail the new implementations and also modifications of the existing classes corresponding to the crack growth process.

5.3.1 Persistence Interface

Persistence interacts with the input and output data. Figure 5.5 shows the UML diagram of *PersistenceAsXml* class which presents those methods that are modified or created in order to facilitate the crack growth procedure.

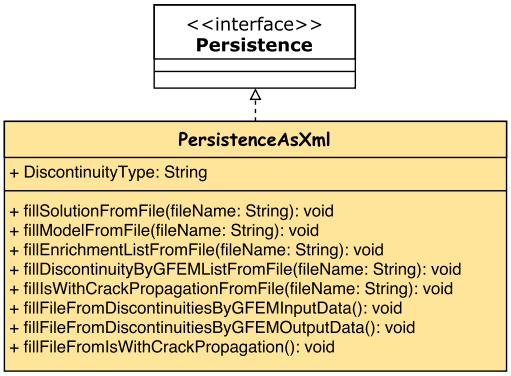


Figure 5.5: UML diagram of the *PersistenceAsXml* class

The method *fillDiscontinuityByGFEMListFromFile* and *fillIsWithCrackPropagationFromFile* are added to the *PersistenceAsXml* class aiming to provide initial information for crack propagation procedure, such as fracture modes and radius of the integral domain for SIF calculation, see A. In addition, *fillIsWithCrackPropagationFromFile* looks into the input file if the crack is stationary or can be propagated during the analysis (see Code 5.1). This code looks into the corresponding part of the XML file and informs the discontinuity type, whether it can be static or quasi-static crack. Also, in the case of crack propagation procedure, it informs if the problem is dealing with single or multiple cracks. Appendix B presents different parts of the *XML* input file for crack propagation procedure with the G/XFEM.

```
public void fillIsWithCrackPropagationFromFile(String fileName) throws Exception {
  try {
    OMElement root = readXmlFile(fileName);
    String ns = root.getNamespace().getNamespaceURI();
    String isWithCrackPropagation;
    String onlyOneCrack;
    String crackPropagationType;
    // Read "<Model ...>"
    OMElement modelElement = root.getFirstChildWithName(new QName(ns, "Model"));
    // Read "<IsWithCrackPropagation ...>"
    OMElement isWithCrackPropElement = modelElement.getFirstChildWithName(new QName(ns, "IsWithCrackPropagation"));
    if (!(isWithCrackPropElement == null)) {
      isWithCrackPropagation = (isWithCrackPropElement.getAttributeValue(new QName(null, "boolean"))).trim();
      ((GFemModel) model).setWithCrackPropagation(Boolean.parseBoolean(isWithCrackPropagation));
      // Read "<OnlyOneCrack ...>"
      OMElement xmlElement = isWithCrackPropElement.getFirstChildWithName(new QName(ns, "OnlyOneCrack"));
      if (!(xmlElement == null)) {
        onlyOneCrack = xmlElement.getText().trim();
        ((GFemModel) model).setOnlyOneCrack(Boolean.parseBoolean(onlyOneCrack));
      }
      // Read "<crackPropagationType ...>"
      xmlElement = isWithCrackPropElement.getFirstChildWithName(new QName(ns, "crackPropagationType"));
      if (!(xmlElement == null)) {
        crackPropagationType = xmlElement.getText().trim();
        ((GFemModel) model).setDiscontinuityType(crackPropagationType);
      }
    }
  }
}
```

Code 5.1: Code block of the *fillIsWithCrackPropagationFromFile(...)* method from *PersistenceAsXml* class

The method *fillSolutionFromFile* from the *PersistenceAsXml* class is responsible to get solution type definition and corresponding parameters from the XML input file. A small part fo this method is shown in Code 5.2, in which for either SteadyState, *StaticEquilibriumPath*, or GlobalLocal, it will initialize a *solution* object. The SteadyState class is responsible to solve static problems for FEM, G/XFEM, Hp-cloud, Mesh-free, etc., methods; the *StaticEquilibriumPath* class is responsible for the nonlinear analysis and quasi-static approach for classical FEM/G/XFEM methods (will be discussed in section 5.3.4).

```
public void fillSolutionFromFile(String fileName) throws Exception {
  try {
    OMElement root = readXmlFile(fileName);
    String ns = root.getNamespace().getNamespaceURI();
    OMElement solElement = root.getFirstChildWithName(new QName(ns, "Solution"));
    if ((solElement.getAttributeValue(new QName(null, "class"))).trim().equals("SteadyState")) {
      solution = new SteadyState();
    } else if ((solElement.getAttributeValue(new QName(null, "class"))).trim().equals("StaticEquilibriumPath")) {
      solution = new StaticEquilibriumPath();
    } else if ((solElement.getAttributeValue(new QName(null, "class"))).trim().equals("GlobalLocal")) {
      solution = new GlobalLocal();
      ((GlobalLocal) solution).setGlobalPath(fileName);
    }
    ...
  }
}
```

Code 5.2: Code block of the *fillIsWithCrackPropagationFromFile(...)* method from *PersistenceAsXml* class

5.3.2 Model Abstract Class

The *Model* abstract class contains the data of the discrete model and provides information for the *Assembler* to assemble the final matrix system (3.1). For current work, several classes are implemented or modified under this abstract class aiming to provide required information for the process. These classes are explained in the following subsections.

5.3.2.1 AnalysisModel Abstract Class

Analysis model is defined by different classes derived from the *AnalysisModel* abstract class, see Fig. 5.6. There are various models already implemented within INSANE code: space frame, plane stress/strain, plate and solid. Therefore, when the *Assembler* starts to assemble the system of equations (3.1), it asks the object of *AnalysisModel* for characteristics about the model, such as how the deformation matrix should be calculated. Beside this, the *GFemAnalysisModel* interface is created to have additional information for G/XFEM analysis. As it can be seen in Fig. 5.6, all three classes of *GFemPlaneStress*, *GFemPlaneStrain* and *GFemReissnerMindlinPlate* are derived from the *AnalysisModel* abstract class and simultaneously implement the methods from the *GFemAnalysisModel* interface.

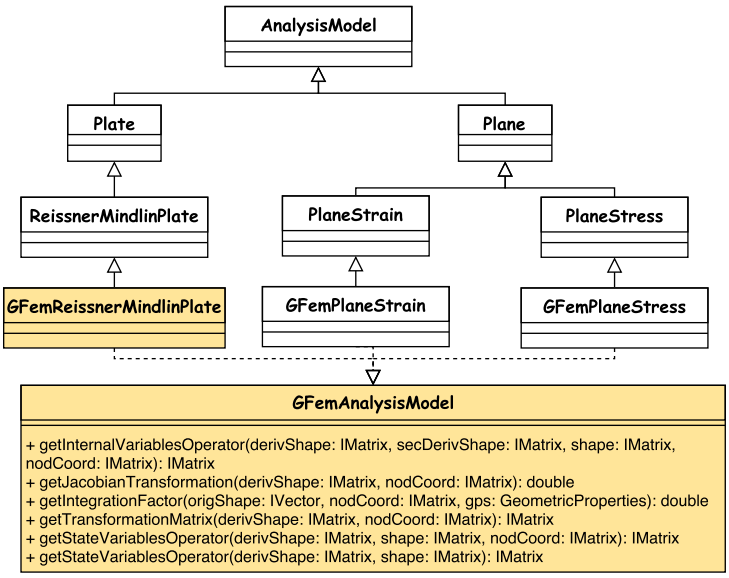


Figure 5.6: UML diagram of the *AnalysisModel* abstract class relations.

5.3.2.2 EnrichmentType Package

Figure 5.7 shows different classes from the *EnrichmentType* package. As it can be seen from this figure, there are various enrichment types that can be used either for classical elasticity problems or LEFM problems.

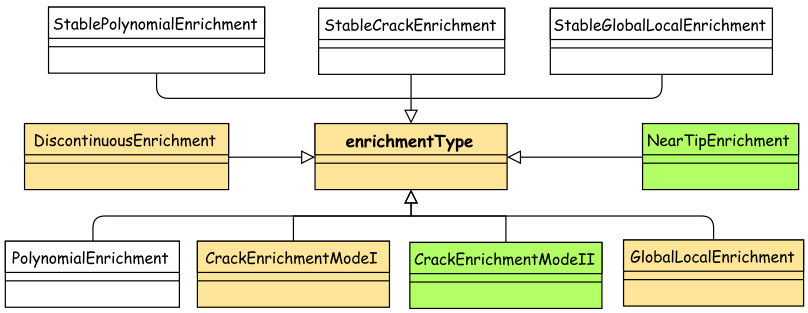


Figure 5.7: *EnrichmentType* package

CrackEnrichmentModeI and *CrackEnrichmentModeII* classes are responsible to enrich the problem with corresponding parts of Eqs. (2.12) and (2.13) regarding to mode-I and mode-II fracture. *NearTipEnrichment* contains all the information from Eq. (2.14). In addition,

CrackEnrichmentModeI, *CrackEnrichmentModeII*, *DiscontinuousEnrichment*, *NearTipEnrichment*, and *GlobalLocalEnrichment* classes are modified to have some necessary methods in order to facilitate their use in the crack propagation procedure. Furthermore, other classes such as the *StableCrackEnrichment* are available to facilitate the use of the stable G/XFEM approach. Figure 5.8 presents the *discontinuousEnrichment* class that contains the necessary information for the Heaviside function calculation. This function will be used along with *NearTipEnrichment* and/or *CrackEnrichment* to facilitate the crack propagation approach.

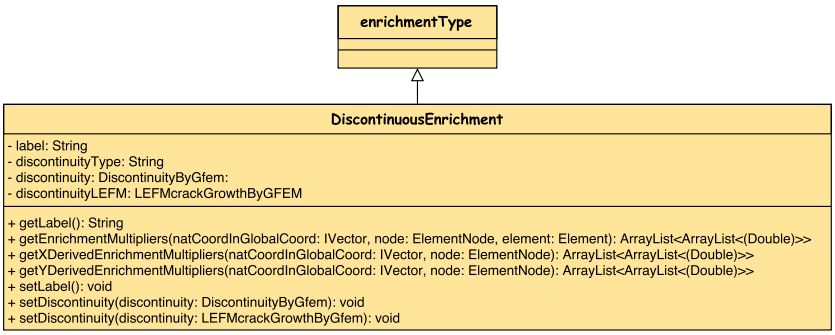


Figure 5.8: UML diagram of the *DiscontinuousEnrichment* class

5.3.2.3 StressIntensityFactors class

Figure 5.9 shows the structure of the *StressIntensityFactors* class. This class is mainly used to calculate the stress intensity factors, either for plane stress/strain or Reissner-Mindlin problems, specifically for **linear elastic materials**. The strategy used to calculate the SIF is the interaction energy integral. It returns the SIFs for different modes, so the crack propagation status and its direction angles can be calculated for the crack analysis process. This class is written based on the formulation from Appendix B and Refs. (Dolbow et al, 2000b,a).

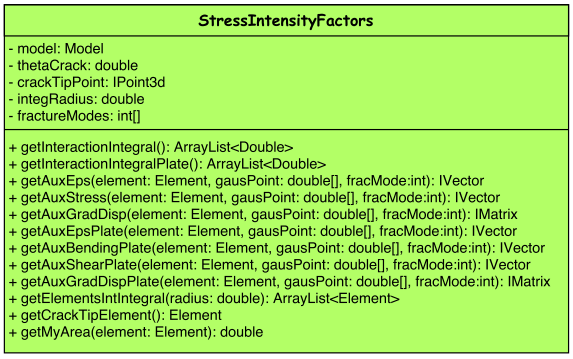


Figure 5.9: Structure of the *StressIntensityFactors* class

5.3.3 Assembler Interface

The task of the *Assembler* interface is to mount the linear equation system from Eq. (3.1), returning the stiffness matrix and its partitions related to free/restrained degrees of freedom. In order to better understand the relationship between the classes described before, the

process of assembling the stiffness matrix of the problem will be described in this section, as shown in Fig. 5.10, as the main part of the *Assembler* interface. An object of the *Solution* class presented as *"Actor"*, asks for one of its attributes, an *Assembler*, i.e., *GFemAssembler*, to perform the assembly of the stiffness matrix **C** (from Eq. (3.1)). The Assembler object, in turn, performs a loop in the list of elements that is stored in the class *GFemModel*, an attribute of the *Assembler*.

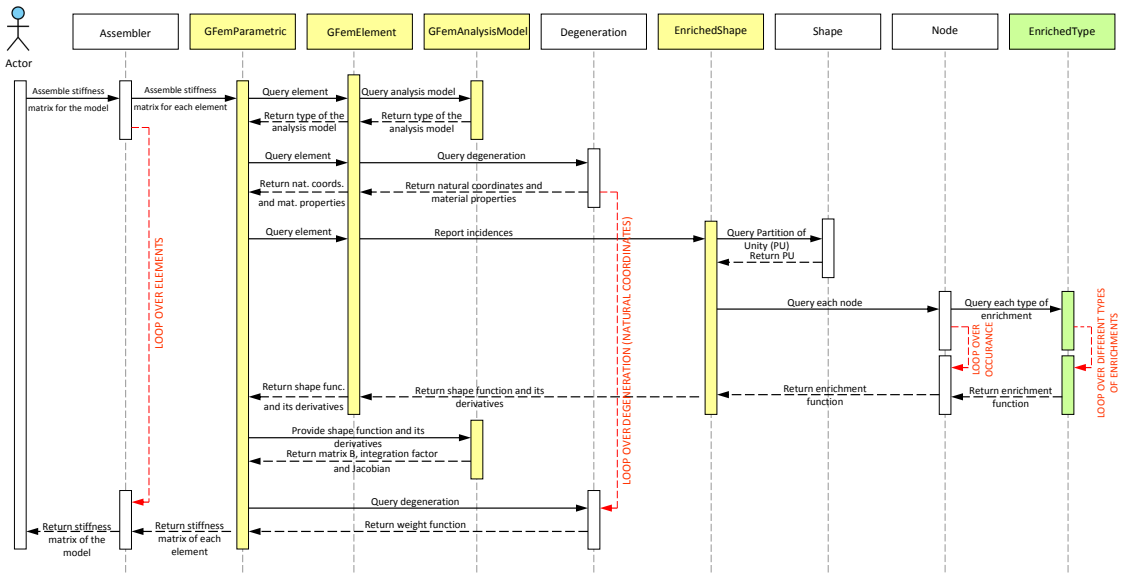


Figure 5.10: Sequence diagram for assembly of the stiffness matrix

Different finite element formulations originally provided within the INSANE environment are implemented by different classes derived from *SolidMech* abstract class (inherited from the *ProblemDriver* abstract class), each one with a specific way to calculate the different parts of the Eq. (3.1). Some of these classes are: parametric (class *Parametric*), Kirchhoff thin plate (class *KirchhoffThinPlate*), and frame (class *Frame*), as it can be seen in Fig. 5.11. The *GFemParametric* class is also designed to carry out the necessary tasks related to the G/XFEM analysis, along with its inheritance the *GFemPhysicallyNonLinear* class to facilitate the nonlinear analysis. Depending on the strategy used to solve the nonlinear analysis via *Newton-Raphson* method, the stiffness matrix can be elastic (*getC()* method), secant (*getTotalC()* method) and tangent (*getIncrementalC()* method). Therefore, in the sequence of stiffness matrix mounting, *Element* calls one of its attributes, the *GFemParametric* object which is responsible for constructing the element's contribution to the stiffness matrix. The *GFemParametric* queries *Element* to obtain certain information that will be used in the construction of the stiffness matrix of the element. The first required information is the type of the analysis model that is provided here by the *GFemAnalysisModel* object, either *GFemPlaneStress/GFemPlaneStrain* or *GFemReissnerMindlinPlate* types. The *KirchhoffThinPlate*, *Frame*, and *Parametric/GFemParametric* classes are derived from the *SolidMech* class since they use different approaches to mount the stiffness matrix. *Parametric/GFemParametric* use a numerical integration approach, while *KirchhoffThinPlate* and *Frame* cases use analytical formulation to mount the stiffness matrix, with respect to their

corresponding formulations. As a consequence, the *PlaneStress/PlaneStrain* and *ReissnerMindlinPlate* classes can use the *Parametric* class to numerically integrate and mount the stiffness matrix. Therefore, these plane/plate cases are derived from the *AnalysisModel* abstract class, as it was shown in Fig. 5.6, and not from *SolidMech* class. Another reason for this is that plane and plate cases can use the same approach to mount the stiffness matrix and only they degrees of freedom and procedure to provide the required derivatives matrices are different.

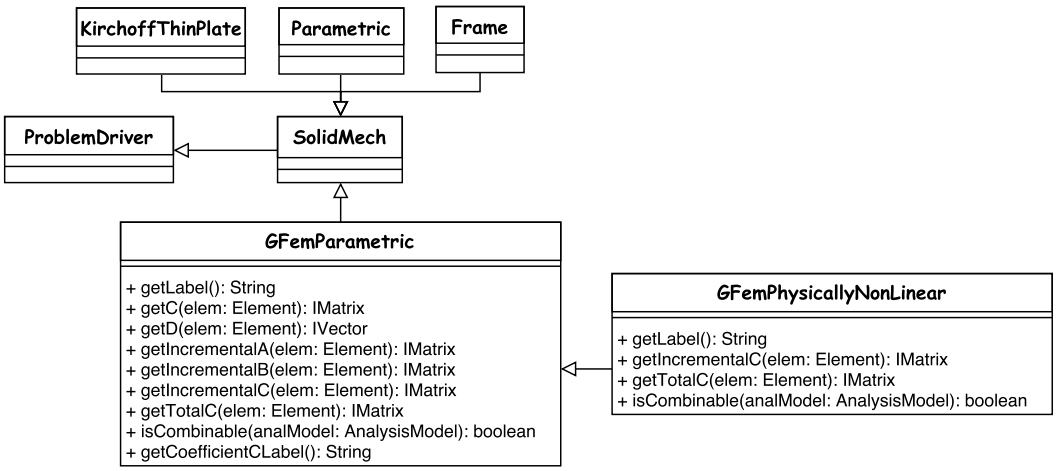


Figure 5.11: UML diagram of the *ProblemDriver* class

Afterwards, the *Element* object queries another attribute, the object *Degeneration* (see Fig. 5.10). The *Degeneration* object has stored the section and material properties and coordinates of the integration points. This list of integration points is used for a loop that runs through each integration point of the current element in order to calculate the portion of the stiffness matrix of the element at each integration point. At each step of this loop, the derivatives of the shape function and its enriched part must be evaluated. This is done by the *EnrichedShape* object, which is also an *Element* attribute. The derivatives of the shape function depends on PU from underlying FE mesh and enrichment types from G/XFEM approach, either continuous or discontinuous functions. The *EnrichedShape* object manages this dependency between these two parts.

For each node, the *EnrichedShape* object is evaluated at the corresponding integration point. The PU is given by a *Shape* object which is a member of the *EnrichedShape*. Enrichment functions are obtained from a list of *EnrichedType* objects, which is the attribute of the *Node* object. This list is stored in *GFemModel* and is accessed through a list of associated objects for the corresponding node. The original and derived shape functions, computed by the *EnrichedShape*, after a loop through the nodes of the element, are sent to the object *GFemParametric* which is responsible for building the array of the element stiffness. *GFemParametric* sends this information to the *GFemAnalysisModel* object, which provides the matrix of derivatives, integration factors, and the Jacobian for a specific analysis model. Finally, the *Degeneration* object is queried to provide the numerical integration weights for

a particular point of the integration. Thus, the *GFemParametric* object can return the stiffness matrix of the element, which uses this portion to form the stiffness matrix $\boldsymbol{C}$, from Eq. (3.1), of the problem.

5.3.4 Solution Abstract Class

Solution abstract class starts the solution process and has the necessary resources for solving the matrix system of the fracture analysis approach. As it can be seen in Fig. 3.4, it shows different classes derived from the class *Solution* to handle either **static** (*SteadyState*, *GlobalLocal*, *ThermoStructural*, *StaticEqulibriumPath*), **dynamic** (*DynamicEqulibriumPath*), or **modal analysis** (*ModalVibration*). The *StaticEqulibriumPath* class is responsible for the nonlinear analysis and it is used here to handle the quasi-static linear elastic fracture analysis based on the G/XFEM method.

Code 5.3 shows the main block (*execute()* method) of the *StaticEqulibriumPath* class. It starts with the solution initialization at step zero, then defining the proportional load for the upcoming steps. There is *do-while* loop that solves the problem step-by-step to reach the maximum number of steps defined by the user.

StaticEqulibriumPath class calls the *LEFMcrackGrowthByGFem* class through *GFemModel* class (line 37 from code 5.3), by calling *update()* and *evaluateCrackPropagationLEFM()* methods, see Fig. 5.12. The *LEFMcrackGrowthByGFem* and *DiscontinuityByGFem* classes are the main core for the quasi-static crack propagation approach based on the G/XFEM methodology for linear and nonlinear material models, respectively. The UML diagram of the *LEFMcrackGrowthByGFem* class is shown in Fig. 5.13. The *DiscontinuityByGFem* class is related to the cohesive crack propagation (Silva, 2016). If the crack propagation status is *true*, then the *update()* method from *GFemModel* class will start to analyze the problem and follow the procedure from SIF calculation (linear elastic material) and adding Heaviside and singular/near-tip enrichment functions for the corresponding nodes. The nodes selection procedure here is based on the definition from section 5.2.1.3. The *update()* method from the *GFemModel* class is presented in Code 5.4. If the solution procedure is quasi-static, i.e., there must be some crack propagation, then lines 4-13 from this code must be followed in the process. As an example, line 4 checks whether the problem type is nonlinear or not, i.e., if this is related to *DiscontinuityByGFem* (cohesive crack propagation) or *LEFMcrackGrowthByGFem* (crack propagation in linear elastic medium) cases. If not, it will call *evaluateCracksPropagation()* method, which is shown in the same block. This method will call *update()* method from *LEFMcrackGrowthByGFem* class which is responsible for the crack propagation process. The *discontinuities* variable is a list of objects from *LEFMcrackGrowthByGFem*/*DiscontinuityByGFem* that represent the cracks of the problem.

```
1 private Step step;
2 private IterativeStrategy[] iterativeStrategy;
3 private LoadCombination loadCombination;
4
5 public void execute() throws Exception {
6     int cont = 0;
```

```
7     step.addObserver(this);
8     step.setIterativeStrategy(iterativeStrategy[cont]);
9
10    step.getAssembler().setXp();
11    this.setCurrentStep(0);
12    step.setLabel(this.getCurrentStep());
13    step.update();
14    step.getAssembler().update();
15    this.setCurrentStep(1);
16
17    step.getAssembler().addLoading(loadCombination);
18    this.step.setProportionalLoad(this.getAssembler().setProportionalLoad());
19    this.step.setConstantLoad(this.getAssembler().setConstantLoad());
20
21    do {
22      if (step.getAssembler() instanceof GFemAssembler) {
23        if (((GFemModel) step.getAssembler().getModel()).isWithCrackPropagation()) {
24          this.step.setProportionalLoad(this.getAssembler().setProportionalLoad());
25          this.step.setConstantLoad(this.getAssembler().setConstantLoad());
26        }
27      }
28      iterativeStrategy[cont].setStep(step);
29      step.setLabel(this.getCurrentStep());
30      step.execute();
31      step.getAssembler().setIterationEnd();
32      if (!(step.getConvergence())) {
33        break;
34      } else {
35        LoadCombination lcomb = step.getAssembler().getModel().getLoadCombinationsList().get(0);
36        step.getAssembler().addLoading(lcomb);
37        step.getAssembler().update();
38        this.setCurrentStep(this.getCurrentStep() + 1);
39      }
40    } while (this.getCurrentStep() <= this.getNumMaxSteps());
41  }
```

Code 5.3: Main code block of the *StaticEqulibriumPath* class

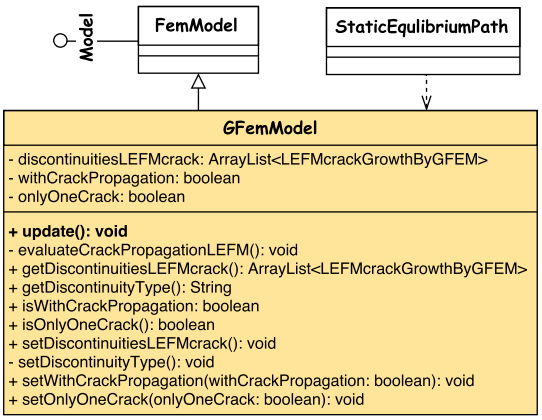


Figure 5.12: Structure of the *GFemModel* class

In the first step of the crack propagation procedure, the *PersistenceAsXml* class is used to fill the data, constructing an object of the *LEFMcrackGrowthByGFem* class and then the *buildNotch()* method is called in order to create the initial crack from the user-inserted data, using the Heaviside function.

```
1   public void update() throws Exception {
2     super.update();
3     if (withCrackPropagation) {
4       if(this.getDiscontinuityType().equals(DiscontinuityByGfem.class.getSimpleName())) {
5         this.evaluateCracksPropagation();//For nonlinear materials
6         if (!onlyOneCrack) {
7           this.evaluateCracksNucleation();
8         } else if (this.discontinuities.size() == 0) {
9           this.evaluateCracksNucleation();
10        }
11      } else if (this.getDiscontinuityType().equals(LEFMcrackGrowthByGfem.class.getSimpleName()))
            {
12        this.evaluateCracksPropagationLEFM();//For linear materials
13      }
```

```
14    }
15  }
16  protected void evaluateCracksPropagationLEFM() {
17    for (int i = 0; i < discontinuities.size(); i++) {
18      discontinuities.get(i).update();
19    }
20  }
```

Code 5.4: Code block of the *update()* method from *GFemModel* class

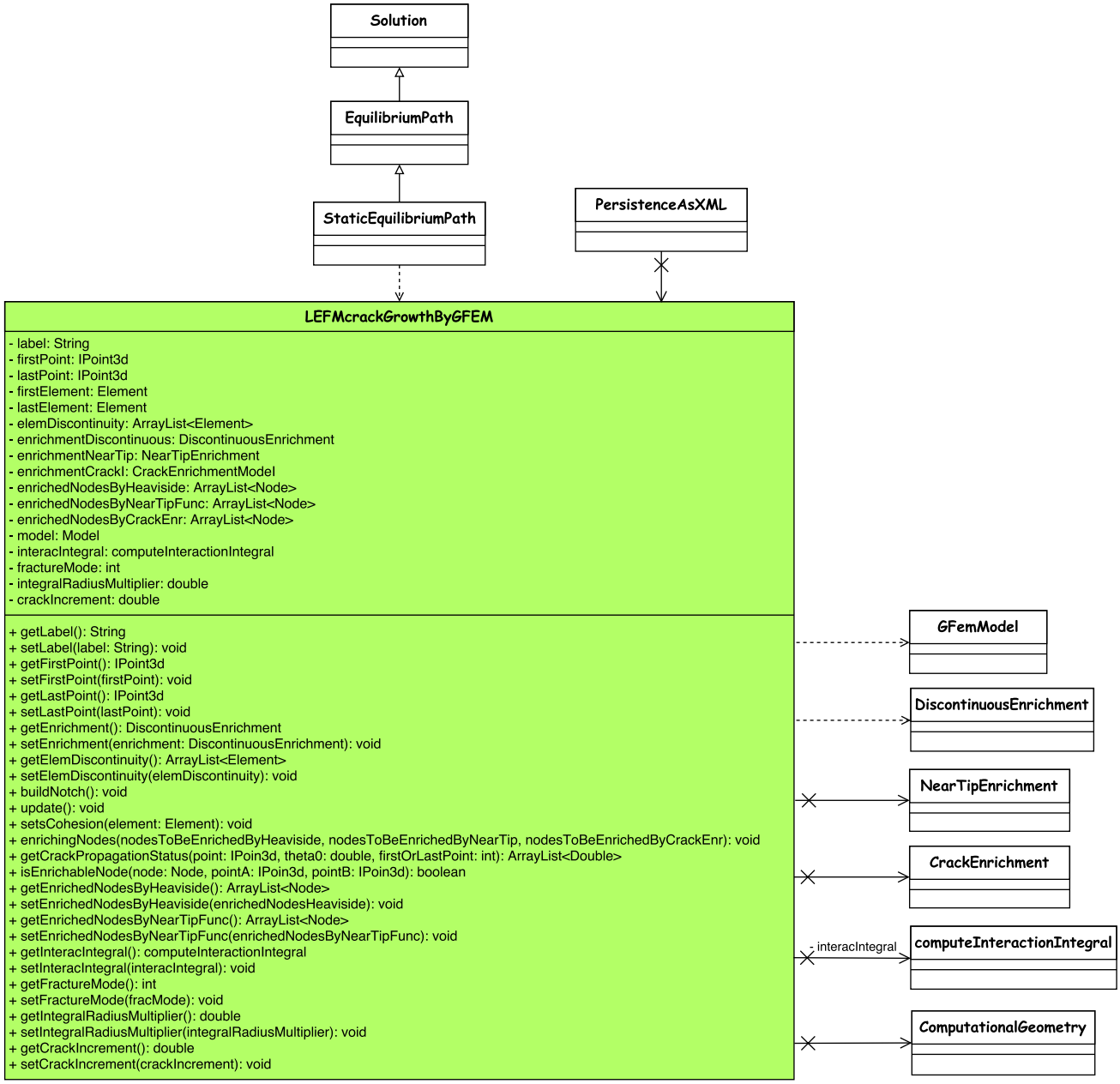


Figure 5.13: UML diagram of the *LEFMcrackGrowthByGFem* class

Moreover, the *stressIntensityFactors* and *LEFMcrackGrowthByGFem* classes use some methods from the *ComputationalGeometry* class, Fig. 5.14, in order to find either the elements inside of the interaction domain integral, intersections of the crack surface with element edges, node selection for Heaviside or singular/near-tip enrichments. For example, we need to find position of all Gauss points with respect to the discontinuity line inside of a nodal support, to see if that node has to be enriched with the Heaviside function or not, in accordance to

the section 5.2.1.3.

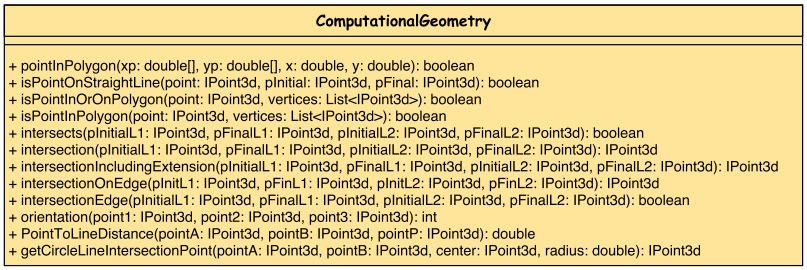


Figure 5.14: UML diagram of the *ComputationalGeometry* class

5.3.5 Crack Propagation Strategy Based on OOP Approach

Figure 5.15 shows the crack propagation modeling flowchart from the current implementation, in which the numbers specify the steps for crack propagation process at each time step. In addition, Code 5.3 showed the main block (*execute()* method) of the *StaticEqulibriumPath* class that is related to steps shown in Fig. 5.15. After defining the model, loading, material properties, and mesh discretization in step 1 and 2, the solution starts for each time step with a *do-while* loop (step 3 from Fig. 5.15 and line 21 from Code 5.3). The iterative approach (line 28 from Code 5.3) and convergence study (lines 32-33 from Code 5.3) is mainly defined for the nonlinear materials, and hence, they don not interfere for the linear materials, which is the case of this PhD thesis. The main solution process takes place with the *execute()* method at line 30 from Code 5.3.

The crack will be propagated if there is a pre-existing crack in the model, by calculating the stress intensity factors at step 4 from Fig. 5.15. Then, the propagation status will be evaluated at step 5 from Fig. 5.15, where the SIF obtained from the model will be compared with fracture toughness of the problem (K_c). If the calculated SIF is greater than or equal to this K_c, the crack can grow, if not it cannot grow under current loading conditions at the current step time. Afterwards, if the crack grows, its direction and also new crack-tip(s) will be calculated in step 6 from from Fig. 5.15. If the crack is not able to propagate at the current time step, the loop goes for the next time step to apply a bigger loading magnitude (step 5 from Fig. 5.15). At the step 7 from Fig. 5.15, if the new crack-tip crosses the problem boundaries, it won't be created and analysis will be followed to the next step until it reaches its final time step without any crack propagation. While, if the new crack-tip is inside of the problem boundaries, then the discontinuous enrichment will be used to include the new crack segment into the model (step 8 from Fig. 5.15) and the whole model will be updated with this new crack-tip (step 9 from Fig. 5.15). These whole steps, from 4 to step 9 of Fig. 5.15, are taken place in line 37 from Code 5.3, by calling the *update()* method from *GFemModel* class, as shown in Code 5.4. The *do-while* loop will be stopped when it reach its maximum time step number (step 10 from Fig. 5.15).

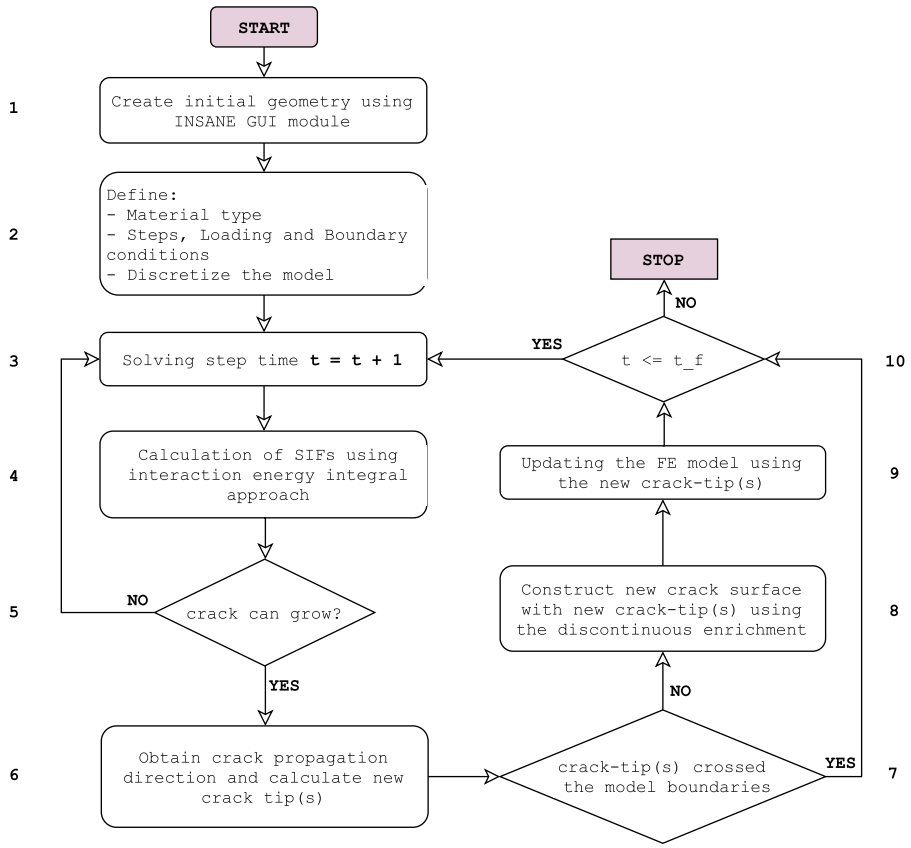


Figure 5.15: Flowchart of the current crack modeling implementation.

5.4 Numerical Examples

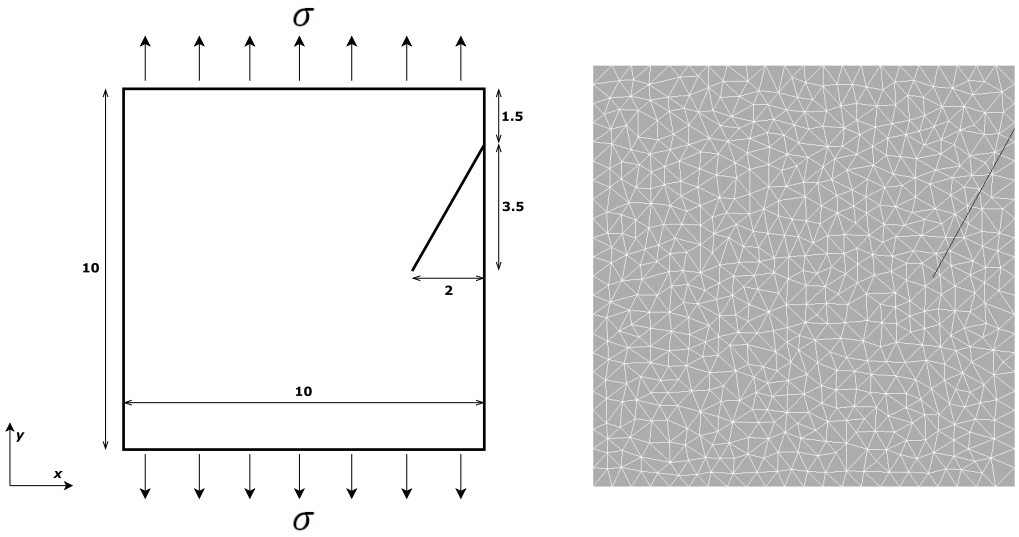
This section presents three linear-elastic problems in $\Re^2$. Section 5.4.1 shows a rectangular plate with an oblique crack, section 5.4.2 presents a plate with an inclined crack under tension loading and a Reissner-Mindlin plate with a crack is analyzed in section 5.4.3. The geometry and boundary conditions are very simple and the goal of choosing them is to demonstrate the capabilities of the G/XFEM method for quasi-static crack propagation process. The number of integration points that are used for these problem were selected big enough to accurately capture the crack prorogation direction within the element boundary, even for elements containing singular enrichment functions. For simplicity, the same number of integration points is used for the all elements with and without singular/Heaviside enrichment functions. The number of integration points for all two problems is considered equal to 8×8.

The domain size of the interaction integral is considered here (see section 5.3.2.3 and Appendix A) by a circle with radius r defined by $r = r_m\, h_{elem}$, in which the element characteristic length, h_{elem}, is the square root of the crack tip element area and r_m is a scalar multiplier (Moës et al, 1999). To have an accurate SIF results, one have to select a proper multiplier r_m. This scalar multiplier can be chosen by performing numerical experiments with different values to have an independent J-integral path. The scalar multiplier r_m for the following three problems is considered equal to 2.0, but the crack increment length has different values for each problem. In all examples, the Heaviside enrichment function (Eq.

(5.2)) is used to insert the discontinuity line within the model boundary, while the singular enrichment functions (Eqs. (2.12) and (2.13), or Eq. (2.14)) are used to capture the stress gradient around the crack tip(s).

5.4.1 A Rectangular Plane Stress Problem with an Oblique Crack

A rectangular plane stress with an oblique crack under a tension stress is shown in Fig. 5.16 along with the discretization mesh. The mesh is composed of 1202 triangular elements, each one with 13 Gauss points. Also, total number of DOFs are equal to 1268. This problem is solved using the single scale G/XFEM approach and its goal is to prove the functionality of the calculation procedure for the crack propagation direction. It is also used to show the possibility of use triangular elements for this kind of analysis. The problem has the following parameters (in consistent units): modulus of elasticity $E = 1.0$, Poisson ratio $\nu = 0.3$, and the tension stress $\sigma = 1.0$.



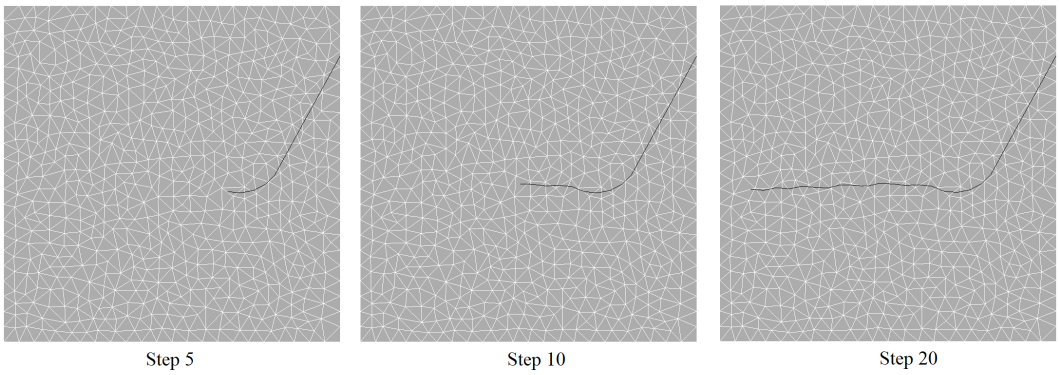
(a) The geometry of the problem. (b) Mesh discretization.

Figure 5.16: *A plate with an oblique crack. The tension stress is equal to* $\sigma = 1.0$.

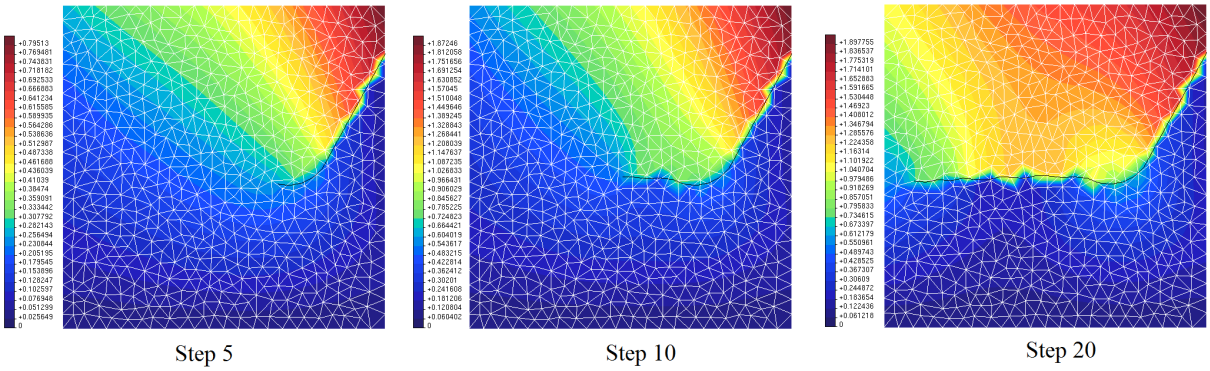
As it was expected, the crack propagates in the mode-I of fracture (cleavage mode) is almost horizontal for this problem, as it can be seen in Fig. 5.17(a). This path is perpendicular to the maximal principal stress which is vertical due to the type applied load shown in Fig. 5.16(a), and it is with accordance to the results from (Bouchard and F. Bay, 2003). In addition, Fig. 5.17(b) is shown the distribution of the displacement in y direction.

5.4.2 Inclined Crack Under Tension

This section presents the results for a problem with an inclined crack, as shown in Fig. 5.18. The objective of this problem is to illustrate the mixed-mode crack propagation using G/XFEM method. The problem is analyzed under plane stress state with the following parameters (in consistent units): modulus of elasticity $E = 1.0$, Poisson ratio $\nu = 0.3$, and



(a) Crack propagation path.



(b) Contour of the displacement in y direction.

Figure 5.17: *Results for the plate with an oblique crack.*

the tension stress $\sigma = 1.0$. The geometrical parameters of this problem are: $W = 3.0$, $2a = 0.35$, and $\beta = 48.5°$. The element size is equal to 0.25 for this problem with total of 144 elements, a uniform mesh of 12×12 elements. The crack increment length considered here is equal to 0.19. Displacement distributions in y direction along with crack propagation path are shown in Fig. 5.19. There are some small fluctuations in the crack propagation path, but this figure clearly shows the mixed-mode propagation path, in accordance with (Ayatollahi and Aliha, 2009).

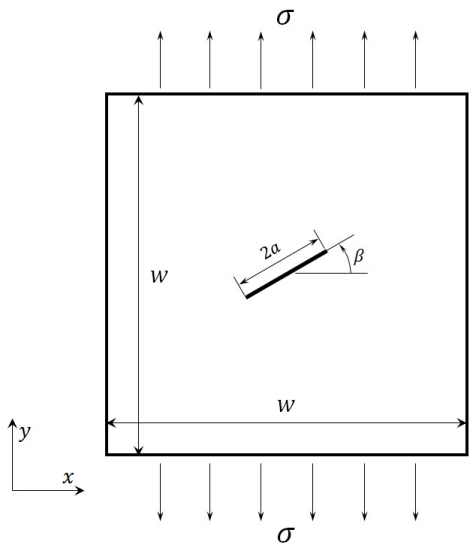
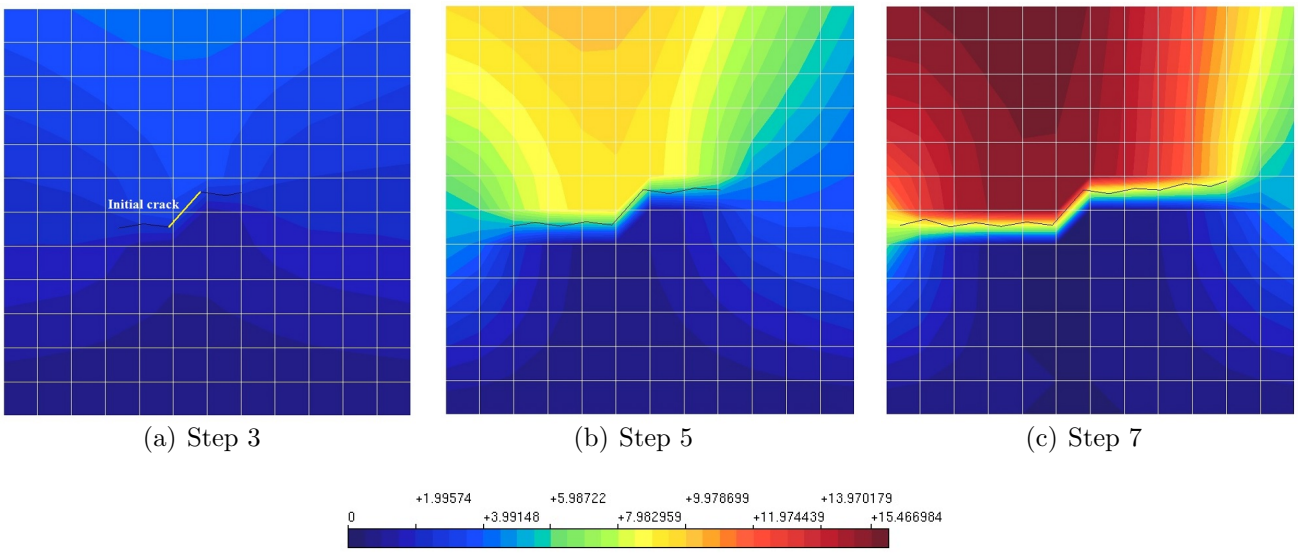


Figure 5.18: *Geometry and loading of the problem with an inclined crack.*



(a) Step 3 (b) Step 5 (c) Step 7

Figure 5.19: Contour of displacement in y direction for inclined crack problem, at different stage of the crack propagation process.

5.4.3 A Reissner-Mindlin Plate with Bending Moment

An infinite plate subjected to a far-field moment M is shown in Fig. 5.20 to have a purely mode-I loading. The aim of this example is to illustrate the crack propagation using G/XFEM method for Reissner-Mindlin plate problems, with $E = 1.0$, $\nu = 0.3$ in consistent units. Thanks to the symmetry about the y axis, only one-half of the plate are modeled with the finite elements.

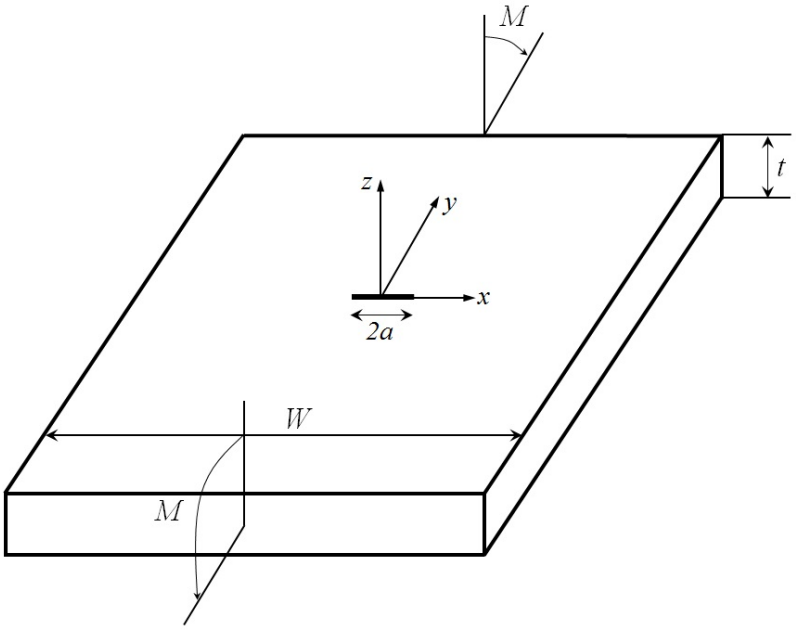
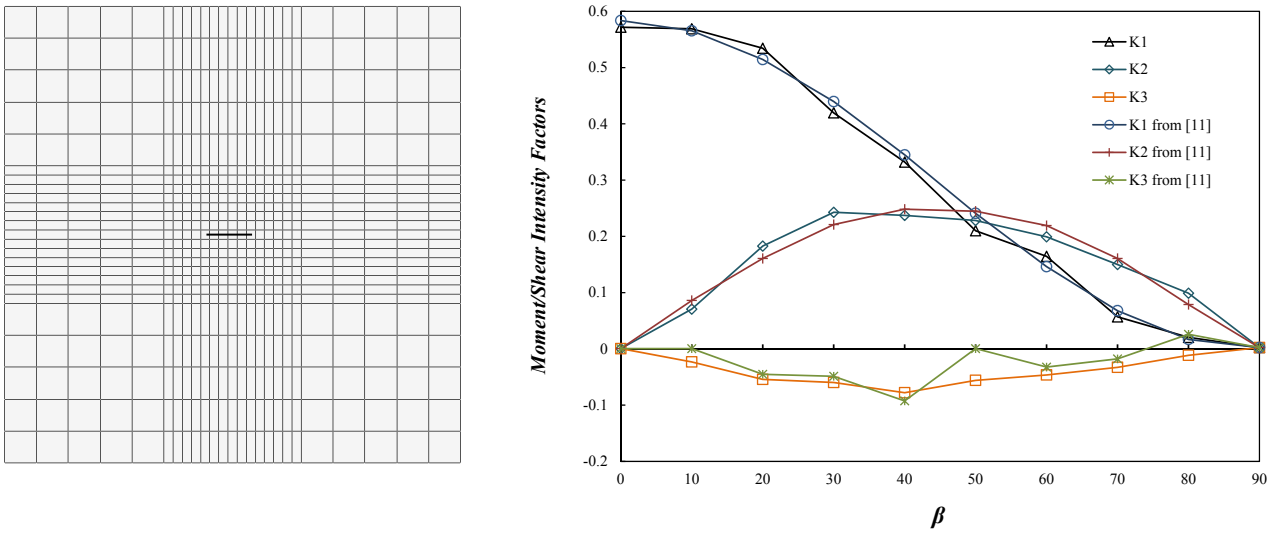


Figure 5.20: Schematic of geometry and loading for Reissner-Mindlin plate under bending. $M = 1.0$, $a = 0.6$, $W = 6$, and thickness $t = 1.0$, in consistent units.

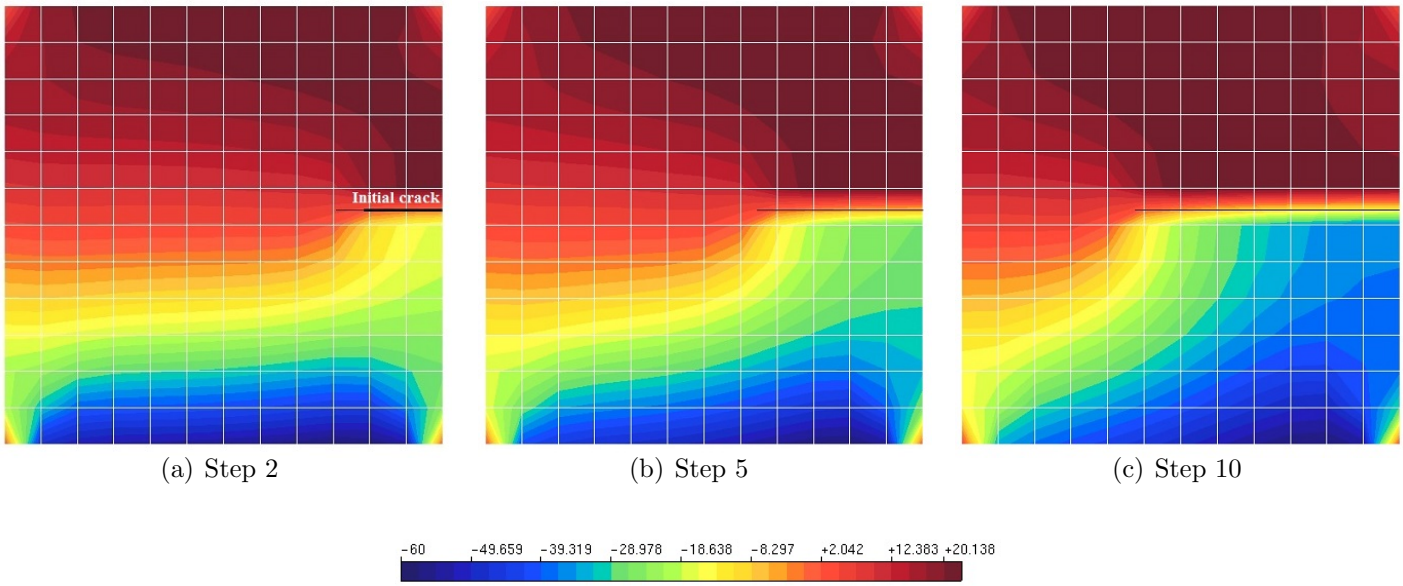
In order to show the capability of the current implementation to extract reasonable moment and shear intensity factors for Reissner-Mindlin plate with an inclined crack, a mesh of 625 quadrilateral elements is used here, see Fig. 5.21(a). Number of integration points for this analysis is set equal to 8×8. Figure 5.21(b) shows the extracted moment and shear intensity factor values for mode-I, mode-II and mode-III, covering a full range of the β values. The β is the angle between the crack line and the x direction. As it can be seen from this figure, the results from current work show a good agreement with those from (Dolbow et al, 2000b).

The element size of this problem also is equal to 0.25 and the model consists of 144 elements, a uniform mesh of 12×12 elements. The crack increment length considered here is equal to 0.125. Figure 5.22 shows the rotation distributions in y direction along with crack propagation path. As it can be seen from this figure, the crack propagation path is along with mode-I propagation, since the loading is pure mode-I.



(a) (b)

Figure 5.21: (a) Discretization used for moment and shear intensity factor calculation, with $\beta = 0^o$ as a schematic. $W = 10$, $a = 0.5$, $t = 1$, and crack geometry is shown in bold. (b) Normalized moment and shear force intensity factors for the cracked Reissner-Mindlin plate with different crack angle, β, for current work and from (Dolbow et al, 2000b).



(a) Step 2 (b) Step 5 (c) Step 10

Figure 5.22: Contour of rotation over y direction for Reissner-Mindlin problem, at different stage of the crack propagation process.

5.5 Summary and Conclusion

This work presented an object-oriented implementation of two-dimensional crack propagation for linear and nonlinear material models into an in-house code called INSANE. The capabilities of the generalized/extended finite element method were used to simulate the discontinuity propagation within the models. Different enrichment types, from Heaviside to singular enrichment functions were used to model the discontinuities and also capture the crack-tip singularities to obtain a better and more accurate crack propagation path. The stress intensity factors are used to obtain the propagation path. The OOP aspects were discussed in detail by providing different UML diagrams in package and class levels as well as different classes interactions aiming to show the whole implementation with regards to these work. Also, various blocks of codes are brought in order to show a part of implementations in a practical way.

The validation of these implementations were presented by different numerical examples for solid mechanics, from plane stress problems to a Reissner-Mindlin problem, aiming to cover all aspects and features of the current implementations. The numerical results presented here clearly show the capability of the current G/XFEM implementations to overcome almost all kinds of two-dimensional crack propagation problems.

Chapter 6

MODELING CRACK PROPAGATION USING TWO-SCALE G/XFEM

6.1 Introduction

This chapter presents a computational framework two-scale/global-local G/XFEM to model crack propagation plane stress/strain and Reissner-Mindlin plate problems. It is noteworthy to say that although there are various investigations on crack propagation for plane stress/strain problems, but there is no work for Reissner-Mindlin plate problem dealing with the crack propagation phenomena using two/multi scale G/XFEM. As discussed before in chapter 5, crack is represented using a discontinuous function along with the asymptotic crack-tip displacement fields and SIFs are also used to either determine the crack propagation direction or propagation status. Several algorithms and strategies have been implemented, within the INSANE. This implementation is presented in detail by solving several linear elastic fracture mechanics problems for both plane stress and Reissner-Mindlin plate problems to demonstrate the robustness and accuracy of the proposed approach. All the explanations and numerical results for this chapter are brought from Malekan et al (2017a).

6.2 Computational Environment

This section presents in detail the new implementations and also modifications of the existing classes corresponding to the crack growth process for two-scale G/XFEM analysis, specifically for *Persistence* and *Solution* classes. Required information for other classes can be found in chapter 5. In the case of the G/XFEM^{gl}, the *Persistence* class must interact with more than one model, i.e., one global and one local model corresponding to the global and local problems, respectively. The Code 5.2 from section 5.3.1 contains also additional lines of codes to include *GlobalLocalQuasiStatic* approaches. This approach is very similar to *GlobalLocal* presented in chapter 4, but includes some necessary methods for the two-scale quasi-static analysis.

```
public IMatrix getCuu() throws Exception {
  IMatrix kr = new IMatrix(this.getSizeOfXu(), this.getSizeOfXu());
  ListIterator<Element> elements = femmodel.getElementsList().listIterator();
  while (elements.hasNext()) {
    Element element = elements.next();
    int[] redEquations = this.getElementEquations(element);
    IMatrix c = element.getC();
    // Adding elements' stiffness matrix to model's stiffness matrix
    for (int i = 0; i < element.getNumberOfDegreesOfFreedom(); i++) {
      int a = redEquations[i];
      for (int j = 0; j < element.getNumberOfDegreesOfFreedom(); j++) {
        int b = redEquations[j];
        if (a > 0 && b > 0) {
          kr.setElement(a - 1, b - 1, kr.getElement(a - 1, b - 1) + c.getElement(i, j));
        }
      }
    }
  }
  // Adding nodal stiffness to model's stiffness matrix
  ListIterator<Node> nodes = this.getModel().getNodesList().listIterator();
  while (nodes.hasNext()) {
    Node node = nodes.next();
    if (node.nodeValuesContainsKey(Node.COEFFICIENTS_C)) {
      for (int i = 0; i < node.getNumberOfDegreesOfFreedom(); i++) {
        int pos = node.getNodeValues(Node.EQUATIONS).getIntPointValue(i);
        if (pos > 0) {
          kr.setElement(pos - 1, pos - 1,
            kr.getElement(pos - 1, pos - 1) + node.getNodeValues(Node.COEFFICIENTS_C).getDoublePointValue(i));
        }
      }
    }
  }
  // Adding stiffness related with Dirichlet and Cauchy BCs, for global-local analysis
  double penaltyFactor = ((GFemModel) this.getModel()).getPenaltyParameter();
  if (penaltyFactor != 0) {
    elements = this.getModel().getElementsList().listIterator();
    while (elements.hasNext()) {
      Element element = elements.next();
      int[] redEquations = this.getElementEquations(element);
      IMatrix cpm = ((GFemElement) element).getCpm();
      cpm.scale(penaltyFactor);
      for (int i = 0; i < element.getNumberOfDegreesOfFreedom(); i++) {
        for (int j = 0; j < element.getNumberOfDegreesOfFreedom(); j++) {
          int a = redEquations[i];
          int b = redEquations[j];
          if (a > 0 && b > 0) {
            kr.setElement(a - 1, b - 1, kr.getElement(a - 1, b - 1) + cpm.getElement(i, j));
          }
        }
      }
    }
  }
  double kappa = ((GFemModel) this.getModel()).getKappaParameter();
  if (kappa != 0) {
    elements = this.getModel().getElementsList().listIterator();
    while (elements.hasNext()) {
      Element element = elements.next();
      int[] redEquations = this.getElementEquations(element);
      IMatrix cpmCauchy = ((GFemElement) element).getCpmCauchy();
      cpmCauchy.scale(kappa);
      for (int i = 0; i <     element.getNumberOfDegreesOfFreedom(); i++) {
        for (int j = 0; j < element.getNumberOfDegreesOfFreedom(); j++) {
          int a = redEquations[i];
          int b = redEquations[j];
          if (a > 0 && b > 0) {
            kr.setElement(a - 1, b - 1, kr.getElement(a - 1, b - 1) + cpmCauchy.getElement(i, j));
          }
        }
      }
    }
  }
  return (kr);
}
```

Code 6.1: Code block of the *getCuu()* method from *GFemAssembler* class

As mentioned in previous chapter, *Assembler* interface is responsible for assembling the linear equation system provided by the discretization of the initial or boundary value problem, for different types of problem. Code 6.1 presents the written code in order to calculate the $\mathbf{C}_{uu}$

matrix (from Eq. (3.2)), in which it also contain the corresponding parts from Dirichlet (block starts with *penaltyFactor*) and Cauchy (block starts with *kappa*) boundary transferring to the local problem boundaries (Alves, 2012).

Similar to previous chapter, the Heaviside function is used to model the crack line within the element boundaries. The *CrackEnrichment* or *NearTipEnrichment* class is used to accurately capture the singularities at the crack-tip and *GlobalLocalEnrichment* class is used to create the numerically-build enrichment functions from the local problem solution.

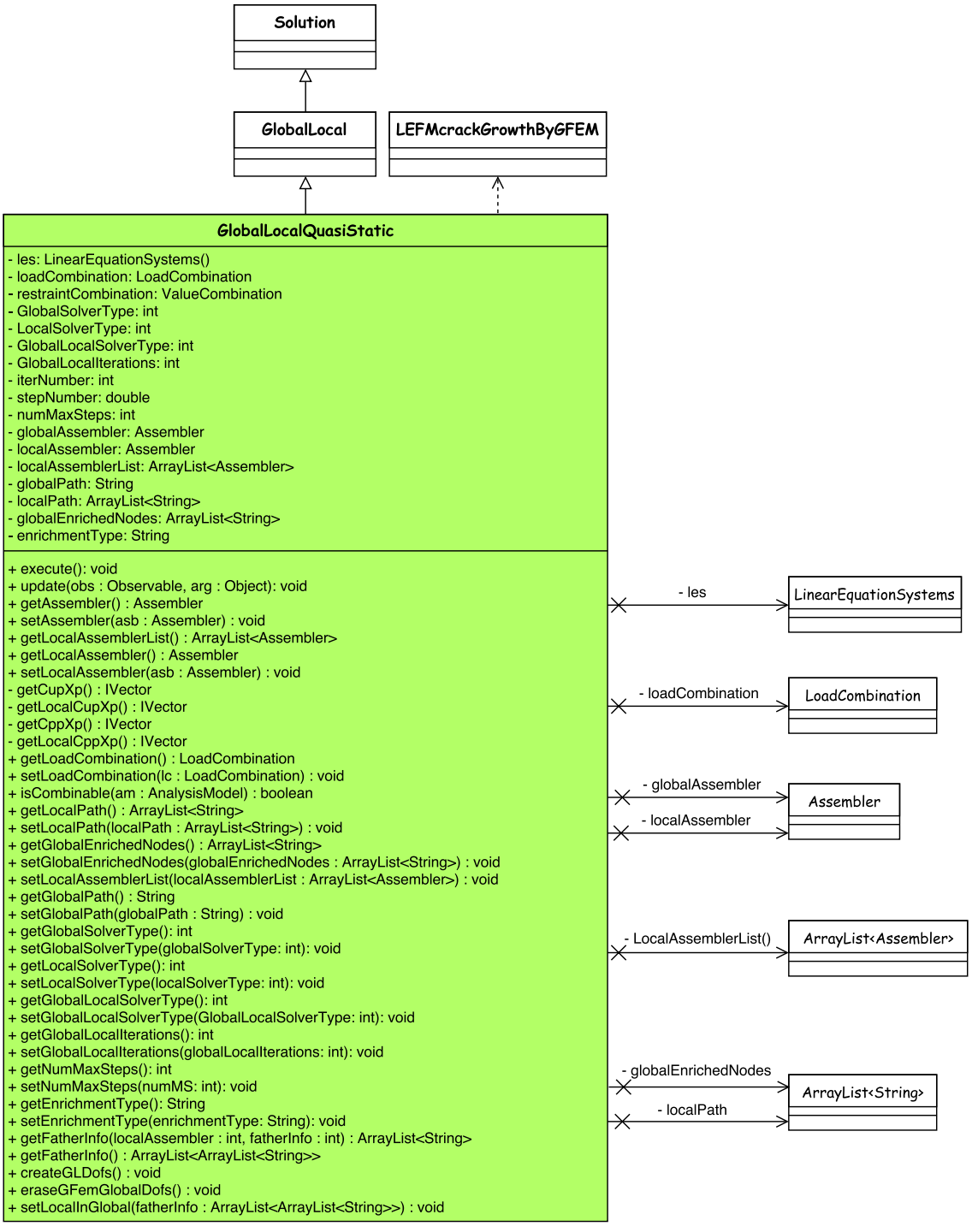


Figure 6.1: Structure and communications of the *GlobalLocalQuasiStatic* class

GlobalLocal class not only performs the static analysis for two-scale G/XFEM problems (Malekan et al, 2016a, 2017b), but also extracts the SIF using the domain-based interaction energy integral (implemented according to Malekan et al (2017d,c) and is shown in Appendix A). Finally the *GlobalLocalQuasiStatic* class, as shown in Fig. 6.1, is created in order to have quasi-static crack propagation using the two-scale G/XFEM method either for plane stress/strain or Reissner-Mindlin plate problems. A part of the *execute()* method of this class is brought in Code 6.2. This part is related to solve the first step of the global-local analysis. There are different solver types, from direct solvers such as *Crout* and *Cholesky* to iterative solvers such as *Preconditioned Conjugate Gradient (PCCG)* and PCCG mixed with the *Babuŭka* approach (see Appendix D). All these solvers are written in the *LinearEquationSystems* class, discussed in section 3.2.3.

```
1  private final LinearEquationSystems les = new LinearEquationSystems();
2  public void execute() throws Exception {
3    // solving global problem, initial global-local step
4    if (globalAssembler.getSizeOfXu() != 0) {
5      for (int i = 0; i < globalAssembler.getModel().getNumberOfElements(); i++) {
6        ArrayList<Element> locElems = new ArrayList<Element>();
7        locElems.add(gloalbAssembler.getModel().getElementsList().get(i));
8        ((GFemElement) globalAssembler.getModel().getElementsList().get(i)).setElementValues("LOCAL_ELEMENTS",
               localElements);
9      }
10     sel.setBVector(new IVector(globalAssembler.getSizeOfXu()));
11     globalAssembler.addLoading(loadCombination);
12     if ((globalAssembler instanceof GFemAssembler) && (this.restraintCombination != null)) {
13       ((GFemAssembler) globalAssembler).addRestraints(restraintCombination);
14     }
15     IVector v = globalAssembler.getNp();
16     les.addToBVector(v);
17     les.addToBVector(globalAssembler.getEp());
18     les.getBVector().scale(mult);
19     les.addToBVector(this.getCupXp());
20     IMatrix cuu = globalAssembler.getCuu();
21     les.setAMatrix(cuu);
22     if (this.GlobalSolverType == 1) {
23     les.choleskySolver(cuu, les.getBVector());
24     } else if (this.GlobalSolverType == 2) {
25       les.croutSolver(cuu, les.getBVector());
26     } else if (this.GlobalSolverType == 3) {
27       les.preconditionedConjugateGradientsSolver(cuu, les.getBVector());
28     } else if (this.GlobalSolverType == 4) {
29       les.babuskaSolverWithPCCGDiag(cuu, les.getBVector());
30     }
31     globalAssembler.setXu(sel.getXVector());
32     les.setAMatrix(globalAssembler.getCpu());
33     les.setXVector(globalAssembler.getXu());
34     les.solverB();
35     les.subFromBVector(globalAssembler.getEu());
36     les.addToBVector(this.getCppXp());
37     // Subtracting possibles dual variables previously set.
38     les.subFromBVector(globalAssembler.getNu());
39     globalAssembler.setNu(les.getBVector());
40     globalAssembler.setXp();
41     globalAssembler.update();
42   }
43   ...
44 }
```

Code 6.2: Code block of the *execue()* method from *GlobalLocalQuasiStatic* class belongs to the first step of global-local analysis

Furthermore, the global-local enrichment function must be used for the two-scale crack propagation procedure using the G/XFEM methodology. In this case, the boundary conditions from the global problem have to be transferred to the local boundaries for each analysis step. This is done using the *EquivalentNodalGeneralValue* class, as shown in Fig. 6.2, proposed in (Alves, 2012) and firstly presented in (Malekan et al, 2017b). The corresponding methods to apply different boundary conditions, such as Dirichlet, Neumann, or Cauchy, are here

modified to have capabilities to solve both plane stress/strain and the Reissner-Mindlin plate problems. A short part of the *getGFemEquivalentNodalValuesFromElement()* method from this class is brought in Code 6.3 for the case of transferring Dirichlet boundary condition to the local problem boundaries. It looks for each one of the element edge separately, and apply the displacement (for Dirichlet case), stress (for Neumann case), both (for Cauchy case), or nothing to them, see Eq. (2.19) from section 2.5.2. These displacement and stress come from the solution of the global problem. Lines 29-33 from the Code 6.3 were previously written in such a way that handle the plane stress/strain problems where each node only has two physical DOFs (two translational DOFs), while they have three DOFs (one translational and two rotational DOFs) in the case of Reissner-Mindlin plate. Therefore, these lines are modified to have a general implementation for both plane stress/strain and Reissner-Mindlin plate problems.

```
1  private Element element;
2  public IVector getGFemEquivalentNodalValuesFromElement() {
3    IVector feq = new IVector(this.element.getNumberOfDegreesOfFreedom());
4    feq.zero();
5    ArrayList<Integer> boundaryInfo = ((ArrayList<Integer>) ((GFemElement) this.element).getElementValues("
         BOUNDARY_INFORMATION"));
6    if (boundaryInfo.size() == 4) {// if quadrilateral (boundaryInfo with 4 values)
7      if (boundaryInfo.get(0) == 3) {
8        IVector feq1 = new IVector(this.element.getNumberOfDegreesOfFreedom());
9        feq1.zero();
10       for (int i = 0; i < integPoints.getNumCol(); i++) {
11         double[] natCoords = integPoints.getColumn(i).toDoubleArray();
12         double[] rst = new double[2];
13         rst[0] = natCoords[0];
14         rst[1] = -1;
15         IMatrix n = this.getStateVariableFunctionMatrix(rst, element.getIncidence());
16         // transform local coordinates of local elem. to global coordinates
17         Shape localShape = this.element.getShape().getOriginalShape();
18         IMatrix cN = this.element.getCartesianNodalCoordsMatrix();
19         IVector glCoordLocalElem = new IVector(cN.getNumCol());
20         IVector femShapeFunction = localShape.getShapeFunction(rst, cN);
21         glCoordLocalElem.mult(femShapeFunction, cN);
22         // use global coords. and transform in local coords. of global elem.
23         IPoint3d point = new IPoint3d(glCoordLocalElem.getElement(0), glCoordLocalElem.getElement(1));
24         GFemElement masterElem = (GFemElement) this.element.getElementValues("GLOBAL_ELEMENT");
25         double[] natCoordMastElem = masterElem.getNaturalCoords(point).getCoords();
26         masterElem.setElementValues(masterElem.STEP_GL, 2);
27         Shape shape = masterElem.getShape();
28         IMatrix shapeFunction = ((EnrichedShape) shape).getShapeFunction(natCoordMastElem, masterElem);
29         IMatrix localDerivates = new IMatrix(IMatrix.identity(3));
30         IMatrix shapeFunc = masterElem.getAnalysisModel().getStateVariablesOperator(localDerivates,
               shapeFunction, masterElem.getLocalCartesianNodalCoordsMatrix());
31         IVector nodDispl = new IVector(masterElem.getStateVariables());
32         IVector dispOnPoint = shapeFunc.mul(nodDispl);
33         IPoint3d p0 = this.element.getIncidence().get(0).getPoint();
34         IPoint3d p1 = this.element.getIncidence().get(1).getPoint();
35         double j = p0.distance(p1) / 2;
36         n.transpose();
37         IVector feqi = n.mul(displacementOnPoint);
38         feqi.scale(integrationPointsWeights.getElement(i) * (j));
39         feq.add(feqi); // add contribution of each Gauss point for this edge
40         feq1.add(feqi);
41       }
42     }
43     ...
44   }
45   ...
46 }
```

Code 6.3: Code block of the *fillIsWithCrackPropagationFromFile()* method from *PersistenceAsXml* class

Figure 6.3 shows the solution process to assemble the stiffness matrix and solve the problem. Similar to chapter 5, the stiffness matrix is mounted by the *FemAssembler* class, and hence *GFemAssembler* class for G/XFEM method, through the *getCuu()* method which is related to the elastic approximations of the constitutive tensor. This method looks into the model

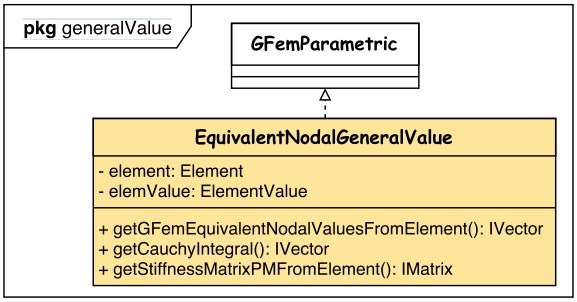


Figure 6.2: UML diagram of the *EquivalentNodalGeneralValue* class

elements according to the problem types, i.e., solid, planes stress/strain, frame, etc. The *ProblemDriver*, by the *GFemParametric* class, build the stiffness matrix of each element. The crack propagation strategy is exactly the same as those presented in section 5.3.5 where except using the *StaticEquilibriumPath* class, the *GlobalLocalQuasiStatic* class handles the whole solution procedure.

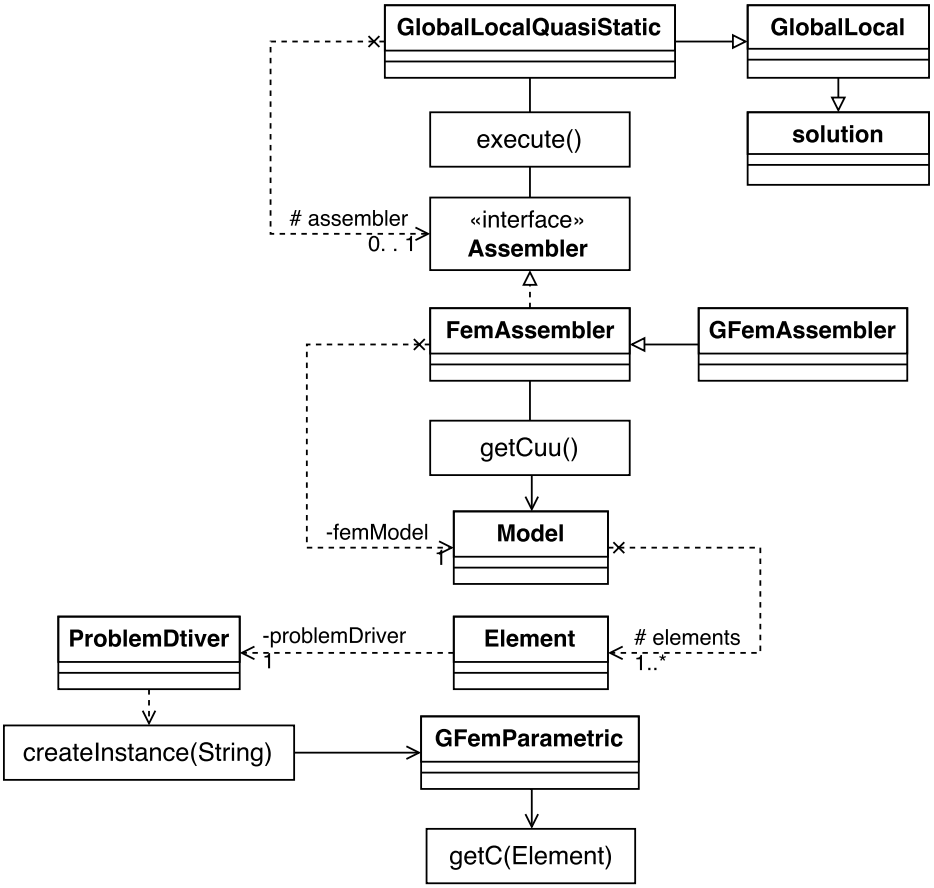


Figure 6.3: UML diagram of the solution process

6.3 Numerical Examples

This section presents three linear-elastic problems in two-dimensional domain. Sections 6.3.1 and 6.3.2 present a single-edge cracked plate under tension and shear loads, respectively.

In addition, a cracked Reissner-Mindlin plate is analyzed in section 6.3.3 for a horizontal and inclined crack configuration. These problems are brought to show the robustness and accuracy of the current G/XFEM^{gl} implementations.

Similar to section 4.3, Dirichlet boundary condition will be applied on the local problem boundaries in order to demonstrate the robustness of the methodology in the worst case scenario. The integration order of G/XFEM^{gl} for both global and local problems is chosen to be enough to reproduce the polynomial approximation. When there is only a polynomial approximation, the number of integration points is equal to the one necessary to accurately reproduce it. When there is the Heaviside (jump) or singular enrichment functions the number of points must be large enough to minimize the integration error. Therefore, the number of integration points that are used for these problem were selected quite big enough to accurately capture the crack propagation direction within the element domain, specifically for local problem, even for elements containing singular enrichment functions. For simplicity, the same number of integration points is used for the all elements with and without singular/Heaviside enrichment functions. Based on our experience with the modeling approach, the number of global-local cycles for each time step is chosen equal to 3. However, one can select larger number of global-local cycles, depending on the problem type.

Definition of the domain size of the interaction integral, r, its scalar multiplier, r_m, and the crack increment length, Δa, are similar to those presented in section 5.4. The scalar multiplier for all two problems is considered equal to 2.0, but the crack increment length has different values for each problem. The crack propagation criteria and the propagation direction are defined based on the stress field around the crack tip, with the help of the SIF calculation procedure. In addition, the Heaviside (Eq. (5.2)) and singular (Eqs. (2.12) and (2.13)) enrichment functions are used for all problems (in the local problems only) in order to model the crack lines and capture the crack-tip singularities within the element domains. Then, the global-local enrichment function is used to enrich the global problem with the solution obtained from local problem, thus the discontinuity effects will be transferred to the global problem via this global-local enrichment function.

6.3.1 A Rectangular Plane Stress Problem with an Edge Crack

This example considers a single-edge cracked plate submitted to a tension stress, as shown in Fig. 6.4. The cracked zone produces singular stress field near the crack tips. The objective of this example is to illustrate the crack propagation under mode-I fracture analysis. The problem is analyzed under plane stress state with (in consistent units): modulus of elasticity $E = 1.0$ and Poisson's ratio $\nu = 0.3$.

According to Tada et al (2000), the reference mode-I SIF for problem shown in Fig. 6.4 is:

$$K_I = [1.12 - 0.231(a/b) + 10.55(a/b)^2 - 21.72(a/b)^3 + 30.39(a/b)^4]\,\sigma\sqrt{\pi a} \qquad (6.1)$$

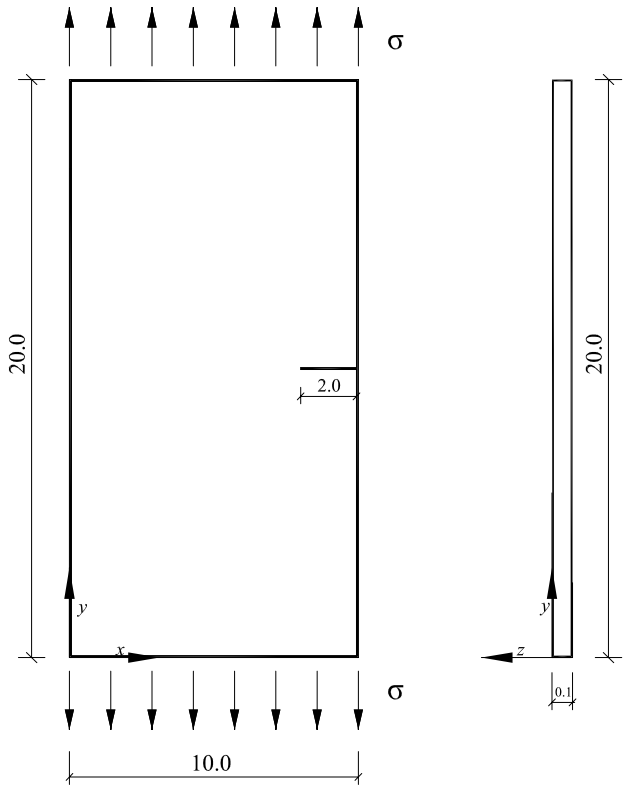


Figure 6.4: *Geometry and loading of the single-edge cracked problem. The tension stress is equal to* $\sigma = 1.0$.

where a is the crack length, b is the plate width, and the expression inside of the brackets is an empirical function which is valid for $a/b \leq 0.6$, as follows (Tada et al, 2000). Analytical SIF from Eq. (6.1) and numerical results, both K_I and K_{II}, from current simulation for the single-edge cracked plate (obtained in the third step of global-local analysis) and for various a/b are shown in Fig. 6.5. Maximum error of the K_I for various a/b values are smaller than 10%. In addition, the K_{II} values are almost near to zero for all a/b values, which in accordance with the pure mode-I loading for this problem.

Figure 6.6 shows the global-local steps and also the local domain discretization along with the global nodes to be enriched with the global-local enrichment function. These three-steps cycle of analysis is repeated for each crack propagation step. There are fourteen global nodes that are enriched with the global-local enrichment function.

The problem is discretized with 78 (a regular mesh of 6×13 elements) and 162 (a regular mesh of 18×9 elements) elements for global and local models, respectively. The element size of the global problem is 1.5, while the local problem has an element size of 0.5. The number of integration points of the two-scale G/XFEM are as follows: 8×8 and 10×10 for global and local problem, respectively. The integration order for global problem is chosen in such a way to have a good initial approximate solution to use for the boundary transferring to the local problem. In addition, the integration order for the local problem is chosen big enough to accurately capture the stress intensity factors for each time step, and hence obtain an accurate crack propagation path. The penalty parameter, η, for Dirichlet boundary

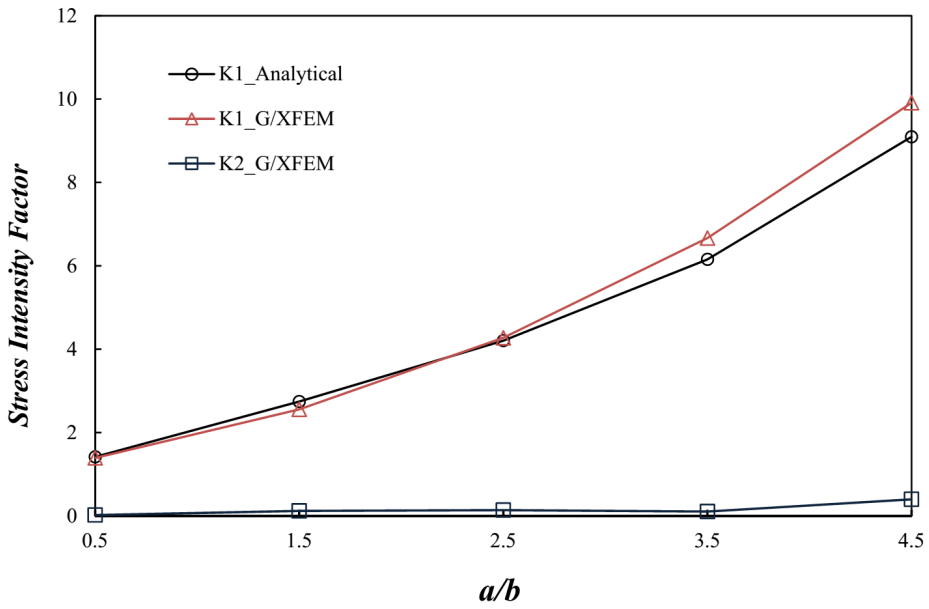


Figure 6.5: *Error between analytical and numerical SIF for single-edge cracked plate.*

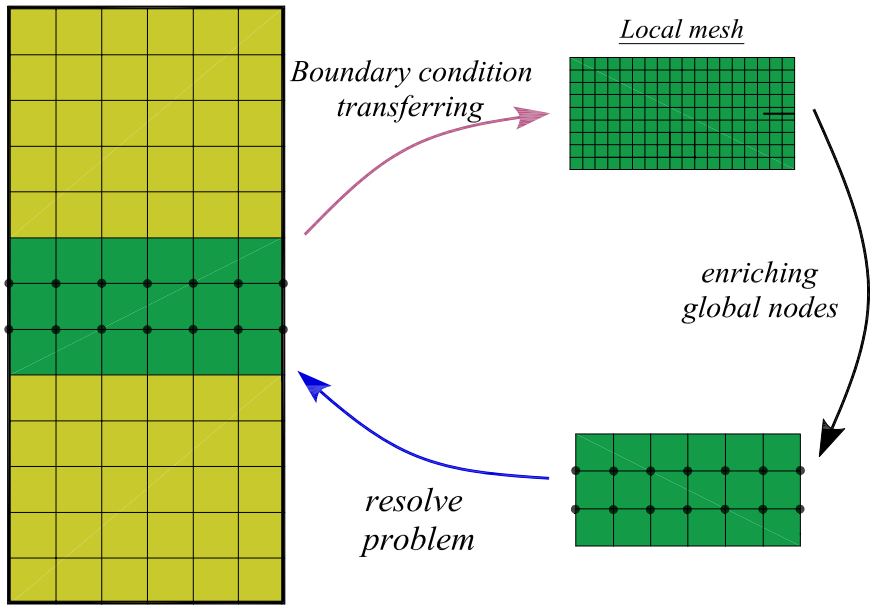


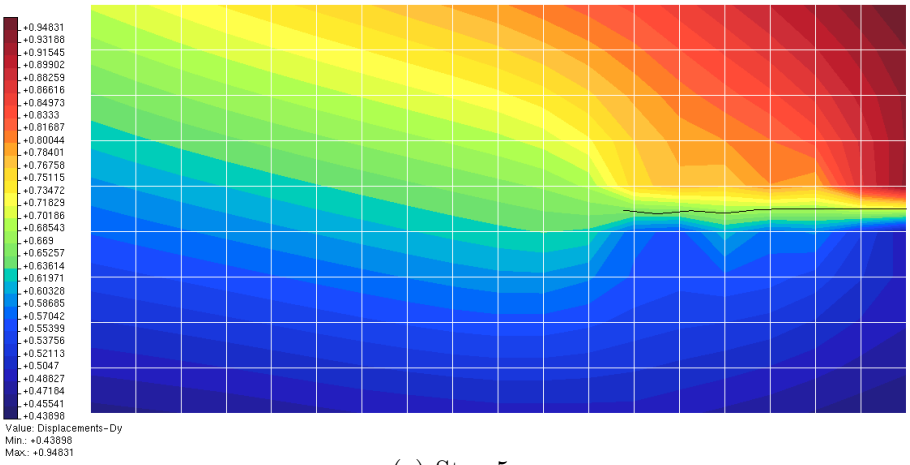
Figure 6.6: Global-local strategy sequences and local domain (in green) discretization. The black markers indicate the nodes to be enriched with the global-local enrichment function.

condition is chosen equal to 1×10^{12}. *Note that the crack propagation only occurs in the local problem and only its effect are transferred to the global problem via the global-local enrichment function.* The crack increment length considered here is equal to 0.325, i.e., almost two-third of the local model element size.

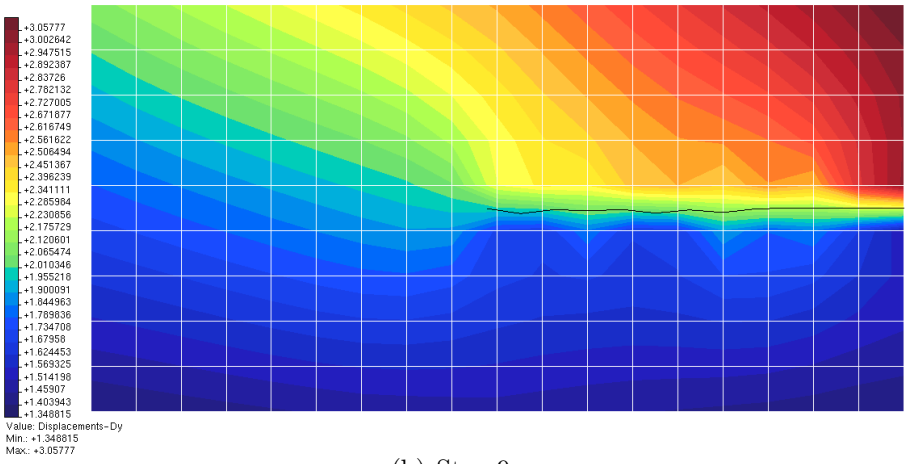
Displacement distributions in y direction along with crack propagation path are shown in Fig. 6.7 where mode-I crack propagation can be clearly seen from that. This figure belongs to the local problem, where the crack propagation procedure is only active in this second scale. Although there are some small fluctuations in the crack propagation path, but it still remains in the mode-I propagation direction as it was expected.

For sake of comparison, the crack growth path for a single scale problem with the same mesh as the global mesh from global-local analysis (a mesh with 78 regular elements), with a number of integration points of 10, are brought in Fig. 6.8. As it can be seen from Figs. 6.7 and 6.8, the global-local analysis delivers a very good crack propagation path in contrast to the single scale analysis, with a number of DOFs equal to 188, while a quite similar crack propagation path can be obtained with a fine mesh (377 quadrilateral elements with an average size of 0.7) and with 825 DOFs, as it is shown in Fig. 6.8(c). The length of the propagating crack is kept the same for these two cases shown in Figs. 6.8(a) and Fig. 6.8(c).

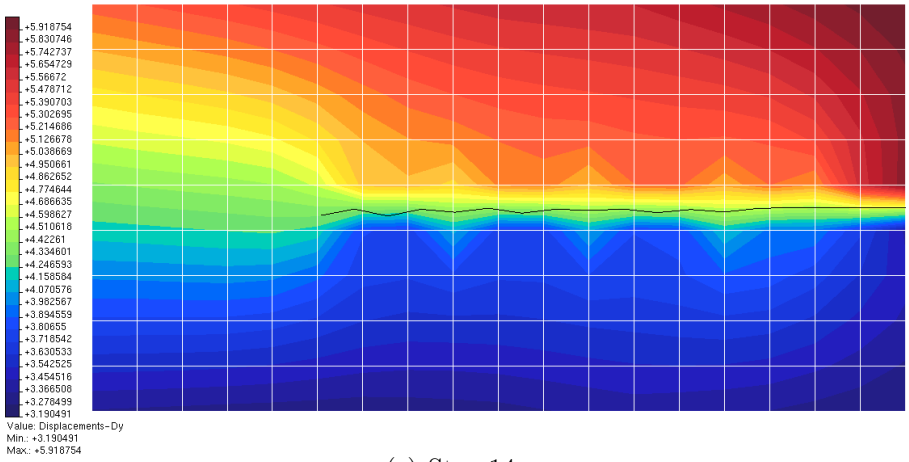
In addition, Fig. 6.9 shows the analytical SIF from Eq. (6.1) and numerical results, both K_I and K_{II}, for the two cases shown in Fig. 6.8. The maximum error for K_I with respect to analytical values are 13% and 40%, for meshes with the element size of 0.7 and 1.5, respectively. In addition, the maximum K_{II}/K_I ratio for these two cases are equal to 0.049 (element size of 0.7) and 0.27 (element size of 1.5), which clearly describes the crack path fluctuation for mesh with bigger element size.



(a) Step 5

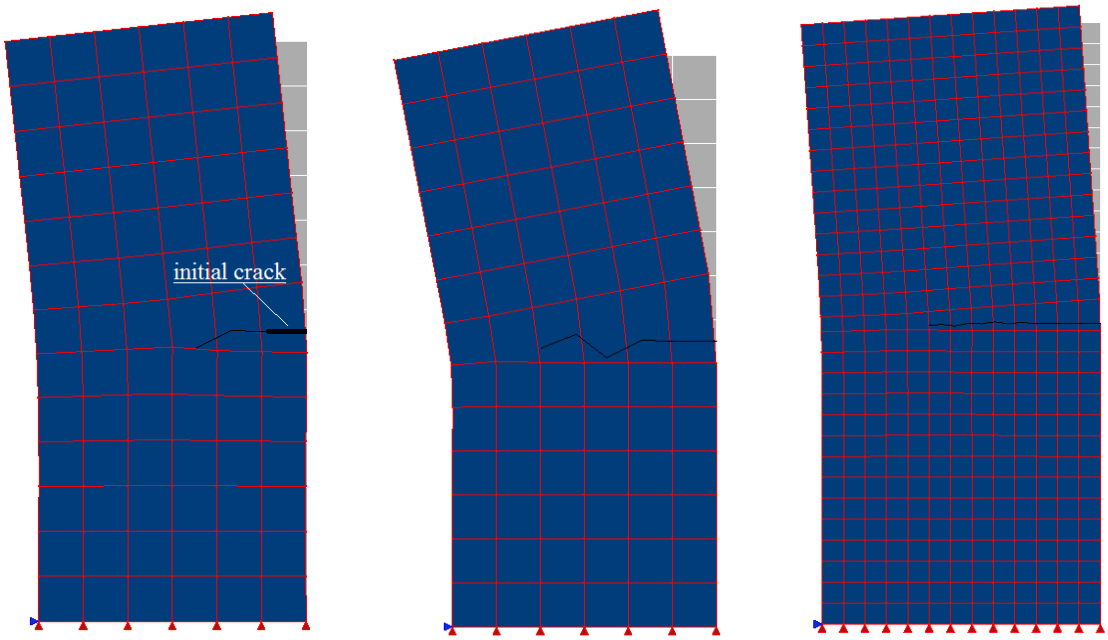


(b) Step 9



(c) Step 14

Figure 6.7: Contour of the displacement in y direction along with the crack propagation path for local problem.



(a) Step 5 (b) Step 9 (c) Step 10 with a fine mesh

Figure 6.8: Deformed shape along with the crack propagation path for a single scale problem, for two different element sizes: one with the same as the global problem, and another with smaller element size.

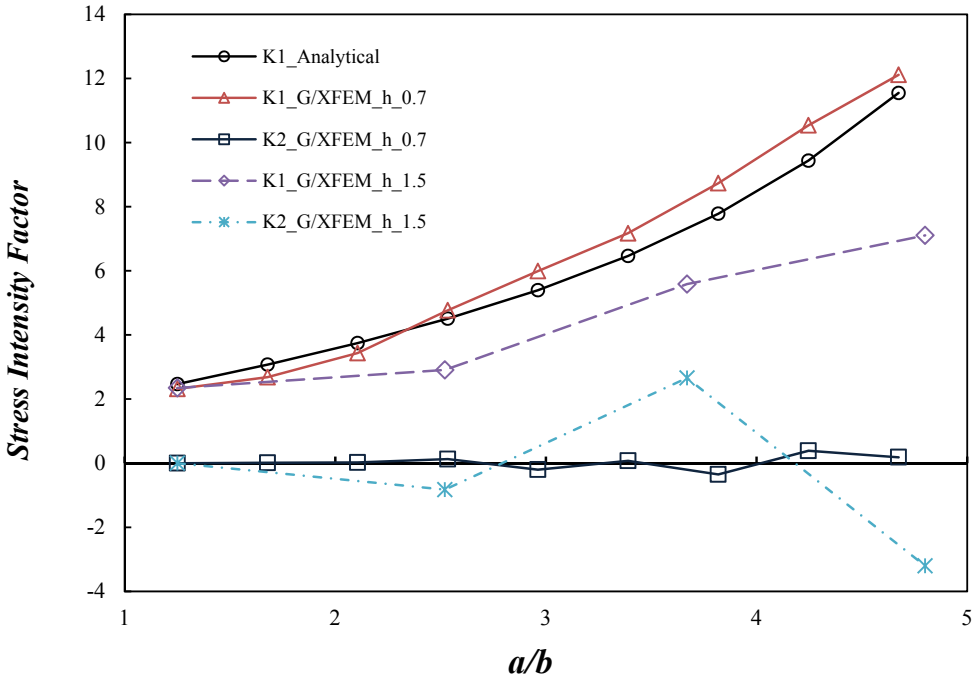


Figure 6.9: *Analytical and numerical results of the stress intensity factor for two cases shown in Fig. 6.8.*

6.3.2 Single-edge Cracked Problem Under Shear Loading

This example corresponds to a rectangular single-edge cracked problem, clamped at the bottom and under the far-field shear stress along the top edge, as it can be seen in Fig. 6.10(a). Material properties are: Young's modulus $E = 3 \times 10^7 N/mm^2$ and Poisson's ratio $\nu = 0.25$. This problem is analyzed under the plane strain condition and it will be compared with the results from (Nguyen-Xuan et al, 2013) aiming to demonstrate the robustness of the current implementation under the mixed-mode fracture condition.

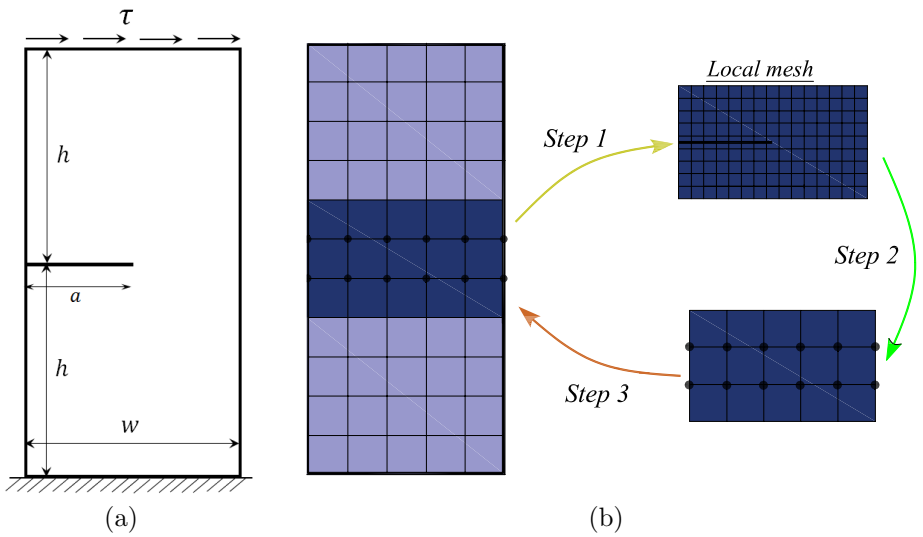


Figure 6.10: *A rectangular single-edge cracked plate under shear loading: (a) geometry and loading, with:* $h = 8$, $W = 7$, $a = 3.5$, *thickness* $t = 1.0$, *and shear loading* $\tau = 1N/mm^2$, *(b) global and local meshes along with local domain (in blue), with emphasizing steps of the global-local analysis. All dimensions are in millimeters.*

The problem is discretized with 55 (a regular mesh of 5×11 elements) and 135 (a regular mesh of 15×9 elements) elements for global and local models, respectively, as it is shown in Fig. 6.10(b). The element size of the global problem is 1.5, while the local problem has an element size of 0.5. The number of integration points of the two-scale G/XFEM are as follows: 8×8 and 10×10 for global and local problem, respectively. The penalty parameter, η, for Dirichlet boundary condition and crack increment length are chosen equal to 1×10^{12} and 0.325, respectively. There are *twelve* global nodes that are enriched with the global-local enrichment function. As it can be seen in Fig. 6.11, a good agreement is obtained for the crack path from *local problem* using the two-scale G/XFEM, with the results comparing with those obtained by Nguyen-Xuan et al (2013).

6.3.3 Reissner-Mindlin Plate with a Crack

An infinite plate is subjected to a far-field moment M as shown in Fig. 6.12 to have a purely mode-I loading. Two different cases are studied here: plate with a horizontal crack, i.e., $\beta = 0$, and with inclined crack with $\beta = 60$ degrees. This problems has the following parameters (in consistent units): modulus of elasticity $E = 1.0$ and Poisson's ratio $\nu = 0.3$.

For horizontal crack, only one-half of the plate are modeled with the finite elements using

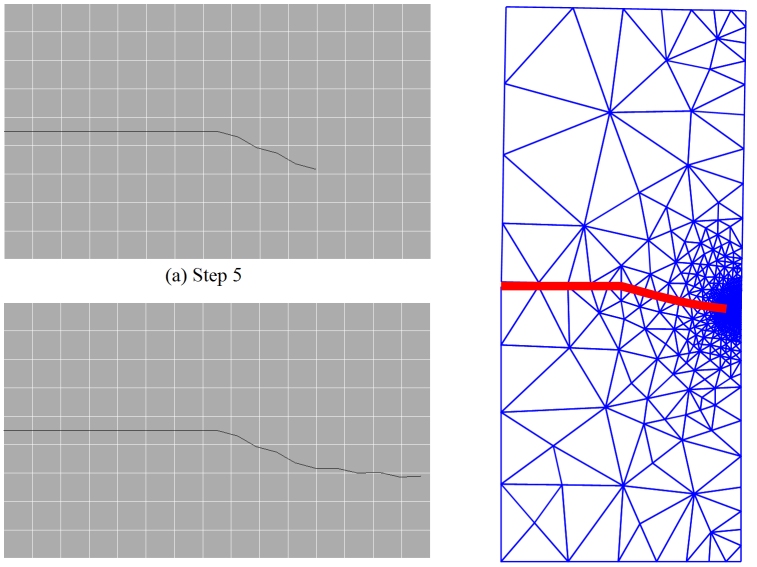


Figure 6.11: Crack propagation trajectory of plate under shear loading: (a) Step 5 and (b) Step 10 for current implementation obtained from local problem, (c) Results from (Nguyen-Xuan et al, 2013).

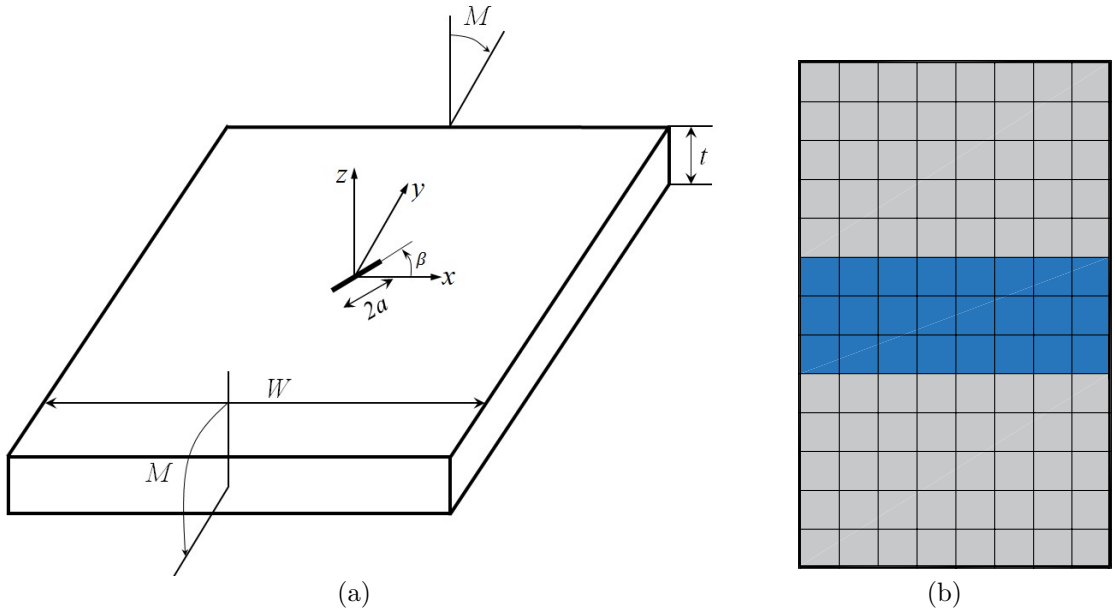


Figure 6.12: Schematic the Reissner-Mindlin plate under bending: (a) geometry and loading with: $a = 0.6$, $W = 6$, and thickness $t = 1.0$, (b) global mesh along with local domain (in blue).

the symmetry about the x_2 axis. In addition, the plate width W is taken to be 20 times the half crack length a in order to have an approximation of the infinite plate, with: $a = 0.5$, $W = 10$, and thickness $t = 1.0$. The length of the plate is chosen equal to the W. The global and local element sizes are equal to 0.7 and 0.25, respectively, with 104 (a regular mesh of 8×13 elements) and 216 (a regular mesh of 24×9 elements) elements for global and local problems, respectively. The crack increment length considered here is equal to 0.23, in which the crack propagation only occurs in the local problem. Figure 6.13 shows the global as well as the local problem discretization along with the global nodes to be enriched with the global-local enrichment function. There are *eighteen* global nodes that are enriched with the global-local enrichment function.

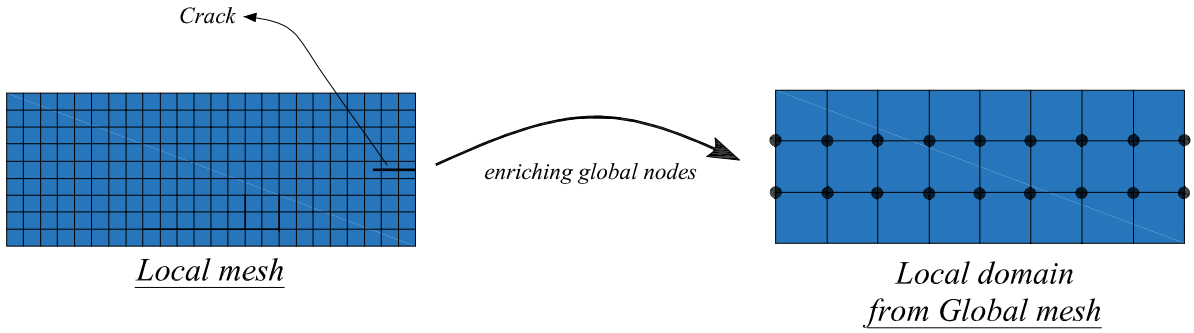


Figure 6.13: Local problem discretization along with the local domain from global mesh, in the case of plate with a horizontal crack. The black markers indicate the nodes to be enriched with the global-local enrichment function.

Table 6.1 gives the comparison of extracted moment intensity factors values for mode-I results from the exact solution, numerical reference from (Dolbow et al, 2000b) using XFEM method, and current G/XFEM and G/XFEM$^{\text{gl}}$ analyses for $\beta = 0$. These results are from two steps of the global-local strategy, one obtained in the global problem (first step) and the other in the global-local enriched problem (third step). It can be observed that the solution obtained with the global-local approximation strategy with only 1044 DOFs is quite accurate as the standard G/XFEM approximation with 2868 DOFs. The scalar multiplier r_m for G/XFEM and G/XFEM$^{\text{gl}}$ are equal to 2.5 and 2.0, respectively.

Table 6.1: Comparison of extracted moment intensity factors values for mode-I, for $a = 0.5$, $t = 5.0$, $\beta = 0$.

Analysis	DOFs	K_I	% of Error
Exact (Joseph and Erdogan, 1991)	—	0.684	—
Dolbow et al (2000b)	3087	0.685	0.15
G/XFEM	2868	0.691	1.02
G/XFEM$^{\text{gl}}$, 1st Step	948	0.441	35.53
G/XFEM$^{\text{gl}}$, 3rd Step	1044	0.668	2.34

Figure 6.14 presents the rotation distributions over y direction, i.e., θ_y, along with crack propagation path. Similar to section 6.3.1, the crack propagation procedure is only active in

the second scale, i.e., the local problem. A quite pure mode-I of the crack propagation can be clearly seen from this, as it was expected from having a purely mode-I far-field moment.

In the case of inclined crack with $\beta = 60$ degrees, the whole plate are modeled with following geometrical parameters: $a = 0.25$, $W = 5$, and thickness $t = 0.5$. Again, the W/a is set equal to 20 to represent an approximation of the infinite plate. The global and local elements are equal to 56 and 216, respectively, with an element size of 0.7 for global and 0.25 for the local problem. The crack increment length considered here is equal to 0.153. Figure 6.15 shows the local problem discretization along with the crack position for plate with the central inclined crack. Similar to the horizontal crack case, there are *eighteen* global nodes that are enriched with the global-local enrichment function.

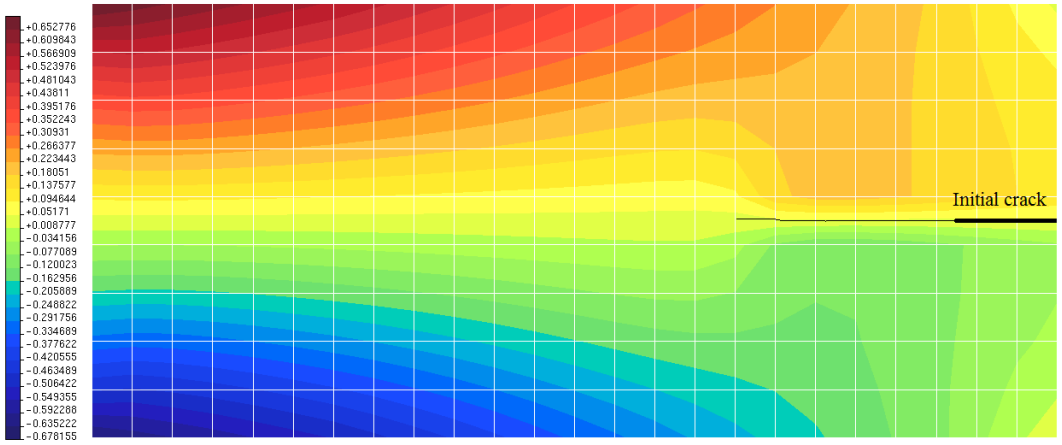
Figure 6.16 presents the crack propagation path for the Reissner-Mindlin plate with an inclined crack. Again, the crack propagation procedure is only active in the second scale, i.e., the local problem. As it can be clearly seen from this figure, the initial inclined crack is propagating horizontally after initial steps of analysis, which is due to the mode-I far-field moment.

6.4 Summary and Conclusion

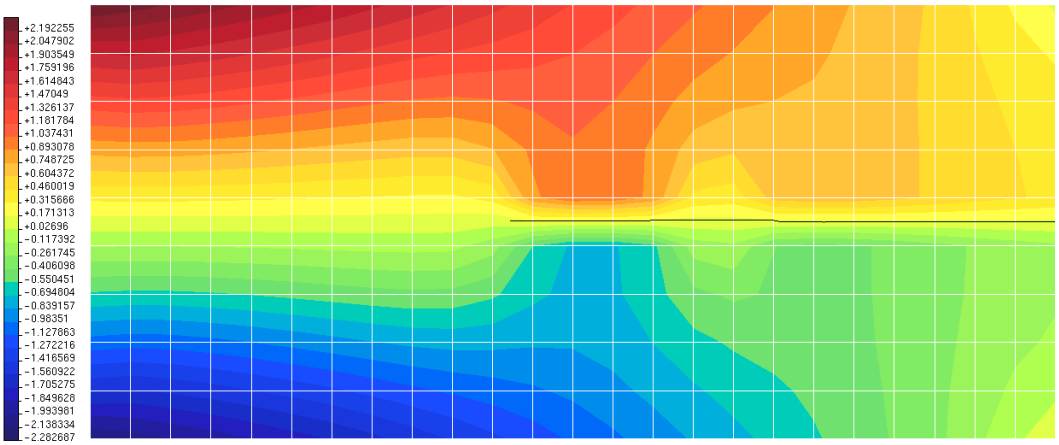
The aim of this chapter was to present a computational framework for crack propagation modeling using the capabilities of the object-oriented programming along with two-scale generalized/extended finite element method. The OOP with its main characteristics such as *abstraction* and *inheritance* greatly helps to include new numerical approaches to solve different kind of problems with small required changes (meshfree, G/XFEM, and nonlocal methods were easily derived from FEM implementations here in INSANE code) and 6.1, in which quasi-static two-static was derived from classical two-scale class, only small changes were done to include the crack propagation strategy. Also, the OOP helps to use different types of problems in a single solution class with the aim of object creation. Moreover, various solvers can be adopted for the solution strategy in this class, as it was shown in Code 6.2 and user easily can mitigate between them with just a single string definition shown in Code B.2. The present object-oriented implementations, as a part of INSANE computational platform, were explained in details and every aspect of them were shown through different UML diagrams, block of codes, both from implementation parts and user input files.

Finally, the validation of the whole implementations were presented by solving two numerical examples for solid mechanics, aiming to cover all aspects of the current work. The first one dealt with a plane stress problem under tension loading, and the second one showed the capability of the proposed computational framework for Reissner-Mindlin plate problem. The main conclusions from the current implementations can be summarized as:

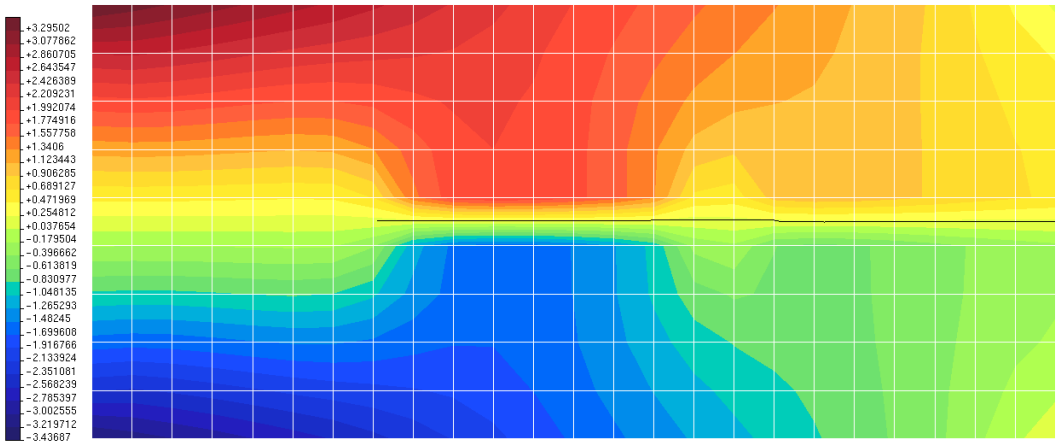
– Heaviside function along with singular enrichment functions are used to facilitate the crack propagation procedure, in the local problem. Also, the global-local enrichment



(a) Step 5



(b) Step 10



(c) Step 13

Figure 6.14: Contour of the rotation over y direction (θ_y) for Reissner-Mindlin plate, along with the crack propagation path for local problem.

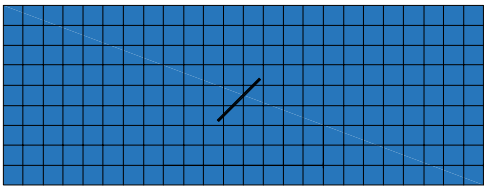
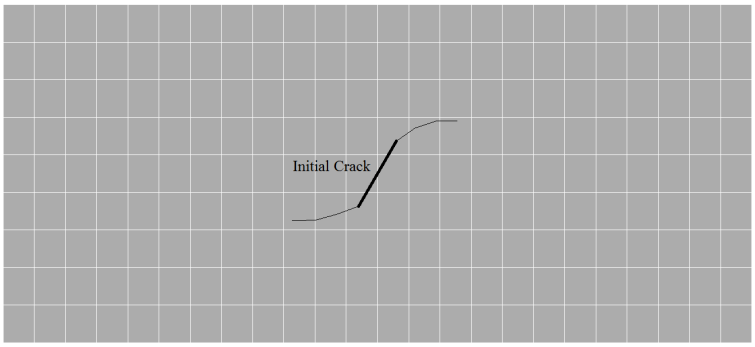
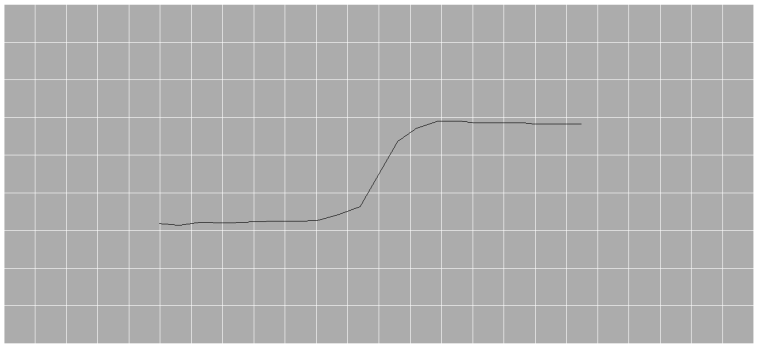


Figure 6.15: Local problem discretization along with the crack position, for Reissner-Mindlin plate with $\beta = 60^o$.



(a) Step 3



(b) Step 9

Figure 6.16: Crack propagation path of the Reissner-Mindlin plate with an inclined crack, from local problem.

function is used to enrich the global problem from the precise solution obtained from the local problem.

– The two-scale crack propagation is established in such a way that the crack only propagates in the local problem and its effects were transferred to the global problem via global-local enrichment functions.

– The numerical results from two examples clearly show the capability of the current OOP-based two-scale G/XFEM implementations to overcome almost all kind of two-dimensional crack propagation problems, with a good accuracy and also easy to expand for three-dimensional or shell problems, where the shell part is a potential research theme to be pursued in the future.

Chapter 7

CONCLUSIONS AND FUTURE WORKS

Detailed conclusions were drawn at the end of each chapter. The research trend and some important conclusions and contributions are summarized below.

7.1 Conclusions

Application of object-oriented programming for FEM has been receiving great attention over the last two decades. There are some main concepts of OOP that makes it easy to use and maintenance, such as: *objects* and *classes*, *data abstraction* and *encapsulation*, *inheritance*, and *polymorphism*. The main objective of this project was *to fracture modeling in thin-walled structure by extending the available capabilities of the G/XFEM method implemented in INSANE in-house code*. Therefore, this work used the capabilities of the OOP to perform a crack propagation study using single and two-scale G/XFEM approaches. The problem studies were plane stress/strain and Reissner-Mindling plates. The reason to choose the Reissner-Mindling plate problem was because of their assumptions which lead to have C^0 continuity, as it was mentioned in chapter 2. Thus, the conventional plane stress/strain elements can be used with some small modifications only to include all three degrees of freedom for Reissner-Mindling plate problem. Following are the main conclusion from this PhD thesis:

- Different enrichment functions were introduced/modified into the INSANE computational platform aiming to facilitate the fracture analysis for G/XFEM and G/XFEM$^{\text{gl}}$ methods, such as: *CrackEnrichmentModeI*, *CrackEnrichmentModeII*, *NearTipEnrichment* and *DiscontinuousEnrichment*.
- The stable G/XFEM shows its capabilities to improve the conditioning and convergence rate of the G/XFEM, for classical and two-scale analyses. Although, current research didn't use this strategy for the crack propagation procedure, but it can be useful which makes the crack propagation feasible with this approach.

- This work introduced a two-scale approach with the G/XFEM method to analyze *Reissner-Mindlin* plate for static analysis, which is an extension to G/XFEM method for plate structures. The two-scale/global-local approach helps to eliminate the need of using fine/very-fine meshes to capture an accurate solution. More details on this issue can be found in (Malekan et al, 2017b). The modifications were made for *GlobaLocal*, *EquivalentNodalValue*, and *GlobalLocalEnrichment* classes.

- A crack propagation procedure was established for classical and two-scale G/XFEM methods aiming to model the fracture for plane stress/strain and Reissner-Mindlin plate. This procedure uses the stress intensity factor as the main parameter to determine the crack propagation status and its direction. The main OOP classes for this part of PhD thesis, that were created/modified, are *LEFMcrackGrowthByGFem*, *GlobalLocalQuasiStatic*, *StressIntensityFactors*, *GFemModel*, *GFemElement*, and *GFemParametric*, among others.

- The OOP with its main characteristics such as *abstraction* and *inheritance* greatly helps to include new numerical approaches to solve different kind of problems with small required changes. Therefore, with the aim of current implementations, different numerical approaches (such as meshfree method) can also use the two-scale analysis scheme, using the *inheritance* concept of the OOP.

Journal and conference publication records that are directly or partially related to this PhD thesis are as follow:

1. M. Malekan, F.B. Barros, R.L.S. Pitangueira, P.D. Alves, S.S. Penna, *A Computational Framework for a Two-scale Generalized/extended Finite Element Method: Generic Imposition of Boundary Conditions*, Engineering Computations, 2017, 34(3), pp. 1-32, `http://dx.doi.org/10.1108/EC-02-2016-0050`

2. M. Malekan, F.B. Barros, *Well-conditioning Global-local Analysis using Stable Generalized/Extended Finite Element Method for Linear Elastic Fracture Mechanics*, Computational Mechanics, 2016, 58(5), pp. 819-831, `http://dx.doi.org/10.1007/s00466-016-1318-7`

3. M. Malekan, F.B. Barros, R.L.S. Pitangueira, P.D. Alves, *An Object-oriented Class Organization for Global-local Generalized Finite Element Method*, Latin American Journal of Solids and Structures, 2016, 13(13), pp. 2529-51, `http://dx.doi.org/10.1590/1679-78252832`

4. M. Malekan, F.B. Barros, *Numerical Analysis of a Main Crack Interactions with Micro Defects/inhomogeneities Using Two-scale Generalized/Extended Finite Element Method*, Computational Mechanics (under review), 2017.

5. M. Malekan, L.L. Silva, F.B. Barros, R.L.S. Pitangueira, S.S. Penna, *Two-dimensional Fracture Modeling with the Generalized/Extended Finite Element Method: An Object-Oriented Programming Approach*, Advances in Engineering Software (under review),

2017.

6. M. Malekan, F.B. Barros, R.L.S. Pitangueira, *A Computational Framework for Crack Propagation with the Two-scale Generalized/Extended Finite Element Method*, Computers & Mathematics with Applications (submitted), 2017.

7. M. Malekan, F.B. Barros, R.P. Silva, *Development and Implementation of a Well-conditioning Approach Toward Generalized/Extended Finite Element Method into an Object-oriented Platform*, CILAMCE2016-0575, XXXVII Latin American Congress on Computational Methods, November 6-9, 2016, Brasilia-DF, Brazil, `http://periodicos.unb.br/index.php/ripe/article/view/23957`

8. M. Malekan, F.B. Barros, R.L.S. Pitangueira, *A Finite Element/Generalized Finite Element Method Code-coupling for Linear Elastic Fracture Mechanics Problems*, CILAMCE2015-0439, XXXVI Latin American Congress on Computational Methods, November 22-25, 2015, Rio de Janeiro, Brazil, `http://dx.doi.org/10.20906/CPS/CILAMCE2015-0439`

9. M. Malekan, F.B. Barros, R.L.S. Pitangueira, P.D. Alves, *Development and Implementation of a Generalized Finite Element Method Object-oriented Class for the Global-Local Analysis*, CILAMCE2014-0142, XXXV Latin American Congress on Computational Methods, November 23-26, 2014, Fortaleza-Ceara, Brazil.

10. M. Malekan, F.B. Barros, L.L. Silva, R.L.S. Pitangueira, *Fracture Modeling in Two-dimensional Problems Using a G/XFEM Object-oriented Implementation*, VI International Symposium on Solid Mechanics - MecSol2017, April 26-28, 2017, Joinville, Brazil.

7.2 Recommendations for Future Works

Although this PhD thesis covers a variety of two-dimensional structural problems, but there are some works that can be done in order to fill some voids of this research, such as:

- The stable G/XFEM can be used to model the crack propagation for both single and two-scale analysis cases. In this manner, some enrichment functions may have to change in order to make the crack propagation feasible with this approach.

- The enrichment functions of the G/XFEM may not be smooth over an enriched element due to presence of the discontinuity inside the element. Therefore, the standard Gauss quadrature rule cannot be used for this type of element, and a modifications are necessary for numerical integration over the element crosses by the discontinuity. In this research, the increasing of the number of Gauss points is used as a solution, however, this approach may result in loss of accuracy. In order to overcome these difficulties, two other techniques can be used : the triangular/quadrilateral partitioning method and the rectangular sub-grids method. Thus, another future line of research

could be implementing one of these techniques in order to make the crack propagation procedure more accurate.

- This research only covers the analysis for two-dimensional and Reissner-Mindlin plate problems, solving either single or two-scale approaches. Since, there is no research work on two-scale G/XFEM analysis for *Shell* structures, this can be a very interesting research theme to be done. Also, this topic can be mixed with the crack propagation implementation from the current project to model the crack propagation in shell structure using the two-scale analysis.

- Although the two-scale/global-local G/XFEM already show their advantages to solve the structural problems, but a three-scale approach can even reduce the computational cost more than the existing two-scale analysis. This can be done by defining a coarse global problem as the first scale, the quite-fine local problem as the second scale, and a very fine sub-local problem as the third scale. The sub-local problem will be used to enrich the local problem, and the local problem will be used to enrich the global problem. With this approach, a small local domain can be chosen, since the third scale can be used to obtain a more precise solution.

- Another promising topic for future studies is using of parallel processing strategy to accelerate solution procedure. Although, the INSANE empowered by incorporating a *SuiteSparse* matrix approach which uses the advantage of sparse matrix to solve the matrix system of equations. But, implementing a parallel processing approach can help two-scale G/XFEM analysis with multiple local problems to solve each local problem with a separated thread/processor.

- The local domain from global problem in two-scale G/XFEM approach was considered as a fixed boundary during the crack propagation process. However, the more efficient way to deal with this kind of problems is to have a moving local problem that updates itself with the crack-tip position.

- The last recommendation but not least, is the modeling the crack front formation, i.e., the orientation of the crack front, with respect the upper/lower surfaces of the problem. There is no study in the literature that considered this behavior for the Reissner-Mindlin plates.

Appendix A

SIF CALCULATION PROCEDURE

Interaction energy integral method is used here to calculate the stress intensity factors. This method requires auxiliary displacement, stress and strain field solutions. Both plane stress and Reissner-Mindlin plate problems will be discussed in this section. These explanations are brought from Malekan and Barros (2017).

A.1 Plane Stress Problems

The J-integral contour for a plane stress problem is defined as Rice (1968):

$$J = \oint_{\Gamma} \left[W\delta_{1j} - \sigma_{ij}\frac{\partial u_i}{\partial x_1} \right] n_j d\Gamma \tag{A.1}$$

where W is the strain energy density defined as $W = \frac{1}{2}\sigma_{ij}\epsilon_{ij}$, σ_{ij} and ϵ_{ij} are the stress and strain tensor, respectively, u_i is the displacement field, n_j is the unit outward normal vector to the contour integral Γ that contains the crack, and δ is the Kronecker delta.

Following Belytschko and Black (1999); Nguyen (2005), assume two states of a cracked body: State (1) $(\sigma_{ij}^{(1)}, \epsilon_{ij}^{(1)}, u_i^{(1)})$ represents the current state and State (2) $(\sigma_{ij}^{(2)}, \epsilon_{ij}^{(2)}, u_i^{(2)})$ is an auxiliary state which will be chosen as the asymptotic fields for modes I and II. The J-integral for the sum of the these two states is:

$$J^{(1+2)} = \oint_{\Gamma} \left[\frac{1}{2}\left(\sigma_{ij}^{(1)} + \sigma_{ij}^{(2)}\right)\left(\epsilon_{ij}^{(1)} + \epsilon_{ij}^{(2)}\right)\delta_{1j} - \left(\sigma_{ij}^{(1)} + \sigma_{ij}^{(2)}\right)\frac{\partial}{\partial x_1}\left(u_i^{(1)} + u_i^{(2)}\right) \right]_j d\Gamma \tag{A.2}$$

expanding Eq. (A.2) results in:

$$J^{(1+2)} = J^{(1)} + J^{(2)} + I^{(1,2)} \tag{A.3}$$

where $J^{(1)}$ and $J^{(2)}$ are obtained from Eq. (A.1) and the states (1) and (2), respectively. $I^{(1,2)}$ is called the interaction integral for states (1) and (2) and defined as:

$$I^{(1,2)} = \oint_{\Gamma} \left[W^{(1,2)} \delta_{1j} - \sigma_{ij}^{(1)} \frac{\partial u_i^{(2)}}{\partial x_1} - \sigma_{ij}^{(2)} \frac{\partial u_i^{(1)}}{\partial x_1} \right] n_j d\Gamma \tag{A.4}$$

in which $W^(1,2)$ is the interaction strain energy defined by:

$$W^{(1,2)} = \sigma_{ij}^{(1)} \epsilon_{ij}^{(2)} = \sigma_{ij}^{(2)} \epsilon_{ij}^{(1)} \tag{A.5}$$

For general mixed mode problems, the following relationship between the value of the J-integral and the stress intensity factors can be generally defined Rice (1968):

$$J = \frac{1}{E'} \left(K_I^2 + K_{II}^2 \right) \tag{A.6}$$

in which $E' = E$ for plane stress and $E' = E/(1-\nu^2)$ for plane strain problems. The J-integral definition in Eq. (A.6) can be rewritten for the combined states (1) and (2), using $K_I = K_I^{(1)} + K_I^{(2)}$ and $K_{II} = K_{II}^{(1)} + K_{II}^{(2)}$, as:

$$J^{(1+2)} = J^{(1)} + J^{(2)} + \frac{2}{E'} \left(K_I^{(1)} K_I^{(2)} + K_{II}^{(2)} K_{II}^{(1)} \right) \tag{A.7}$$

Equations (A.3) and (A.7) provide this new description to the interaction integral:

$$I^{(1,2)} = \frac{2}{E'} \left(K_I^{(1)} K_I^{(2)} + K_{II}^{(2)} K_{II}^{(1)} \right) \tag{A.8}$$

$K_I^{(1)}$ and $K_{II}^{(1)}$ can be obtained by choosing an appropriate auxiliary field for state (2). For an auxiliary field of mode I (with $K_I^{(2)} = 1$ and $K_{II}^{(2)} = 0$), the interaction integral $I^{(1,2)}$ is expressed as $I^{(1)}_{\text{mode } I}$:

$$K_I^{(1)} = \frac{E'}{2} I^{(1)}_{\text{mode } I} \tag{A.9}$$

Similarly, to a mode II auxiliary field (with $K_I^{(2)} = 0$ and $K_{II}^{(2)} = 1$), $I^{(1,2)}$ is expressed as $I^{(1)}_{\text{mode } II}$ and:

$$K_{II}^{(1)} = \frac{E'}{2} I^{(1)}_{\text{mode } II} \tag{A.10}$$

Aiming to simplify the numerical integration of Eq. (A.4), it can be converted into an area integral by multiplying the integrand with a smooth bounded weighting function q, described below. Then for each contour Γ, assuming the crack faces are stress free and straight in the interior of the region A and bounded by the outer prescribed contour C_0, the interaction integral may be written as:

$$I^{(1,2)} = \oint_{C} \left[W^{(1,2)} \delta_{1j} - \sigma_{ij}^{(1)} \frac{\partial u_i^{(2)}}{\partial x_1} - \sigma_{ij}^{(2)} \frac{\partial u_i^{(1)}}{\partial x_1} \right] q \, m_j \, d\Gamma \tag{A.11}$$

with:

$$q = \begin{cases} 1 & \text{on } \Gamma \\ \\ 0 & \text{on } C_0 \end{cases} \tag{A.12}$$

where $C = \Gamma + C_+ + C_- + C_0$ and $\boldsymbol{m_j}$ is the outward unit normal vector, equal to n_j on $C_0 \cup C_+ \cup C_-$ to the contour C, see Fig. A.1(a). The divergence theorem is used, which gives the following equation for the interaction integral in domain form:

$$I^{(1,2)} = \oint_A \left[\sigma_{ij}^{(1)} \frac{\partial u_i^{(2)}}{\partial x_1} + \sigma_{ij}^{(2)} \frac{\partial u_i^{(1)}}{\partial x_1} - W^{(1,2)} \delta_{1j} \right] \frac{\partial q}{\partial x_j} dA \tag{A.13}$$

Following expressions can be adopted to compute the displacement, stress and strain fields for auxiliary states:

Table A.1: Expressions to calculate auxiliary states.

Field	Mode I	Mode II
u_x	$\frac{1}{2\mu}\sqrt{\frac{r}{2\pi}} \cos\frac{\theta}{2}\left(\kappa - 1 + 2\sin^2\frac{\theta}{2}\right)$	$\frac{1}{2\mu}\sqrt{\frac{r}{2\pi}} \sin\frac{\theta}{2}\left(\kappa + 1 + 2\cos^2\frac{\theta}{2}\right)$
u_y	$\frac{1}{2\mu}\sqrt{\frac{r}{2\pi}} \sin\frac{\theta}{2}\left(\kappa + 1 - 2\cos^2\frac{\theta}{2}\right)$	$\frac{1}{2\mu}\sqrt{\frac{r}{2\pi}} \cos\frac{\theta}{2}\left(\kappa - 1 - 2\sin^2\frac{\theta}{2}\right)$
σ_{xx}	$\frac{1}{\sqrt{2\pi r}} \cos\frac{\theta}{2}\left(1 - \sin\frac{\theta}{2}\sin\frac{3\theta}{2}\right)$	$-\frac{1}{\sqrt{2\pi r}} \sin\frac{\theta}{2}\left(2 + \cos\frac{\theta}{2}\cos\frac{3\theta}{2}\right)$
σ_{yy}	$\frac{1}{\sqrt{2\pi r}} \cos\frac{\theta}{2}\left(1 + \sin\frac{\theta}{2}\sin\frac{3\theta}{2}\right)$	$\frac{1}{\sqrt{2\pi r}} \sin\frac{\theta}{2}\cos\frac{\theta}{2}\cos\frac{3\theta}{2}$
τ_{xy}	$\frac{1}{\sqrt{2\pi r}} \cos\frac{\theta}{2}\sin\frac{\theta}{2}\cos\frac{3\theta}{2}$	$\frac{1}{\sqrt{2\pi r}} \cos\frac{\theta}{2}\left(1 - \sin\frac{\theta}{2}\sin\frac{3\theta}{2}\right)$

where μ and κ are material constants defined by:

$$\mu = \frac{E}{2(1+\nu)}; \qquad \kappa = \begin{cases} 3 - 4\nu & \text{plane strain} \\ \\ \dfrac{3-\nu}{1+\nu} & \text{plane stress} \end{cases} \tag{A.14}$$

By having the displacement, one can obtain the strain fields by using the following expression:

$$\epsilon_{ij}^{(2)} = \frac{1}{2}\left(u_{i,j}^{(2)} + u_{j,i}^{(2)}\right) \tag{A.15}$$

Since all terms in Eq. (A.13) were defined, the interaction integral can be calculated numerically over the domain elements (Fig. A.1(b)) using:

$$I^{(1,2)} = \sum_{\substack{\text{elements} \\ \text{over domain A}}} \sum_{gp=1}^{N_{gp}} \left\{ \left[\sigma_{ij}^{(1)} \frac{\partial u_i^{(2)}}{\partial x_1} + \sigma_{ij}^{(2)} \frac{\partial u_i^{(1)}}{\partial x_1} - W^{(1,2)} \delta_{1j} \right] \frac{\partial q}{\partial x_j} \right\} w_{gp} \, \text{detJ} \tag{A.16}$$

where N_{gp} is the number of Gauss points, w_{gp} is the Gauss point weight, and $detJ$ is the determinant of the Jacobian.

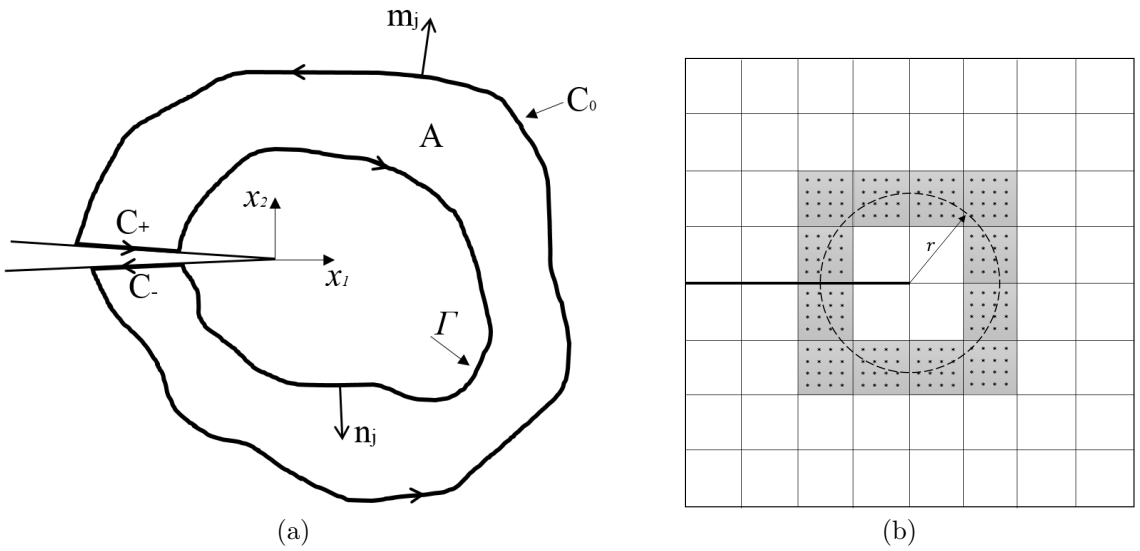


Figure A.1: J-integral domain definition: (a) Conventions at crack tip. Domain A is enclosed by Γ, C_+, C_-, and C_0. Unit normal $m_j = n_j$ on C_+, C_-, and C_0 and $m_j = -n_j$ on Γ. (b) Elements selected around the crack tip, over a circle with radius of r, for the interaction energy integral calculation.

A.2 Reissner-Mindlin plate problem

The J-integral contour for a Reissner-Mindlin plate problem is defined as Dolbow et al (2000b):

$$J = \oint_{\Gamma} \left\{ W\delta_{1\beta} - \left[M_{\alpha\beta}\psi_{\alpha,1} + Q_{\beta}w_{,1} \right] \right\} n_{\beta} d\Gamma \tag{A.17}$$

where $M_{\alpha\beta}$ is the bending moment, Q_β is the shear, w is the transverse displacement and ψ_α is section rotation about the x_α axes that define the middle plane of the plate, in which α and β ranging over the values 1,2.

After following similar mathematical to the section A.1, one can reach following interaction energy integral equation for the Reissner-Mindlin plate:

$$I = \oint_{\Gamma} \left\{ W\delta_{1\beta} - \left[M^{(1)}_{\alpha\beta}\psi^{(2)}_{\alpha,1} + M^{(2)}_{\alpha\beta}\psi^{(1)}_{\alpha,1} + Q^{(1)}_{\beta}w^{(2)}_{,1} + Q^{(2)}_{\beta}w^{(1)}_{,1} \right] \right\} n_j d\Gamma \tag{A.18}$$

in which, similar to the section A.1, State (1) represents the current state and State (2) is an auxiliary state. The interaction strain energy, W, is defined by:

$$W^{(1,2)} = M^{(1)} : \epsilon^{(2)}_b + Q^{(1)} \cdot \epsilon^{(2)}_s = M^{(2)} : \epsilon^{(1)}_b + Q^{(2)} \cdot \epsilon^{(1)}_s \tag{A.19}$$

where ϵ_b and ϵ_s are bending and shear strains, respectively. Referring to Fig. A.1(a) and Eq. (A.12), Eq. (A.18) can be rewritten for a crack with traction free faces as:

$$I = \oint_{C} \left\{ -W\delta_{1\beta} + \left[M^{(1)}_{\alpha\beta}\psi^{(2)}_{\alpha,1} + M^{(2)}_{\alpha\beta}\psi^{(1)}_{\alpha,1} + Q^{(1)}_{\beta}w^{(2)}_{,1} + Q^{(2)}_{\beta}w^{(1)}_{,1} \right] \right\} m_\beta \, q \, dC \tag{A.20}$$

Applying the divergence theorem to the integral over A, we obtain:

$$I = \int_A \left\{ \left[M^{(1)}_{\alpha\beta} \psi^{(2)}_{\alpha,1} + M^{(2)}_{\alpha\beta} \psi^{(1)}_{\alpha,1} + Q^{(1)}_{\beta} w^{(2)}_{,1} + Q^{(2)}_{\beta} w^{(1)}_{,1} \right] - W\delta_{1\beta} \right\} q_\beta \, dA \tag{A.21}$$

The above integral can be reduced depending on whether the quantity of interest is K_I, K_{II}, or K_{III}, as certain terms in the auxiliary fields vanish for each case. For example, for K_I and K_{II} the integral takes the form:

$$I = \int_A \left\{ \left[Q^{(1)}_{\beta} w^{(2)}_{,1} + Q^{(2)}_{\beta} w^{(1)}_{,1} \right] - W\delta_{1\beta} \right\} q_\beta \, dA \tag{A.22}$$

whereas for K_{III} the integral is:

$$I = \int_A \left\{ \left[M^{(1)}_{\alpha\beta} \psi^{(2)}_{\alpha,1} + M^{(2)}_{\alpha\beta} \psi^{(1)}_{\alpha,1} \right] - W\delta_{1\beta} \right\} q_\beta \, dA \tag{A.23}$$

The auxiliary state for the displacement fields in Reissner-Mindlin plate theory can be found in Sosa (1986) as a power series in $\sqrt{r}$ and are shown in the following Table.

Table A.2: Auxiliary states for the displacement fields of Reissner-Mindlin plate.

Field	Mode I	Mode II	Mode III
w	$\frac{6r\sqrt{2r}}{Et^3}\left[\frac{1}{3}(7+\nu)C_{\frac{3\theta}{2}} - (1-\nu)C_{\frac{\theta}{2}}\right]$	$\frac{6r\sqrt{2r}}{Et^3}\left[-\frac{1}{3}(5+3\nu)S_{\frac{3\theta}{2}} + (1-\nu)S_{\frac{\theta}{2}}\right]$	$\frac{6\sqrt{2r}}{5t\mu}S_{\frac{\theta}{2}}$
ψ_1	$\frac{6\sqrt{2r}}{Et^3}C_{\frac{\theta}{2}}\left[4-(1+\nu)C_\theta\right]$	$\frac{6\sqrt{2r}}{Et^3}S_{\frac{\theta}{2}}\left[4+(1+\nu)C_\theta\right]$	$\frac{16r\sqrt{2r}}{5Et^3}\left[-S_{\frac{\theta}{2}} - (1+3\nu)C_{\frac{\theta}{2}}S_\theta\right]$
ψ_2	$\frac{6\sqrt{2r}}{Et^3}\left[4S_{\frac{\theta}{2}} - (1+\nu)C_{\frac{\theta}{2}}S_\theta\right]$	$\frac{6\sqrt{2r}}{Et^3}\left[-2C_{\frac{\theta}{2}}(1-\nu) + (1+\nu)S_{\frac{\theta}{2}}S_\theta\right]$	$\frac{16r\sqrt{2r}}{5Et^3}C_{\frac{\theta}{2}}\left[1+(1+3\nu)C_\theta\right]$

in which C_θ and S_θ represent $\cos\theta$ and $\sin\theta$ functions, respectively. Moreover, the auxiliary bending moments and shears in polar coordinates are as follows (Sosa, 1986):

$$M_{rr} = -\frac{K_1}{4\sqrt{2r}}\left(\cos\frac{3\theta}{2} - 5\cos\frac{\theta}{2}\right) + \frac{3K_2}{4\sqrt{2r}}\left(\sin\frac{3\theta}{2} - \frac{5}{3}\sin\frac{\theta}{2}\right) \tag{A.24}$$

$$M_{\theta\theta} = \frac{K_1}{4\sqrt{2r}}\left(\cos\frac{3\theta}{2} + 3\cos\frac{\theta}{2}\right) - \frac{3K_2}{4\sqrt{2r}}\left(\sin\frac{3\theta}{2} + \sin\frac{\theta}{2}\right) \tag{A.25}$$

$$M_{r\theta} = \frac{K_1}{4\sqrt{2r}}\left(\sin\frac{3\theta}{2} + \sin\frac{\theta}{2}\right) + \frac{3K_2}{4\sqrt{2r}}\left(\cos\frac{3\theta}{2} + \frac{1}{3}\cos\frac{\theta}{2}\right) \tag{A.26}$$

$$Q_r = \frac{K_3}{\sqrt{2r}}\sin\frac{\theta}{2} \tag{A.27}$$

$$Q_\theta = \frac{K_3}{\sqrt{2r}}\cos\frac{\theta}{2} \tag{A.28}$$

The quantities M_{11}, M_{22}, M_{12}, Q_1, and Q_2 can be calculated from the following transformation formulas:

$$M_{11} = M_{rr}\cos^2\theta + M_{\theta\theta}\sin^2\theta - 2M_{r\theta}\sin\theta\cos\theta \tag{A.29}$$

$$M_{22} = M_{rr}\sin^2\theta + M_{\theta\theta}\cos^2\theta + 2\,M_{r\theta}\sin\theta\cos\theta \tag{A.30}$$

$$M_{12} = (M_{rr} - M_{\theta\theta}) \sin\theta \cos\theta + 2\, M_{r\theta}(\cos^2\theta - \sin^2\theta) \tag{A.31}$$

$$Q_1 = Q_r \cos\frac{\theta}{2} - Q_\theta \sin\theta \tag{A.32}$$

$$Q_2 = Q_r \sin\frac{\theta}{2} + Q_\theta \cos\theta \tag{A.33}$$

Thus, we can have the following relations for auxiliary bending moments and shears:

$$M_{11} = \frac{K_1}{\sqrt{2r}} \cos\frac{\theta}{2}\left(1 - \sin\frac{\theta}{2}\sin\frac{3\theta}{2}\right) - \frac{K_2}{\sqrt{2r}} \sin\frac{\theta}{2}\left(2 + \cos\frac{\theta}{2}\cos\frac{3\theta}{2}\right) \tag{A.34}$$

$$M_{22} = \frac{K_1}{\sqrt{2r}} \cos\frac{\theta}{2}\left(1 - \sin\frac{\theta}{2}\sin\frac{3\theta}{2}\right) - \frac{K_2}{\sqrt{2r}} \sin\frac{\theta}{2}\cos\frac{\theta}{2}\cos\frac{3\theta}{2} \tag{A.35}$$

$$M_{12} = \frac{K_1}{\sqrt{2r}} \sin\frac{\theta}{2}\cos\frac{\theta}{2}\cos\frac{3\theta}{2} + \frac{K_2}{\sqrt{2r}} \cos\frac{\theta}{2}\left(1 - \sin\frac{\theta}{2}\sin\frac{3\theta}{2}\right) \tag{A.36}$$

$$Q_1 = -\frac{K_3}{\sqrt{2r}} \sin\frac{\theta}{2} \tag{A.37}$$

$$Q_2 = \frac{K_3}{\sqrt{2r}} \cos\frac{\theta}{2} \tag{A.38}$$

As an example, K_2 and K_3 must be set equal to zero in all equation in order to calculate auxiliary moment intensity factor of mode I. The process of evaluating the mixed-mode intensity factors must be carried out with a judicious choice of the auxiliary moment and shear force intensity factors to evaluating the interaction energy integral. From the Eq. (A.17) and the energy release rate formulation, one can obtain the following expression:

$$I = \frac{24\pi}{Et^3}\left[K_I K_I^{(2)} + K_{II} K_{II}^{(2)}\right] + \frac{12\pi}{10\mu t} K_{III} K_{III}^{(2)} \tag{A.39}$$

where, to extract K_I, the following values is chosen$K_i^{(2)} = 1$, $K_{II}^{(2)} = 0$, and $K_{III}^{(2)} = 0$. Then, the moment intensity factor K_I can be calculated as:

$$K_I = \frac{Et^3}{24\pi} I \tag{A.40}$$

Appendix B

XML INPUT FILE

This section presents a detail explanation on different parts of the *XML* input file for both classical and two-scale analysis. Code B.1 shows the input data for *Solution* type definition for classical analysis using *StaticEquilibriumPath* solver class. It needs to define total number of steps, required parameters for iterative approach such as: maximum number of iterations; tolerance; convergence type (Force:1, Displacement:2, Both:3); and equilibrium type (1, 2, 3), and iterative strategy (such as displacement control and cylindrical arc-length control).

```
<Solution class="StaticEquilibriumPath">
  <NumMaxSteps>10</NumMaxSteps>
  <Step class="StandardNewtonRaphson">
    <NumMaxIterations>100</NumMaxIterations>
    <Tolerance>1.0E-08</Tolerance>
    <ConvergenceType>1</ConvergenceType>
    <EquilibriumType>1</EquilibriumType>
  </Step>
  <IterativeStrategyList>
    <IterativeStrategy class="DisplacementControl" LoadFactor="0.1">
      <NodeControl>10</NodeControl>
      <DirectionControl>y</DirectionControl>
    </IterativeStrategy>
  </IterativeStrategyList>
</Solution>
```

Code B.1: XML input file for *Solution* definition

Two-scale analysis is done using *GlobalLocal* and *GlobalLocalQuasiStatic* for static and quasi-static crack propagation problems, respectively, as shown in Code B.2. *GlobalEnrichedNodes* defines global node labels to be enriched by the global-local enrichment strategy. *LocalEnricherList* specifies whether any node of GlobalEnrichedNodes is belong to two-overlapped local element, if so it must be specified by number '1' in this tag. *LocalPath* is related to the local file name, which it only differs by a string defined in this tag from the global file name. *GlobalSolverType*, *LocalSolverType*, and *GlobalLocalSolverType* specify the solver type at each step of the global-local analysis for step 1, step 2, and step 3, respectively (see Fig. 2.4 for step definition). *GlobalLocalIterations* defines how many iterations must be followed for each static or quasi-static step. For example, if this values is set equal to 2, the global-local steps will be repeated two times for the same loading magnitude to have a more precise solution. Finally, the *NumMaxSteps* defines number of quasi-static steps for the crack propagation procedure. This tag must be used only with the *GlobalLocalQuasiStatic* solver class.

```
<Solution class="GlobalLocalQuasiStatic">
  <GlobalEnrichedNodes>9 18</GlobalEnrichedNodes>
  <LocalEnricherList>0 0</LocalEnricherList>
  <LocalPath>-1</LocalPath>
  <GlobalSolverType>4</GlobalSolverType>
  <LocalSolverType>4</LocalSolverType>
  <GlobalLocalSolverType>4</GlobalLocalSolverType>
  <GlobalLocalIterations>3</GlobalLocalIterations>
  <NumMaxSteps>4</NumMaxSteps>
</Solution>
```

Code B.2: XML input file for *quasi-static global-local* analysis definition

Code B.3 defines the material properties, for either linear or nonlinear analysis. It is a list which means the model can be created with multiple material mediums. The tag *Fracture-Toughness* is used for crack propagation in order to verify the crack propagation with the literature, if there is any.

```
<MaterialList>
  <Material class="LinearElasticIsotropic" label="Material">
    <Elasticity>1.0</Elasticity>
    <Poisson>0.3</Poisson>
    <FractureToughness>0.025</FractureToughness>
  </Material>
</MaterialList>
```

Code B.3: XML input file for *Material* definition

The enrichment definition is brought with the explanation in the Code B.4. This block of codes contain two different enrichment types: singular and polynomial for classical and stable G/XFEM analyses, respectively. *InitialPoint* is related to the crack-tip point, *Lambda* and Q are singular parameters shown in Eq. (2.12), and *Theta* is the crack orientation with respect to the positive x-coordinate direction from global coordinate system. The parameters in *StablePolynomialEnrichment* type are related to order of polynomials that is used to enrich the problem.

```
<EnrichmentList>
  <Enrichment class="CrackEnrichmentModeI" label="crack">
    <InitialPoint>8.0 0.0 0</InitialPoint>
    <Lambda>0.5</Lambda>
    <Q>0.333333333</Q>
    <Theta>3.14159265358</Theta>
  </Enrichment>
  <Enrichment class="StablePolynomialEnrichment" label="poly">
    <X>1 0 2 0</X>
    <Y>0 1 0 2</Y>
    <Z>0 0 0 0</Z>
  </Enrichment>
</EnrichmentList>
```

Code B.4: XML input file for *Enrichment* definition

Code B.5 is shown the nodal definition tags. It contains the nodal: coordinate, labels, restraint types (*false*:free to move, *true*:fixed), enrichment type definition (can hold multiple enrichment types for a single node), and scaling factor which mostly related to the polynomial enrichment function.

```
<Node label="1">
  <Coord>0.000000 1.000000 0.000000</Coord>
  <NodeValues>
    <DOFLabels>Dx Dy</DOFLabels>
```

```
5       <Restraints>false false</Restraints>
6       <EnrichmentType>poly</EnrichmentType>
7       <ScalingFactor>1</ScalingFactor>
8     </NodeValues>
9   </Node>
```

Code B.5: XML input file for *Node* definition

The main part to specify the linear or nonlinear materials is in the element definition, as shown in the Code B.6. *Incidence* defines the nodal incidence of the element and *AnalysisModel* specifies the problem type, from plane stress (*GFemPlaneStress*) and plane strain (*GFemPlaneStrain*) to Reissner-Mindlin plate (*GFemReissnerMindlinPlate*), for G/XFEM analysis. Gaussian quadrature order is defined with *IntegrationOrder* tag. *ConstitutiveModel* defines if the material mode is linear (*LinearElasticConstModel*) or nonlinear (*MLFOCM_SCM_FD*) and *ElmDegenerations* sets the element section properties. *GlobalElement* tag is used in the local problem definition in order to specify this local element belongs to which global element. *BoundaryInformation* defines the boundary condition type to be applied on the corresponding local problem boundary, from the global solution comes from the first global-local step. Each number is related to corresponding element edge: 0 for apply nothing on the boundary, 3 for Neumann boundary, 4 for Dirichlet boundary, and 5 for Cauchy boundary transferring types.

```
1  <Element class="ParametricElement.Quadrilateral.Q4" label="E1">
2    <Incidence>1 2 3 4</Incidence>
3    <AnalysisModel>GFemPlaneStress</AnalysisModel>
4    <!-- or "GFemReissnerMindlinPlate" for Reissner-Mindlin problem-->
5    <IntegrationOrder>6 6 0</IntegrationOrder>
6    <ConstitutiveModel>LinearElasticConstModel</ConstitutiveModel>
7    <!-- or "MLFOCM_SCM_FD" for Nonlinear materials -->
8    <ElmDegenerations>Section</ElmDegenerations>
9    <GlobalElement>F10</GlobalElement>
10   <BoundaryInformation>3 0 0 3</BoundaryInformation>
11 </Element>
```

Code B.6: XML input file for *Element* definition

Code B.7 has required tags to define the discontinuity type, crack line, and other parameters for the static and quasi-static analysis based on G/XFEM method. This code defines the crack geometry by setting two points, representing a straight line. Lines 5-6 and 11-13 are parameters related to the singular (crack) or near-tip enrichment functions. Lines 15-18 fill some parameters specifically for LEFM crack propagation, such as: *fractureModes* to define mode(s) of fracture (1, 2, 3, 4, 5, and 6 for mode-I, mode-II, mode-III, mode-I and II, mode-I and III, and mode-II and III, respectively), *InteractIntegRadiusMultiplier* to define the interaction integral multiplier, i.e. r_m, *SingleOrDoubleCrackTip* to define whether both crack-tips are inside of the model boundaries or not, *crackIncrement* which defines the δa, from section 5.2.2. The tag *IsWithCrackPropagation* specifies if the problem is a static ('*false*') analysis or quasi-static ('*true*'). Finally, the *OnlyOneCrack* forces the problem to be analyzed with a single ('true') or multiple cracks ('false'). This belongs only for nonlinear material analysis procedure.

```
1  <DiscontinuityList>
2    <Discontinuity class="LEFMcrackGrowthByGfem" label="D1">
3      <FirstPoint>
```

```
      <coordinateFirst>10.0 2.0 0.0</coordinateFirst>
      <firstTheta0>0.0</firstTheta0>
      <firstQ>0.3333333</firstQ>
      <firstLambda>0.5</firstLambda>
    </FirstPoint>
    <SecondPoint>
      <coordinateSecond>8.0 2.0 0.0</coordinateSecond>
      <secondTheta0>3.14159265359</secondTheta0>
      <secondQ>0.3333333</secondQ>
      <secondLambda>0.5</secondLambda>
    </SecondPoint>
    <fractureModes>4</fractureModes>
    <InteractIntegRadiusMultiplier>2.0</InteractIntegRadiusMultiplier
        >
    <SingleOrDoubleCrackTip>2</SingleOrDoubleCrackTip>
    <crackIncrement>0.75</crackIncrement>
  </Discontinuity>
</DiscontinuityList>
<IsWithCrackPropagation boolean="false">
  <OnlyOneCrack>true</OnlyOneCrack>
  <crackPropagationType>LEFMcrackGrowthByGfem</crackPropagationType>
</IsWithCrackPropagation>
```

Code B.7: XML input file for *Discontinuity* definition

Appendix C

SuiteSparse MATRIX APPROACH

A **Sparse** matrix data structure uses the advantage of sparse matrix to solve the matrix system of equations. In order to optimize computational time, the *sparse matrix* can be held in some form of compact data structure that avoids storing the numerically zero entries in the matrix. The two most common formats for sparse direct methods are the triplet matrix and the compressed-column matrix (or the compressed-row matrix) (Davis, 2004). The simplest sparse matrix data structure is a list of the non-zero entries in arbitrary order, also called the triplet form. However, in the compressed-column format, which is used in INSANE computational framework, each column is represented as a list of values and their corresponding row indices for each column. To create this data structure, the first step counts the number of entries in each column of the matrix, and the column pointer array, as well as its corresponding value, is constructed as the cumulative sum of the column counts. The entries are placed in their appropriate columns in a second step.

Popular choice to solve static finite element problems ($A\boldsymbol{x} = b$) are direct methods that rely on numerical factorizations. Among various direct methods, one can use the unsymmetric multifrontal method in particular (Davis and Duff, 2004; Pais et al, 2012; Zheng and Luo, 2014; Davis et al, 2016). The UMFPACK (Davis, 2009) code is written based on the multifrontal method. This code is chosen to accelerate the processor part mainly because it is well-designed and its source codes are easily available. It already has been used by different commercial and in-house codes, such as ANSYS (2015); NASTRAN (2015), and FreeFem++(Hecht, 2012).

A typical multifrontal algorithm consists of the symbolic and the numerical factorization. In the symbolic factorization stage, an elimination tree is formed. The numerical factorization includes pivoting, assembling, and updating is then performed on each of the frontal matrices. For a frontal matrix, the following transformation is performed (Yu et al, 2011).

$$\begin{bmatrix} P & Q \\ R & C \end{bmatrix} = \begin{bmatrix} L_1 & 0 \\ L_2 & C - L_2U_2 \end{bmatrix} \begin{bmatrix} U_1 & U_2 \\ 0 & 1 \end{bmatrix} \tag{C.1}$$

where block P contains all of the pivots in the current supernode, blocks Q and R contain

Algorithm C.1: General multifrontal numerical factorization approach used in UMFPACK (Davis, 2004; Yu et al, 2011)

```
1: procedure NUMERICAL FACTORIZATION
2:     Initialization
3:     Symbolic factorization and form the elimination tree
4:     for all frontal matrices do
5:         while factorize the frontal do
6:             Assemble using update matrices and original matrix elements
7:             Perform row pivoting
8:             Factor the pivot to get a part of L1 and U1
9:             Update part of L2
10:        end while
11:        Update U2 and C
12:    end for
13: end procedure
```

the corresponding nonzero rows and columns of the pivots, $P = L_1U_1$, $L_2 = RU_1^{-1}$, and $U_2 = L_1^{-1}Q$. Figure C.1 shows a schema of the matrices notations for the multifrontal method used in UMFPACK. Also, algorithm C.1 is a simplified version of the unsymmetric multifrontal method used by UMFPACK.

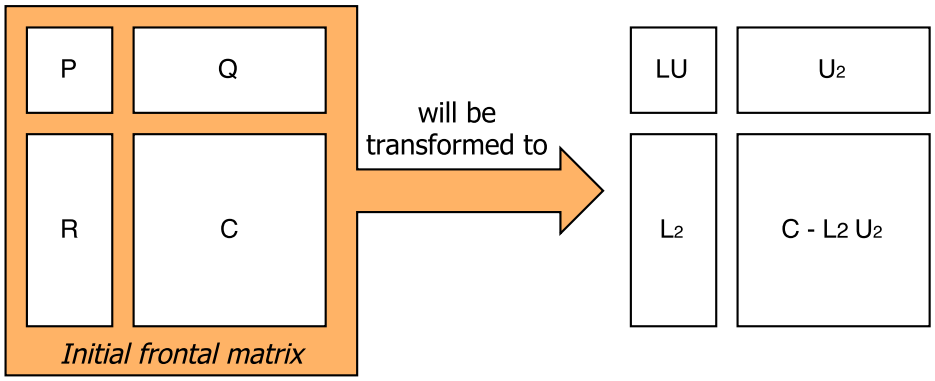


Figure C.1: Matrices notations for the multifrontal method

Figure C.2 shows the structure of the *SparseMatrix* class that extends the *IMatrix* class (Andrade and Silva, 2015). The main implementation for conducting and utilizing the UMFPACK capabilities is written in this class. After filling the required parts of this class, the corresponding libraries related to the UMFPACK will be called in order to solve the system of equations.

The example presented in section 4.3.1 were also solved using the *SparseMatrix* approach. As mentioned earlier, using the *SuiteSparse* library is a powerful tool to accelerate solving procedure of matrix system of equations. Table C.1 clearly shows the advantage of using the LU factorization algorithm provided by the *SuiteSparse* library over a so-called conventional approaches (Babuška iterative approach). In INSANE, this last one also corresponds to the LU factorization, but it is totally written in *Java* and the stiffness matrix is fully stored.

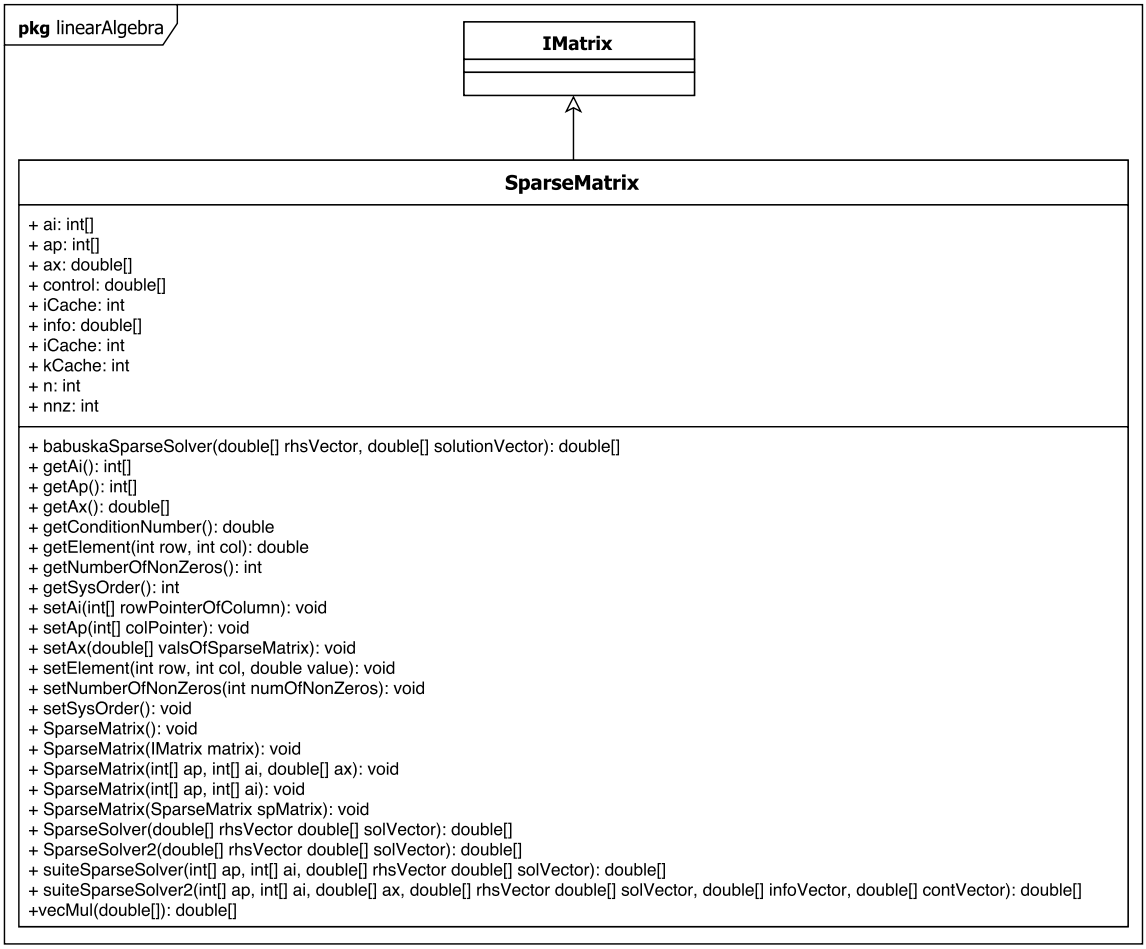


Figure C.2: UML diagram of the *SparseMatrix* class

Only corresponding data to the enriching problem with the singular enrichment are brought in this Table.

Table C.1: Comparing CPU time between conventional and using the *SuiteSparse* library

	G/XFEM			***SG/XFEM***		
h size	2	1	0.5	2	1	0.5
DOF	389	718	1940	389	718	1940
CPU time (msec) Babuška iterative method	145	722	6369	139	667	6081
CPU time (msec) SuiteSparse method	8	25	48	8	23	44

Appendix D

PCCG MIXED WITH THE *BABUŠKA* APPROACH

The conjugate gradient (CG) method was first proposed by Hestenes and Stiefel in 1952 (Hestenes and Stiefel, 1952) as a direct solver. Moreover, it is one of the most effective and simple methods that can solve the linear system of equations in an iterative fashion. It and is based on the idea that the solution of linear system of equations $\boldsymbol{Ax} = \boldsymbol{b}$ can be transformed to the minimization of the potential $\Pi = \frac{1}{2}\boldsymbol{x}^T\boldsymbol{Ax} - \boldsymbol{bx}$ (Bathe, 1996). However, the CG method is not always more efficient than direct methods such as LU and Cholesky factorization methods. The convergence rate of the CG method greatly depends on the condition number of the matrix $\boldsymbol{A}$. If the conditioning of matrix $\boldsymbol{A}$ is very poor, the CG method may lead to higher computational cost than a direct solver. Consequently, in industrial applications, a preconditioner $\boldsymbol{C}$ is almost always used with the conjugate gradient method to improve the conditioning of the matrix. This approach is denoted the preconditioned conjugate gradient method (PCCG). According (Bathe, 1996; Kim et al, 2015), the following two steps with their detail sub-steps is the preconditioned conjugate gradient algorithm:

(1) Initialization: Choose the initial guess $\boldsymbol{x}_0$:

(i). $\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{Ax}_0$

(ii). $\bar{\boldsymbol{r}}_0 = \boldsymbol{C}^{-1}\boldsymbol{r}_0$

(iii). $\boldsymbol{p}_0 = \bar{\boldsymbol{r}}_0$

(2) Preconditioned conjugate gradient algorithm: While $\| \boldsymbol{b} - \boldsymbol{Ax}_i \| / \| \boldsymbol{b} \| \geq TOL$ for $i = 0, 1, 2, ...$ repeat the following procedure. Here, TOL denotes the level of tolerance set for convergence.

(i). $\lambda_i = \dfrac{\bar{\boldsymbol{r}}_i^T \boldsymbol{r}_i}{\boldsymbol{p}_i^T \boldsymbol{Ap}_i}$

(ii). $\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \lambda_i \boldsymbol{p}_i$

Algorithm D.1: Algorithm of the Babuŭka procedure (Strouboulis et al, 2000b; Barros, 2002)

1: Initialization
2: $T_{ij} = \frac{\delta_{ij}}{K_{ij}}$
3: $\bar{\boldsymbol{F}} = \boldsymbol{TF}$
4: $\bar{\boldsymbol{K}} = \boldsymbol{TKT}$ (normalizing the principal diagonal)
5: $\boldsymbol{K_\epsilon} = \bar{\boldsymbol{K}} + \epsilon \boldsymbol{I}$
6: $\bar{\boldsymbol{U}}_0 = \boldsymbol{K}_\epsilon^{-1} \bar{\boldsymbol{F}}$
7: $\boldsymbol{r}_0 = \bar{\boldsymbol{F}} - \boldsymbol{K}\bar{\boldsymbol{U}}_0$
8: $\boldsymbol{e}_0 = \boldsymbol{K}_\epsilon^{-1} \boldsymbol{r}_0$
9: $i = 1$
10: **while** $\| \frac{\boldsymbol{e}_i \bar{\boldsymbol{K}} \boldsymbol{e}_i}{\boldsymbol{U}_i \bar{\boldsymbol{K}} \boldsymbol{U}_i} \| < TOL$ **do**
11: $\quad \boldsymbol{r}_i = \boldsymbol{r}_0 - \sum_{j=0}^{i-1} \bar{\boldsymbol{K}} \boldsymbol{e}_j$
12: $\quad \boldsymbol{e}_i = \boldsymbol{K}_\epsilon^{-1} \boldsymbol{r}_i$
13: $\quad \bar{\boldsymbol{U}}_i = \bar{\boldsymbol{U}}_0 + \sum_{j=0}^{i-1} \boldsymbol{e}_j$
14: $\quad i = i + 1$
15: **end while**
16: $\boldsymbol{U} = \boldsymbol{T}\bar{\boldsymbol{K}}$

(iii). $\boldsymbol{r}_{i+1} = \boldsymbol{r}_i + \lambda_i \boldsymbol{A} \boldsymbol{p}_i$

(iv). $\bar{\boldsymbol{r}}_{i+1} = \boldsymbol{C}^{-1} \boldsymbol{r}_{i+1}$

(v). $\alpha_i = \frac{\bar{\boldsymbol{r}}_{i+1}^T \boldsymbol{r}_{i+1}}{\bar{\boldsymbol{r}}_i^T \boldsymbol{r}_i}$

(vi). $\boldsymbol{p}_{i+1} = \bar{\boldsymbol{r}}_{i+1} + \alpha_i \boldsymbol{p}_i$

In general, as the eigenvalues of the preconditioner $\boldsymbol{C}$ get closer to those of the matrix $\boldsymbol{A}$, the convergence of the PCCG method improves (Bathe, 1996).

The PCCG mixed with the *Babuška* approach is shown in Code D.1. This approach is already implemented into the INSANE computational framework, under the class *LinearEquationSyestems*. The tolerance for Babuška procedure is set equal to 1×10^{-10}. Also, the tolerance for preconditioned conjugate gradient method inside of Babuška procedure is set equal to 1×10^{-12}. The PCCG method is used first to obtain U_0 and e_0. Then, it is used inside of while loop from Algorithm D.1 to obtain the e_i.

```
1  public IVector babuskaSolverWithPCCGDiag(IMatrix k, IVector f) {
2    IMatrix ke = new IMatrix(k.getNumRow(), k.getNumCol());
3    IVector tij = new IVector(k.getNumRow());
4    tij.setAllElements(1);
5    IMatrix kn = (IMatrix) k.clone();
6    kn.sqrt();
7    tij.divideBy(kn);
8    IVector fn = new IVector(tij.getSize());
9    fn = tij.vecDotVec(f);
10   ke.normPrincDiag(tij, k);
11   kn.set(ke);
12   babuskaPerturb = 1e-10;
13   ke.addPerturbDiag(babuskaPerturb);
14   IVector u0 = preconditionedConjugateGradientsSolver(ke, fn);
15   IVector r0 = (IVector) fn.clone();
16   IVector temp1 = kn.mul(u0);
17   r0.sub(temp1);
18   IVector e0 = preconditionedConjugateGradientsSolver(ke, r0);
19
20   IVector ri = (IVector) r0.clone();
21   IVector ui = (IVector) u0.clone();
22   IVector ei = (IVector) e0.clone();
23   babuskaTol = 1e-10;
24   double var = kn.mul(e0).dot(e0) / kn.mul(u0).dot(u0);
25
26   int i = 1;
27   while (Math.abs(var) > babuskaTol) {
28     ri.sub(kn.mul(ei));
29     ei = preconditionedConjugateGradientsSolver(ke, ri);
30     ui.add(ei);
31     var = (kn.mul(ei).dot(ei) / kn.mul(ui).dot(ui));
32     i += 1;
33   }
34   x = (IVector) tij.vecDotVec(ui);
35   return x;
36 }
```

Code D.1: Code block of the *babuskaSolverWithPCCGDiag(...)* method from *LinearEquationSystems* class

References

Ahmed A, van der Meer F, Sluys L (2012) A geometrically nonlinear discontinuous solid-like shell element (dsls) for thin shell structures. Computer Methods in Applied Mechanics and Engineering 201-204:191–207 4

Alves PD (2012) Estratégia global-local aplicada ao método dos elementos finitos generalizados. Master's thesis, Universidade Federal de Minas Gerais x, 17, 22, 74, 75

Alves PD, Barros FB, Pitangueira RLS (2013) An object oriented approach to the generalized finite element method. Advances in Engineering Software 59:1–18 6, 11, 22

Andrade MP, Silva RP (2015) Implementation of numerical linear algebra libraries via native interface in the insane platform (in Portuguese). In: XXXVI Latin American Congress on Computational Methods, CILAMCE2015, Rio de Janeiro, Brazil 106

ANSYS (2015) Ansys user's manual, ansys inc. PA, USA 105

Areias PMA, Belytschko T (2005) Non-linear analysis of shells with arbitrary evolving cracks using xfem. International Journal for Numerical Methods in Engineering 62:384–415 4

Areias PMA, Song JH, Belytschko T (2006) Analysis of fracture in thin shells by overlapping paired elements. Computer Methods in Applied Mechanics and Engineering 195:5343–5360 4

Ayatollahi M, Aliha M (2009) Analysis of a new specimen for mixed mode fracture tests on brittle materials. Engineering Fracture Mechanics 76(11):1563–1573 67

Babuška I, Banerjee U (2011) Stable generalized finite element method (sgfem). Tech. rep., Technical Report ICES REPORT 11-07, The Institute for Computational Engineering and Sciences, The University of Texas at Austin, Austin, Texas, USA 3, 13

Babuška I, Banerjee U (2012) Stable generalized finite element method (SGFEM). Computer methods in applied mechanics and engineering 201-204:91–111 3, 13, 38

Babuška I, Caloz G, Osborn JE (1994) Special finite element method for a classe of second order elliptic problems whith rough coefficients. SIAM Journal on Numerical Analysis 31(4):745–981 2

Bangerth W, Hartmann R, Kanschat G (2007) deal.ii - a general-purpose object-oriented finite element library. ACM Transactions on Mathematical Software 33(4):1–27 6

Barbieri E, Meo M (2012) A fast object-oriented matlab implementation of the reproducing kernel particle method. Computational Mechanics 49(5):581–602 6

Barcellos CS, Mendonca PTR, Duarte CA (2009) A ck continuous generalized finite element formulation applied to laminated kirchhoff plate model. Computational Mechanics 44:377–393 4

Barros FB (2002) Meshless methods and generalized finite element method in structural nonlinear analysis (in Portuguese). PhD thesis, EESC - USP, Sao Carlos, SP, Brazil 109

Bathe KJ (1996) Finite element procedures. Prentice-Hall, Inc. 108, 109

Bazănt ZP (1976) Instability, ductility, and size effects in strain-softening concrete. Journal of Engineering Mechanics 102(2):331–344 10

Béchet E, Minnebo H, Mos N, Burgardt B (2005) Improved implementation and robustness study of the x-fem for stress analysis around cracks. International Journal for Numerical Methods in Engineering 64:1033–1056 3, 13, 41, 46

Belytschko T, Black T (1999) Elastic crack growth in finite elements with minimal remeshing. International Journal for Numerical Methods in Engineering 45:601–620 2, 3, 12, 51, 95

Belytschko T, Lu YY, Gu L (1993) Crack propagation by element free Galerkin methods. Advanced Computational Methods for Material Modelling - ASME 268:191–205 2

Belytschko T, Moës N, Usui S, Parimi C (2001) Arbitrary discontinuities in finite elements. International Journal for Numerical Methods in Engineering 50:993–1013 51

Biglari FR, O'Dowd KMNNP (2002) Numerical simulation of crack growth based on void volume fraction. In: Proceedings of ISME2002 Conference, Tehran, Iran, 53

Bordas S, Nguyen PV, Dunant C, Guidoum A, Nguyen-Dang H (2007) An extended finite element library. International Journal for Numerical Methods in Engineering 71:703–732 6

Bordas S, Rabczuk T, Zi G (2008) Three-dimensional crack initiation, propagation, branching and junction in non-linear materials by an extended meshfree method without asymptotic enrichment. Engineering Fracture Mechanics 75:943–960 3

de Borst R, Sluys LJ, Muhlhaus HB, Pamin J (1993) Fundamental issues in finite element analyses of localisation of deformation. Engineering Computations 10(2):99–121 1, 10

Bouchard P, F Bay YC (2003) Numerical modelling of crack propagation: automatic remeshing and comparison of different criteria. Computer Methods in Applied Mechanics and Engineering 192:3887–3908 66

Cai Y, Zhuang X, Augarde C (2010) A new partition of unity finite element free from the linear dependence problem and possessing the delta property. Computer Methods in Applied Mechanics and Engineering 199(17-20):1036–1043 12

Camacho GT, Ortiz M (1996) Computational modelling of impact damage in brittle materials. International Journal of Solids and Structures 33(20-22):2899–2938 10

Chamrová R, Patzák B (2010) Object-oriented programming and the extended finite-element method. Proceedings of the Institution of Civil Engineers - Engineering and Computational Mechanics 163:271–278 6

Chessa J, Wang H, Belytschko T (2003) On the construction of blending elements for local partition of unity enriched finite elements. International Journal for Numerical Methods in Engineering (57):1015–1038 3, 12

Cook RD, Malkus DS, Plesha ME, Witt RJ (2002) Concepts and Applications of Finite Element Analysis. John Wiley and Sons, New York, New York, USA 1, 10

Dadvand P, Rossi R, Onate E (2010) An object-oriented environment for developing finite element codes for multi-disciplinary applications. Archives of Computational Methods in Engineering 17(3):253–297 6

Davis TA (2004) A column pre-ordering strategy for the unsymmetric-pattern multifrontal method. ACM Transations on Mathematical Software 30(2):165–195 105, 106

Davis TA (2009) Umfpack version 5.4.3 user guide 105

Davis TA, Duff I (2004) Algorithm 832: UMFPACK V4.3-an unsymmetric-pattern multifrontal method. ACM Transations on Mathematical Software 30(2):196–199 105

Davis TA, Rajamanickam S, Sid-Lakhdar WM (2016) A survey of direct methods for sparse linear systems. Acta Numerica 25:383–566 105

Dolbow J, Moës N, Belytschko T (2000a) Discontinuous enrichment in finite elements with a partition of unity method. Finite Elements in Analysis and Design 36:235–260 58

Dolbow J, Moës N, Belytschko T (2000b) Modeling fracture in mindlin-reissner plates with the extended finite element method. International Journal of Solids and Structures 37:7161–7183 xii, 3, 20, 58, 69, 70, 86, 98

Dolbow JE (1999) An extended finite element method with discontinuous enrichment for applied mechanics. PhD thesis, Northwestern University 52

Duarte CA, Babuška IM (2005) A global-local approach for the construction of enrichment functions for the generalized fem and its application to propagating three-dimensional cracks. Technical report, ECCOMAS Thematic Conference on Meshless Methods, technical Report 06 14

Duarte CA, Kim DJ (2008) Analysis and applications of a generalized finite element method with global-local enrichment functions. Computer Methods in Applied Mechanics and Engineering 197:487–504 16, 34

Duarte CA, Oden JT (1995) Hp clouds - a meshless method to solve boundary-value problem. Tech. rep., TICAM, The University of Texas at Austin, technical Report 11

Duarte CA, Oden JT (1996) An h-p adaptive method using cloud. Computer Methods in Applied Mechanics and Engineering 139:237–262 2

Duarte CA, Babuška I, Oden JT (2000) Generalized finite element methods for three-dimensional structural mechanics problems. Computers & Structures 77(2):215–232 11, 12

Duarte CA, Hamzeh ON, Liszka TJ, Tworzydlo WW (2001) A generalized finite element method for the simulation of three-dimensional dynamic crack propagation. Computer Methods in Applied Mechanics and Engineering 190:2227–2262 3

Duarte CA, Kim DJ, Babuška I (2007) A global-local approach for the construction of enrichment functions for the generalized fem and its application to three-dimensional cracks. In: Leitão V, Alves C, Duarte CA (eds) Advances in Meshfree Techniques, pp 1–26 4

Dunant C (2009) Experimental and modelling study of alkali-silica-reaction in concrete. PhD thesis, École Polytechnique Fédérale de Lausanne, France 6

Dunant C, Vinh PN, Belgasmia M, Bordas S, Guidoum A (2007) Architecture tradeoffs of integrating a mesh generator to partition of unity enriched object-oriented finite element software. Revue Européenne de Mécanique Numérique 16(2):237–258 6

Erdogan F, Sih G (1963) On the crack extension in plates under plane loading and transverse shear. Transaction of ASME, Journal of Basic Engineering 85:519–525 53

Evangelista F, Roesler JR, Duarte CA (2013) Two-scale approach to predict multi-site cracking potential in 3-d structures using the generalized finite element method. International Journal of Solids and Structures 50:1991–2002 5

Faria BRL (2014) Element Free Galerkin: Integração nodal conforme e estabilizada (SCNI) com método da penalidade e dos multiplicadores de lagrange. Master's thesis, Universidade Federal de Minas Gerais 22

Fonseca F (2008) Sistema computacional para anàlise dinmica geometricamente não-linear através do método dos elementos finitos. Master's thesis, Universidade Federal de Minas Gerais 22

Fonseca FT, Pitangueira RLS (2007) An object oriented class organization for dynamic geometrically nonlinear. In: CMNE/CILAMCE (Congress on Numerical Methods in Engineering), Porto, Portugal 6, 22

Forde B, Foschi R, Stiemer S (1990) Object-oriented finite element analysis. Computers & Structures 34(3):355–374 5

Fries TP (2008) A corrected xfem approximation without problems in blending elements. International Journal for Numerical Methods in Engineering 75(5):503–532 3, 13, 48

Fries TP, Belytschko T (2006) The intrinsic xfem: a method for arbitrary discontinuities without additional unknowns. International Journal for Numerical Methods in Engineering 68(13):1358–1385 3

Fries TP, Belytschko T (2010) The extended/generalized finite element method: An overview of the method and its applications. International Journal for numerical methods in engineering 84:253–304 5, 11

Gori L, Penna SS, da Silva Pitangueira RL (2017) A computational framework for constitutive modelling. Computers & Structures 187:1–23 22

Gracie R, Wang H, Belytschko T (2008) Blending in the extended finite element method by discontinuous galerkin and assumed strain methods. International Journal for Numerical Methods in Engineering 74:1645–1669 3, 12

Gupta V (2014) Improved conditioning and accuracy of a two-scale generalized finite element method for fracture mechanics. PhD thesis, University of Illinois at Urbana-Champaign 13, 37, 41

Gupta V, Duarte CA (2016) On the enrichment zone size for optimal convergence rate of the generalized/extended finite element method. Computers & Mathematics with Applications 72(3):481–493 12

Gupta V, Duarte CA, Babuška I, Banerjee U (2013) A stable and optimally convergent generalized FEM (SGFEM) for linear elastic fracture mechanics. Computer methods in applied mechanics and engineering 266:23–39 x, 13, 14, 38

Gupta V, Duarte CA, Babuška I, Banerjee U (2015) Stable gfem (sgfem): Improved conditioning and accuracy of gfem/xfem for three-dimensional fracture mechanics. Computer methods in applied mechanics and engineering 289:355–386 13, 14, 38

Gurson AL (1997) Continuum theory of ductile rupture by void nucleation and growth: Part i. yield criteria and flow rules for porous ductile media. Journal of Engineering Materials and Technology 99(1):2–15 53

Hecht F (2012) New development in freefem++. Journal of Numerical Mathematics 20(3-4):251–265 105

Hestenes M, Stiefel E (1952) Methods of conjugate gradients for solving linear systems. Journal of Research of the National Bureau of Standards 49:409–436 108

Hirai I, Uchiyama Y, Mizuta Y, Pilkey WD (1985) An exact zooming method. Finite Elements in Analysis and Design 1:61–69 4

Hosseini S, Malekan M, Pitangueira RLS, Silva RP (2017) Imposition of dirichlet boundary conditions in element free galerkin method through an object-oriented implementation. Latin American Journal of Solids and Structures 14(6):1020–1042 22

Huang R, Sukumar N, Prevost JH (2003) Modeling quasi-static crack growth with the extended finite element method part ii: Numerical applications. International Journal of Solids and Structures 40(7539-7552) 54

Hughes TJR (2000) The Finite Element Method: Linear Static and Dynamic Finite Element Analysis. Dover, New York, USA 1

Hussain MA, Pu SL, Underwood JH (1974) Strain energy release rate for a crack under combined mode i and mode ii. Fracture Analysis, ASTM STP 560, ASTM, Philadelphia, PA pp 2–28 53

Ingraffea AR, Saouma V (1985) Numerical modelling of discrete crack propagation in reinforced and plain concrete. In: Fracture Mechanics of Concrete, Martinus Nijhoff Publishers, Dordrecht, pp 171–225 1, 10

Joseph P, Erdogan F (1991) Bending of a thin reissner plate with a through crack. Journal of Applied Mechanics 58(3):842–846 86

Khoei AR (2015) Extended Finite Element Method: Theory and Applications. John Wiley & Sons, Inc. 54

Kim DJ, Pereira JP, Duarte CA (2010) Analysis of three-dimensional fracture mechanics problems: A two-scale approach using coarse-generalized fem meshes. International Journal for Numerical Methods in Engineering 81:335–365 16, 30

Kim DJ, Duarte CA, Sobh NA (2011) Parallel simulations of three-dimensional cracks using the generalized finite element method. Computational Mechanics 47(3):265–282 3

Kim DJ, Duarte CA, , Proença SPB (2012) A generalized finite element method with global-local enrichment functions for confined plasticity problems. Computational Mechanics 50:1–16 16

Kim DJ, Hong SG, Duarte CA (2015) Generalized finite element analysis using the preconditioned conjugate gradient method. Applied Mathematical Modelling 39(19):5837–5848 108

Laborde P, Renard JPY, Salan M (2005) High-order extended finite element method for cracked domains. International Journal for Numerical Methods in Engineering 64:354–381 13, 41, 46

Lage C (1998) The application of object-oriented methods to boundary elements. Computer Methods in Applied Mechanics and Engineering 157:2015–213 6

Larsson R, Mediavilla J, Fagerstrm M (2011) Dynamic fracture modeling in shell structures based on xfem. International Journal for Numerical Methods in Engineering 86:499–527 4

Lasry J, Pommier J, Renard Y, Salaun M (2010) eXtended finite element methods for thin cracked plates with kirchhoff-love theory. International Journal for Numerical Methods in Engineering 84:1115–1138 4

Leon SE, Paulino G, Pereira A, Menezes I, Lages E (2011) A unified library of nonlinear solution schemes. Applied Mechanics Reviews 64(4):040,803(26 pages) 6

Mackie RI (2009) Advantages of object-oriented finite-element analysis. Proceedings of the Institution of Civil Engineers - Engineering and Computational Mechanics 162(1):23–29 5

Maiti SK, Smith RA (1984) Comparison of the criteria for mixed mode brittle fracture based on the preinstability stress-strain field. part ii: Pure shear and uniaxial compressive loading. International Journal of Fracture 24:5–22 53

Malekan M, Barros FB (2016) Well-conditioning global-local analysis using stable generalized/extended finite element method for linear elastic fracture mechanics. Computational Mechanics 58(5):819–831 6, 14, 22, 27

Malekan M, Barros FB (2017) Numerical analysis of a main crack interactions with micro-defects/inhomogeneities using two-scale generalized/extended finite element method. Theoretical and Applied Fracture Mechanics (under review) 95

Malekan M, Barros FB, Pitangueira RLS, Alves PD (2016a) An object-oriented class organization for global-local generalized finite element method. Latin American Journal of Solids and Structures 13(13):2529–2551 6, 22, 75

Malekan M, Barros FB, Silva RP (2016b) Development and implementation of a well-conditioning approach toward generalized/extended finite element method into an object-oriented platform. In: Proceedings of the XXXVII Iberian Latin-American Congress on Computational Methods in Engineering, ABMEC, Brasilia, DF, Brazil 22, 27

Malekan M, Barros FB, Pitangueira RL (2017a) A computational framework for crack propagation with the two-scale generalized/extended finite element method. Computers & Mathematics with Applications (submitted) pp 1–27 49, 72

Malekan M, Barros FB, Pitangueira RLS, Alves PD, Penna SS (2017b) A computational framework for a two-scale generalized/extended finite element method: generic imposition of boundary conditions. Engineering Computations 34(3):988–1019 6, 22, 75, 92

Malekan M, Barros FB, Silva LL, Pitangueira RL (2017c) Fracture modeling in two-dimensional problems using a g/xfem object-oriented implementation. In: Proceedings of the VI International Symposium on Solid Mechanics - MecSol 2017, Joinville, Brazil 49, 75

Malekan M, Silva LL, Barros FB, Pitangueira RL, Penna SS (2017d) Fracture modeling with the generalized/extended finite element method: An object-oriented programming approach. Advances in Engineering Software (under review) pp 1–32 49, 75

Melenk JM, Babuška I (1996) The partition of unity finite element method: Basic theory and applications. Computer Methods in Applied Mechanics and Engineering 39:289–314 11

Mendonca PTR, Barcellos CS, Torres DAF (2011) Analysis of anisotropic mindlin plate model by continuous and non-continuous GFEM. Finite Elements in Analysis and Design 47:698–717 4

Menk A, Bordas SPA (2011) A robust preconditioning technique for the extended finite element method. International Journal for Numerical Methods in Engineering 85:1609–1632 3, 13

Miranda A, Castro J, Martha L, Bittencourt T (2003) Fatigue life and crack path predictions in generic 2D structural components. Engineering Fracture Mechanics 70:1259–1279 54

Moës N, Dolbow J, Belytschko T (1999) A finite element method for crack growth without remeshing. International Journal for Numerical Methods in Engineering 46:131–150 3, 65

Monaghan JJ (1982) Why particle methods work. SIAM Journal on Scientific and Statistical Computing 3:422–433 2

Mostofizadeh S, Fagerstrm M, Larsson R (2013) Dynamic crack propagation in elastoplastic thin-walled structures: Modelling and validation. International Journal for Numerical Methods in Engineering 96:63–86 4

NASTRAN (2015) Msc nastran reference manual, msc software inc, USA 105

Nguyen VP (2005) An object oriented approach to the xfem with applications to fracture mechanics. Master's thesis, EMMC-Hochiminh University of Technology 95

Nguyen-Xuan H, Liu G, Bordas S, Natarajan S, Rabczuk T (2013) An adaptive singular es-fem for mechanics problems with singular field of arbitrary order. Computer Methods in applied Mechanics and Engineering 253:252–273 xiii, 84, 85

Noor AK (1986) Global-local methodologies and their application to nonlinear analysis. Finite Elements in Analysis and Design 2:333–346 4, 14

Oden JT, Duarte CA, Zienkiewicz OC (1998) A new cloud-based hp finite element method. Computer Methods in Applied Mechanics and Engineering 153:117–126 11, 12

Onate E (2013) Structural Analysis with the Finite Element Method, Linear Statics - Vol. 2: Beams, Plates and Shells, 1st edn. Springer 18

Ortiz M, Leroy Y, Needleman A (1987) A finite element method for localized failure analysis. Computer Methods in Applied Mechanics and Engineering 61(2):189–214 2

Pais MJ, Yeralan SN, Davis TA, Kim NH (2012) An exact reanalysis algorithm using incremental cholesky factorization and its application to crack growth modeling. International Journal for Numerical Methods in Engineering 91:1358–1364 105

Patzák B, Bittnar Z (2001) Design of object oriented finite element code. Advances in Engineering Software 32(10-11):759–767 6

Peixoto R, Anacleto F, Ribeiro G, Pitangueira R, Penna S (2016) A solution strategy for non-linear implicit bem formulation using a unified constitutive modelling framework. Engineering Analusis with Boundary Elements 64:295–310 22

Peixoto RG (2016) Anàlise de degradação material, bifurcação e transição entre descontinuidades fracas e fortes através do método dos elementos de contorno. PhD thesis, Universidade Federal de Minas Gerais 22

Pereira H, Pitangueira RL, Barros FB (2015) Comparison the performance of lagrange multiplier method and penalty method for imposition of essential boundary conditions in the hp-cloud meshless method. In: International Symposium on Solid Mechanics - MecSol 2015, Belo Horizonte, Brazil 22

Pereira HCR (2015) Estratégia de enriquecimento para métodos baseados na partição da unidade. Master's thesis, Universidade Federal de Minas Gerais 22

Pereira JP, Duarte CA, Jiao X, Guoy D (2009) Generalized finite element method enrichment functions for curved singularities in 3d fracture mechanics problems. Computational Mechanics 44(1):73–92 12

Pereira JPA, Kim DJ, Duarte CA (2011) A two-scale approach for the analysis of propagating three-dimensional fractures. Computational Mechanics 49(1):99–121 3, 5

Plews JA, Duarte C (2014) Bridging multiple structural scales with a generalized finite element method. International Journal for Numerical Methods in Engineering 5

Rabczuk T, Bordas S, Zi G (2010) On three-dimensional modelling of crack growth using partition of unity methods. Computers & Structures 88:1391–1411 3

Renard Y, Pommier J (2017) Getfem++ documentation 6

Rice J (1968) A path independent integral and the approximate analysis of strain concentration by notches and cracks. Journal of Applied Mechanics, Transactions ASME 35:379–386 95, 96

Rojas-Diaz R, Sukumar N, Saez A, Garcia-Sanchez F (2011) Fracture in magnetoelectroelastic materials using the extended finite element method. International Journal for Numerical Methods in Engineering 88(12):1238–1259 46

Sethian JA (1999) Level set methods and fast marching methods: evolving interfaces in computational geometry, fluid mechanics, computer vision, and material science. Cambridge University Press 50

Shibanuma K, Utsunomiya T (2009) Reformulation of xfem based on pufem for solving problem caused by blending elements. Finite Element in Analysis and Design 45(11):806–816 12

Shibanuma K, Utsunomiya T, Aihara S (2014) An explicit application of partition of unity approach to XFEM approximation for precise reproduction of a priori knowledge of solution. International Journal for Numerical Methods in Engineering 97(8):551–581 13

Silva LL (2016) Sistema gráfico interativo para análise de nucleação e propagação de trincas. Master's thesis, Universidade Federal de Minas Gerais 61

Silva RP (2012) Ańalise não-linear de estruturas de concreto por meio do método element free galerkin. PhD thesis, Universidade Federal de Minas Gerais 22

Song JH, Belytschko T (2009) Dynamic fracture of shells subjected to impulsive loads. Journal of Applied Mechanics 76(5):051,301–9 4

Sosa J H; Eischen (1986) Computation of stress intensity factors for plate bending via a path-independent integral. Engineering Fracture Mechanics 25(4):451–462 99

Steele CR, Balch CD (2009) Introduction to the theory of plates lecture note 18

Strouboulis T, Babuška I, Copps K (2000a) The design and analysis of the generalized finite element method. Computer Methods in Applied Mechanics and Engineering 181(1-3):43–69 2, 5, 38

Strouboulis T, Copps K, Babuška I (2000b) The generalized finite element method: an example of its implementation and illustration of its performance. International Journal for Numerical Methods in Engineering 47:1401–1417 6, 109

Strouboulis T, Copps K, Babuška I (2001) The generalized finite element method. Computer Methods in Applied Mechanics and Engineering 190:4081–4193 2, 5, 6

Sukumar N, Prévost J (2003) Modeling quasi-static crack growth with the extended finite element method part i: Computer implementation. International Journal of Solids and Structures 40(7513-7537) 3, 5

Swenson DV, Ingraffea AR (1988) Modeling mixed-mode dynamic crack propagation using finite elements: Theory and applications. Computational Mechanics 3(5):381–397 10

Szabo B, Babuška I (1991) Finite Element Analysis. John Wiley & Sons, Inc. 12, 31, 34, 41

Tada H, Paris PC, Irwin GR (2000) The Stress Analysis of Cracks Handbook, 3rd edn. ASME Press 78, 79

Tarancón JE, Vercher A, Giner E, Fuenmayor FJ (2009) Enhanced blending elements for xfem applied to linear elastic fracture mechanics. International Journal for Numerical Methods in Engineering 77:126–148 3, 12

Tian R (2013) Extra-dof-free and linearly independent enrichments in gfem. Computer Methods in Applied Mechanics and Engineering 266:1–22 12

W K Liu SJ, Zhang FY (1995) Reproducing kernel particle methods. International Journal for Numerical Methods in Engineering 20:1081–1106 2

Wolff KP (2010) Implementação computacional de um modelo de fissuração para o concreto baseado no método dos elementos finitos estendido (xfem). Master's thesis, Universidade Federal de Minas Gerais 22

Wu JY, Li FB (2015) An improved stable xfem (is-xfem) with a novel enrichment function for the computational modeling of cohesive cracks. Computer Methods in Applied Mechanics and Engineering 295:77–107 12

Xu D, Liu Z, Liu X, Zeng Q, Zhuang Z (2014) Modeling of dynamic crack branching by enhanced extended finite element method. Computational Mechanics 54(2):489–502 3

Yu CD, Wang W, Pierce D (2011) A CPU-GPU hybrid approach for the unsymmetric multifrontal method. Parallel Computing 37:759–770 105, 106

Zeng Q, Liu Z, Xu D, Wang H, Zhuang Z (2016) Modeling arbitrary crack propagation in coupled shell/solid structures with x-fem. International Journal for Numerical Methods in Engineering 106:1018–1040 4

Zheng AX, Luo XQ (2014) Numerical study of quasi-static crack growth problems based on extended finite element method. Journal of Shanghai Jiaotong University 19(6):736–746 105

Zheng Q, Liu Z, Xu D, Zhuang Z (2014) Modeling stationary and moving cracks in shells by x-fem with cb shell elements. Computational Mechanics 57(7):1276?284 4

Zheng Q, Liu Z, Xu D, Wang H, Zhuang Z (2015) Modeling arbitrary crack propagation in coupled shell/solid structures with x-fem. International Journal for Numerical Methods in Engineering 4

Zhuang Z, Cheng BB (2011) A novel enriched cb shell element method for simulating arbitrary crack growth in pipes. Science China Technological Sciences 54:1520–1531 4

Zimmermann T, Dubois-Pelerin Y, Bomme P (1992) Object-oriented finite element programming: I. governing principles. Computer methods in applied mechanics and engineering 98:291–303 5