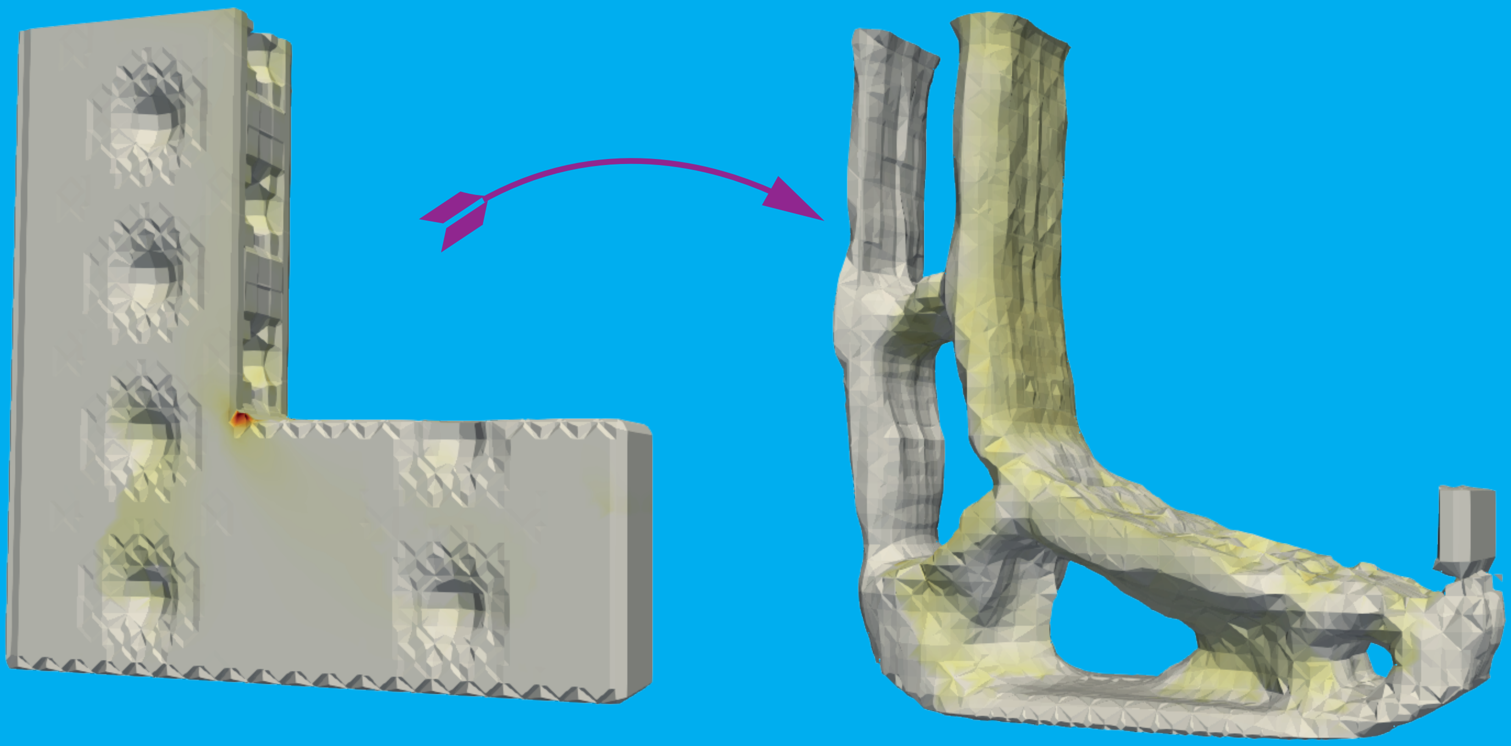
An Interface-enriched Topology Optimization for Mitigating the Effect of Surface Flaws in 3D Brittle Structures

Shangru Liu



TUDelft

An Interface-enriched Topology Optimization for Mitigating the Effect of Surface Flaws in 3D Brittle Structures

by

Shangru Liu

to obtain the degree of Master of Science in Offshore & Dredging Engineering
at the Delft University of Technology,
to be defended publicly on October 18, 2022.

Student number:	5249732	
Project duration:	August 3, 2021 – October 18, 2022	
Thesis committee:	Dr. C. L. Walters,	TU Delft, chair
	Dr. A. M. Aragón,	TU Delft, supervisor
	Dr. L. F. P. Noël,	TU Dleft
	Ir. Y. Yan,	TU Delft

Acknowledgements

The two years I spent in Delft were by no means an easy journey. I would like to give my deepest acknowledgements to everyone who helped and guided me through the time.

First, I would like to express my deepest gratitude to Dr. Alejandro Aragón. You are the person that I have spent most of my time with in TU Delft. I started to take your course Advanced FEM in the third quarter of my first year and then Enriched FEM, and finally, this graduation project on fracture-based topology optimization. It's you that offer me the opportunity to study FEM, topology optimization, and fracture mechanics systematically. I never imagined I could achieve this far before I met you.

I would also like to express my gratitude to Dr. Carey Walters. I am fond of your kindness in offering lectures and answering questions (and your efficiency in answering emails ;p). Alejandro is the person that guided me through the field of FEM, but you are the person that opened the door for me. I enjoyed the time programming ANSYS in your Ultimate Strength class!

Ir. Yuheng Yan, I would like to thank you not only for being my committee member but also for being a good friend of mine. You have helped me with my coding issues since the Advanced FEM, and also helped a lot me in understanding Hybrida.

I am also grateful to Dr. Lise Noël. Although I spent very little time with you, and you might not even know me before my defense, I actually joined your Topology Optimization course and applied for your PhD position on cardiac tissue modeling. Therefore, even though I haven't talked to you directly, I benefited greatly from your information.

I would also like to thank my best friends, Tzu-yao, Xinmiao, and Iting. You gave me lots of accompany during my difficult times. Especially Tzu-yao, more than a fellow, you are almost my teacher in Python and maths.

Finally, I need to express my deepest gratitude to my parents. You raise me up, to walk on stormy seas.

Shangru Liu
Utrecht, October 2022

Abstract

In this 3D fracture-based topology optimization framework, the author extended the 2D framework on tailoring fracture resistance for brittle materials (Zhang et al., 2022) to 3D. In the optimization, the topology is described by radial basis functions interpolated level set function, and the problem is solved by the Interface-enriched Generalized Finite Element Method (IGFEM). Cracks are assumed to exist on enriched nodes that are added on the boundary of the geometry to increase the accuracy of approximation. The first part of the work assumes cracks to be semi-circular with the crack plane perpendicular to the boundary, and the crack opening direction to be either parallel to the XZ- or YZ-plane of the global coordinates. An extended framework that assumes crack opening direction perpendicular to the surface first principal stress is also developed at the end. The energy release rates (ERRs) of the cracks are evaluated with the topological derivative method, which requires only a stress field of the geometry and weight functions that relate the stress and stress intensity factors (SIFs). The weight functions are found by a finite element analysis on a cuboid with a crack. This approach is computationally efficient because it eliminates the need of actually modeling and meshing the crack planes in the geometry during optimization. Moreover, a 3D stress recovery technique (or stress improvement procedure, SIP) is used to recover the nodal stress non-locally to improve the accuracy. The objective function is then established with the p-mean aggregation of the ERRs. Finally, Numerical examples in 3D, including the famous L-bracket benchmark problem, are performed to prove the correctness and capacity of the framework. In conclusion, this extended framework shows more flexibility and provides more information to the optimized design than the 2D framework by considering an added dimension both in analyzed geometry and crack shape, i.e., the effect of the anisotropy of cracks can be captured.

Keywords: topology optimization · 3D · level set · interface-enriched generalized finite element method · energy release rate · topological derivative

Contents

1

Introduction

Flaws in structures may reduce their reliability and may also lead to catastrophic scenarios in brittle structures-sudden brittle failure. A well-known example in the early 1940s is the failures of Liberty ships, where brittle fractures were spotted on 1500 out of 2700 ships, and 12 of them were completely broken in half. The primary reason for failures is a decrease in fracture toughness of the steel under low temperature. Other than that, design flaws such as sharp corners, and flaws generated due to welding, were spotted as origins of cracks (Zhang, 2016). Fracture-induced failures are still a concern even in recent decades. Epp (2022) found that around 60% of 275 failure cases of bridges and buildings that happened 1990-2019 are fracture-related. Therefore, enhancing reliability is also essential for modern structures, besides chasing for larger sizes and lighter weights.

Both experimental and numerical methods are applied in structural development to enhance their reliability. To give an example, nondestructive testing and examination analysis is often used to detect material defects to anticipate fatigue life (Vukelić & Vizentin, 2017). However, these experiments require test samples and human manipulation, which are costly and time-consuming. To avoid this, numerical methods, especially the finite element method (FEM), are used to simulate structural behaviors. Various parameters, such as displacement, stress, and stress-intensity factors, are employed to check the reliability of designs. Though effective as they are, engineers have to check through multiple criteria and iterate their designs, which is still inefficient. Therefore, despite verifying designs numerically, it would be wise to also use numerical methods, such as topology optimization, to generate the initial design. Failure-related parameters, including stress, damage, and fracture, can be included in the topology optimization frameworks to increase the failure resistance of structures.

By embedding stress in topology optimization frameworks, it is possible to modify the structure to control the stress limit. Yang and Chen (1996) were the first to perform peak von Mises stress minimization in three-dimensional (3D) problems with density-based topology optimization. To include more variety in the approach, Giraldo-Londoño and Paulino (2020) introduced a unified yield function so that multiple criteria, including von Mises, Drucker-Prager, Tresca, Mohr Coulomb, Bresler-Pister, and Willan-Warnke can be added to topology optimization formulation. Other than that, limiting the first principal stress with the bi-directional evolutionary structural optimization method (BESO) in 3D is also possible (Chen et al., 2021). One of the obstacles to the development of 3D stress-based topology optimization is the high computational cost because stress is a local quantity that exists throughout the computational domain (Yang & Chen, 1996). Therefore, Yang and Chen (1996) applied aggregation functions Kreisselmeier-Steinhauser (KS) (Kreisselmeier & Steinhauser, 1980) and p-norm (Duysinx & Sigmund, 1998) to aggregate all stress values to a single term in the topology optimization formulation. Other than these methods, P-mean aggregation, which aggregates and average the total value by the number of data points, is also available (Duysinx & Sigmund, 1998; Holmberg et al., 2013). However, since these methods ”globalize” the stress and the maximum stress becomes dominant, they become ineffective in capturing local quantities. To improve on this issue, Wang and Qian (2018) proposed to aggregate Heaviside functions, which have value 0 or 1 determined by whether the material has failed. On the contrary, Senhora et al. (2020) proved that it is possible in 3D topology optimization to keep the locality of stress and evaluate them in the objective function by means of an augmented Lagrangian approach instead of aggregation. Other than working on the formulations, Wang et al.

(2022) recently adopted a multi-resolution topology optimization scheme for 3D stress minimization, which decouples the mesh and topology description model to decrease computational cost. Jeong et al. (2013) provided an algorithm with perturbed structural displacement to calculate sensitivity to reduce computational costs by not accessing the internal finite element information.

Damage-based topology optimization started with the work of Bendsøe and Kikuchi (1988), which restricts the amount of damage in optimization with a continuum damage model. However, few 3D optimizations have been carried out ever since. Amir and Sigmund (2013) performed 3D truss topology optimization on concrete by modelling it with a gradient-enhanced strain-softening continuum damage model. Chen et al. (2020) considered a fatigue damage constraint based on Palmgren/Miner linear damage accumulation hypothesis.

Unlike stress and damage criteria, fracture criteria consider a joint effect of stress, flaw size, and material fracture toughness. They are, therefore, more comprehensive in considering the existence of flaws within the target structures (Anderson, 2017). This leads to the advantage of fracture-based topology optimization-it can optimize structures while allowing cracks to nucleate and exist on the structures. There are three major strategies to define the cracks in the context of fracture-based topology optimization, namely, stationary and usually pre-defined cracks, evolving cracks, and distributing cracks throughout the geometry.

In the first strategy, cracks are usually placed on critical regions with stress concentration. Kang et al. (2017) placed a crack directly into the mesh in a non-design region. They minimized an objective function composed of the energy release rate (ERR) of the crack evaluated by the J-integral method (Rice, 1968) and the compliance of the structure with the solid isotropic material with penalization (SIMP) method. Klarbring et al. (2018) applied this idea to 3D adhesively bonded patches used for repairing fractured structures to minimize the sum of ERRs of cracks. Both works used standard FEM, which requires a fine adaptive mesh around the crack. As an improvement, Hu et al. (2019) used XFEM and defined crack with a level set function to reduce the complexity of modelling and meshing. Adding to this strategy, da Silva et al. (2022) proposed to include a breaking region in the design, which would fail first under an overload condition, but the structure should not fail. However, they provided their framework with stress criteria instead of fracture. Besides traditional gradient-based topology optimization methods, Gu et al. (2016) and Gu et al. (2017) modified the strain energy of cracked structures with a greedy algorithm to avoid complex math in sensitivity analysis and geometry updates. Peridynamics-based topology optimization was used by Habibian et al. (2021), Kefal et al. (2019), Lahe Motlagh and Kefal (2021), and Sohouli et al. (2020) to optimize cracked structures for ease of including cracks in the peridynamics model. In general, the above methods are effective for reducing the possibility of failure due to damage at particular regions. However, they hardly include nature of crack nucleation and propagation, and cannot enhance fracture resistance over the entire structure.

In the second strategy, the nature of fracturing processes is modelled, including crack initiation, propagation, and complete failure of structures. In the context of brittle elastic materials, the phase-field method (Francfort & Marigo, 1998; Miehe et al., 2010) is frequently used for modeling cracks. Xia et al. (2018) maximized the mechanical work that a quasi-brittle composite structure (2D and 3D) can undertake during fracturing processes by modifying the distribution of inclusion material with BESO. This framework only takes the bulk brittle fracture into account, so they soon extended their framework by taking into account the interfacial damage, which is the major cause of microcracks nucleation and propagation (Da et al., 2018). They then further extended their framework by adding constraints over unit cells periodically, so to optimize periodic quasi-brittle composites (Da & Yvonnet, 2020). Despite using BESO, Russ and Waisman (2019) constrained fracture surface energy by combing SIMP with phase-field. Li et al. (2021) also showed the possibility of changing their original framework (Da et al., 2018) to SIMP-based topology optimization. Different from Russ and Waisman (2019), who optimized single phase material, Li et al. (2021) optimized composites, and 3D designs. They found out that the final outputs of SIMP and BESO are close, but SIMP can provide faster convergence for homogeneous design (Li et al., 2021). To further diverse their frameworks and attain a final result with clear boundaries, Wu et al. (2020) changed the framework of Da et al. (2018) to level set-based topology optimization, and developed a new topological derivative that considers time-dependent gradient information about displacement when updating the geometry. The same group then modified this framework by adding time-dependent phase-field variables in the topological derivative so to improve the accuracy of sensitivity (Wu et al., 2021). Besides, a 3D optimization on a dental bridge was provided. Desai et al. (2022)

also worked on a level set-phase field framework, but added a constraint on compliance, and solve the optimization problem with an adjoint Lagrangian equation. A major contribution is that they applied PETSc scientific calculation tool (Balay et al., 2019) to parallelise their program to solve large-scale 3D problems. However, even after parallelisation, they found the computational time of solving phase-field method to be long.

Both strategies above can only allow cracks to nucleate and propagate from a critical region, which neglects the possibility of cracks nucleating from elsewhere in the structure. However, the computational cost will be huge if applied the idea with the above methods. For example, modeling cracks everywhere in the mesh require fine mesh throughout the domain, which will significantly increase the solving time. Besides, J-integral has to be performed multiple times. Challis et al. (2008) avoided this issue by considering crack propagation as a change in the energy-density of the boundary. They applied the virtual crack extension technique on every boundary node to simulate crack propagation inward to the geometry, but ERR defined under linear elastic fracture mechanics context was altered.

To solve the computational burden of meshing and evaluating J-integral, Silva et al. (2011) developed a topological derivative method, which can evaluate the ERR of a crack with the stress field of the target structure, the weight functions relate the stress field and stress intensity factors (SIFs), and crack parameters. The same group later applied this method to 2D shape optimization with a constraint on maximum allowable ERRs on the shape boundary aggregated with both p-norm and KS functions, respectively (Alidoost et al., 2022). As a more comprehensive framework, Zhang et al. (2022) applied the topological derivative method on topology optimization. They used a level set function interpolated by the radial basis functions (RBFs) (Wendland, 1995) to describe the topology. The level set-based interface-enriched Generalized Finite Element Method (IGFEM) (Soghrati et al., 2012) was used to solve the structural response to avoid remeshing during optimization and achieve an accurate displacement field in the entire computational domain. Cracks were placed on all enriched nodes on the solid-void interface (geometry boundary) and perpendicular to the interface. Then, p-mean aggregation was used to collect all ERRs into the objective function. Moreover, a non-local stress recovery technique for 2D (or stress improvement procedure, SIP) is used to attain a more accurate stress field than directly calculated stress from FEA (Payen & Bathe, 2012). Finally, the method of moving asymptotes (MMA) is used to update the design variables. With the topological derivative method, the computational cost of tailoring surface ERRs can be reduced roughly to the same magnitude of surface stress optimization. However, this comparably computationally inexpensive framework has not yet been applied to 3D problems.

This work extends the work of Zhang et al. (2022) to 3D topology optimization to further reveal the capacity of the framework. We follow the same procedures in topology description, FEA, and geometry update. Similar to Zhang et al. (2022), we assume cracks nucleate perpendicularly to the solid-void interface, but are half-penny-shaped (semi-circular). Besides, in the first part of the work (subsection 2.1.3-section 2.2), crack opening directions are parallel to the XZ- or YZ-plane of the global coordinates for the sack of keeping acceptable simplicity and computational cost in the first stage of the project. An extended method that assumes the crack openings to be perpendicular to the surface first principal stress is developed in section 2.3. ERRs are evaluated at the deepest point of each crack by a 3D topological derivative method (Alidoost et al., 2020) and aggregated with a p-mean function. The weight functions used by the topological derivative method are evaluated with FEA on a cuboid with a crack and stored as offline numerical functions. A 3D stress recovery technique (Sharma et al., 2018) is used to evaluate the stress field. The framework is verified by finite difference analysis. A shape optimization on a cube under triaxial load is tested under both initial and extended frameworks. An optimal ellipsoidal and spherical hole with a uniformly distributed ERR field are found respectively. The topology optimization of the 3D L-bracket benchmark problem with a sharp right angle corner with high stress and ERR concentration was performed. The concentration was smoothed out by replacing the right angle corner with a fillet.

2

An Interface-enriched Topology Optimization for Mitigating the Effect of Surface Flaws in 3D Brittle Structures

This chapter gives the major content of the thesis. First, a description of the static problem that has to be solved during optimization and implementation of the Interface-enriched Generalized Finite Element Method (IGFEM) are given. Then, formulation and numerical examples of topology optimization framework with crack openings parallel to the XZ- and YZ-planes are given. This formulation is extended by allowing crack openings perpendicular to the surface first principal stress in the end. Finally, a summary and conclusions to the work are provided.

2.1. Formulation

2.1.1. Static problem description

Define a computational domain $\Omega \subset \mathbb{R}^3$, which consists of solid domain ω_s and void domain Ω_v, such that $\Omega = \Omega_s \cup \Omega_v$ and $\Omega_s \cap \Omega_v = \emptyset$. The domain has closure $\overline{\Omega} = \overline{\Omega}_s \cup \overline{\Omega}_v$. The Dirichlet boundary condition is applied on the boundary of the computational domain $\partial\Omega \equiv \Gamma = \overline{\Omega} \setminus \Omega$ at region Γ^u with displacement $\bar{\boldsymbol{u}}$. The Neumann boundary condition is applied on the boundary of the solid domain $\partial\Omega_s \equiv \Gamma_s = \overline{\Omega}_s \setminus \Omega_s$ at region Γ^t with load $\bar{\boldsymbol{t}}$. The solid domain consists of a linear elastic material with Young's modulus E_1 and Poisson's ratio ν_1. The Young's modulus of the void material is $E_2 \ll E_1$. The Young's modulus of the void material should be non-zero to avoid singular stiffness matrix, but is set to a small value to minimize its effect on the problem.

In each iteration of optimization, a static problem in the following general form has to be solved to find a displacement field $\boldsymbol{u}$ that satisfies the equilibrium equation,

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

and boundary conditions,

$$\begin{cases} \boldsymbol{u} = \bar{\boldsymbol{u}} & \text{on } \Gamma^u, \\ \boldsymbol{\sigma} \cdot \boldsymbol{n} = \bar{\boldsymbol{t}} & \text{on } \Gamma^t, \end{cases} \tag{2.2}$$

where $\nabla\cdot$, $\boldsymbol{\sigma}$, $\boldsymbol{b}$, $\boldsymbol{n}$ are the divergence operator, stress tensor, body force, and outward normal vector of the solid boundary Γ_s, respectively.

This problem can also be written in a weak form: Find $\boldsymbol{u}$ in an admissible displacement space $\boldsymbol{U}$ such that:

$$\mathscr{B}(\boldsymbol{u}, \boldsymbol{w}) = \mathscr{L}(\boldsymbol{w}), \quad \forall \boldsymbol{w} \in \boldsymbol{W}, \tag{2.3}$$

where $\boldsymbol{W}$ is a test function space that satisfies the Dirichlet boundary condition. According to the virtual work principle in elasticity, the bilinear and linear functions in the above equation are equivalent to

$$\mathscr{B}(\boldsymbol{u}, \boldsymbol{w}) = \int_\Omega \boldsymbol{\epsilon}(\boldsymbol{w}) : \boldsymbol{\sigma}(\boldsymbol{u}) \, \mathrm{d}\Omega, \tag{2.4}$$

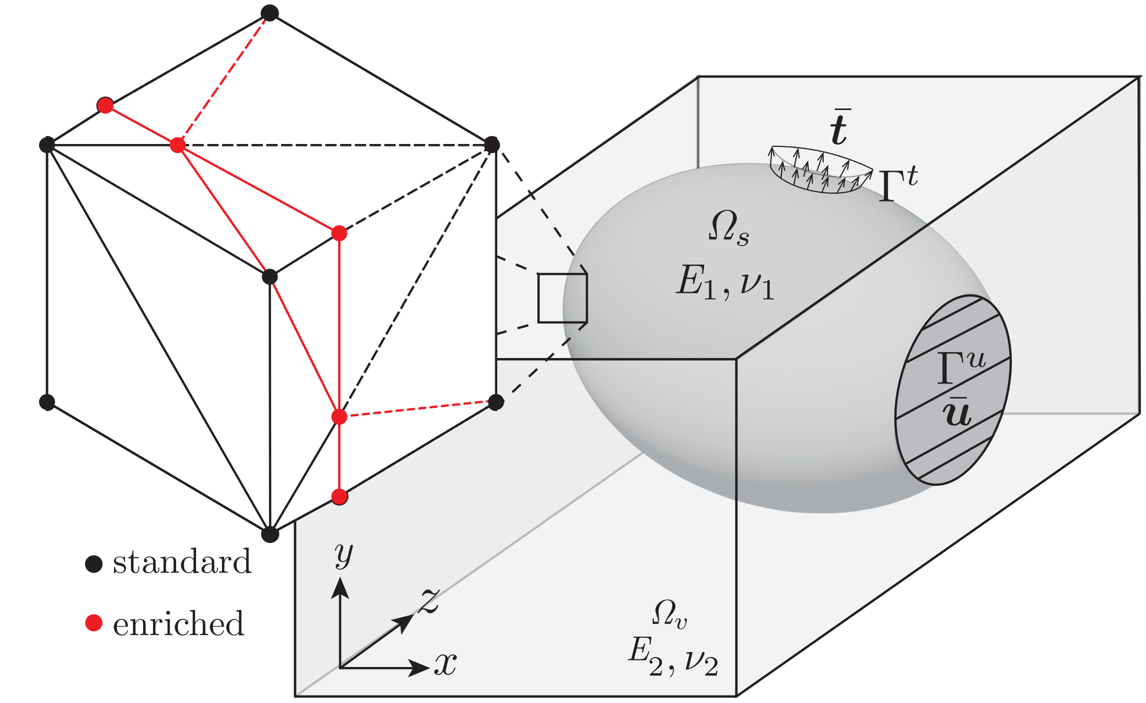


Figure 2.1: Computational domain Ω consists of solid domain Ω_s and void material domain Ω_v. Dirichlet and Neumann boundary conditions are applied on Γ^u and Γ^t, respectively. The insertion shows a set of triangular elements that form a cube and are cut by the material boundary (red solid lines). The solid and dashed lines indicate the element edges outside and inside the Ω_s. The black lines and dots show original elements; the red lines and dots show the newly created edges of enriched elements and enriched nodes, respectively.

and

$$\mathscr{L}(\boldsymbol{w}) = \int_{\Omega} \boldsymbol{w} \cdot \boldsymbol{b} \, \mathrm{d}\Omega + \int_{\Gamma^t} \boldsymbol{w} \cdot \bar{\boldsymbol{t}}\mathrm{A}, \tag{2.5}$$

respectively, where $\boldsymbol{\epsilon}(\boldsymbol{w}) = \frac{1}{2}(\nabla \boldsymbol{w} + \nabla \boldsymbol{u}^{\intercal})$ is the small linearized strain tensor, and $\boldsymbol{\sigma}(\boldsymbol{u}) = \boldsymbol{D} : \boldsymbol{\epsilon}(\boldsymbol{u})$ according to the generalized Hooke's law, where $\boldsymbol{D}$ is the constitutive tensor.

The domain Ω is then discretized as $\Omega^h = \cup_{j \in \iota_0} e_j$ with finite elements, where e_j is the j^{th} element and ι_0 is the set of all original elements. Writing Equation 2.3 in discretized form:

$$\sum_j \int_{e_j} \boldsymbol{\varepsilon}(\boldsymbol{w}^h) : \boldsymbol{D} : \boldsymbol{\varepsilon}(\boldsymbol{u}^h) \, \mathrm{d}\Omega = \sum_j \int_{e_j} \boldsymbol{w}^h \cdot \boldsymbol{b} \, \mathrm{d}\Omega + \sum_j \int_{\partial e_j \cap \Gamma^t} \boldsymbol{w}^h \cdot \bar{\boldsymbol{t}} \, \mathrm{d}\Gamma, \quad \forall \boldsymbol{w} \in \boldsymbol{W}, \tag{2.6}$$

where $\boldsymbol{u}^h \in \boldsymbol{U}^h$ is the trial function and $\boldsymbol{w}^h \in \boldsymbol{W}^h$ is the test function.

As shown in Figure 2.1, the Interface-Generalized Finite Element Method (IGFEM) does not necessarily require a matching mesh. Enriched nodes are created at the intersections of element edges and material interfaces. The cut elements are split into integration (enriched) elements for numerical integration. The displacement field $\boldsymbol{u}^h$ is approximated by a function with a standard FEM term and enrichment term:

$$\boldsymbol{u}^h = \underbrace{\sum_{i \in \iota_h} N_i(\boldsymbol{x}) \boldsymbol{U}_i}_{\text{standard FEM}} + \underbrace{\sum_{i \in \iota_w} \psi_i(\boldsymbol{x}) \boldsymbol{\alpha}_i}_{\text{enrichment}}, \tag{2.7}$$

where ι_h is the index set of all original mesh nodes in Ω^h, N_i is the Lagrangian shape function and $\boldsymbol{U}_i$ is the degrees of freedoms (DOFs) of the i^{th} mesh node; ι_w is the index set of all enriched nodes, ψ_i is the enrichment function associated with the enriched DOFs $\boldsymbol{\alpha}_i$.

An isoparametric procedure is followed to find the local stiffness matrix $\boldsymbol{k}_e$ and force vector $\boldsymbol{f}_e$ of the integration elements:

$$\boldsymbol{k}_e = \int_e \boldsymbol{B}^{\mathrm{T}}\boldsymbol{D}\boldsymbol{B} j_e \mathrm{d}\boldsymbol{\xi}, \tag{2.8}$$

and

$$\boldsymbol{f}_e = \int_e \begin{bmatrix}\boldsymbol{N}\\ \boldsymbol{\psi}\end{bmatrix} \boldsymbol{b} j_e \mathrm{d}\boldsymbol{\xi} + \int_{\partial e \cap \Gamma^t} \begin{bmatrix}\boldsymbol{N}\\ \boldsymbol{\psi}\end{bmatrix} \bar{\boldsymbol{t}} j_e \mathrm{d}\partial\boldsymbol{\xi}, \tag{2.9}$$

where $\boldsymbol{B} = [\Delta_\xi^{\mathrm{T}}\boldsymbol{N}^{\mathrm{T}}\boldsymbol{J}^{-T} \quad \Delta_\xi^{\mathrm{T}}\boldsymbol{\psi}^{\mathrm{T}}\boldsymbol{J}_e^{-T}]$ is the strain-displacement matrix, $\boldsymbol{\xi} = (\xi_1, \xi_2, \xi_3)$ is the natural coordinates, and $\boldsymbol{N}$ and $\boldsymbol{\psi}$ are the vectors of the standard and enrichment shape functions stack of the element, respectively, j_e is the determinant of Jacobian. $\boldsymbol{J}$ and $\boldsymbol{J}_e$ are the Jacobian of the isoparametric mapping for the parent and integration elements. The differential operator Δ_ξ is defined by

$$\Delta_\xi \equiv \begin{bmatrix} \frac{\partial}{\partial\xi_1} & 0 & 0 & \frac{\partial}{\partial\xi_2} & 0 & \frac{\partial}{\partial\xi_3} \\ 0 & \frac{\partial}{\partial\xi_2} & 0 & \frac{\partial}{\partial\xi_1} & \frac{\partial}{\partial\xi_3} & 0 \\ 0 & 0 & \frac{\partial}{\partial\xi_3} & 0 & \frac{\partial}{\partial\xi_2} & \frac{\partial}{\partial\xi_1}. \end{bmatrix}^{\mathrm{T}} \tag{2.10}$$

The shape functions and procedures of finding $\boldsymbol{k}_e$ and $\boldsymbol{f}_e$ of non-cut elements are the same as in standard FEM. The global stiffness matrix $\boldsymbol{K}$ and force vector $\boldsymbol{F}$ of all elements could therefore be given:

$$\boldsymbol{K} = \mathop{\mathbb{A}}_{j} \boldsymbol{k}_j, \quad \boldsymbol{F} = \mathop{\mathbb{A}}_{j} \boldsymbol{f}_j, \tag{2.11}$$

where $\mathbb{A}$ represents the finite element assembly operator.

One advantage of IGFEM over XFEM is that it keeps the physical meaning of DOFs on the original mesh nodes as standard FEM (Soghrati et al., 2012). This enables prescribing the Dirichlet boundary conditions on enriched nodes by solving a local problem (van den Boom et al., 2019)(Zhang et al., 2022). Figure 2.2 shows a scenario that occurs in this work, an element e which has Dirichlet boundary condition applied on the shaded surface circumference by node $\boldsymbol{x}_1$, $\boldsymbol{x}_2$ and $\boldsymbol{x}_3$, but split by a material interface marked in red. Enriched nodes $\boldsymbol{x}_5$, $\boldsymbol{x}_6$ and $\boldsymbol{x}_7$ are created. In order to apply the displacement $\bar{\boldsymbol{u}}$, a local problem is solved:

$$\begin{cases} \boldsymbol{\alpha}_5 = \bar{\boldsymbol{u}}(\boldsymbol{x}_5) - N_1(\boldsymbol{x}_5)\boldsymbol{U}_1 - N_2(\boldsymbol{x}_5)\boldsymbol{U}_2, \\ \boldsymbol{\alpha}_6 = \bar{\boldsymbol{u}}(\boldsymbol{x}_6) - N_2(\boldsymbol{x}_6)\boldsymbol{U}_2 - N_3(\boldsymbol{x}_6)\boldsymbol{U}_3, \end{cases} \tag{2.12}$$

where $\boldsymbol{U}_1 = \bar{\boldsymbol{u}}(\boldsymbol{x}_1)$ and $\boldsymbol{U}_2 = \bar{\boldsymbol{u}}(\boldsymbol{x}_2)$.

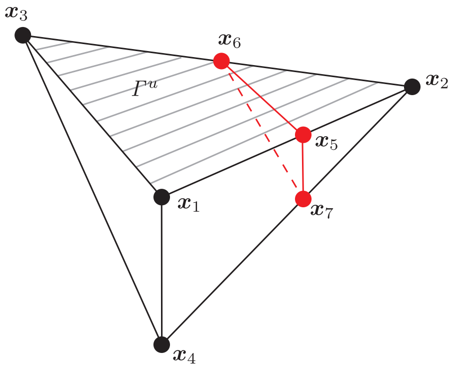


Figure 2.2: An element cut by material interface and forms enriched nodes $\boldsymbol{x}_5$, $\boldsymbol{x}_6$, and $\boldsymbol{x}_7$. Dirichlet boundary condition is applied on Γ^u which is on the element surface enclosed by $\boldsymbol{x}_1$, $\boldsymbol{x}_2$, and $\boldsymbol{x}_3$.

As mentioned in Equation 2.9, tractions on the boundary Γ^t can be assembled into $\boldsymbol{f}$ with the same elementwise procedure as in standard FEM. The scenario shown in Figure 2.3 is the case that appears in this work, where the Neumann boundary condition is applied on the material interface within the cut element. A 2D triangular surface element connecting $\boldsymbol{x}_5$, $\boldsymbol{x}_6$ and $\boldsymbol{x}_7$ is created to compute the traction numerically. However, the dimension of the surface element does not match with the space $\mathbb{R}^3$ so the inverse and determinant of its Jacobian matrix are not available. To cope with this problem, we employ Devloo (1997)'s method to reform the Jacobian matrix, which has the form:

$$\boldsymbol{J}_e = \begin{bmatrix} J_\xi & \boldsymbol{V}_1 \cdot \tilde{\boldsymbol{V}}_2 J_\eta \\ 0 & \boldsymbol{V}_2 \cdot \tilde{\boldsymbol{V}}_2 J_\eta \end{bmatrix}, \tag{2.13}$$

where J_ξ, J_η, $\boldsymbol{V}_1$, $\boldsymbol{V}_2$ and $\tilde{\boldsymbol{V}}_2$ are coefficients and vectors used to approximate the Jacobian matrix.

Detailed expression, explanation, and derivative of Devloo's Jacobian can be found in Appendix E. For more detailed explanations on IGFEM, please reference to Soghrati et al. (2012) and Aragón et al. (2020).

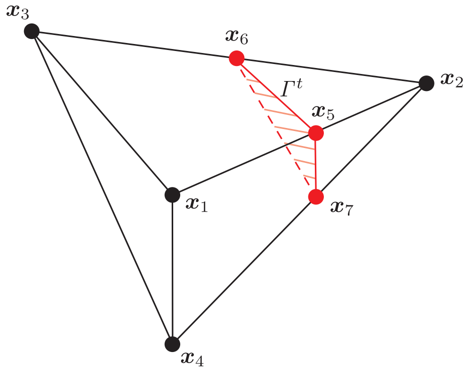


Figure 2.3: A 2D triangular integration element enclosed by $\boldsymbol{x}_5$, $\boldsymbol{x}_6$ and $\boldsymbol{x}_7$ is created at the material interface (red shaded area). Neumann boundary condition is applied on Γ^t which is also the red shaded area.

2.1.2. Topology description

Level set function, first introduced by Sethian and Wiegmann (2000), is used to express the geometry boundary Γ, which has the implicit form

$$\begin{cases} \phi(\boldsymbol{x}) = 0, & \text{if } \boldsymbol{x} \in \Gamma_s, \\ \phi(\boldsymbol{x}) < 0, & \text{if } \boldsymbol{x} \in \Omega_s, \\ \phi(\boldsymbol{x}) > 0, & \text{if } \boldsymbol{x} \in \Omega_v. \end{cases} \tag{2.14}$$

For example, the position of the enriched nodes $\boldsymbol{x}_n$ can be defined by the linear interpolation of the positions ($\boldsymbol{x}_l$ and $\boldsymbol{x}_m$) and level-set values ($\phi(\boldsymbol{x}_l)$ and $\phi(\boldsymbol{x}_m)$) of the neighboring original nodes on the same edge as

$$\boldsymbol{x}_n = \boldsymbol{x}_l - \frac{\phi_l}{\phi_m - \phi_l}(\boldsymbol{x}_m - \boldsymbol{x}_l), \tag{2.15}$$

where $\phi_{l,m} = \phi(\boldsymbol{x}_{l,m})$. The relationship is illustrated in Figure 2.4.

In this work, compactly supported radial basis functions (RBFs) are used to interpolate the level set function $\phi(\boldsymbol{x})$:

$$\phi(\boldsymbol{x}) = \sum_{i=1}^{N_r} \vartheta_i(\boldsymbol{x}) s_i = \boldsymbol{\Theta}^\intercal \boldsymbol{s}, \tag{2.16}$$

where N_r is the total number of RBFs, s_i is the design variable, and $\boldsymbol{s} = [s_1 \;\; s_2 \;\; \dots \;\; s_{N_r}]$ is the stack of all design variables. The compactly supported RBFs ϑ_i is defined by

$$\vartheta_i(\boldsymbol{x}) = max(0, 1 - r_i)^4 (4r_i + 1), \tag{2.17}$$

where radius r_i is expressed as

$$r_i(\boldsymbol{x}, \boldsymbol{x}_i) = \frac{||\boldsymbol{x} - \boldsymbol{x}_i||}{r_s}, \tag{2.18}$$

and r_s is the radius of the support. $\boldsymbol{\Theta} = [\vartheta_1(\boldsymbol{x}) \;\; \vartheta_2(\boldsymbol{x}) \;\; \dots \;\; \vartheta_i(\boldsymbol{x})]$ is the stack of RBFs.

The compactly supported RBFs is chosen for three reasons (Wang & Wang, 2006):

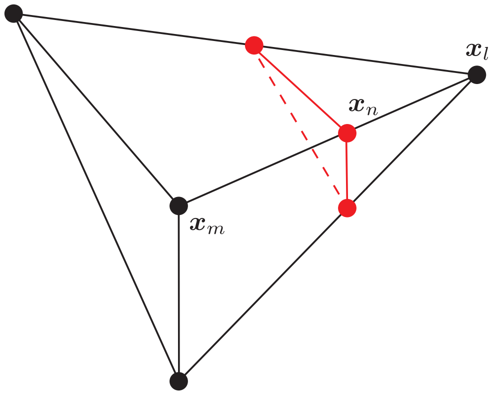


Figure 2.4: The coordinate of $\boldsymbol{x}_n$ can be expressed with the coordinates and level-set values of $\boldsymbol{x}_l$ and $\boldsymbol{x}_m$

1. The dimension of the design space and the mesh size can be defined separately because the RBF grid and finite element mesh are decoupled. This also helps to avoid the FEM approximation being affected by the design space.
2. The RBFs are smooth continuous functions and, therefore, can provide a smooth level set function and act as a filter of the design domain.
3. The influence of each design variable can be extended across several elements by giving proper support radius r_i. This helps the optimizer to modify the level set values more and therefore helps increase the convergence rate.

2.1.3. Topology optimization formulation

The following topology optimization problem is structured:

$$\begin{aligned} minimize \quad & J(\mathbf{s}) \\ subject\ to \quad & \boldsymbol{KU} = \boldsymbol{F}, \\ & V_s \leq V_c, \end{aligned} \tag{2.19}$$

where $J(\mathbf{s})$ is the objective function , $\mathbf{s}$ is design variable vectors, V_s is the volume of the geometry, V_c is the maximum allowable volume. The objective function is a p-mean aggregation function of the energy release rates:

$$J = (\frac{1}{N}\sum_{i=1}^{N} G_i^p)^{\frac{1}{p}}, \tag{2.20}$$

where N is the number of cracks, which equals the number of enriched nodes on the tensioned boundaries; G_i is the energy release rate of the crack on the i^{th} enriched node; p is a constant integer.

Evaluation of energy release rate

Alidoost et al. (2020) provide an expression of evaluating energy release rate G of half-penny-shaped cracks with the topological derivative method:

$$G = \pi\epsilon(\frac{\bar{K}_I^2 + \bar{K}_{II}^2}{\bar{E}} + \frac{\bar{K}_{III}^2}{2\nu}), \tag{2.21}$$

where ϵ is the crack length, $\bar{E} = E/(1-\nu^2)$, E is the Young's modulus. $\bar{K}_{I,II,III}$ are normalized stress intensity factors of modes I, II and III evaluated with

$$\begin{bmatrix} \bar{K}_I \\ \bar{K}_{II} \\ \bar{K}_{III} \end{bmatrix} = \underbrace{\begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}}_{\boldsymbol{H}} \begin{bmatrix} \sigma_{\theta\theta} \\ \sigma_{r\theta} \\ \sigma_{z\theta} \end{bmatrix}, \tag{2.22}$$

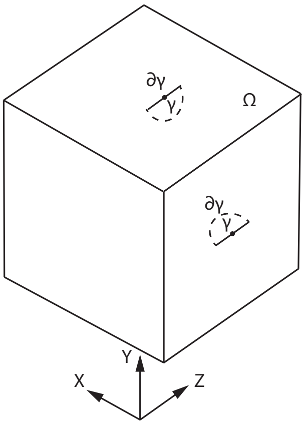


Figure 2.5: Domain (Ω) with Cracks (γ). Most cracks are assumed to be parallel to the XZ-plane, but cracks on a surface that parallel to the XZ-plane are assumed to be parallel to the YZ-plane.

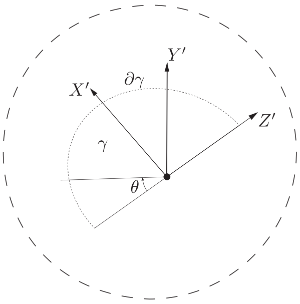


Figure 2.6: Local coordinate system defined according to crack plane γ

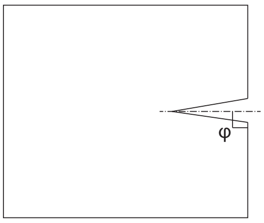


Figure 2.7: Cross-sectional view of a crack. The crack plane is assumed to be perpendicular to the geometry surface

where $\boldsymbol{H}$ is a matrix of weight functions. The weight functions are evaluated in a FEA with domain integral and M_1 integral as the method given by Nagai et al. (2013). Plots of all weight functions h_{11}-h_{33} are shown in Figure 2.8; detailed calculation processes are given in Appendix F.

The above formulations could further be modified to enable linear algebric operations:

$$G = \frac{\pi\epsilon}{2\mu\bar{E}}\boldsymbol{K'}^{\mathsf{T}}\boldsymbol{K'}, \tag{2.23}$$

where

$$\boldsymbol{K'} = \begin{bmatrix} \sqrt{2\mu} & 0 & 0 \\ 0 & \sqrt{2\mu} & 0 \\ 0 & 0 & \sqrt{\bar{E}} \end{bmatrix} \begin{bmatrix} \bar{K_I} \\ \bar{K_{II}} \\ \bar{K_{III}} \end{bmatrix} = \underbrace{\begin{bmatrix} \sqrt{2\mu}h_{11} & \sqrt{2\mu}h_{12} & \sqrt{2\mu}h_{13} \\ \sqrt{2\mu}h_{21} & \sqrt{2\mu}h_{22} & \sqrt{2\mu}h_{23} \\ \sqrt{\bar{E}}h_{31} & \sqrt{\bar{E}}h_{32} & \sqrt{\bar{E}}h_{33} \end{bmatrix}}_{\boldsymbol{H'}} \begin{bmatrix} \sigma_{\theta\theta} \\ \sigma_{r\theta} \\ \sigma_{z\theta} \end{bmatrix}. \tag{2.24}$$

The stress could be expressed in global coordinates by adding a transformation matrix:

$$\begin{bmatrix} \sigma_{\theta\theta} \\ \sigma_{r\theta} \\ \sigma_{z\theta} \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\boldsymbol{M}_1} \boldsymbol{R} \underbrace{\begin{bmatrix} \sigma_{xx} & \sigma_{xy} & \sigma_{xz} \\ \sigma_{yx} & \sigma_{yy} & \sigma_{yz} \\ \sigma_{zx} & \sigma_{zy} & \sigma_{zz} \end{bmatrix}}_{\boldsymbol{\sigma}} \boldsymbol{R}^{\mathsf{T}} \underbrace{\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}}_{\boldsymbol{M}_2}, \tag{2.25}$$

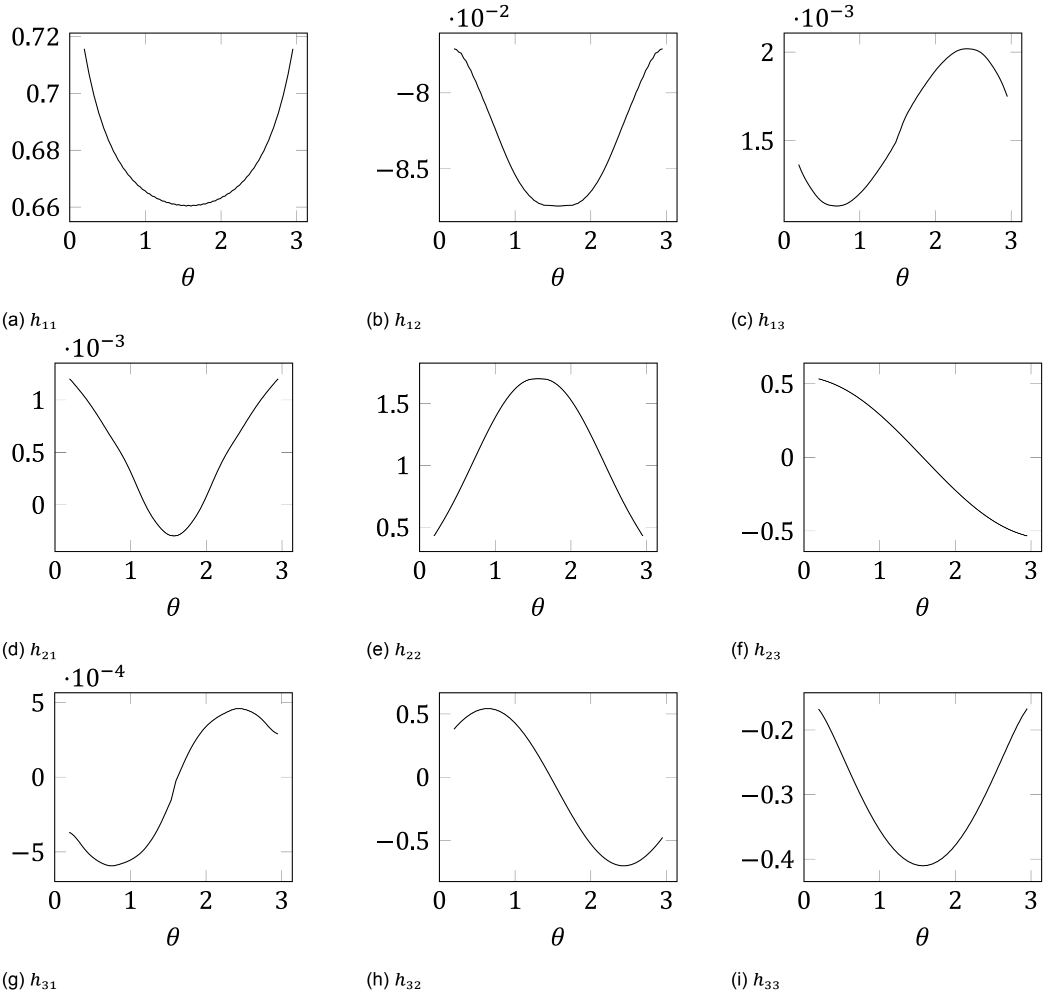
where stress transformation $\boldsymbol{R}$ is given in Appendix A. To keep acceptable complexity in analytical expressions and computational cost, crack opening direction (direction of Z' axis in Figure 2.6) is assumed to be either parallel to the $XZ-$plane or $YZ-$plane, $\boldsymbol{R} = \boldsymbol{R}_{xz}$ if the crack opening direction is defined parallel to the $XZ-$plane, $\boldsymbol{R} = \boldsymbol{R}_{yz}$ if the crack opening direction is defined parallel to the $YZ-$plane. Besides, as illustrated in Figure 2.7, all the cracks are assumed to initiate perpendicular to the surfaces of the geometry ($\varphi = 90°$).

According to Alidoost et al. (2020), for crack size less than 5% of the geometry, the error of the ERR calculated by the topological derivative method is less than 3% for crack front position $\frac{5\pi}{18} \leq \theta \leq \frac{13\pi}{18}$ and less than 8% for $\theta \leq \frac{2\pi}{9}$ and $\theta \geq \frac{7\pi}{9}$ than the boundary element method.

After introducing the linear algebraic formulation, equation (2.21) can be changed to

$$G = \frac{\pi\epsilon}{2\mu\bar{E}}\boldsymbol{M}_2^{\mathsf{T}}\boldsymbol{R}\,\boldsymbol{\sigma}^{\mathsf{T}}\boldsymbol{R}^{\mathsf{T}}\boldsymbol{M}_1^{\mathsf{T}}\boldsymbol{H'}^{\mathsf{T}}\boldsymbol{H'}\boldsymbol{M}_1\boldsymbol{R}\,\boldsymbol{\sigma}\boldsymbol{R}^{\mathsf{T}}\boldsymbol{M}_2. \tag{2.26}$$

This expression can be used in the 2D framework by replacing $\boldsymbol{R}$, $\boldsymbol{\sigma}$ and $\boldsymbol{H'}$ with the corresponding expression in 2D. The expression will then be equivalent to the ERR evaluation expression given by Zhang et al. (2022).



(a) h_{11}

(b) h_{12}

(c) h_{13}

(d) h_{21}

(e) h_{22}

(f) h_{23}

(g) h_{31}

(h) h_{32}

(i) h_{33}

Figure 2.8: Plot of weight functions in $\boldsymbol{H}$

Note that we follow the hypothesis that cracks extend to the direction that is perpendicular to the maximum tension direction in ideal brittle materials (Erdogan & Sih, 1963). Therefore, only the cracks with positive mode-I SIF K_I are considered in the calculation of energy release rates and aggregated into Equation 2.20.

Evaluation of Stress

A stress recovery technique (or stress improvement procedure (SIP)) is used in this work to smoothen the stress field and obtain faster convergence than directly calculated stress from FEA (Payen & Bathe, 2012; Sharma et al., 2018; Zhang et al., 2022). The 2D SIP formulation was first derived by Payen and Bathe (2012) based on the Hu-Washizu principle (Chan, 1968), and Sharma et al. (2018) extended this method to 3D.

To determine the recovered stress in each element, a patch of neighbouring elements is needed as the calculation domain ε. The stress is determined by first satisfying equilibrium in a weak sense in the domain and then projecting the directly calculated stress from finite element analysis results to the domain. Figure 2.9a gives an illustration of the process. A patch of elements in darker grey is used to determine the recovered stress of the red element.

The recovered stress of elements is determined by $\boldsymbol{\sigma}_e = \boldsymbol{E}_\sigma \hat{\boldsymbol{\sigma}}$, where $\hat{\boldsymbol{\sigma}}$ is a 60×1 stress coefficient vector determined by

$$\left(\sum_{\{e\in\epsilon\}}\begin{bmatrix}\int_e \bar{\boldsymbol{E}}_\sigma^{\mathrm{T}}\boldsymbol{E}_\sigma \mathrm{d}e \\ \int_e \boldsymbol{E}_\zeta^{\mathrm{T}}\boldsymbol{\partial}_\sigma \boldsymbol{E}_\sigma \mathrm{d}e\end{bmatrix}\right)\hat{\boldsymbol{\sigma}} = \left(\sum_{\{e\in\epsilon\}}\begin{bmatrix}\int_e \bar{\boldsymbol{E}}_\sigma^{\mathrm{T}}\boldsymbol{\sigma}_e^h \mathrm{d}e \\ -\int_e \boldsymbol{E}_\zeta^{\mathrm{T}}\boldsymbol{b}\mathrm{d}e\end{bmatrix}\right), \tag{2.27}$$

where $\bar{\boldsymbol{E}}_\sigma$, $\boldsymbol{E}_\sigma$, $\boldsymbol{E}_\zeta$ are interpolation matrices, $\boldsymbol{\sigma}_e^h$ is the directly calculated stress of element e, $\boldsymbol{\partial}_\sigma$ is a differential operator. Coefficients in Equation 2.27 are given in Appendix B.

As illustrated in Figure 2.9b, to determine the nodal stress $\boldsymbol{\sigma}(\boldsymbol{x}_i)$, a patch of elements connected to the node is needed as the calculation domain ε_i. Nodal stress $\boldsymbol{\sigma}(\boldsymbol{x}_i)$ can then be recovered by

$$\boldsymbol{\sigma}(x_i) = \frac{\sum_{e\in\varepsilon_i} \boldsymbol{\sigma}_e(\boldsymbol{x}_i)}{N_i}, \tag{2.28}$$

where $\boldsymbol{\sigma}_e(\boldsymbol{x}_i) = \boldsymbol{E}_\sigma(\boldsymbol{x}_i)\hat{\boldsymbol{\sigma}}_e$ is the elemental stress of element e determined at node i, and N_i is the number of elements in the patch.

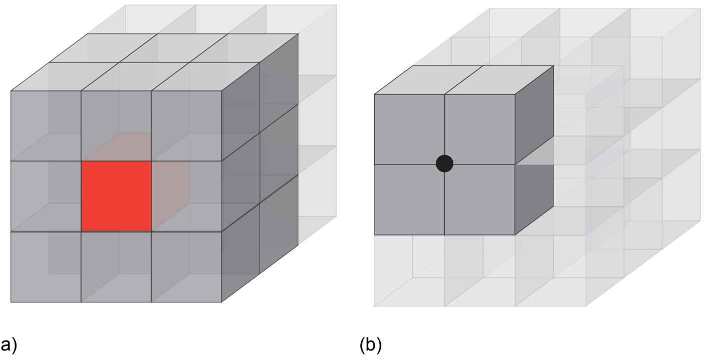


Figure 2.9: (a):target element (red) and its element patch (gray) to calculate elemental stress; (b): surface node (black) and its element patch to recover nodal stress

Sensitivity

An adjoint Lagrangian function $L = J + \boldsymbol{\lambda}^\intercal(\boldsymbol{KU} - \boldsymbol{F})$ is established with the adjoint vector $\boldsymbol{\lambda}$. The sensitivity of the objective function is then expressed as the derivative of L with respect to the j^{th} design variable s_j:

$$\begin{aligned}\frac{dL}{ds_j} &= \frac{\partial J}{\partial s_j} + \frac{\partial J}{\partial \boldsymbol{U}}\frac{\partial \boldsymbol{U}}{\partial s_j} + \boldsymbol{\lambda}^\intercal(\frac{\partial(\boldsymbol{KU})}{\partial s_j} - \frac{\partial \boldsymbol{F}}{\partial s_j}) \\ &= \frac{\partial J}{\partial s_j} + (\frac{\partial J}{\partial \boldsymbol{U}} + \boldsymbol{\lambda}^\intercal\boldsymbol{K})\frac{\partial \boldsymbol{U}}{\partial s_j} + \boldsymbol{\lambda}^\intercal(\frac{\partial \boldsymbol{K}}{\partial s_j}\boldsymbol{U} - \frac{\boldsymbol{F}}{\partial s_j}).\end{aligned} \tag{2.29}$$

The adjoint equation below has to be solved to determine the adjoint vector $\boldsymbol{\lambda}$:

$$\frac{\partial J}{\partial \boldsymbol{U}} + \boldsymbol{\lambda}^\intercal\boldsymbol{K} = 0. \tag{2.30}$$

The detailed processes of deriving $\frac{\partial J}{\partial s_j}$, $\frac{\partial J}{\partial \boldsymbol{U}}$, $\frac{\partial \boldsymbol{K}}{\partial s_j}$ and $\frac{\partial \boldsymbol{F}}{\partial s_j}$ are given in Appendix C.

2.2. Numerical examples

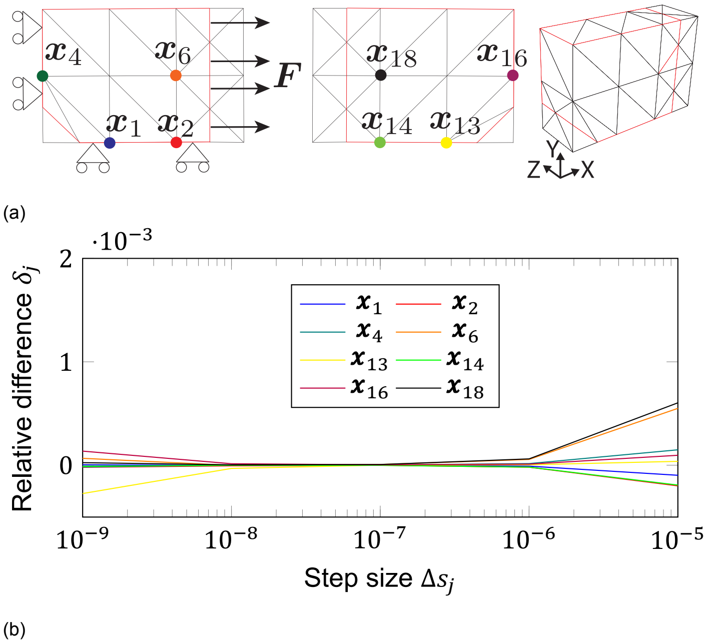
One verification example and two optimization examples are carried out in this section. No units are specified in this work, so any consistent unit system can be applied. The crack length should be less than 1% of the domain size to keep the validity of the topological derivative method, so all cracks are set to 0.01 in the examples. Power factor $p = 8$ is used in all the objective functions. No hole nucleation method is used during the optimization, so holes are placed into the initial design.

2.2.1. Sensitivity verification

The sensitivity of the topology optimization formulation given in section subsection 2.1.3 is verified with finite difference analysis (van den Boom et al., 2021). The analytical expression of sensitivity $\frac{\mathrm{d}L}{\mathrm{d}s_j}$ is compared with the numerical calculated finite difference derivative J_j', which is determined as the relative differences:

$$\delta_j = \frac{J'_j - \frac{\mathrm{d}L}{\mathrm{d}s_j}}{\frac{\mathrm{d}L}{\mathrm{d}s_j}}. \tag{2.31}$$

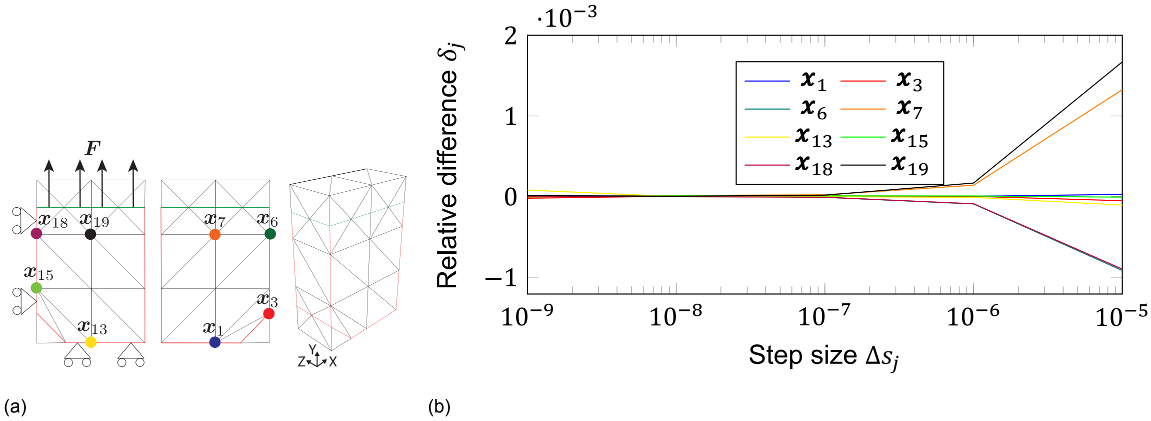
Figure 2.10-Figure 2.15 show the geometry used for sensitivity verification and plots of the results. The geometry and mesh are shown in (a), where the solid part is marked with red and green lines. The surfaces with crack openings parallel to the XZ-plane are marked in red, while the surfaces with crack openings parallel to the YZ-plane are marked in green. The Young's modulus of the solid and void materials are $E_s = 1$ and $E_v = 10^{-6}$. A distributed load $F = 1$ is applied. The relative differences with respect to different finite difference step size Δs_j are shown in (b). The relative differences reach their minimum values $\delta_{j,min} \approx 8 \times 10^{-6} - 7 \times 10^{-5}$ at step size around 10^{-7}, which proves the correctness of the analytical sensitivity.



(a)

(b)

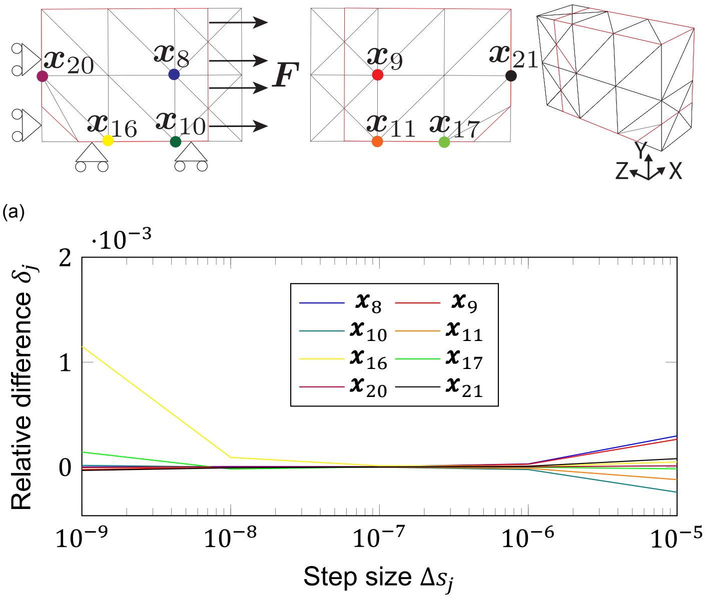
Figure 2.10: (a): Geometry uses for sensitivity verification in X-direction, where the solid part is marked in red. All crack openings are parallel to the XZ-plane. (b): Relative difference δ_j with respect to step size Δs_j on different design variables s_j.



(a)

(b)

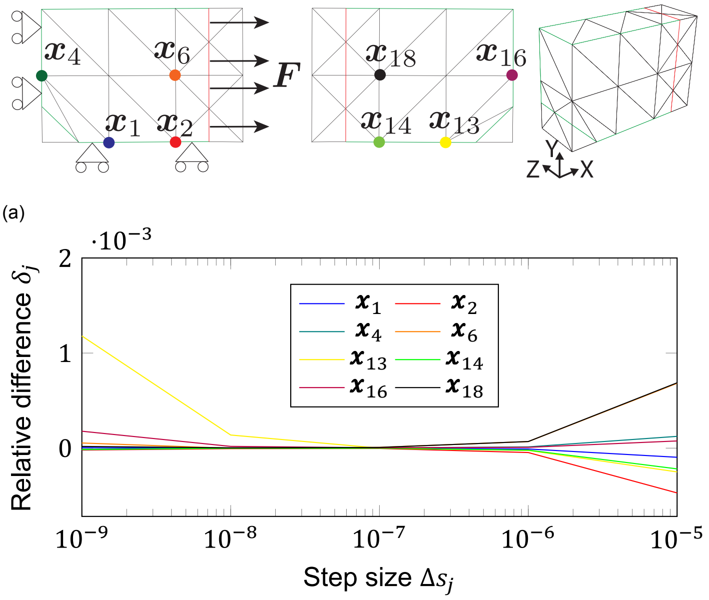
Figure 2.11: (a): Geometry uses for sensitivity verification in Y-direction, where the solid part is marked in red, including surface with crack openings parallel to the XZ-plane. Top surface with crack openings parallel to the YZ-plane is marked in green. (b): Relative difference δ_j with respect to step size Δs_j on different design variables s_j.



(a)

(b)

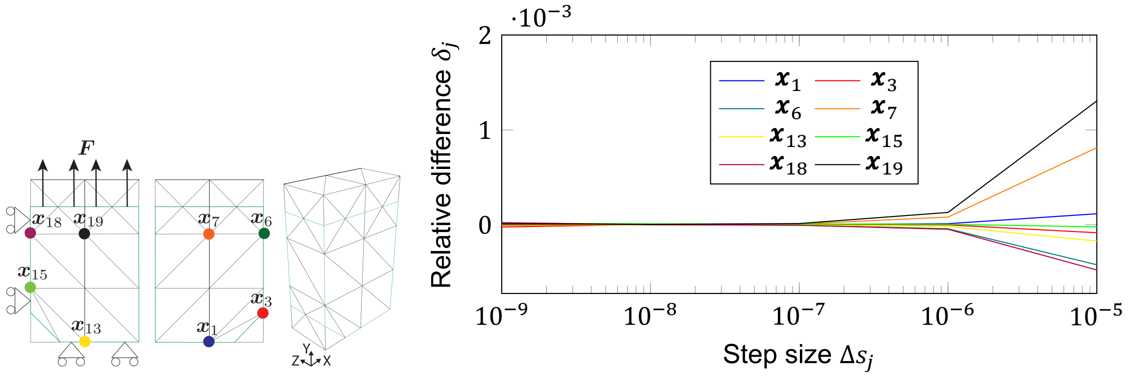
Figure 2.12: (a):Geometry uses for sensitivity verification in Z-direction, where the solid part is marked in red. All crack openings are parallel to the XZ-plane. (b): Relative difference δ_j with respect to step size Δs_j on different design variables s_j.



(a)

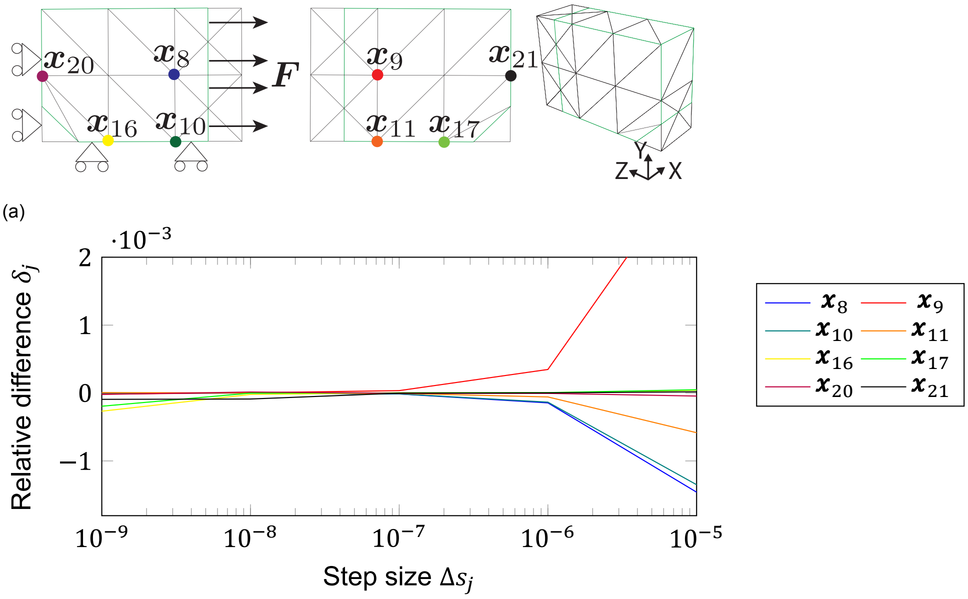
(b)

Figure 2.13: (a):Geometry uses for sensitivity verification in X-direction, where the solid part is marked in green, including surface with crack openings parallel to the YZ-plane. Right surface with crack openings parallel to the XZ-plane is marked in red. (b): Relative difference δ_j with respect to step size Δs_j on different design variables s_j.



(a) (b)

Figure 2.14: (a): Geometry uses for sensitivity verification in Y-direction, where the solid part is marked in green. (b): Relative difference δ_j with respect to step size Δs_j on different design variables s_j.



(a)

(b)

Figure 2.15: (a): Geometry uses for sensitivity verification in Z-direction, where the solid part is marked in green. (b): Relative difference δ_j with respect to step size Δs_j on different design variables s_j

2.2.2. Cube with an ellipsoid

Cube with an ellipsoid is a triaxially tensioned shape optimization example. It is a $2 \times 2 \times 2$ cube loaded with with unit tractions $\boldsymbol{t}_1$, $\boldsymbol{t}_2$ and $\boldsymbol{t}_3$ on the surfaces as shown in Figure 2.16. As illustrated in Figure 2.17, only $1/8$ of the geometry is used by taking the advantage of symmetry conditions. The geometry is immersed into a background computational domain of $1.1 \times 1.1 \times 1.1$ discretized by $20 \times 20 \times 20 \times 6$ tetrahedral elements. The volume constraint of solid material is $V_c = 0.94$; 140 iterations are used.

The solid part of the initial and final design and their energy release rate distributions of are shown in Figure 2.18. The initial design has a tetrahedral hole in the middle, where the energy release rates concentrate on the three corners and three edges of the hole. Due to the anisotropy of the assumption of crack-opening directions, the bottom edge has higher energy release rates than the two other edges. Figure 2.19 shows the convergence of objective function and volume constraint. The designs' cross-section views at iterations 20, 60, and 120 are also shown. The optimization converges after around 120 iterations. The energy release rates over the material interface are smoothed out during

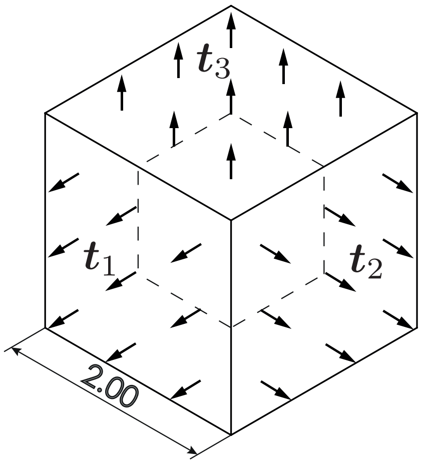


Figure 2.16: Isometric view of cube under triaxial tension load

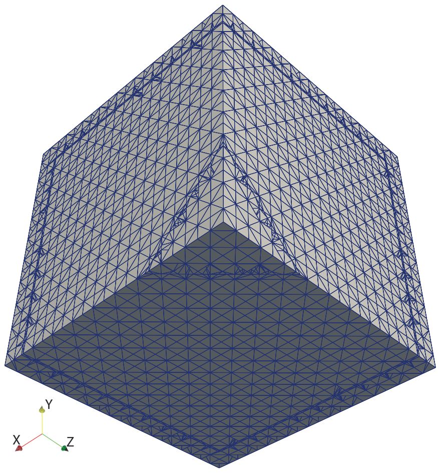


Figure 2.17: 1/8 of the full geometry immersed in the computational domain

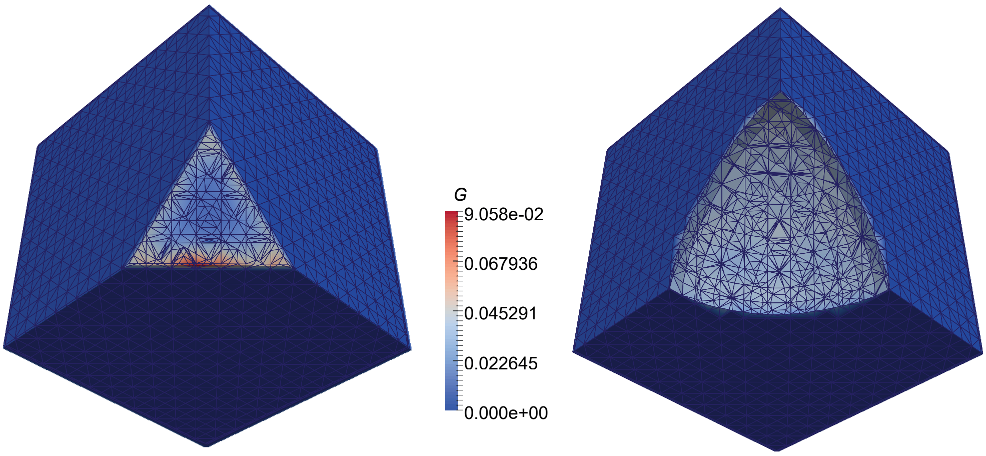


Figure 2.18: (a) Initial design with a tetrahedral hole, where concentration of energy release rates is spotted on three corners and three edges; (b) Final design with 1/8 of a cube with an ellipsoid, where the energy release rates distribute uniformly over the material interface

the optimization and the design comes to an end with $1/8$ of an ellipsoidal hole with a larger radius in Y-direction (hole radius: $r_x = 0.5011$, $r_y = 0.5429$, $r_z = 0.4998$). As a common engineering sense, a spherical hole has the best mechanical property in undertaking triaxial load. The hole converged to an ellipsoid instead because the anisotropic assumption results in the hole having higher ERRs in the lower region (the region closer to the XZ-plane) when the radius of the hole equals in three directions. To smoothen the ERR field, the algorithm shrinks the radius in the lower part and therefore results in an ellipsoid. This is a drawback of the current framework. However, it also shows that the anisotropy of cracks can affect the optimization, which means a potentially different design is needed to have the best fracture resistance property than optimizing the stress distribution.

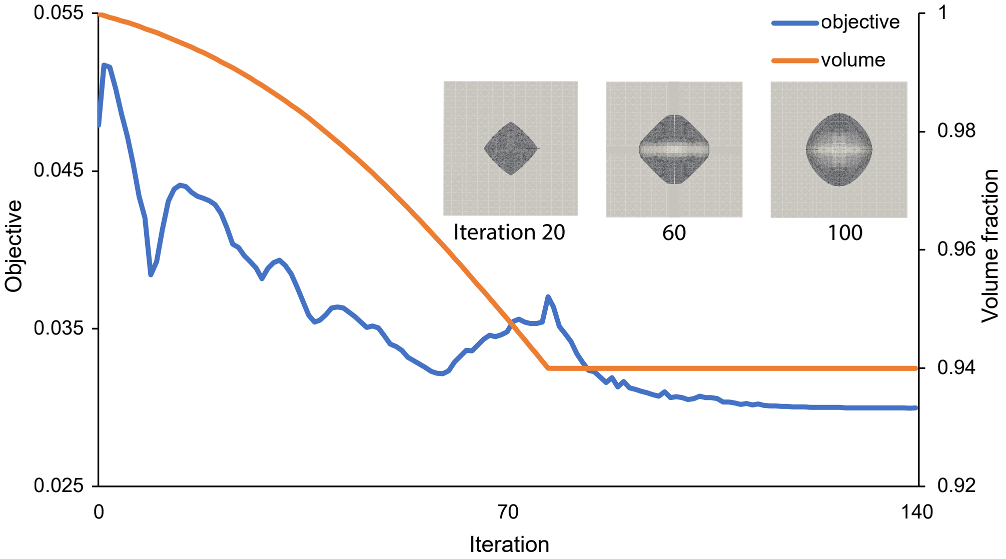


Figure 2.19: The objective function converges at about 120 iterations. Designs at iteration 20, 60, and 120 are shown

2.2.3. L-bracket

A 3D L-bracket is investigated to examine the capacity of the method on large-scale problems. The dimensions of the L-bracket are given in figure Figure 2.20. The top surface of the L-bracket is fixed, and a vertical downward load $\boldsymbol{F} = -2.6$ is applied at the red shaded region. Only half of the structure was modelled by making use of the symmetry condition. The structure is immersed into a background mesh of $40 \times 40 \times 8 \times 6$ as shown in Figure 2.21. Several holes are pre-seeded in the structure because the no hole-seeding algorithm was applied in this work.

The initial design and optimized design are shown in Figure 2.22, and the optimization process is shown in Figure 2.23. In the initial design, a high ERR concentration is spotted in the sharp corner of the structure. The framework smoothed out the corner and replaced it with a fillet. Moreover, a smooth surface transition is achieved throughout the final design. The framework reduced the volume fraction to the set value 0.65 and the value of the objective function to 0.0581. The optimization approximates convergence after around 150th iteration. However, oscillations still exist, which means the final convergence is not reached. Smaller move limit and more iterations are needed to stabilize the process and achieve a final convergence. The surface of the final design is not smooth, which indicates a larger radius of the RBFs and a smaller mesh size is needed to improve the filtering effect of RBFs.

Huge computational demand is required in this example. The author ran the computation on a processor with Intel Xeon CPUs (Gold 6150, 2.7 GHz) and the computational time for 200 iterations is 221 hours. The long computational time is because all the parameters are calculated node- and element-wise with for loops in Python. Vectorization for this example is problematic because support for multi-dimensional sparse matrix in Python is limited and the amount of data is too large to store in a Numpy array. Hence, the time taken to write and read the data is significant. A practical solution is to use the multi-processing, for example, with Python parallel computation libraries such as PETSc and MPI4Py.

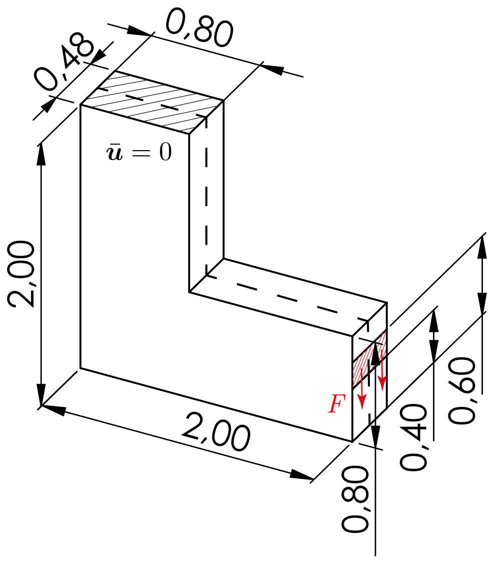
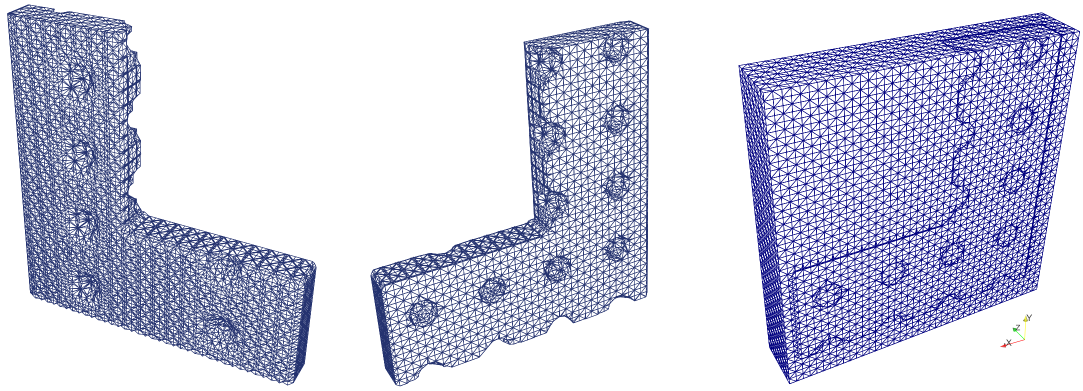


Figure 2.20: 3D L-bracket fixed on the top surface and with a load $\boldsymbol{F}$ applied on the right end



(a) View from front-right

(b) Cross sectional view

(c) Immersed view

Figure 2.21: 3D L-bracket views from different angles

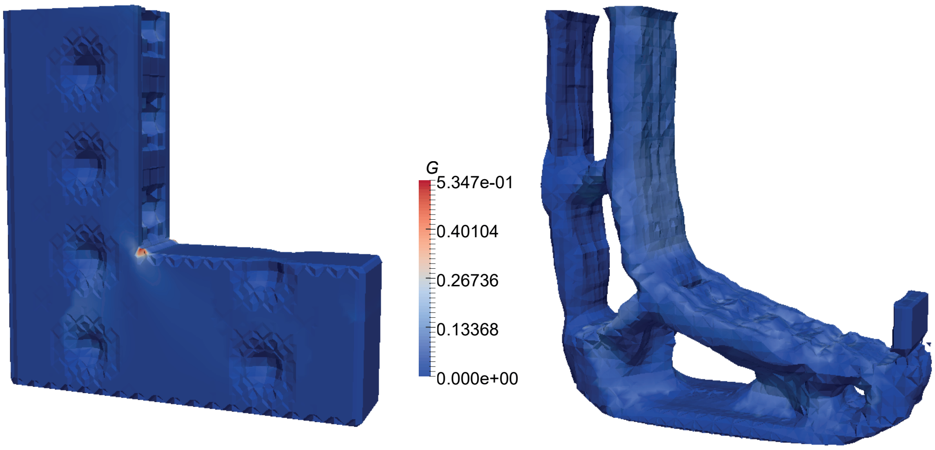


Figure 2.22: Initial design (left) has high ERR concentration at the sharp corner while the final design (right) smoothed it out

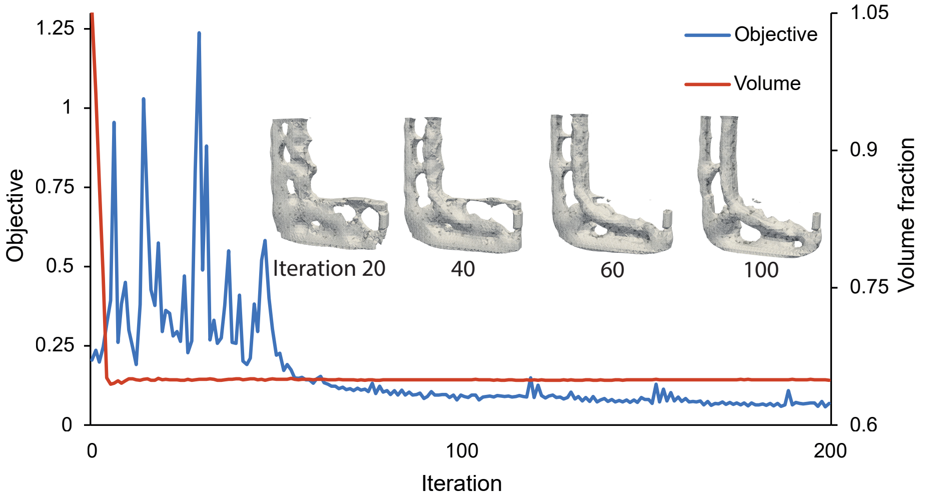


Figure 2.23: Optimization process of L-bracket. Designs of the 20th, 40th, 60th, and 100th iterations are shown.

2.3. Extension: crack opening direction determined by surface principal stress

The opening direction of cracks are assumed to be parallel to the XZ- and YZ-plane in section 2.1 and section 2.2. However, this assumption bought anisotropic property into the structure, and is unrealistic considering the natural growth of cracks. Therefore, to further extend our framework, a method to determine the crack opening directions according to the surface first principal stress direction is developed in this chapter.

2.3.1. Formulation

In Equation 2.25, we established a process to transform stress tensor expressed in global coordinates system to local system determined by the crack plane and the XZ- and YZ-plane. To enable surface principal stress determination, only one more transformation matrix needs to be added, and Equation 2.25 is extended to

$$\begin{bmatrix}\sigma_{\theta\theta}\\ \sigma_{r\theta}\\ \sigma_{z\theta}\end{bmatrix} = \underbrace{\begin{bmatrix}0&1&0\\1&0&0\\0&0&1\end{bmatrix}}_{\boldsymbol{M}_1}\boldsymbol{R}_t\boldsymbol{R}\underbrace{\begin{bmatrix}\sigma_{xx}&\sigma_{xy}&\sigma_{xz}\\ \sigma_{yx}&\sigma_{yy}&\sigma_{yz}\\ \sigma_{zx}&\sigma_{zy}&\sigma_{zz}\end{bmatrix}}_{\boldsymbol{\sigma}}\boldsymbol{R}^{\intercal}\boldsymbol{R}_t^{\intercal}\underbrace{\begin{bmatrix}0\\1\\0\end{bmatrix}}_{\boldsymbol{M}_2}. \tag{2.32}$$

Consider the coordinates system as shown in Figure 2.6 and Figure 2.24, the stress transformation matrix $\boldsymbol{R}_t$ has the form:

$$\boldsymbol{R}_t = \begin{bmatrix}1&0&0\\0&cos(\theta_p)&-sin(\theta_p)\\0&sin(\theta_p)&cos(\theta_p)\end{bmatrix}, \tag{2.33}$$

where θ_p is the angle between the direction of the surface 1st principal stress and the Y' axis. According to Sun et al. (2009), it is evaluated by

$$\theta_p = \pm\frac{1}{2}\tan^{-1}\left(-\frac{2\sigma_{y'z'}}{\sigma_{z'z'}-\sigma_{y'y'}}\right), \tag{2.34}$$

where the sign ± is defined according to whether if the result leads to the 1st principal stress in that direction, $\sigma_{y'z'}$, $\sigma_{y'y'}$, and $\sigma_{z'z'}$ are the surface stresses in the local coordinates.

Based on these stress transformation, the ERR evaluation expression Equation 2.26 is further extended to

$$G = \frac{\pi\epsilon}{2\mu\bar{E}}\boldsymbol{M}_2^{\intercal}\boldsymbol{R}_t\boldsymbol{R}\boldsymbol{\sigma}^{\intercal}\boldsymbol{R}^{\intercal}\boldsymbol{R}_t^{\intercal}\boldsymbol{M}_1^{\intercal}\boldsymbol{H}'^{\intercal}\boldsymbol{H}'\boldsymbol{M}_1\boldsymbol{R}_t\boldsymbol{R}\,\boldsymbol{\sigma}\boldsymbol{R}^{\intercal}\boldsymbol{R}_t^{\intercal}\boldsymbol{M}_2. \tag{2.35}$$

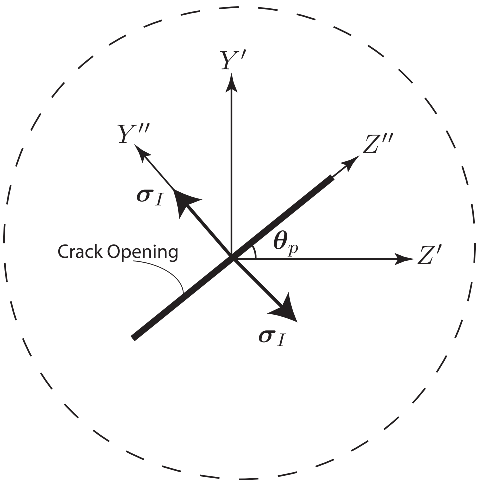
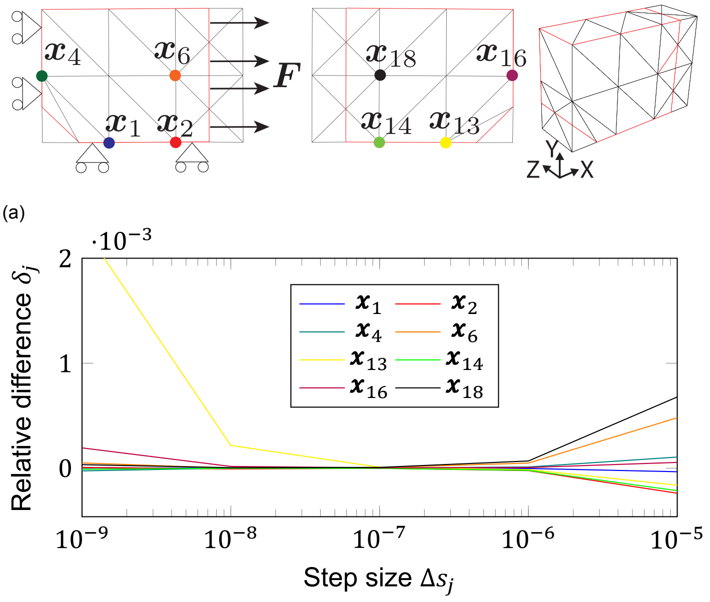


Figure 2.24: Crack opening direction is perpendicular to the surface first principal stress σ_I. The Y' and Z' are the original local coordinate system and Y'', Z'' are the transformed coordinates system. The direction of X'' stays the same as X'.

The general topology optimization formulation and sensitivity stays the same as Equation 2.19, Equation 2.20, Equation 2.29 and Equation 2.30. The extended sensitivity formulation for Equation 2.35 is given in Appendix G.

2.3.2. Sensitivity verification

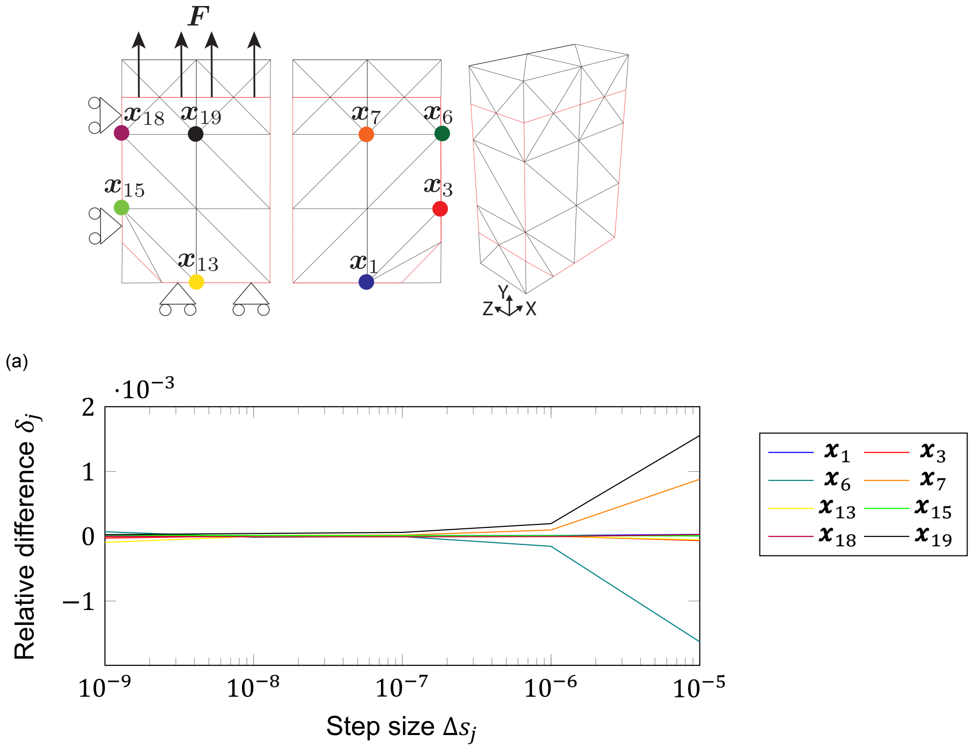
The same method and model of finite difference analysis as in subsection 2.2.1 are used to verify the analytical sensitivity of the extended framework. It is found that the relative differences can also reach their minimum values $\delta_{j,min} \approx 8 \times 10^{-6} - 7 \times 10^{-5}$ at step size around 10^{-7}.



(a)

(b)

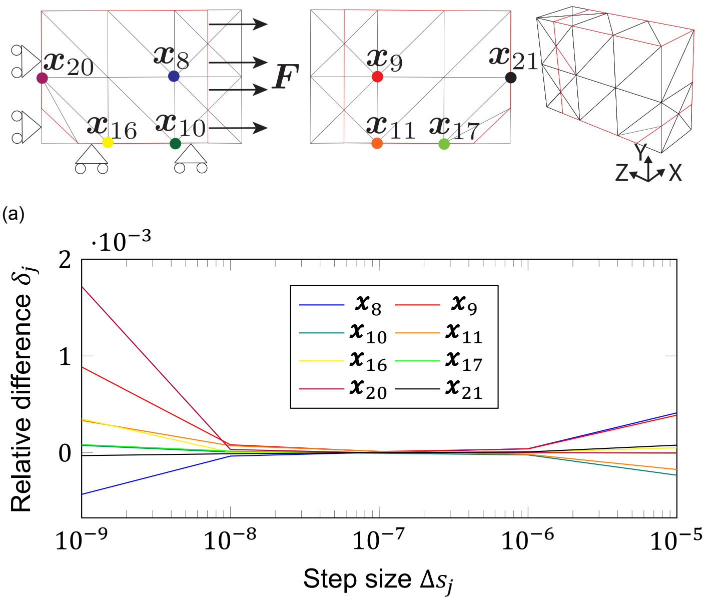
Figure 2.25: (a): Geometry uses for sensitivity verification in X-direction, where the solid part is marked in red. All crack openings are perpendicular to the surface first principal stress. (b): Relative difference δ_j with respect to step size Δs_j on different design variables s_j.



(a)

(b)

Figure 2.26: (a): Geometry uses for sensitivity verification in Y-direction, where the solid part is marked in red. All crack openings are perpendicular to the surface first principal stress. (b): Relative difference δ_j with respect to step size Δs_j on different design variables s_j.



(a)

(b)

Figure 2.27: (a):Geometry uses for sensitivity verification in Z-direction, where the solid part is marked in red. All crack openings are parallel to the XZ-plane. (b): Relative difference δ_j with respect to step size Δs_j on different design variables s_j.

2.3.3. Numerical example: cube with a sphere

The same triaxially loaded cube with a hole optimization as in subsection 2.2.2 is performed to examine the effectiveness of the extended framework. The solid part of the initial and final designs, and their ERRs distributions are shown in Figure 2.28. Figure 2.29 shows the convergence of objective function and volume constraint. The cross-sectional view of the design at iterations 10, 50, and 160 are also shown. The optimization approaches convergence after around 100 iterations. The ERRs over the material interface are smoothed out during the optimization, and the design comes to an end with 1/8 of a nearly perfect spherical hole. The hole radius of X, Y, and Z directions are $r_x = 0.5179$, $r_y = 0.5117$, and $r_z = 0.5103$, respectively, so the average radius is 0.5133; the relative differences of radius in three directions are 0.90%, -0.3%, and 0.58%, respectively, which is obviously better than the result of the former framework. This happens because the surface first principal stress distributes evenly in three directions, and the surface first principal stress has a dominant contribution to the calculation of ERRs, so the ERR field also distributes evenly in three directions. This doesn't mean that the anisotropic property of cracks is disappeared, but it is counteracted and therefore provides an isotropic cube overall.

This example shows that the anisotropic problem of the former framework can be solved by assuming crack openings perpendicular to the surface first principal stresses. This framework also fits the hypothesis that crack planes extend to the direction perpendicular to the maximum tensile stress.

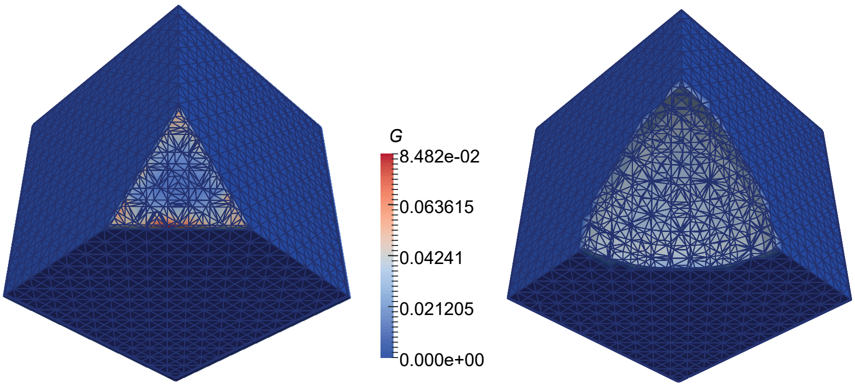


Figure 2.28: (a) Initial design with a tetrahedral hole, where concentration of energy release rates is spotted on three corners and three edges; (b) Final design with $1/8$ of a cube with a sphere, where the energy release rates distribute uniformly over the material interface

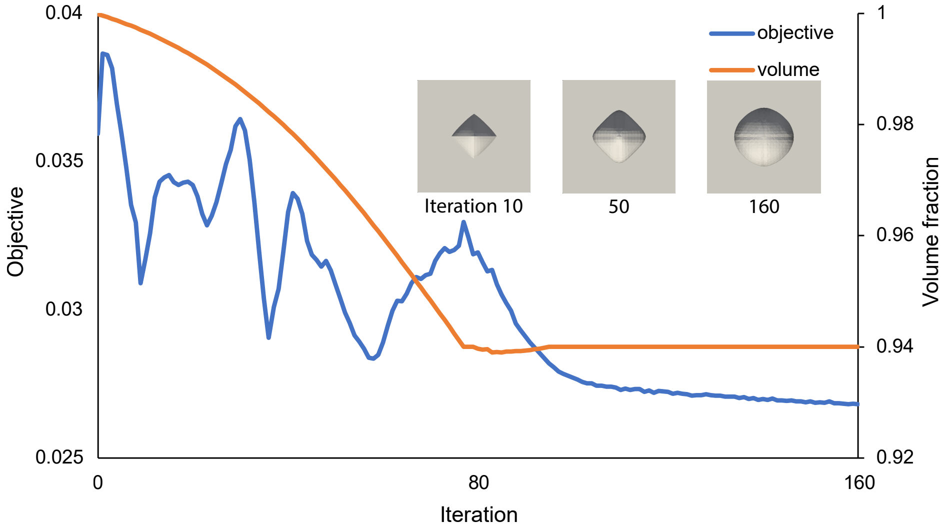


Figure 2.29: The objective function approaches convergence after around 100 iterations. Designs at iteration 10, 50, and 160 are shown

2.4. Summary and Conclusions

This work extended the fracture-based topology optimization framework that tailors the fracture resistance of structures in 2D (Zhang et al., 2022) to 3D. The new 3D framework is based on IGFEM. The design domain is immersed into a background mesh and expressed by a level set function interpolated by the compactly supported radial basis functions. Cracks are assumed to nucleate on all the enriched nodes on the geometry surface and perpendicular to the surface. The topological derivative method is used to evaluate the energy release rates. This method is numerically efficient because it only requires a single FEA to determine the ERRs through weight functions and the stress fields. It avoids the need of modeling and meshing the crack planes in the geometry. The weight functions were evaluated with FEA and stored as offline numerical expressions. A stress recovery technique in 3D is used to find a more accurate stress field than directly calculated with FEA and to reduce the effect of the stress overestimation problem of enriched FEMs. P-mean aggregation function is used to collect the local ERRs into a single term in the objective function to reduce computational cost. The analytical sensitivity of objective function was derived and examined with the finite difference analysis.

Two numerical examples were given in the first framework of the thesis, in which crack openings were assumed to be parallel to the XZ- and YZ-plane. In the first example, an octahedral void within a triaxially loaded cube was modified to an ellipsoidal void. The ERRs distributed evenly throughout the surface of the ellipsoidal void. The radius of the ellipsoid in the Y-direction turned out to be larger. This was caused by the anisotropy bought into the model by assuming all the crack openings parallel to the XZ-plane. Higher energy release rates were spotted in X- and Z-directions than Y-direction if the void is spherical, so the algorithm extended the void in Y-direction to smoothen the field. Even though this anisotropy resulted in an ellipsoid, this demonstrates the capacity of this framework. It shows that both stress and crack geometry can affect the optimized design.

L-bracket optimization was performed to examine the capacity of this framework in large-scale problems. The algorithm smoothed out the sharp rectangular corner with a high ERR concentration and provided a final design with a smooth transition throughout the surfaces. However, a high computational cost was consumed.

An extended framework assumes the opening directions of cracks perpendicular to the surface first principal stress was provided in the end. The same cube under triaxial load model was optimized. The extended framework provided a better sphere compared to the first. The differences among the radii in three axial directions are less than 1%. This proves that by assuming crack openings perpendicular to the surface first principal stress, the anisotropy of cracks can be counteracted globally. This framework is effective in optimizing ideal brittle structures, in which the hypothesis that crack planes extend to the direction perpendicular to the maximum tensile stress is applicable.

Several final conclusions and prospective investigations were drawn:

1. The 3D framework is more versatile than the 2D because it can perform optimization with considerations on crack radius and crack opening direction. As has been found in the cube with an ellipsoid example, the anisotropy of cracks does have an effect on the design. Therefore, it is recommended to investigate the influence of customizing the crack opening directions according to different criteria. For example, since the 3D topological derivative method not only takes the mode I fracture into consideration, but the mode II and III as well, it is possible to consider shear-dominant criteria rather than tension-dominant. This helps to perform optimization on a variety of materials.

2. This framework assumes the crack plane to be perpendicular to the geometry surface, crack opening direction either parallel to certain planes or perpendicular to the surface first principal stress, and evaluates ERR with the deepest point in the crack front. However, these assumptions do not necessarily represent the largest ERR on the evaluation point on the surface of the geometry, which is the actual crack propagating direction. To cope with this, the critical plane method should be applied to the framework.

3. In this work, the weight functions are stored numerically. However, analytical weight functions would be more beneficial when the critical plane method is implemented, or multiple points in the crack fronts are considered. This is because the critical method needs to solve a maximum value problem, and this problem needs to be embedded into the topology optimization formulation and

sensitivity. It will be more efficient to use analytical expressed weight functions to find sensitivity than numerical ones.

4. Cracks were assumed to be semi-circular in this work, and the topological derivative method was applied accordingly. This assumption holds with most smooth surfaces because the dimension of cracks is significantly smaller than the surfaces. However, this assumption is violated when sharp edges or corners appeare in the design. For example, on the edges of the octahedral void in the cube example, the actual shape of the cracks should have an opening of 90° instead of semi-circular. In order to improve and generalize the framework, the evaluation of energy release rates of cracks in different shapes should be investigated. This also enables the possibility of evaluating the cracks within the geometry, which in many cases are assumed to be circular or elliptical.

5. Rather than placing cracks throughout the geometry surface and using the same radius for all cracks, it is possible to only place cracks at certain regions, or use different crack radii at different regions. This is meaningful when the damage in the structure is different. For example, if the structure is partially exposed to a corrosive environment, it is important to enhance the fracture resistance of that part.

6. Even though the stiffness of the void elements is set to a value significantly smaller than the solid elements to avoid singular stiffness matrix, the void elements could have an influence on the optimization because the void elements exert minor pressure and traction on the surface. However, the method provided by van den Boom et al. (2019) to set the material properties of the void elements to zero and delete the extra DOFs could be applied to this framework to solve the problem. Furthermore, this method should also help to reduce the calculation cost to solve the linear static problem because it reduces the size of the stiffness matrix.

7. The stress recovery technique used can help attain more accurate stress field than directly calculated stress from FEA, but the large matrices and complex computation process used have a higher computational demand. This is also one of the most time-consuming processes in this framework. Parallel computing tools such as PETSc and MPI4Py could be utilized to distribute the workload on the processors.

Surface fracture analysis is a problem the highly relies on the quality of the boundary. By combing IGFEM with level set topology optimization, we were able to provide an accurately described boundary for evaluating ERRs throughout the surface of the target geometry, and enhancing its fracture resistance accordingly.

3 Reflection on Project

This graduation project worth 60 ECTS. The author spent 14 months (3.8.2021-20.10.2022, with a month off) to finish. There were delays and unplanned tasks occurred. However, considering his relatively low starting point, the duration is acceptable. This chapter discusses the roadmap and programming challenges (vectorization & parallelisation) in the project.

3.1. Roadmap & Gantt-chart

This project consists of four modules: IGFEM, topology optimization, fracture mechanics, complex Python programming. The author starts with a basic understanding of FEM, understanding of engineering mechanics, basic skill in Python programming, and without any understanding of topology optimization. The roadmap in Figure 3.1 shows his progress during the year.

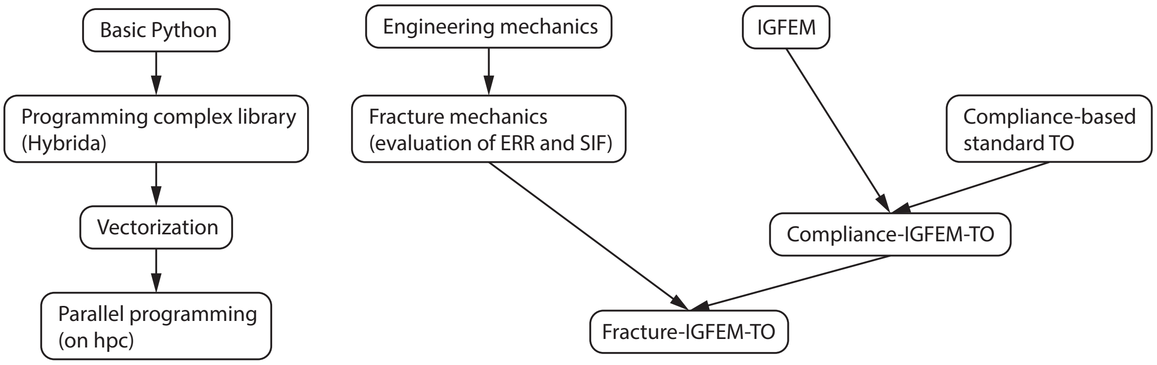

Basic Python
Programming complex library (Hybrida)
Vectorization
Parallel programming (on hpc)
Engineering mechanics
Fracture mechanics (evaluation of ERR and SIF)
IGFEM
Compliance-based standard TO
Compliance-IGFEM-TO
Fracture-IGFEM-TO

Figure 3.1: Project roadmap. The first row is the staring point of the author. The topology optimization (TO) module starts from zero understanding.

Page 27-28 shows the Gantt chart of the project. This Gantt chart is not the original one, but an actual timeline of the execution of the project. The unplanned tasks occurred are marked in yellow. The first unplanned task, develop 3D immersed Neumann boundary condition, happened because there was no 3D immersed Neumann boundary condition available in the group. The second unplanned task, vectorization, happened because the original code was too slowly. The other three unplanned events, ERR evaluation by considering crack opening direction based on the surface first principal stress, compliance constraint, and code parallelisation, are counted as extensions as they are out of the original scope of the project.

Delays are marked in red. The major delays are due to overtime debugging. The unexpected debugging time on evaluating weight functions costs 1 month, and debugging time on getting the right TO code costs 2 months. The first delay happened due to a lack of understanding of discrete-element

FEM (DEFEM) and the geometric engine of Hybrida, which is the enriched-FEM library that this project is built on. The model made did not have a fine enough crack shape for accurate evaluation. The author's understanding of DEFEM and Hybrida developed over time but this task was not re-executed because it is not the most important part of the project. The second delay happened because of unscientific ways in treating bugs and doing research. First, the author tried to verify the framework by performing full-scale numerical examples instead of using the minimal model and finite difference analysis node-by-node and element-by-element. Second, he tried to speed up the code without verifying the correctness of the code first. Third, the understanding towards the MeshPy library and FEM of the author was not thorough, so that he tried to compare the results of different coarse meshes on the same geometry, which produced unreliable analysis and wasted time. The delay of the L-bracket is caused by long calculation time. Note that the event, weight functions evaluation with ANSYS, is a solution to cope with the delay. It was considered at the start of the project and would be executed when the evaluation of weight functions with DEFEM failed. Those events extend to the end of the project, indicating that they are unable to be finished before graduation.

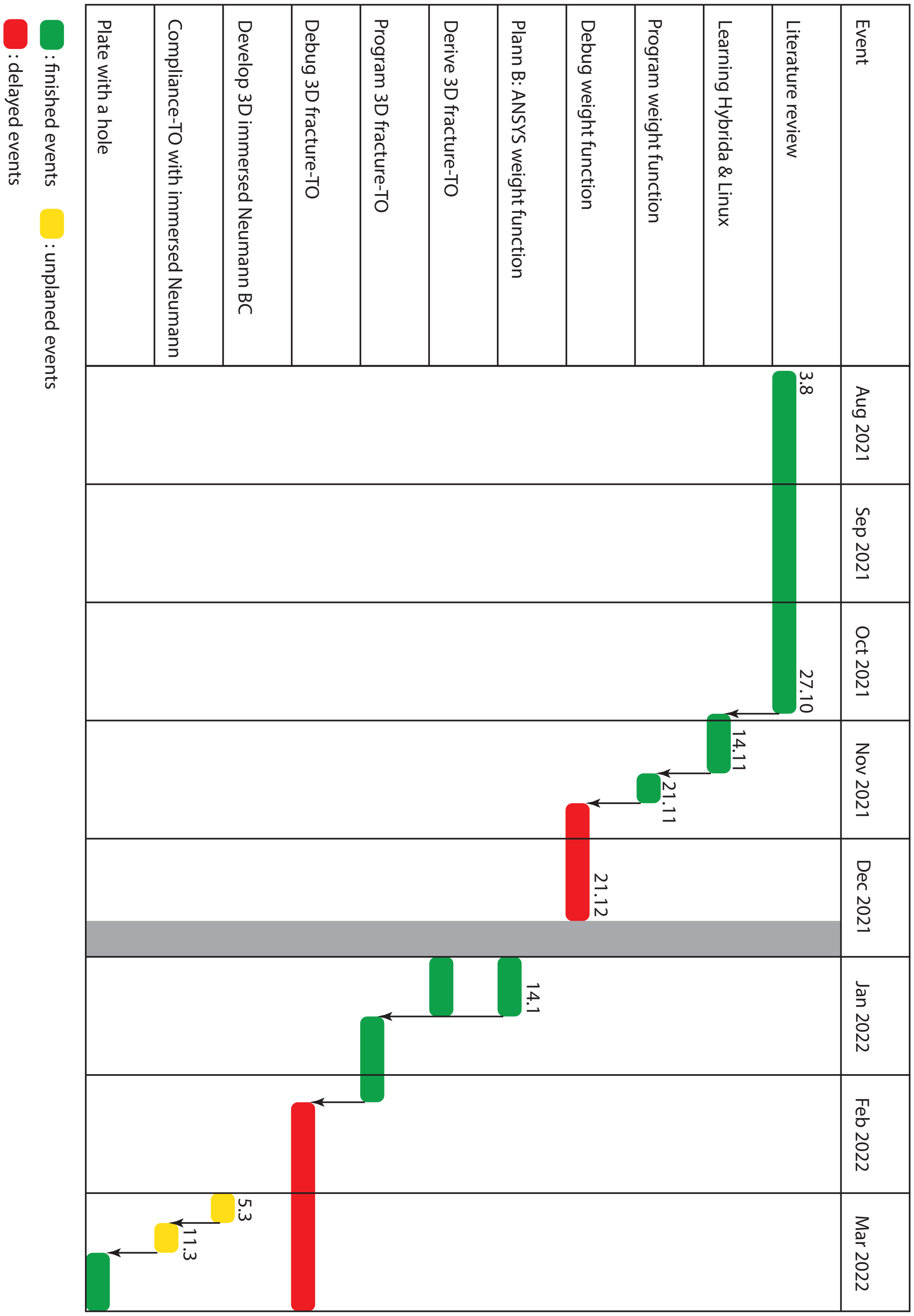
3.2. Computational challenges

Long computational time has been hindering the project since the beginning of coding. For example, evaluation of weight function takes around 8 hours, the optimization of cube with sphere takes around 69 hours, and the optimization of L-bracket takes 221 hours at least. Therefore, to continue 3D large-scale optimization, speeding up the code is the first thing.

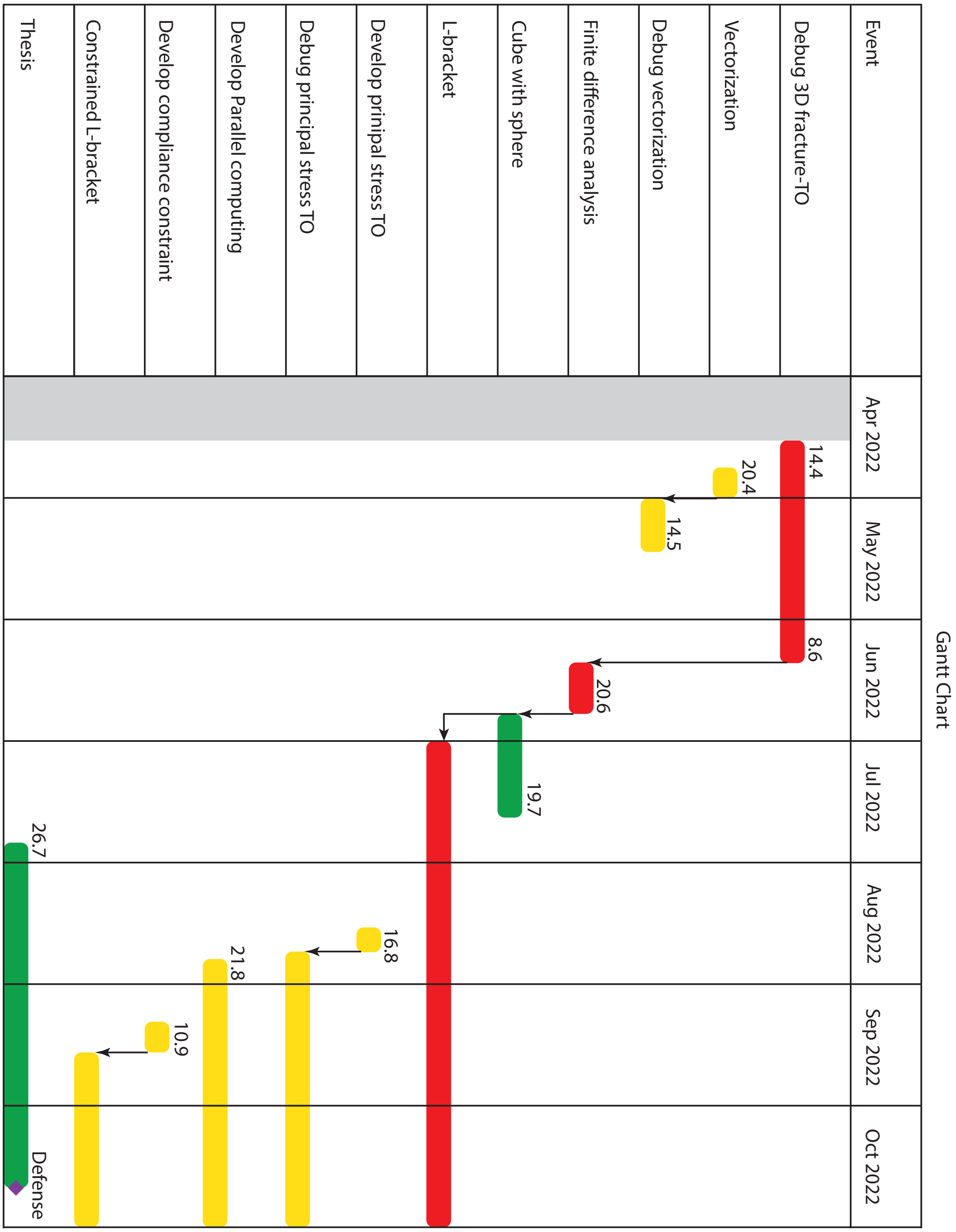
The author has investigated vectorization method with Numpy and several parallel computation packages for python, including Concurrent.futures, MPI4Py, DASK, and PETSc. Table 3.1 gives a non-quantify comparison of the advantages and disadvantages of the packages based on the ease of use on applying to Hybrida based on the author's experience. In summary, for simple for-loops and on single processor, process pool methods are the best because the automatic scheduler provided can exploit the capacity of the processor. However, passing complex Python classes to the process pools is difficult, which is the case for Hybrida. In this case, directly chunking the for-loops and sending a certain number of loops to multiple processors is more recommended. Method on direct parallelisation is referenced to Zaccone (2015). This requires minimal changes to the code, but it also has a barrier in data transmission and storage. Therefore, a shared-storage parallelisation based on OpenMPI and MPI4Py is worth investigating.

Method	Numpy vectorization	Direct parallelization with MPI4Py
Advantage	Fast in calculation	1. Easy to implement 2. Can be applied on unlimited number of processors
Disadvantage	1. Expensive in storing the data 2. Code needs to be re-written 3. Can only work on one processor	Non-automatic, may result in waste of processor time
Method	Process pool	PETSc
Advantage	Dynamic & automatic	Good at solving algebraic systems with MPI
Disadvantage	1. Difficult to pass complex arguments 2. Need to design objects that contain the loop	Need to use PETSc's data structure
Method	DASK	
Advantage	1. Scalable 2. DASK array can greatly reduce the size of matrices	
Disadvantage	1. Difficult to pass complex arguments 2. Building process of complex nested for-loops is inefficient	

Table 3.1: Comparison among different parallelisation methods/packages


Event
Aug 2021
Sep 2021
Oct 2021
Nov 2021
Dec 2021
Jan 2022
Feb 2022
Mar 2022
Literature review
3.8
27.10
Learning Hybrida & Linux
14.11
Program weight function
21.11
Debug weight function
21.12
Plann B: ANSYS weight function
14.1
Derive 3D fracture-TO
Program 3D fracture-TO
Debug 3D fracture-TO
Develop 3D immersed Neumann BC
5.3
Compliance-TO with immersed Neumann
11.3
Plate with a hole
🟩 : finished events
🟨 : unplaned events
🟥 : delayed events

Gantt Chart


Gantt Chart
Event
Apr 2022
May 2022
Jun 2022
Jul 2022
Aug 2022
Sep 2022
Oct 2022
Debug 3D fracture-TO
Vectorization
Debug vectorization
Finite difference analysis
Cube with sphere
L-bracket
Develop prinipal stress TO
Debug principal stress TO
Develop Parallel computing
Develop compliance constraint
Constrained L-bracket
Thesis
14.4
20.4
14.5
8.6
20.6
19.7
16.8
21.8
10.9
26.7
Defense

A

Coordinate Transformation Matrix

A.1. Normal vector of surface

Vector $\boldsymbol{n} = (u_x, u_y, u_z)$ is a unit vector normal to the geometry surface and pointing inward to it:

$$\boldsymbol{n} = \frac{\sum_{i=1}^{N_s} \boldsymbol{d}_i}{N_s}, \tag{A.1}$$

where N_s is the number of surface elements sharing the node; $\boldsymbol{d}_i = (d_{ix}, d_{iy}, d_{iz})$ is the unit directional vector of each surface element. $\boldsymbol{d}_i$ is defined as follows:

$$\boldsymbol{d}_i = sign \frac{\boldsymbol{d}_{01} \times \boldsymbol{d}_{02}}{|\boldsymbol{d}_{01} \times \boldsymbol{d}_{02}|}, \tag{A.2}$$

$$sign = \begin{cases} -1, & \text{if } \boldsymbol{d}_i \text{ is pointing inward to the geometry,} \\ 1, & \text{if } \boldsymbol{d}_i \text{ is pointing outward to the geometry,} \end{cases} \tag{A.3}$$

where $\boldsymbol{d}_{01} = \boldsymbol{x}_1 - \boldsymbol{x}_0$ and $\boldsymbol{d}_{02} = \boldsymbol{x}_2 - \boldsymbol{x}_0$ are vectors connecting node 0, 1 and node 0, 2 of a surface triangular element, respectively. $\boldsymbol{x}_0$, $\boldsymbol{x}_1$, $\boldsymbol{x}_2$ are the coordinates of the nodes.

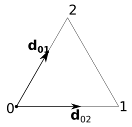


Figure A.1: Triangular element with directional Vectors

A.2. Crack opening parallel to XZ-plane

Two definitions was made to define cracks parallel to XZ-plane. First, the deepest crack front direction X' is align with the normal of the surface $\boldsymbol{n}$. Second, the crack opening direction is parallel to the XZ-plane, which indicates that the Z' axis of the local coordinates is parallel to the XZ-plane and perpendicular to the unit normal vector of XZ-plane. These then define a system equation:

$$\begin{cases} \boldsymbol{n_z} \cdot \boldsymbol{n} = 0, \\ \boldsymbol{n_z} \cdot (0, 1, 0) = 0, \\ |\boldsymbol{n_z}| = 1, \end{cases} \tag{A.4}$$

where $\boldsymbol{n}_z$ is the unit directional vector of Z' axis, and was found as

$$\boldsymbol{n_z} = \left(-\frac{u_z}{\sqrt{u_x^2 + u_z^2}}, 0, \frac{u_x}{\sqrt{u_x^2 + u_z^2}} \right). \tag{A.5}$$

The unit directional vector of Y' is the cross product of the unit directional vector of X' and Z', $\boldsymbol{n}_y = -\boldsymbol{n} \times \boldsymbol{n}_z$, which leads to

$$\boldsymbol{n}_y = \frac{1}{\sqrt{u_x^2 + u_z^2}\sqrt{u_x^2 + u_y^2 + u_z^2}} \left(-u_x u_y,\, u_x^2 + u_z^2,\, -u_y u_z\right). \tag{A.6}$$

Assembling up the unit directional vectors, the transformation equation between global and local coordinates can be set up:

$$\boldsymbol{R}\begin{bmatrix} u_x & \frac{-u_x u_y}{\sqrt{u_x^2+u_z^2}\sqrt{u_x^2+u_y^2+u_z^2}} & -\frac{u_z}{\sqrt{u_x^2+u_z^2}} \\ u_y & \frac{u_x^2+u_z^2}{\sqrt{u_x^2+u_z^2}\sqrt{u_x^2+u_y^2+u_z^2}} & 0 \\ u_z & \frac{-u_y u_z}{\sqrt{-u_y u_z}\sqrt{u_x^2+u_y^2+u_z^2}} & \frac{u_x}{\sqrt{u_x^2+u_z^2}} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{A.7}$$

By solving the above equation, the transformation matrix $\boldsymbol{R}$ could be found:

$$\boldsymbol{R} = \begin{bmatrix} \frac{u_x}{u_x^2+u_y^2+u_z^2} & \frac{u_y}{u_x^2+u_y^2+u_z^2} & \frac{u_z}{u_x^2+u_y^2+u_z^2} \\ \frac{-u_x u_y}{\sqrt{u_x^2+u_z^2}\sqrt{u_x^2+u_y^2+u_z^2}} & \frac{\sqrt{u_x^2+u_z^2}}{\sqrt{u_x^2+u_y^2+u_z^2}} & \frac{-u_y u_z}{\sqrt{u_x^2+u_z^2}\sqrt{u_x^2+u_y^2+u_z^2}} \\ -\frac{u_z}{\sqrt{u_x^2+u_z^2}} & 0 & \frac{u_x}{\sqrt{u_x^2+u_z^2}} \end{bmatrix}. \tag{A.8}$$

A.3. Crack Opening Parallel to YZ-plane

Two definitions was made to define cracks parallel to YZ-plane. First, the deepest crack front direction X' is align with the normal of the geometry surface $\boldsymbol{n}$. Second, the crack opening direction is parallel to the YZ-plane, which indicates that the Z' axis of the crack is parallel to the YZ-plane and perpendicular to the unit normal vector of YZ-plane. These then define a system equation:

$$\begin{cases} \boldsymbol{n_z} \cdot \boldsymbol{n} = 0, \\ \boldsymbol{n_z} \cdot (1, 0, 0) = 0, \\ |\boldsymbol{n_z}| = 1, \end{cases} \tag{A.9}$$

where $\boldsymbol{n}_z$ is the unit direction vector of Z' axis, and was found as:

$$\boldsymbol{n_z} = \left(0,\, -\frac{u_z}{\sqrt{u_y^2 + u_z^2}},\, \frac{u_y}{\sqrt{u_y^2 + u_z^2}}\right). \tag{A.10}$$

The unit directional vector of Y' is the cross product of the unit directional vector of X' and Z', $\boldsymbol{n}_y = -\boldsymbol{n} \times \boldsymbol{n}_z$, which leads to

$$\boldsymbol{n_y} = \frac{1}{\sqrt{u_y^2 + u_z^2}\sqrt{u_x^2 + u_y^2 + u_z^2}} \left(-\left(u_y^2 + u_z^2\right),\, u_x u_y,\, u_x u_z\right) \tag{A.11}$$

Assembling up the unit directional vectors, the transformation equation between global and local coordinates could be set up:

$$\boldsymbol{R}\begin{bmatrix} u_x & \frac{-u_y^2+u_z^2}{\sqrt{u_y^2+u_z^2}\sqrt{u_x^2+u_y^2+u_z^2}} & 0 \\ u_y & \frac{u_x u_y}{\sqrt{u_y^2+u_z^2}\sqrt{u_x^2+u_y^2+u_z^2}} & -\frac{u_z}{\sqrt{u_y^2+u_z^2}} \\ u_z & \frac{u_x u_z}{\sqrt{u_y^2+u_z^2}\sqrt{u_x^2+u_y^2+u_z^2}} & \frac{u_y}{\sqrt{u_y^2+u_z^2}} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{A.12}$$

By solving the above equation, the transformation matrix $\boldsymbol{R}$ could be found:

$$\boldsymbol{R} = \begin{bmatrix} \frac{u_x}{u_x^2+u_y^2+u_z^2} & \frac{u_y}{u_x^2+u_y^2+u_z^2} & \frac{u_z}{u_x^2+u_y^2+u_z^2} \\ -\frac{\sqrt{u_y^2+u_z^2}}{\sqrt{u_x^2+u_y^2+u_z^2}} & \frac{u_x u_y}{\sqrt{u_y^2+u_z^2}\sqrt{u_x^2+u_y^2+u_z^2}} & \frac{u_x u_z}{\sqrt{u_y^2+u_z^2}\sqrt{u_x^2+u_y^2+u_z^2}} \\ 0 & -\frac{u_z}{\sqrt{u_y^2+u_z^2}} & \frac{u_y}{\sqrt{u_x^2+u_z^2}} \end{bmatrix}. \tag{A.13}$$

B

3D Stress Recovery Technique

The interpolation matrices and differential operator of the stress recovery technique in Equation 2.27 are:

$$\boldsymbol{E}_\sigma = \begin{bmatrix} \boldsymbol{e}_\sigma & & & & & \\ & \boldsymbol{e}_\sigma & & & & \\ & & \boldsymbol{e}_\sigma & & & \\ & & & \boldsymbol{e}_\sigma & & \\ & & & & \boldsymbol{e}_\sigma & \\ & & & & & \boldsymbol{e}_\sigma \end{bmatrix}, \tag{B.1}$$

$$\boldsymbol{e}_\sigma = \begin{bmatrix} 1 & x & y & z & xy & yz & zx & x^2 & y^2 & z^2 \end{bmatrix}, \tag{B.2}$$

$$\partial_\sigma = \begin{bmatrix} \frac{\partial}{\partial x} & 0 & 0 & \frac{\partial}{\partial y} & 0 & \frac{\partial}{\partial z} \\ 0 & \frac{\partial}{\partial y} & 0 & \frac{\partial}{\partial x} & \frac{\partial}{\partial z} & 0 \\ 0 & 0 & \frac{\partial}{\partial z} & 0 & \frac{\partial}{\partial y} & \frac{\partial}{\partial x} \end{bmatrix}, \tag{B.3}$$

$$\boldsymbol{E}_\zeta = \begin{bmatrix} 1 & x & y & z & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & x & y & z & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & x & y & z \end{bmatrix}, \tag{B.4}$$

$$\bar{\boldsymbol{E}}_{\sigma}^{\mathsf{T}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ x & 0 & 0 & -y & 0 & 0 \\ y & 0 & 0 & 0 & 0 & 0 \\ z & 0 & 0 & 0 & 0 & 0 \\ 2xy & 0 & 0 & -y^2 & 0 & 0 \\ yz & 0 & 0 & 0 & 0 & 0 \\ x^2 & 0 & 0 & -xy & yz & -xz \\ 0 & x & 0 & 0 & 0 & 0 \\ 0 & y & 0 & 0 & -z & 0 \\ 0 & z & 0 & 0 & 0 & 0 \\ 0 & 2xy & 0 & -x^2 & 0 & 0 \\ 0 & 2yz & 0 & 0 & -z^2 & 0 \\ 0 & xz & 0 & 0 & 0 & 0 \\ 0 & y^2 & 0 & -xy & -yz & xz \\ 0 & 0 & x & 0 & 0 & 0 \\ 0 & 0 & y & 0 & 0 & 0 \\ 0 & 0 & z & 0 & -y & 0 \\ 0 & 0 & xy & 0 & 0 & 0 \\ 0 & 0 & 2yz & 0 & -y^2 & 0 \\ 0 & 0 & 2xz & 0 & 0 & -x^2 \\ 0 & 0 & z^2 & xy & -yz & -xz \\ 0 & y & 0 & -x & 0 & 0 \\ 0 & 0 & 0 & -z & 0 & 0 \\ 0 & 0 & 0 & 2yz & 0 & -z^2 \\ 0 & 0 & 0 & 2xz & -z^2 & 0 \\ 0 & x^2 & 0 & 0 & 0 & 0 \\ y^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -z^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & -x & 0 \\ 0 & 0 & 0 & 0 & 2xy & -x^2 \\ 0 & 0 & 0 & -x^2 & 2xz & 0 \\ 0 & 0 & 0 & 0 & -x^2 & 0 \\ 0 & 0 & y^2 & 0 & 0 & 0 \\ 0 & z^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & z & 0 & 0 & -x \\ 0 & 0 & 0 & 0 & 0 & -y \\ x & 0 & 0 & 0 & 0 & -z \\ 0 & 0 & 0 & 0 & -y^2 & 2xy \\ 0 & 0 & 0 & -y^2 & 0 & 2xy \\ 0 & 0 & x^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -y^2 \\ z^2 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{B.5}$$

C

Sensitivity

This section is aiming at finding the derivative terms within the adjoint equation Equation 2.30.

C.1. $\frac{\partial J}{\partial s_j}$

The derivative of J with respect to the design variable s_j is given as

$$\frac{\partial J}{\partial s_j} = \left(\frac{1}{N_{node}}\right)^{\frac{1}{p}} \frac{1}{p} \left(\sum_{i=1}^{N_{node}} G_i^p\right)^{\frac{1}{p}-1} \left(\sum_{i=1}^{N_{node}} p G_i^{p-1} \frac{\partial G_i}{\partial s_j}\right), \tag{C.1}$$

where

$$\begin{aligned} \frac{\partial G_i}{\partial s_j} = \frac{\pi\epsilon}{\mu\bar{E}} \Bigg(& \boldsymbol{M}_2^{\intercal} \frac{\partial \boldsymbol{R}}{\partial s_j} \boldsymbol{\sigma}^{\intercal} \boldsymbol{R}^{\intercal} \boldsymbol{M}_1^{\intercal} \boldsymbol{H}'^{\intercal} \boldsymbol{H}' \boldsymbol{M}_1 \boldsymbol{R}\,\boldsymbol{\sigma} \boldsymbol{R}^{\intercal} \boldsymbol{M}_2 \\ & + \boldsymbol{M}_2^{\intercal} \boldsymbol{R}\, \frac{\partial \boldsymbol{\sigma}^{\intercal}}{\partial s_j} \boldsymbol{R}^{\intercal} \boldsymbol{M}_1^{\intercal} \boldsymbol{H}'^{\intercal} \boldsymbol{H}' \boldsymbol{M}_1 \boldsymbol{R}\,\boldsymbol{\sigma} \boldsymbol{R}^{\intercal} \boldsymbol{M}_2 \\ & + \boldsymbol{M}_2^{\intercal} \boldsymbol{R}\,\boldsymbol{\sigma}^{\intercal} \frac{\partial \boldsymbol{R}^{\intercal}}{\partial s_j} \boldsymbol{M}_1^{\intercal} \boldsymbol{H}'^{\intercal} \boldsymbol{H}' \boldsymbol{M}_1 \boldsymbol{R}\,\boldsymbol{\sigma} \boldsymbol{R}^{\intercal} \boldsymbol{M}_2 \Bigg). \end{aligned} \tag{C.2}$$

According to Equation A.8 and Equation A.13, $\frac{\partial \boldsymbol{R}}{\partial s_j}$ could be solved with

$$\frac{\partial \boldsymbol{R}}{\partial s_j} = \frac{\partial \boldsymbol{R}}{\partial u_x}\frac{\partial u_x}{\partial s_j} + \frac{\partial \boldsymbol{R}}{\partial u_y}\frac{\partial u_y}{\partial s_j} + \frac{\partial \boldsymbol{R}}{\partial u_z}\frac{\partial u_z}{\partial s_j}, \tag{C.3}$$

where $\boldsymbol{R}$ is equivalent to $\boldsymbol{R}_{XZ}$ or $\boldsymbol{R}_{YZ}$. For cracks with opening direction parallel to XZ-plane:

$$\frac{\partial \boldsymbol{R}_{XZ}}{\partial u_x} = \begin{bmatrix} \frac{-u_x^2+u_y^2+u_z^2}{\left(u_x^2+u_y^2+u_z^2\right)^2} & -\frac{2u_xu_y}{\left(u_x^2+u_y^2+u_z^2\right)^2} & -\frac{2u_xu_z}{\left(u_x^2+u_y^2+u_z^2\right)^2} \\ \frac{u_y\left(u_x^4-u_y^2u_z^2-u_z^4\right)}{\left(u_x^2+u_z^2\right)^{\frac{3}{2}}\left(u_x^2+u_y^2+u_z^2\right)^{\frac{3}{2}}} & \frac{u_xu_y^2}{\left(u_x^2+u_z^2\right)^{\frac{1}{2}}\left(u_x^2+u_y^2+u_z^2\right)^{\frac{3}{2}}} & \frac{u_xu_yu_z\left(2u_x^2+u_y^2+2u_z^2\right)}{\left(u_x^2+u_z^2\right)^{\frac{3}{2}}\left(u_x^2+u_y^2+u_z^2\right)^{\frac{3}{2}}} \\ \frac{u_xu_z}{\left(u_x^2+u_z^2\right)^{\frac{3}{2}}} & 0 & \frac{u_z^2}{\left(u_x^2+u_z^2\right)^{\frac{3}{2}}} \end{bmatrix}, \tag{C.4}$$

$$\frac{\partial \boldsymbol{R}_{XZ}}{\partial u_y} = \begin{bmatrix} -\frac{2u_xu_z}{\left(u_x^2+u_y^2+u_z^2\right)^2} & \frac{u_x^2-u_y^2+u_z^2}{\left(u_x^2+u_y^2+u_z^2\right)^2} & -\frac{2u_yu_z}{\left(u_x^2+u_y^2+u_z^2\right)^2} \\ -\frac{u_x\sqrt{u_x^2+u_z^2}}{\left(u_x^2+u_y^2+u_z^2\right)^{\frac{3}{2}}} & -\frac{u_y\sqrt{u_x^2+u_z^2}}{\left(u_x^2+u_y^2+u_z^2\right)^{\frac{3}{2}}} & -\frac{u_z\sqrt{u_x^2+u_z^2}}{\left(u_x^2+u_y^2+u_z^2\right)^{\frac{3}{2}}} \\ 0 & 0 & 0 \end{bmatrix}, \tag{C.5}$$

$$\frac{\partial \boldsymbol{R}_{XZ}}{\partial u_z}=\begin{bmatrix} -\frac{2u_x u_z}{\left(u_x^2+u_y^2+u_z^2\right)^2} & -\frac{2u_y u_z}{\left(u_x^2+u_y^2+u_z^2\right)^2} & \frac{u_x^2+u_y^2-u_z^2}{\left(u_x^2+u_y^2+u_z^2\right)^2} \\ \frac{u_x u_y u_z\left(2u_x^2+u_y^2+2u_z^2\right)}{\left(u_x^2+u_z^2\right)^{\frac{3}{2}}\left(u_x^2+u_y^2+u_z^2\right)^{\frac{3}{2}}} & \frac{u_y^2 u_z}{\left(u_x^2+u_z^2\right)^{\frac{1}{2}}\left(u_x^2+u_y^2+u_z^2\right)^{\frac{3}{2}}} & -\frac{u_y\left(u_x^4+u_y^2u_z^2-u_z^4\right)}{\left(u_x^2+u_z^2\right)^{\frac{3}{2}}\left(u_x^2+u_y^2+u_z^2\right)^{\frac{3}{2}}} \\ -\frac{u_x^2}{\left(u_x^2+u_z^2\right)^{\frac{3}{2}}} & 0 & -\frac{u_x u_z}{\left(u_x^2+u_z^2\right)^{\frac{3}{2}}} \end{bmatrix}. \tag{C.6}$$

For cracks with opening direction parallel to YZ-plane:

$$\frac{\partial \boldsymbol{R}_{YZ}}{\partial u_x}=\begin{bmatrix} \frac{-u_x^2+u_y^2+u_z^2}{\left(u_x^2+u_y^2+u_z^2\right)^2} & -\frac{2u_x u_y}{\left(u_x^2+u_y^2+u_z^2\right)^2} & -\frac{2u_x u_z}{\left(u_x^2+u_y^2+u_z^2\right)^2} \\ \frac{u_x\sqrt{u_y^2+u_z^2}}{\left(u_x^2+u_y^2+u_z^2\right)^{\frac{3}{2}}} & \frac{u_y\sqrt{u_y^2+u_z^2}}{\left(u_x^2+u_y^2+u_z^2\right)^{\frac{3}{2}}} & \frac{u_z\sqrt{u_y^2+u_z^2}}{\left(u_x^2+u_y^2+u_z^2\right)^{\frac{3}{2}}} \\ 0 & 0 & 0 \end{bmatrix}, \tag{C.7}$$

$$\frac{\partial \boldsymbol{R}_{YZ}}{\partial u_y}=\begin{bmatrix} -\frac{2u_x u_y}{\left(u_x^2+u_y^2+u_z^2\right)^2} & \frac{u_x^2-u_y^2+u_z^2}{\left(u_x^2+u_y^2+u_z^2\right)^2} & -\frac{2u_y u_z}{\left(u_x^2+u_y^2+u_z^2\right)^2} \\ -\frac{u_x^2 u_y}{\left(u_x^2+u_z^2\right)^{\frac{1}{2}}\left(u_x^2+u_y^2+u_z^2\right)^{\frac{3}{2}}} & \frac{u_x\left(u_x^2uz^2-u_y^4+u_z^4\right)}{\left(u_y^2+u_z^2\right)^{\frac{3}{2}}\left(u_x^2+u_y^2+u_z^2\right)^{\frac{3}{2}}} & -\frac{u_x u_y u_z\left(u_x^2+2u_y^2+2u_z^2\right)}{\left(u_y^2+u_z^2\right)^{\frac{3}{2}}\left(u_x^2+u_y^2+u_z^2\right)^{\frac{3}{2}}} \\ 0 & \frac{u_y u_z}{\left(u_y^2+u_z^2\right)^{\frac{3}{2}}} & \frac{u_z^2}{\left(u_y^2+u_z^2\right)^{\frac{3}{2}}} \end{bmatrix}, \tag{C.8}$$

$$\frac{\partial \boldsymbol{R}_{YZ}}{\partial u_z}=\begin{bmatrix} -\frac{2u_x u_z}{\left(u_x^2+u_y^2+u_z^2\right)^2} & -\frac{2u_y u_z}{\left(u_x^2+u_y^2+u_z^2\right)^2} & \frac{u_x^2+u_y^2-u_z^2}{\left(u_x^2+u_y^2+u_z^2\right)^2} \\ -\frac{u_x^2 u_z}{\left(u_x^2+u_z^2\right)^{\frac{1}{2}}\left(u_x^2+u_y^2+u_z^2\right)^{\frac{3}{2}}} & -\frac{u_x u_y u_z\left(u_x^2+2u_y^2+2u_z^2\right)}{\left(u_y^2+u_z^2\right)^{\frac{3}{2}}\left(u_x^2+u_y^2+u_z^2\right)^{\frac{3}{2}}} & \frac{u_x\left(u_x^2uy^2+u_y^4-u_z^4\right)}{\left(u_y^2+u_z^2\right)^{\frac{3}{2}}\left(u_x^2+u_y^2+u_z^2\right)^{\frac{3}{2}}} \\ 0 & -\frac{u_y^2}{\left(u_y^2+u_z^2\right)^{\frac{3}{2}}} & -\frac{u_y u_z}{\left(u_y^2+u_z^2\right)^{\frac{3}{2}}} \end{bmatrix}. \tag{C.9}$$

$\frac{\partial u_x}{\partial s_j}$, $\frac{\partial u_y}{\partial s_j}$, $\frac{\partial u_z}{\partial s_j}$ could be further solved with

$$\begin{bmatrix} \frac{\partial u_x}{\partial s_j} \\ \frac{\partial u_y}{\partial s_j} \\ \frac{\partial u_z}{\partial s_j} \end{bmatrix}=\begin{bmatrix} \sum \frac{\partial u_x}{\partial d_{ix}}\frac{\partial d_{ix}}{\partial s_j}+\sum \frac{\partial u_x}{\partial d_{iy}}\frac{\partial d_{iy}}{\partial s_j}+\sum \frac{\partial u_x}{\partial d_{iz}}\frac{\partial d_{iz}}{\partial s_j} \\ \sum \frac{\partial u_y}{\partial d_{ix}}\frac{\partial d_{ix}}{\partial s_j}+\sum \frac{\partial u_y}{\partial d_{iy}}\frac{\partial d_{iy}}{\partial s_j}+\sum \frac{\partial u_y}{\partial d_{iz}}\frac{\partial d_{iz}}{\partial s_j} \\ \sum \frac{\partial u_z}{\partial d_{ix}}\frac{\partial d_{ix}}{\partial s_j}+\sum \frac{\partial u_z}{\partial d_{iy}}\frac{\partial d_{iy}}{\partial s_j}+\sum \frac{\partial u_z}{\partial d_{iz}}\frac{\partial d_{iz}}{\partial s_j} \end{bmatrix}, \tag{C.10}$$

where

$$\frac{\partial u_x}{\partial d_{ix}}=\frac{\left(\sum d_{iy}\right)^2+\left(\sum d_{iz}\right)^2}{N[\left(\sum d_{ix}\right)^2+\left(\sum d_{iy}\right)^2+\left(\sum d_{iz}\right)^2]^{\frac{3}{2}}}, \tag{C.11}$$

$$\frac{\partial u_y}{\partial d_{iy}}=\frac{\left(\sum d_{ix}\right)^2+\left(\sum d_{iz}\right)^2}{N[\left(\sum d_{ix}\right)^2+\left(\sum d_{iy}\right)^2+\left(\sum d_{iz}\right)^2]^{\frac{3}{2}}}, \tag{C.12}$$

$$\frac{\partial u_z}{\partial d_{iz}}=\frac{\left(\sum d_{ix}\right)^2+\left(\sum d_{iy}\right)^2}{N[\left(\sum d_{ix}\right)^2+\left(\sum d_{iy}\right)^2+\left(\sum d_{iz}\right)^2]^{\frac{3}{2}}}, \tag{C.13}$$

$$\frac{\partial u_x}{\partial d_{iy}}=\frac{\partial u_y}{\partial d_{ix}}=-\frac{\sum d_{ix}\sum d_{iy}}{N[\left(\sum d_{ix}\right)^2+\left(\sum d_{iy}\right)^2+\left(\sum d_{iz}\right)^2]^{\frac{3}{2}}}, \tag{C.14}$$

$$\frac{\partial u_x}{\partial d_{iz}}=\frac{\partial u_z}{\partial d_{ix}}=-\frac{\sum d_{ix}\sum d_{iz}}{N[\left(\sum d_{ix}\right)^2+\left(\sum d_{iy}\right)^2+\left(\sum d_{iz}\right)^2]^{\frac{3}{2}}}, \tag{C.15}$$

$$\frac{\partial u_y}{\partial d_{iz}}=\frac{\partial u_z}{\partial d_{iy}}=-\frac{\sum d_{iy}\sum d_{iz}}{N[\left(\sum d_{ix}\right)^2+\left(\sum d_{iy}\right)^2+\left(\sum d_{iz}\right)^2]^{\frac{3}{2}}}. \tag{C.16}$$

To solve $\frac{\partial d_{ix,y,z}}{\partial s_j}$, let $\boldsymbol{D} = \boldsymbol{d}_{01} \times \boldsymbol{d}_{02}$. Therefore, equation A.2 becomes

$$\boldsymbol{d}_i = sign\frac{\boldsymbol{D}}{|\boldsymbol{D}|} = sign\frac{1}{\sqrt{D_x^2 + D_y^2 + D_z^2}}\left(D_x, D_y, D_z\right), \tag{C.17}$$

and

$$\begin{bmatrix} \frac{\partial d_{ix}}{\partial s_j} \\ \frac{\partial d_{iy}}{\partial s_j} \\ \frac{\partial d_{iz}}{\partial s_j} \end{bmatrix} = sign\begin{bmatrix} \frac{\partial d_{ix}}{\partial D_x}\frac{\partial D_x}{\partial s_j} + \frac{\partial d_{ix}}{\partial D_y}\frac{\partial D_y}{\partial s_j} + \frac{\partial d_{ix}}{\partial D_z}\frac{\partial D_z}{\partial s_j} \\ \frac{\partial d_{iy}}{\partial D_x}\frac{\partial D_x}{\partial s_j} + \frac{\partial d_{iy}}{\partial D_y}\frac{\partial D_y}{\partial s_j} + \frac{\partial d_{iy}}{\partial D_z}\frac{\partial D_z}{\partial s_j} \\ \frac{\partial d_{iz}}{\partial D_x}\frac{\partial D_x}{\partial s_j} + \frac{\partial d_{iz}}{\partial D_y}\frac{\partial D_y}{\partial s_j} + \frac{\partial d_{iz}}{\partial D_z}\frac{\partial D_z}{\partial s_j} \end{bmatrix}, \tag{C.18}$$

where,

$$\frac{\partial d_{ix}}{\partial D_x} = \frac{D_y^2 + D_z^2}{\left(D_x^2 + D_y^2 + D_z^2\right)^{\frac{3}{2}}}, \tag{C.19}$$

$$\frac{\partial d_{iy}}{\partial D_y} = \frac{D_x^2 + D_z^2}{\left(D_x^2 + D_y^2 + D_z^2\right)^{\frac{3}{2}}}, \tag{C.20}$$

$$\frac{\partial d_{iz}}{\partial D_z} = \frac{D_x^2 + D_y^2}{\left(D_x^2 + D_y^2 + D_z^2\right)^{\frac{3}{2}}}, \tag{C.21}$$

$$\frac{\partial d_{ix}}{\partial D_y} = \frac{\partial d_{iy}}{\partial D_x} = -\frac{D_x D_y}{\left(D_x^2 + D_y^2 + D_z^2\right)^{\frac{3}{2}}}, \tag{C.22}$$

$$\frac{\partial d_{ix}}{\partial D_z} = \frac{\partial d_{iz}}{\partial D_x} = -\frac{D_x D_z}{\left(D_x^2 + D_y^2 + D_z^2\right)^{\frac{3}{2}}}, \tag{C.23}$$

$$\frac{\partial d_{iy}}{\partial D_z} = \frac{\partial d_{iz}}{\partial D_y} = -\frac{D_y D_z}{\left(D_x^2 + D_y^2 + D_z^2\right)^{\frac{3}{2}}}, \tag{C.24}$$

$$\frac{\partial \boldsymbol{D}}{\partial s_j} = \frac{\partial \boldsymbol{d}_{01}}{\partial s_j} \times \boldsymbol{d}_{02} + \boldsymbol{d}_{01} \times \frac{\partial \boldsymbol{d}_{02}}{\partial s_j}, \tag{C.25}$$

$$\frac{\partial \boldsymbol{d}_{01}}{\partial s_j} = \frac{\partial \boldsymbol{x}_1}{\partial s_j} - \frac{\partial \boldsymbol{x}_0}{\partial s_j}, \tag{C.26}$$

$$\frac{\partial \boldsymbol{d}_{02}}{\partial s_j} = \frac{\partial \boldsymbol{x}_2}{\partial s_j} - \frac{\partial \boldsymbol{x}_0}{\partial s_j}. \tag{C.27}$$

The derivative of nodal stress with respect to design variable s_j is given by

$$\frac{\partial \boldsymbol{\sigma}(\boldsymbol{x}_i)}{\partial s_j} = \frac{\partial \boldsymbol{\sigma}(\boldsymbol{x}_i)}{\partial \boldsymbol{x}_n}\frac{\partial \boldsymbol{x}_n}{\partial s_j} = \frac{1}{N_i}\frac{\partial\left(\sum \boldsymbol{E}_\sigma(\boldsymbol{x}_i)\hat{\boldsymbol{\sigma}}_e\right)}{\partial \boldsymbol{x}_n}\frac{\partial \boldsymbol{x}_n}{\partial s_j} = \frac{1}{N_i}\sum\left(\frac{\partial \boldsymbol{E}_\sigma(\boldsymbol{x}_i)}{\partial \boldsymbol{x}_n}\hat{\boldsymbol{\sigma}}_e + \boldsymbol{E}_\sigma(\boldsymbol{x}_i)\frac{\partial \hat{\boldsymbol{\sigma}}_e}{\partial \boldsymbol{x}_n}\right)\frac{\partial \boldsymbol{x}_n}{\partial s_j}. \tag{C.28}$$

The derivatives of $\boldsymbol{E}_\sigma$ with respect to $\boldsymbol{x}_n$ are given by

$$\frac{\partial \boldsymbol{E}_\sigma}{\partial \boldsymbol{x}_n} = \begin{bmatrix} \frac{\partial \boldsymbol{e}_\sigma}{\partial \boldsymbol{x}_n} & & & & & \\ & \frac{\partial \boldsymbol{e}_\sigma}{\partial \boldsymbol{x}_n} & & & & \\ & & \frac{\partial \boldsymbol{e}_\sigma}{\partial \boldsymbol{x}_n} & & & \\ & & & \frac{\partial \boldsymbol{e}_\sigma}{\partial \boldsymbol{x}_n} & & \\ & & & & \frac{\partial \boldsymbol{e}_\sigma}{\partial \boldsymbol{x}_n} & \\ & & & & & \frac{\partial \boldsymbol{e}_\sigma}{\partial \boldsymbol{x}_n} \end{bmatrix}, \tag{C.29}$$

where $\frac{\partial \boldsymbol{E}_\sigma}{\partial \boldsymbol{x}_n}$ represents $\frac{\partial \boldsymbol{E}_\sigma}{\partial x_n}$, $\frac{\partial \boldsymbol{E}_\sigma}{\partial y_n}$ and $\frac{\partial \boldsymbol{E}_\sigma}{\partial z_n}$; $\frac{\partial \boldsymbol{e}_\sigma}{\partial \boldsymbol{x}_n}$ represents $\frac{\partial \boldsymbol{e}_\sigma}{\partial x_n}$, $\frac{\partial \boldsymbol{e}_\sigma}{\partial y_n}$ and $\frac{\partial \boldsymbol{e}_\sigma}{\partial z_n}$. $\frac{\partial \boldsymbol{e}_\sigma}{\partial \boldsymbol{x}_n}$ are given by

$$\frac{\partial \boldsymbol{e}_\sigma}{\partial x_n} = \begin{bmatrix} 0 & 1 & 0 & 0 & y & 0 & z & 2x & 0 & 0 \end{bmatrix}, \tag{C.30}$$

$$\frac{\partial \boldsymbol{e}_\sigma}{\partial y_n} = \begin{bmatrix} 0 & 0 & 1 & 0 & x & z & 0 & 0 & 2y & 0 \end{bmatrix}, \tag{C.31}$$

$$\frac{\partial \boldsymbol{e}_\sigma}{\partial z_n} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & y & x & 0 & 0 & 2z \end{bmatrix}. \tag{C.32}$$

To find $\frac{\partial \hat{\boldsymbol{\sigma}}_e}{\partial \boldsymbol{x}_n}$, derivative of Equation 2.27 need to be taken. For simplicity, let

$$\boldsymbol{A}_e = \begin{bmatrix} \int_e \bar{\boldsymbol{E}}_\sigma^{\top} \boldsymbol{E}_\sigma \mathrm{d}e \\ \int_e \boldsymbol{E}_\zeta^{\top} \boldsymbol{\partial}_\sigma \boldsymbol{E}_\sigma \mathrm{d}e \end{bmatrix}, \ \boldsymbol{B}_{e1} = \int_e \bar{\boldsymbol{E}}_\sigma^{\top} \boldsymbol{\sigma}_e^h \mathrm{d}e, \ \boldsymbol{B}_{e2} = -\int_e \boldsymbol{E}_\zeta^{\top} \boldsymbol{b} \mathrm{d}e. \tag{C.33}$$

Therefore, the corresponding derivative of Equation 2.27 is expressed as

$$\left(\sum_{e \in \varepsilon_i} \frac{\partial \boldsymbol{A}_e}{\partial \boldsymbol{x}_n} \right) \hat{\boldsymbol{\sigma}}_e + \left(\sum_{e \in \varepsilon_i} \boldsymbol{A}_e \right) \frac{\partial \hat{\boldsymbol{\sigma}}_e}{\partial \boldsymbol{x}_n} = \sum_{e \in \varepsilon_i} \begin{bmatrix} \frac{\partial \boldsymbol{B}_{e1}}{\partial \boldsymbol{x}_n} \\ \frac{\partial \boldsymbol{B}_{e2}}{\partial \boldsymbol{x}_n} \end{bmatrix}. \tag{C.34}$$

According to Equation C.34, the derivative of $\hat{\boldsymbol{\sigma}}_e$ with respect to $\boldsymbol{x}_n$ is given by

$$\frac{\partial \hat{\boldsymbol{\sigma}}_e}{\partial \boldsymbol{x}_n} = \left(\sum_{e \in \varepsilon_i} \boldsymbol{A}_e \right)^{-1} \{ \sum_{e \in \varepsilon_i} \begin{bmatrix} \frac{\partial \boldsymbol{B}_{e1}}{\partial \boldsymbol{x}_n} \\ \frac{\partial \boldsymbol{B}_{e2}}{\partial \boldsymbol{x}_n} \end{bmatrix} - \left(\sum_{e \in \varepsilon_i} \frac{\partial \boldsymbol{A}_e}{\partial \boldsymbol{x}_n} \right) \hat{\boldsymbol{\sigma}}_e \}, \tag{C.35}$$

where $\sum_{e \in \varepsilon_i} \boldsymbol{A}_e$, $\boldsymbol{B}_{e1}$ and $\boldsymbol{B}_{e2}$ can be found by substituting Appendix B into the expressions.

The last term, the derivative of $\boldsymbol{x}_n$ with respect to the design variable s_j is found by

$$\frac{\partial \boldsymbol{x}_n}{\partial s_j} = \frac{\partial \boldsymbol{x}_n}{\partial \phi_l} \frac{\partial \phi_l}{\partial s_j} + \frac{\partial \boldsymbol{x}_n}{\partial \phi_m} \frac{\partial \phi_m}{\partial s_j}, \tag{C.36}$$

where, by taking derivative of Equation 2.15,

$$\frac{\partial \boldsymbol{x}_n}{\partial \phi_l} = -\frac{\phi_m}{(\phi_m - \phi_l)^2} (\boldsymbol{x}_m - \boldsymbol{x}_l), \tag{C.37}$$

and

$$\frac{\partial \boldsymbol{x}_n}{\partial \phi_m} = -\frac{\phi_l}{(\phi_m - \phi_l)^2} (\boldsymbol{x}_m - \boldsymbol{x}_l). \tag{C.38}$$

According to Equation 2.16,

$$\frac{\partial \phi_l}{\partial s_j} = \vartheta_j (\boldsymbol{x}_l), \tag{C.39}$$

and

$$\frac{\partial \phi_m}{\partial s_j} = \vartheta_j (\boldsymbol{x}_m). \tag{C.40}$$

C.2. $\frac{\partial J}{\partial \boldsymbol{U}}$

The derivative of J with respect to displacement $\boldsymbol{U}$ is

$$\frac{\partial J}{\partial \boldsymbol{U}} = \left(\frac{1}{N_{node}} \right)^{\frac{1}{p}} \frac{1}{p} \left(\sum_{i=1}^{N_{node}} G_i^p \right)^{\frac{1}{p}-1} \left(\sum_{i=1}^{N_{node}} p G_i^{p-1} \frac{\partial G_i}{\partial \boldsymbol{U}} \right), \tag{C.41}$$

where

$$\frac{\partial G_i}{\partial \boldsymbol{U}} = \frac{\pi\epsilon}{\mu\bar{E}}\left(\boldsymbol{M}_2^{\top}\boldsymbol{R}\,\frac{\partial \boldsymbol{\sigma}^{\top}}{\partial \boldsymbol{U}}\boldsymbol{R}^{\top}\boldsymbol{M}_1^{\top}\boldsymbol{H}'\boldsymbol{M}_1\boldsymbol{R}\,\boldsymbol{\sigma}\boldsymbol{R}^{\top}\boldsymbol{M}_2\right). \tag{C.42}$$

The derivative of nodal stress with respect to displacement $\boldsymbol{U}$ is

$$\frac{\partial \boldsymbol{\sigma}\,(\boldsymbol{x}_i)}{\partial \boldsymbol{U}} = \frac{1}{N_i}\frac{\partial\left(\sum \boldsymbol{E}_{\sigma}\,(\boldsymbol{x}_i)\,\hat{\boldsymbol{\sigma}}_e\right)}{\partial \boldsymbol{U}} = \boldsymbol{E}_{\sigma}\,(\boldsymbol{x}_i)\sum \frac{\partial \hat{\boldsymbol{\sigma}}_e}{\partial \boldsymbol{U}}, \tag{C.43}$$

where

$$\frac{\partial \hat{\boldsymbol{\sigma}}_e}{\partial \boldsymbol{x}_n} = \left(\sum_{e\in\varepsilon_i}\boldsymbol{A}_e\right)^{-1}\{\sum_{e\in\varepsilon_i}\begin{bmatrix}\frac{\partial \boldsymbol{B}_{e1}}{\partial \boldsymbol{U}}\\ \frac{\partial \boldsymbol{B}_{e2}}{\partial \boldsymbol{U}}\end{bmatrix} - \left(\sum_{e\in\varepsilon_i}\frac{\partial \boldsymbol{A}_e}{\partial \boldsymbol{U}}\right)\hat{\boldsymbol{\sigma}}_e\} = \left(\sum_{e\in\varepsilon_i}\boldsymbol{A}_e\right)^{-1}\sum_{e\in\varepsilon_i}\begin{bmatrix}\frac{\partial \boldsymbol{B}_{e1}}{\partial \boldsymbol{U}}\\ \frac{\partial \boldsymbol{B}_{e2}}{\partial \boldsymbol{U}}\end{bmatrix}. \tag{C.44}$$

The second therm $\frac{\partial \boldsymbol{A}_e}{\partial \boldsymbol{U}}$ is deleted because according to Equation C.33, $\boldsymbol{A}_e$ is not related to $\boldsymbol{U}$.

C.3. $\frac{\partial \boldsymbol{K}}{\partial s_j}$

According to Equation 2.8, the derivative of element stiffness $\boldsymbol{k}_e$ with respect to $\boldsymbol{x}_n$ is

$$\frac{\partial \boldsymbol{k}_e}{\partial \boldsymbol{x}_n} = \int_e\left(\frac{\partial j_e}{\partial \boldsymbol{x}_n}\boldsymbol{B}^{\top}\boldsymbol{D}\boldsymbol{B} + j_e\frac{\partial \boldsymbol{B}^{\top}}{\partial \boldsymbol{x}_n}\boldsymbol{D}\boldsymbol{B} + j_e\boldsymbol{B}^{\top}\boldsymbol{D}\frac{\partial \boldsymbol{B}}{\partial \boldsymbol{x}_n}\right)\mathrm{d}\boldsymbol{\xi}, \tag{C.45}$$

where

$$\frac{\partial \boldsymbol{B}}{\partial \boldsymbol{x}_n} = \left[\frac{\partial \Delta_{\boldsymbol{\xi}}^{\top}\boldsymbol{N}^{\top}}{\partial \boldsymbol{x}_n}\boldsymbol{J}^{-T} + \Delta_{\boldsymbol{\xi}}^{\top}\boldsymbol{N}^{\top}\frac{\partial \boldsymbol{J}^{-T}}{\partial \boldsymbol{x}_n} \quad \frac{\partial \Delta_{\boldsymbol{\xi}}^{\top}\boldsymbol{\psi}^{\top}}{\partial \boldsymbol{x}_n}\boldsymbol{J}_e^{-T} + \Delta_{\boldsymbol{\xi}}^{\top}\boldsymbol{\psi}^{\top}\frac{\partial \boldsymbol{J}_e^{-T}}{\partial \boldsymbol{x}_n}\right] = \left[0 \quad \Delta_{\boldsymbol{\xi}}^{\top}\boldsymbol{\psi}^{\top}\frac{\partial \boldsymbol{J}_e^{-T}}{\partial \boldsymbol{x}_n}\right]. \tag{C.46}$$

The first element is eliminated because the derivative of the shape function derivatives of linear tetrahedral element $\frac{\partial \Delta_{\boldsymbol{\xi}}^{\top}\boldsymbol{N}^{\top}}{\partial \boldsymbol{\xi}_p} = 0$, which means $\frac{\partial \Delta_{\boldsymbol{\xi}}^{\top}\boldsymbol{N}^{\top}}{\partial \boldsymbol{x}_n} = \frac{\partial \Delta_{\boldsymbol{\xi}}^{\top}\boldsymbol{N}^{\top}}{\partial \boldsymbol{\xi}_p}\frac{\partial \boldsymbol{\xi}_p}{\partial \boldsymbol{x}}\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{x}_n} = 0$, and the Jacobian of the parent element is not related to the coordinates of enriched nodes so $\frac{\partial \boldsymbol{J}}{\partial \boldsymbol{x}_n} = 0$. $\boldsymbol{\xi}_p$ represents the natural coordinates of the mapped parent element. The first term of the second element is also eliminated because the derivative of the enrichment derivatives is also zero $\left(\frac{\partial \Delta_{\boldsymbol{\xi}}^{\top}\boldsymbol{\psi}^{\top}}{\partial \boldsymbol{\xi}_p} = 0\right)$ and $\frac{\partial \Delta_{\boldsymbol{\xi}}^{\top}\boldsymbol{\psi}^{\top}}{\partial \boldsymbol{x}_n} = \frac{\partial \Delta_{\boldsymbol{\xi}}^{\top}\boldsymbol{\psi}^{\top}}{\partial \boldsymbol{\xi}_e}\frac{\partial \boldsymbol{\xi}_e}{\partial \boldsymbol{x}}\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{x}_n} = 0$, where $\boldsymbol{\xi}_e$ represents the natural coordinates of the mapped integration element.

In Equation C.45 and Equation C.46, the the derivative of the determinant and inverse of the Jacobian matrix of the integration elements are needed. According to Magnus and Neudecker (2019), the derivative of a determinant of a Jacobian matrix can be found by the trace of the multiplication of the Jacobian's adjugate $\left(\mathrm{adj}\,(\boldsymbol{J}_e) = j_e\boldsymbol{J}_e^{-T}\right)$ and the derivative of the Jacobian, which is

$$\frac{\partial j_e}{\partial \boldsymbol{x}_n} = \mathrm{tr}\left(\mathrm{adj}\,(\boldsymbol{J}_e)\,\frac{\partial \boldsymbol{J}_e}{\partial \boldsymbol{x}_e}\right) \tag{C.47}$$

The derivative of the inverse Jacobian can be found by taking the derivative of $\boldsymbol{J}_e\boldsymbol{J}_e^{-1} = \boldsymbol{I}$:

$$\frac{\partial \boldsymbol{J}_e\boldsymbol{J}_e^{-1}}{\partial \boldsymbol{x}_n} = \frac{\partial \boldsymbol{J}_e}{\partial \boldsymbol{x}_n}\boldsymbol{J}_e^{-1} + \boldsymbol{J}_e\frac{\partial \boldsymbol{J}_e^{-1}}{\partial \boldsymbol{x}_n} = \frac{\partial \boldsymbol{I}}{\partial \boldsymbol{x}_n} = 0. \tag{C.48}$$

Therefore, the derivative of the inverse Jacobian can also be expressed in terms of derivative of the Jacobian:

$$\frac{\partial \boldsymbol{J}_e^{-1}}{\partial \boldsymbol{x}_n} = -\boldsymbol{J}_e^{-1}\frac{\partial \boldsymbol{J}_e}{\partial \boldsymbol{x}_n}\boldsymbol{J}_e^{-1}, \tag{C.49}$$

where $\frac{\partial \boldsymbol{J}_e}{\partial \boldsymbol{x}_n}$ can be found in Appendix D.

C.4. $\frac{\partial \boldsymbol{f}_e}{\partial \boldsymbol{x}_n}$

Taking derivative of Equation 2.9 with respect to $\boldsymbol{x}_n$:

$$\frac{\partial \boldsymbol{f}_e}{\partial \boldsymbol{x}_n} = \int_e \left(\begin{bmatrix} \frac{\partial \boldsymbol{N}}{\partial \boldsymbol{x}_n} \\ \boldsymbol{0} \end{bmatrix} j_e \boldsymbol{b} + \begin{bmatrix} \boldsymbol{N} \\ \boldsymbol{\psi} \end{bmatrix} \frac{\partial j_e}{\partial \boldsymbol{x}_n} \boldsymbol{b} \right) \mathrm{d}\boldsymbol{\xi} + \int_{\partial e \cap \Gamma^{\mathrm{T}}} \left(\begin{bmatrix} \frac{\partial \boldsymbol{N}}{\partial \boldsymbol{x}_n} \\ \boldsymbol{0} \end{bmatrix} j_e \bar{\boldsymbol{t}} + \begin{bmatrix} \boldsymbol{N} \\ \boldsymbol{\psi} \end{bmatrix} \frac{\partial j_e}{\partial \boldsymbol{x}_n} \bar{\boldsymbol{t}} \right) \mathrm{d}\partial \boldsymbol{\xi}. \tag{C.50}$$

Because body force is neglected in this work, so further derivation of the first term will not be given. $\frac{\partial \boldsymbol{N}}{\partial \boldsymbol{x}_n}$ is given in Appendix D; $\frac{\partial \boldsymbol{\psi}}{\partial \boldsymbol{x}_n} = 0$ because the enrichment functions in reference coordinates are not affected by the position of the enriched nodes. The derivative of determinant of Jacabian is given in Appendix E.

D

Isoparametric Mapping of integration Elements and Normal Jacobian

In FEM, the functions evaluated on integration elements are done by numerical methods, so isoparametric mapping needs to be applied to map the integration element to the natural coordinates. The global coordinate $\boldsymbol{x}$ can be expressed with and natural coordinates of the integration and parent elements:

$$\boldsymbol{x} = \boldsymbol{x}_e^{\mathsf{T}} \boldsymbol{N}_e(\boldsymbol{\xi}_e) = \boldsymbol{x}_p^{\mathsf{T}} \boldsymbol{N}(\boldsymbol{\xi}_p), \tag{D.1}$$

where $\boldsymbol{N}_e$ and $\boldsymbol{N}$ are matrices consist of the linear Lagrangian shape functions of the integration and parent elements separately; $\boldsymbol{x}_e$ and $\boldsymbol{x}_p$ are the global coordinates of the nodes in integration and global elements.

The Jacobians of both both mappings are

$$\boldsymbol{J}_e = \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\xi_e}} = \boldsymbol{x}_e^{\mathsf{T}} \frac{\partial \boldsymbol{N}_e}{\partial \boldsymbol{\xi}_e} \tag{D.2}$$

and

$$\boldsymbol{J} = \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\xi}_p} = \boldsymbol{x}_p^{\mathsf{T}} \frac{\partial \boldsymbol{N}}{\partial \boldsymbol{\xi}_p}, \tag{D.3}$$

respectively.

The derivative of $\boldsymbol{J}_e$ with respect to $\boldsymbol{x}_n$ is

$$\frac{\partial \boldsymbol{J}_e}{\partial \boldsymbol{x}_n} = \frac{\partial \boldsymbol{x}_e^{\mathsf{T}}}{\partial \boldsymbol{x}_n} \frac{\partial \boldsymbol{N}_e}{\partial \boldsymbol{\xi}_e} + \boldsymbol{x}_e^{\mathsf{T}} \frac{\partial^2 \boldsymbol{N}_e}{\partial \boldsymbol{\xi}_e \partial \boldsymbol{x}_n} = \frac{\partial \boldsymbol{x}_e^{\mathsf{T}}}{\partial \boldsymbol{x}_n} \frac{\partial \boldsymbol{N}_e}{\partial \boldsymbol{\xi}_e}, \tag{D.4}$$

where $\frac{\partial \boldsymbol{x}_e}{\partial \boldsymbol{x}_n}$ is a matrix with one at the elements of enriched node n.

In Equation C.50, $\frac{\partial \boldsymbol{N}}{\partial \boldsymbol{x}_n}$ is required and $\frac{\partial \boldsymbol{N}}{\partial \boldsymbol{x}_n}$ can be further split by chain rule:

$$\frac{\partial \boldsymbol{N}}{\partial \boldsymbol{x}_n} = \frac{\partial \boldsymbol{N}}{\partial \boldsymbol{\xi}_p} \frac{\partial \boldsymbol{\xi}_p}{\partial \boldsymbol{x}} \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{x}_n}, \tag{D.5}$$

where $\frac{\partial \boldsymbol{\xi}_p}{\partial \boldsymbol{x}}$ can be found by the inverse mapping relationship in Equation D.1. For a parent tetrahedral element, Equation D.1 can be expanded as

$$\begin{aligned}\boldsymbol{x} &= \begin{bmatrix} x_{j,1} & x_{k,1} & x_{l,1} & x_{m,1} \\ x_{j,2} & x_{k,2} & x_{l,2} & x_{m,2} \\ x_{j,3} & x_{k,3} & x_{l,3} & x_{m,3} \end{bmatrix} \begin{bmatrix} 1-\xi_1-\xi_2-\xi_3 \\ \xi_1 \\ \xi_2 \\ \xi_3 \end{bmatrix} \\ &= \begin{bmatrix} x_{j,1}+\xi_1(x_{k,1}-x_{j,1})+\xi_2(x_{l,1}-x_{j,1})+\xi_3(x_{m,1}-x_{j,1}) \\ x_{j,2}+\xi_1(x_{k,2}-x_{j,2})+\xi_2(x_{l,2}-x_{j,2})+\xi_3(x_{m,2}-x_{j,2}) \\ x_{j,3}+\xi_1(x_{k,3}-x_{j,3})+\xi_2(x_{l,3}-x_{j,3})+\xi_3(x_{m,3}-x_{j,3}) \end{bmatrix} \\ &= \begin{bmatrix} x_{j,1} \\ x_{j,2} \\ x_{j,3} \end{bmatrix} + \underbrace{\begin{bmatrix} x_{k,1}-x_{j,1} & x_{l,1}-x_{j,1} & x_{m,1}-x_{j,1} \\ x_{k,2}-x_{j,2} & x_{l,2}-x_{j,2} & x_{m,2}-x_{j,2} \\ x_{k,3}-x_{j,3} & x_{l,3}-x_{j,3} & x_{m,3}-x_{j,3} \end{bmatrix}}_{\boldsymbol{M}} \begin{bmatrix} \xi_1 \\ \xi_2 \\ \xi_3 \end{bmatrix}. \end{aligned} \tag{D.6}$$

Therefore, the natural coordinates $\boldsymbol{\xi}_p$ in parent element can be expressed by

$$\boldsymbol{\xi}_p = \boldsymbol{M}^{-1}\boldsymbol{x} - \boldsymbol{M}^{-1} \begin{bmatrix} x_{j,1} \\ x_{j,2} \\ x_{j,3} \end{bmatrix}. \tag{D.7}$$

Then, Equation D.5 can further be derived:

$$\frac{\partial \boldsymbol{N}}{\partial \boldsymbol{x}_n} = \frac{\partial \boldsymbol{N}}{\partial \boldsymbol{\xi}_p} \boldsymbol{M}^{-1} \frac{\partial \boldsymbol{x}_e}{\partial \boldsymbol{x}_n} \boldsymbol{N}_e. \tag{D.8}$$

E

Devloo’s Jacobian and its derivative

As has been explained in Section 2.1.1, a surface element is needed to integrate Equation 2.9. However, the Jacobian matrix of the element is a 3×2 matrix

$$\boldsymbol{J}_e = \boldsymbol{x}_e^{\mathrm{T}} \frac{\partial \boldsymbol{N}_e}{\partial \boldsymbol{\xi}_e} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial x}{\partial \eta} \\ \frac{\partial y}{\partial \xi} & \frac{\partial y}{\partial \eta} \\ \frac{\partial z}{\partial \xi} & \frac{\partial z}{\partial \eta} \end{bmatrix}, \tag{E.1}$$

which does not have a determinant nor inverse matrix.

Devloo (1997) provided a method for solving the issue:

For triangular elements in 3D spaces, defines:

$$J_\xi = \sqrt{(\frac{\partial x}{\partial \xi})^2 + (\frac{\partial y}{\partial \xi})^2 + (\frac{\partial z}{\partial \xi})^2}, \tag{E.2}$$

$$J_\eta = \sqrt{(\frac{\partial x}{\partial \eta})^2 + (\frac{\partial y}{\partial \eta})^2 + (\frac{\partial z}{\partial \eta})^2}, \tag{E.3}$$

$$\boldsymbol{V}_1 = \frac{1}{J_\xi}(\frac{\partial x}{\partial \xi}, \frac{\partial y}{\partial \xi}, \frac{\partial z}{\partial \xi}), \tag{E.4}$$

$$\tilde{\boldsymbol{V}}_2 = \frac{1}{J_\eta}(\frac{\partial x}{\partial \eta}, \frac{\partial y}{\partial \eta}, \frac{\partial z}{\partial \eta}), \tag{E.5}$$

$$\boldsymbol{V}_2 = \frac{\tilde{\boldsymbol{V}}_2 - (\tilde{\boldsymbol{V}}_2 \cdot \boldsymbol{V}_1)\boldsymbol{V}_1}{||\tilde{\boldsymbol{V}}_2 - (\tilde{\boldsymbol{V}}_2 \cdot \boldsymbol{V}_1)\boldsymbol{V}_1||}, \tag{E.6}$$

$$\boldsymbol{V}_3 = \boldsymbol{V}_1 \times \boldsymbol{V}_2. \tag{E.7}$$

With the above definitions, the Jacobian matrix is transformed to

$$\boldsymbol{J}_e = \begin{bmatrix} J_\xi & \boldsymbol{V}_1 \cdot \tilde{\boldsymbol{V}}_2 J_\eta \\ 0 & \boldsymbol{V}_2 \cdot \tilde{\boldsymbol{V}}_2 J_\eta \end{bmatrix}. \tag{E.8}$$

Therefore, the Jacobian number is $j_e = det(\boldsymbol{J}_e) = \boldsymbol{V}_2 \cdot \tilde{\boldsymbol{V}}_2 J_\xi J_\eta$. Its derivative is calculated as follows:

$$\frac{\partial j_e}{\partial \boldsymbol{x}_n} = \frac{\partial \boldsymbol{V}_2}{\partial \boldsymbol{x}_n} \tilde{\boldsymbol{V}}_2 J_\xi J_\eta + \boldsymbol{V}_2 \frac{\partial \tilde{\boldsymbol{V}}_2}{\partial \boldsymbol{x}_n} J_\xi J_\eta + \boldsymbol{V}_2 \tilde{\boldsymbol{V}}_2 \frac{\partial J_\xi}{\partial \boldsymbol{x}_n} J_\eta + \boldsymbol{V}_2 \tilde{\boldsymbol{V}}_2 J_\xi \frac{\partial J_\eta}{\partial \boldsymbol{x}_n}, \tag{E.9}$$

where

$$\frac{\partial J_\xi}{\partial \boldsymbol{x}_n} = \frac{1}{J_\xi}[\frac{\partial x}{\partial \xi} \frac{\partial}{\partial \boldsymbol{x}_n}(\frac{\partial x}{\partial \xi}) + \frac{\partial y}{\partial \xi} \frac{\partial}{\partial \boldsymbol{x}_n}(\frac{\partial y}{\partial \xi}) + \frac{\partial z}{\partial \xi} \frac{\partial}{\partial \boldsymbol{x}_n}(\frac{\partial z}{\partial \xi})], \tag{E.10}$$

$$\frac{\partial J_\eta}{\partial \boldsymbol{x}_n} = \frac{1}{J_\eta}[\frac{\partial x}{\partial \eta}\frac{\partial}{\partial \boldsymbol{x}_n}(\frac{\partial x}{\partial \eta}) + \frac{\partial y}{\partial \eta}\frac{\partial}{\partial \boldsymbol{x}_n}(\frac{\partial y}{\partial \eta}) + \frac{\partial z}{\partial \eta}\frac{\partial}{\partial \boldsymbol{x}_n}(\frac{\partial z}{\partial \eta})], \tag{E.11}$$

$$\frac{\partial \tilde{\boldsymbol{V}}_2}{\partial \boldsymbol{x}_n} = -\frac{1}{J_\eta^2}\frac{\partial J_\eta}{\partial \boldsymbol{x}_n}(\frac{\partial x}{\partial \eta}, \frac{\partial y}{\partial \eta}, \frac{\partial z}{\partial \eta}) + J_\xi[\frac{\partial}{\partial \boldsymbol{x}_n}(\frac{\partial x}{\partial \eta}) + \frac{\partial}{\partial \boldsymbol{x}_n}(\frac{\partial y}{\partial \eta}) + \frac{\partial}{\partial \boldsymbol{x}_n}(\frac{\partial z}{\partial \eta})], \tag{E.12}$$

$$\frac{\partial \boldsymbol{V}_2}{\partial \boldsymbol{x}_n} = (\frac{\boldsymbol{I} - \boldsymbol{V}_2 \cdot \boldsymbol{V}_2^\intercal}{||\tilde{\boldsymbol{V}}_2 - (\tilde{\boldsymbol{V}}_2 \cdot \boldsymbol{V}_1)\boldsymbol{V}_1||})[\frac{\partial \tilde{\boldsymbol{V}}_2}{\partial \boldsymbol{x}_n} - (\frac{\partial \tilde{\boldsymbol{V}}_2}{\partial \boldsymbol{x}_n} \cdot \boldsymbol{V}_1 + \tilde{\boldsymbol{V}}_2 \cdot \frac{\partial \boldsymbol{V}_1}{\partial \boldsymbol{x}_n})\boldsymbol{V}_1 - (\tilde{\boldsymbol{V}}_2 \cdot \boldsymbol{V}_1)\frac{\partial \boldsymbol{V}_1}{\partial \boldsymbol{x}_n}], \tag{E.13}$$

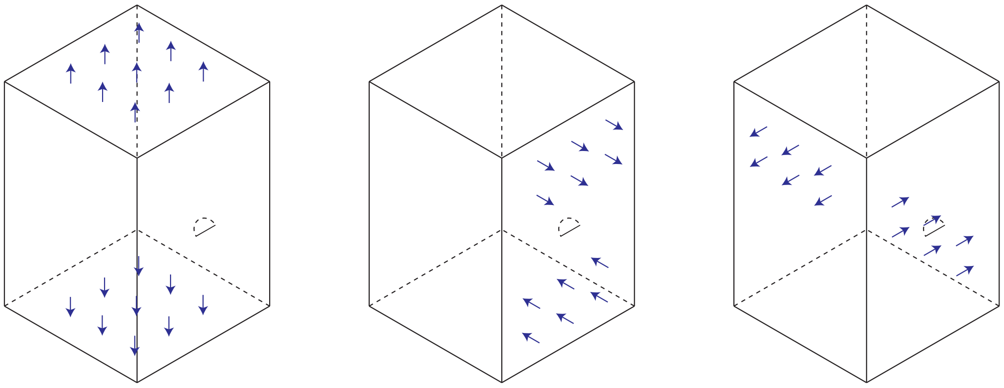
$$\frac{\partial \boldsymbol{V}_1}{\partial \boldsymbol{x}_n} = -\frac{1}{J_\xi^2}\frac{\partial J_\xi}{\partial \boldsymbol{x}_n}(\frac{\partial x}{\partial \xi}, \frac{\partial y}{\partial \xi}, \frac{\partial z}{\partial \xi}) + J_\xi[\frac{\partial}{\partial \boldsymbol{x}_n}(\frac{\partial x}{\partial \xi}) + \frac{\partial}{\partial \boldsymbol{x}_n}(\frac{\partial y}{\partial \xi}) + \frac{\partial}{\partial \boldsymbol{x}_n}(\frac{\partial z}{\partial \xi})], \tag{E.14}$$

$$\boldsymbol{I} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}. \tag{E.15}$$

F

Weight Function Evaluation

In order to evaluate the weight functions a cuboid with a crack is modeled as shown in figure Figure F.1. Three types of load (mode-I, II, and III) are applied to find the SIFs of the crack. An intact cuboid with the same dimension and boundary conditions is also modeled to find the stress field. Weight functions that relate the SIFs and the stress field can then be found.



(a) Mode I loading

(b) Mode II loading

(c) Mode III loading

Figure F.1: Test structure under mode-I, II, III loading

The domain integral method is used to evaluate the energy release rates and the M_1 integral method (or ’interaction integral’) is used to extract SIFs (Nakamura & Parks, 1989)(Nagai et al., 2013)(Zhang et al., 2019). An integration domain $V(s)$ for evaluating the fracture parameters at the crack front position s is illustrated in Figure F.2. The domain has length 2η and radius r, and is enclosed by circular surface at $s+\eta$ and $s-\eta$.

The local interaction energy release rate defined in terms of M_1 integral:

$$J(s) = \frac{\int_V (\sigma_{ij}^{(1)} \frac{\partial u_j^{(2)}}{\partial x_1} + \sigma_{ij}^{(2)} \frac{\partial u_j^{(1)}}{\partial x_1} - \sigma_{j1}^{(1)} \epsilon_{j1}^{(2)} \delta_{1i}) \cdot \frac{\partial q_1}{\partial x_i} \mathrm{d}V}{\int_{s-\eta}^{s+\eta} \mu(s) \mathrm{d}s}, \tag{F.1}$$

where σ, ϵ, and u represents stress, strain, and displacement separately; q_1 is a test function continuous in $V(s)$ and has zero value outside; μ is a function represents the virtual crack advance at point s as illustrated in Figure F.3; superscript (1) indicates actual field and (2) indicates auxiliary field.

The local interaction energy release rate can also be expressed in terms of the SIFs:

$$J(s) = \frac{1-\nu^2}{E}(2K_I^{(1)}K_I^{(2)} + 2K_{II}^{(1)}K_{II}^{(2)}) + \frac{1+\nu}{E}(2K_{III}^{(1)}K_{III}^{(2)}). \tag{F.2}$$

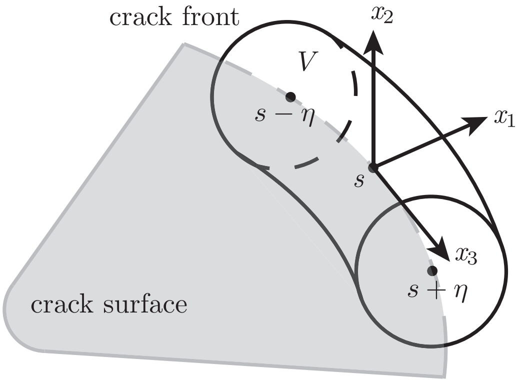


Figure F.2: Integration domain at the crack front

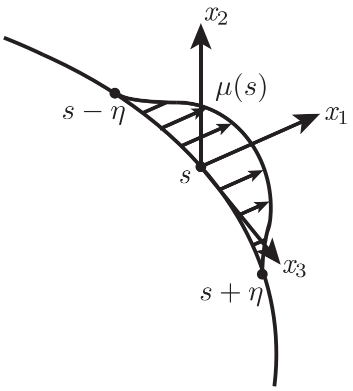


Figure F.3: Virtual crack advance function μ

By setting the SIFs of the auxiliary fields as (a) $[K_I^{(2)}=1,\ K_{II}^{(2)}=0,\ K_{III}^{(2)}=0]$, (b) $[K_I^{(2)}=0,\ K_{II}^{(2)}=1,\ K_{III}^{(2)}=0]$, and (c) $[K_I^{(2)}=0,\ K_{II}^{(2)}=0,\ K_{III}^{(2)}=1]$, the SIFs can be found by

$$\begin{cases} 2\frac{1-\nu^2}{E}K_I^{(1)}=J_a, \\ 2\frac{1-\nu^2}{E}K_{II}^{(1)}=J_b, \\ 2\frac{1+\nu}{E}K_{III}^{(1)}=J_c. \end{cases} \tag{F.3}$$

Under three different load modes, three sets of SIFs and stress fields can be found. The weight functions in $\boldsymbol{H}$ can then be solve by

$$\frac{1}{\sqrt{\pi\epsilon}}\begin{bmatrix} K_I^{(i)} & K_I^{(ii)} & K_I^{(iii)} \\ K_{II}^{(i)} & K_{II}^{(ii)} & K_{II}^{(iii)} \\ K_{III}^{(i)} & K_{III}^{(ii)} & K_{III}^{(iii)} \end{bmatrix}=\boldsymbol{H}\begin{bmatrix} \sigma_{\theta\theta}^{(i)} & \sigma_{\theta\theta}^{(ii)} & \sigma_{\theta\theta}^{(iii)} \\ \sigma_{r\theta}^{(i)} & \sigma_{r\theta}^{(ii)} & \sigma_{r\theta}^{(iii)} \\ \sigma_{z\theta}^{(i)} & \sigma_{z\theta}^{(ii)} & \sigma_{z\theta}^{(iii)} \end{bmatrix}. \tag{F.4}$$

G

Sensitivity for Extended Formulation

G.1. $\frac{\partial G_i}{\partial s_j}$

After adding the stress transformation matrix for the surface 1st principal stress, the sensitivity of ERR is extended to

$$\begin{aligned}\frac{\partial G_i}{\partial s_j} = \frac{\pi\epsilon}{\mu\bar{E}}\Bigg(&\boldsymbol{M}_2^{\mathsf{T}}\frac{\partial \boldsymbol{R}_t}{\partial s_j}\boldsymbol{R}\,\boldsymbol{\sigma}^{\mathsf{T}}\boldsymbol{R}^{\mathsf{T}}\boldsymbol{R}_t^{\mathsf{T}}\boldsymbol{M}_1^{\mathsf{T}}\boldsymbol{H'}^{\mathsf{T}}\boldsymbol{H'}\boldsymbol{M}_1\boldsymbol{R}_t\boldsymbol{R}\,\boldsymbol{\sigma}\boldsymbol{R}^{\mathsf{T}}\boldsymbol{R}_t^{\mathsf{T}}\boldsymbol{M}_2\\ &+\boldsymbol{M}_2^{\mathsf{T}}\boldsymbol{R}_t\frac{\partial \boldsymbol{R}}{\partial s_j}\,\boldsymbol{\sigma}^{\mathsf{T}}\boldsymbol{R}^{\mathsf{T}}\boldsymbol{R}_t^{\mathsf{T}}\boldsymbol{M}_1^{\mathsf{T}}\boldsymbol{H'}^{\mathsf{T}}\boldsymbol{H'}\boldsymbol{M}_1\boldsymbol{R}_t\boldsymbol{R}\,\boldsymbol{\sigma}\boldsymbol{R}^{\mathsf{T}}\boldsymbol{R}_t^{\mathsf{T}}\boldsymbol{M}_2\\ &+\boldsymbol{M}_2^{\mathsf{T}}\boldsymbol{R}_t\boldsymbol{R}\,\frac{\partial \boldsymbol{\sigma}^{\mathsf{T}}}{\partial s_j}\boldsymbol{R}^{\mathsf{T}}\boldsymbol{R}_t^{\mathsf{T}}\boldsymbol{M}_1^{\mathsf{T}}\boldsymbol{H'}^{\mathsf{T}}\boldsymbol{H'}\boldsymbol{M}_1\boldsymbol{R}_t\boldsymbol{R}\,\boldsymbol{\sigma}\boldsymbol{R}^{\mathsf{T}}\boldsymbol{R}_t^{\mathsf{T}}\boldsymbol{M}_2\\ &+\boldsymbol{M}_2^{\mathsf{T}}\boldsymbol{R}_t\boldsymbol{R}\,\boldsymbol{\sigma}^{\mathsf{T}}\frac{\partial \boldsymbol{R}^{\mathsf{T}}}{\partial s_j}\boldsymbol{R}_t\boldsymbol{M}_1^{\mathsf{T}}\boldsymbol{H'}^{\mathsf{T}}\boldsymbol{H'}\boldsymbol{M}_1\boldsymbol{R}_t\boldsymbol{R}\,\boldsymbol{\sigma}\boldsymbol{R}^{\mathsf{T}}\boldsymbol{R}_t^{\mathsf{T}}\boldsymbol{M}_2\\ &+\boldsymbol{M}_2^{\mathsf{T}}\boldsymbol{R}_t\boldsymbol{R}\,\boldsymbol{\sigma}^{\mathsf{T}}\boldsymbol{R}^{\mathsf{T}}\frac{\partial \boldsymbol{R}_t^{\mathsf{T}}}{\partial s_j}\boldsymbol{M}_1^{\mathsf{T}}\boldsymbol{H'}^{\mathsf{T}}\boldsymbol{H'}\boldsymbol{M}_1\boldsymbol{R}_t\boldsymbol{R}\,\boldsymbol{\sigma}\boldsymbol{R}^{\mathsf{T}}\boldsymbol{R}_t^{\mathsf{T}}\boldsymbol{M}_2\Bigg),\end{aligned}\tag{G.1}$$

where

$$\begin{aligned}\frac{\partial \boldsymbol{R}_t}{\partial s_j} &= \frac{\partial \boldsymbol{R}_t}{\partial \theta_p}\left(\frac{\partial \theta_p}{\partial \sigma_{y'z'}}\frac{\partial \sigma_{y'z'}}{\partial s_j}+\frac{\partial \theta_p}{\partial \sigma_{y'y'}}\frac{\partial \sigma_{y'y'}}{\partial s_j}+\frac{\partial \theta_p}{\partial \sigma_{z'z'}}\frac{\partial \sigma_{z'z'}}{\partial s_j}\right)\\ &= \frac{\partial \boldsymbol{R}_t}{\partial \theta_p}\left(\begin{bmatrix}\frac{\partial \theta_p}{\partial \sigma_{y'z'}} & \frac{\partial \theta_p}{\partial \sigma_{y'y'}} & \frac{\partial \theta_p}{\partial \sigma_{z'z'}}\end{bmatrix}\begin{bmatrix}\frac{\partial \sigma_{y'z'}}{\partial s_j}\\ \frac{\partial \sigma_{y'y'}}{\partial s_j}\\ \frac{\partial \sigma_{z'z'}}{\partial s_j}\end{bmatrix}\right).\end{aligned}\tag{G.2}$$

In the above equation,

$$\frac{\partial \boldsymbol{R}_t}{\partial \theta_p} = \begin{bmatrix}0 & 0 & 0\\ 0 & -\sin\left(\theta_p\right) & -\cos\left(\theta_p\right)\\ 0 & \cos\left(\theta_p\right) & -\sin\left(\theta_p\right)\end{bmatrix},\tag{G.3}$$

$$\frac{\partial \theta_p}{\partial \sigma_{y'z'}} = \mp\frac{\sigma_{z'z'}-\sigma_{y'y'}}{\left(\sigma_{z'z'}-\sigma_{y'y'}\right)^2+4\sigma_{y'z'}^2},\tag{G.4}$$

$$\frac{\partial \theta_p}{\partial \sigma_{y'y'}} = \mp\frac{\sigma_{y'z'}}{\left(\sigma_{z'z'}-\sigma_{y'y'}\right)^2+4\sigma_{y'z'}^2},\tag{G.5}$$

$$\frac{\partial \theta_p}{\partial \sigma_{y'y'}} = \pm\frac{\sigma_{y'z'}}{\left(\sigma_{z'z'}-\sigma_{y'y'}\right)^2+4\sigma_{y'z'}^2}.\tag{G.6}$$

For the derivative of the local stresses in Equation G.2, first consider the derivation of the stresses from the global stresses in an algebraic manner:

$$\begin{bmatrix}\sigma_{y'z'}\\ \sigma_{y'y'}\\ \sigma_{z'z'}\end{bmatrix} = \underbrace{\begin{bmatrix}0&0&1\\0&1&0\\0&0&0\end{bmatrix}}_{\boldsymbol{M}_3}\boldsymbol{R\sigma R}^\top \underbrace{\begin{bmatrix}0\\1\\0\end{bmatrix}}_{\boldsymbol{M}_2} + \underbrace{\begin{bmatrix}0&0&0\\0&0&0\\0&0&1\end{bmatrix}}_{\boldsymbol{M}_4}\boldsymbol{R\sigma R}^\top \underbrace{\begin{bmatrix}0\\0\\1\end{bmatrix}}_{\boldsymbol{M}_5}. \tag{G.7}$$

Therefore, the derivatives of the local stresses are evaluated by

$$\begin{aligned}\begin{bmatrix}\frac{\partial\sigma_{y'z'}}{\partial s_j}\\ \frac{\partial\sigma_{y'y'}}{\partial s_j}\\ \frac{\partial\sigma_{z'z'}}{\partial s_j}\end{bmatrix} =& \boldsymbol{M}_3\frac{\partial\boldsymbol{R}}{\partial s_j}\boldsymbol{\sigma R}^\top\boldsymbol{M}_2 + \boldsymbol{M}_3\boldsymbol{R}\frac{\partial\boldsymbol{\sigma}}{\partial s_j}\boldsymbol{R}^\top\boldsymbol{M}_2 + \boldsymbol{M}_3\boldsymbol{R\sigma}\frac{\partial\boldsymbol{R}^\top}{\partial s_j}\boldsymbol{M}_2\\ &+\boldsymbol{M}_4\frac{\partial\boldsymbol{R}}{\partial s_j}\boldsymbol{\sigma R}^\top\boldsymbol{M}_5 + \boldsymbol{M}_4\boldsymbol{R}\frac{\partial\boldsymbol{\sigma}}{\partial s_j}\boldsymbol{R}^\top\boldsymbol{M}_5 + \boldsymbol{M}_4\boldsymbol{R\sigma}\frac{\partial\boldsymbol{R}^\top}{\partial s_j}\boldsymbol{M}_5.\end{aligned} \tag{G.8}$$

G.2. $\frac{\partial G_i}{\partial U}$

By adding the stress transformation matrix $\boldsymbol{R}_t$, Equation C.42 is extended to

$$\begin{aligned}\frac{\partial G_i}{\partial s_j} = \frac{\pi\epsilon}{\mu\bar{E}}\Bigg(&\boldsymbol{M}_2^\top\frac{\partial\boldsymbol{R}_t}{\partial\boldsymbol{U}}\boldsymbol{R\,\sigma}^\top\boldsymbol{R}^\top\boldsymbol{R}_t^\top\boldsymbol{M}_1^\top\boldsymbol{H}'\boldsymbol{M}_1\boldsymbol{R}_t\boldsymbol{R\,\sigma R}^\top\boldsymbol{R}_t^\top\boldsymbol{M}_2\\ &+\boldsymbol{M}_2^\top\boldsymbol{R}_t\boldsymbol{R}\,\frac{\partial\boldsymbol{\sigma}^\top}{\partial\boldsymbol{U}}\boldsymbol{R}^\top\boldsymbol{R}_t^\top\boldsymbol{M}_1^\top\boldsymbol{H}'\boldsymbol{M}_1\boldsymbol{R}_t\boldsymbol{R\,\sigma R}^\top\boldsymbol{R}_t^\top\boldsymbol{M}_2\\ &+\boldsymbol{M}_2^\top\boldsymbol{R}_t\boldsymbol{UR\,\sigma}^\top\boldsymbol{R}^\top\frac{\partial\boldsymbol{R}_t^\top}{\partial\boldsymbol{U}}\boldsymbol{M}_1^\top\boldsymbol{H}'\boldsymbol{M}_1\boldsymbol{R}_t\boldsymbol{R\,\sigma R}^\top\boldsymbol{R}_t^\top\boldsymbol{M}_2\Bigg),\end{aligned} \tag{G.9}$$

where

$$\frac{\partial\boldsymbol{R}_t}{\partial U} = \frac{\partial\boldsymbol{R}_t}{\partial\theta_p}\left(\begin{bmatrix}\frac{\partial\theta_p}{\partial\sigma_{y'z'}} & \frac{\partial\theta_p}{\partial\sigma_{y'y'}} & \frac{\partial\theta_p}{\partial\sigma_{z'z'}}\end{bmatrix}\begin{bmatrix}\frac{\partial\sigma_{y'z'}}{\partial\boldsymbol{U}}\\ \frac{\partial\sigma_{y'y'}}{\partial\boldsymbol{U}}\\ \frac{\partial\sigma_{z'z'}}{\partial\boldsymbol{U}}\end{bmatrix}\right), \tag{G.10}$$

$$\begin{bmatrix}\frac{\partial\sigma_{y'z'}}{\partial\boldsymbol{U}}\\ \frac{\partial\sigma_{y'y'}}{\partial\boldsymbol{U}}\\ \frac{\partial\sigma_{z'z'}}{\partial\boldsymbol{U}}\end{bmatrix} = \boldsymbol{M}_3\boldsymbol{R}\frac{\partial\boldsymbol{\sigma}}{\partial\boldsymbol{U}}\boldsymbol{R}^\top\boldsymbol{M}_2 + \boldsymbol{M}_4\boldsymbol{R}\frac{\partial\boldsymbol{\sigma}}{\partial\boldsymbol{U}}\boldsymbol{R}^\top\boldsymbol{M}_5. \tag{G.11}$$

Bibliography

Alidoost, K., Feng, M., Geubelle, P. H., & Tortorelli, D. A. (2020). Energy release rate approximation for small surface cracks in three-dimensional domains using the topological derivative. *Journal of Applied Mechanics*, *87*(4), 041004.

Alidoost, K., Fernandez, F., Geubelle, P. H., & Tortorelli, D. A. (2022). Fracture-based shape optimization built upon the topological derivative. *Computer Methods in Applied Mechanics and Engineering*, *395*, 114994.

Amir, O., & Sigmund, O. (2013). Reinforcement layout design for concrete structures based on continuum damage and truss topology optimization. *Structural and Multidisciplinary Optimization*, *47*(2), 157–174.

Anderson, T. L. (2017). *Fracture mechanics: Fundamentals and applications*. Boca Raton: CRC press.

Aragón, A. M., Liang, B., Ahmadian, H., & Soghrati, S. (2020). On the stability and interpolating properties of the hierarchical interface-enriched finite element method. *Computer Methods in Applied Mechanics and Engineering*, *362*, 112671.

Balay, S., Abhyankar, S., Adams, M., Brown, J., Brune, P., Buschelman, K., Dalcin, L., Dener, A., Eijkhout, V., Gropp, W., et al. (2019). Petsc users manual.

Bendsøe, M. P., & Kikuchi, N. (1988). Generating optimal topologies in structural design using a homogenization method. *Computer methods in applied mechanics and engineering*, *71*(2), 197–224.

Challis, V. J., Roberts, A. P., & Wilkins, A. H. (2008). Fracture resistance via topology optimization. *Structural and Multidisciplinary Optimization*, *36*(3), 263–271.

Chan, A. (1968). Variational methods in elasticity and plasticity. k. washizu. pergamon, oxford, 1968. 350 pp. illustrated. 120s. *The Aeronautical Journal*, *72*(694), 889–889.

Chen, A., Cai, K., Zhao, Z.-L., Zhou, Y., Xia, L., & Xie, Y. M. (2021). Controlling the maximum first principal stress in topology optimization. *Structural and Multidisciplinary Optimization*, *63*(1), 327–339.

Chen, Z., Long, K., Wen, P., & Nouman, S. (2020). Fatigue-resistance topology optimization of continuum structure by penalizing the cumulative fatigue damage. *Advances in Engineering Software*, *150*, 102924.

Da, D., & Yvonnet, J. (2020). Topology optimization for maximizing the fracture resistance of periodic quasi-brittle composites structures. *Materials*, *13*(15), 3279.

Da, D., Yvonnet, J., Xia, L., & Li, G. (2018). Topology optimization of particle-matrix composites for optimal fracture resistance taking into account interfacial damage. *International Journal for Numerical Methods in Engineering*, *115*(5), 604–626.

da Silva, G. A., Beck, A. T., & Sigmund, O. (2022). Structural topology optimization with predetermined breaking points. *Computer Methods in Applied Mechanics and Engineering*, *400*, 115610.

Desai, J., Allaire, G., & Jouve, F. (2022). Topology optimization of structures undergoing brittle fracture. *Journal of Computational Physics*, *458*, 111048.

Devloo, P. R. B. (1997). Pz: An object oriented environment for scientific programming. *Computer methods in applied mechanics and engineering*, *150*(1-4), 133–153.

Duysinx, P., & Sigmund, O. (1998). New developments in handling stress constraints in optimal material distribution, 4906.

Epp, P. Y. (2022). *Learning from failure: Development and discussion of a database of structural failures* (Doctoral dissertation). University of British Columbia.

Erdogan, F., & Sih, G. (1963). On the crack extension in plates under plane loading and transverse shear.

Francfort, G. A., & Marigo, J.-J. (1998). Revisiting brittle fracture as an energy minimization problem. *Journal of the Mechanics and Physics of Solids*, *46*(8), 1319–1342.

Giraldo-Londoño, O., & Paulino, G. H. (2020). A unified approach for topology optimization with local stress constraints considering various failure criteria: Von mises, drucker–prager, tresca, mohr–

coulomb, bresler–pister and willam–warnke. *Proceedings of the Royal Society A*, *476*(2238), 20190861.

Gu, G. X., Dimas, L., Qin, Z., & Buehler, M. J. (2016). Optimization of composite fracture properties: Method, validation, and applications. *Journal of Applied Mechanics*, *83*(7).

Gu, G. X., Wettermark, S., & Buehler, M. J. (2017). Algorithm-driven design of fracture resistant composite materials realized through additive manufacturing. *Additive manufacturing*, *17*, 47–54.

Habibian, A., Sohouli, A., Kefal, A., Nadler, B., Yildiz, M., & Suleman, A. (2021). Multi-material topology optimization of structures with discontinuities using peridynamics. *Composite Structures*, *258*, 113345.

Holmberg, E., Torstenfelt, B., & Klarbring, A. (2013). Stress constrained topology optimization. *Structural and Multidisciplinary Optimization*, *48*(1), 33–47.

Hu, J., Yao, S., Gan, N., Xiong, Y., & Chen, X. (2019). Fracture strength topology optimization of structural specific position using a bi-directional evolutionary structural optimization method. *Engineering Optimization*, *52*, 583–602.

Jeong, S. H., Park, S. H., Choi, D.-H., & Yoon, G. H. (2013). Toward a stress-based topology optimization procedure with indirect calculation of internal finite element information. *Computers & Mathematics with Applications*, *66*(6), 1065–1081.

Kang, Z., Pai, L., & Li, M. (2017). Topology optimization considering fracture mechanics behaviors at specified locations. *Structural and Multidisciplinary Optimization*, *55*(5), 1847–1864.

Kefal, A., Sohouli, A., Oterkus, E., Yildiz, M., & Suleman, A. (2019). Topology optimization of cracked structures using peridynamics. *Continuum Mechanics and Thermodynamics*, *31*(6), 1645–1672.

Klarbring, A., Torstenfelt, B., Edlund, U., Schmidt, P., Simonsson, K., & Ansell, H. (2018). Minimizing crack energy release rate by topology optimization. *Structural and Multidisciplinary Optimization*, *58*(4), 1695–1703.

Kreisselmeier, G., & Steinhauser, R. (1980). Systematic control design by optimizing a vector performance index. In *Computer aided design of control systems* (pp. 113–117). Elsevier.

Lahe Motlagh, P., & Kefal, A. (2021). Comparative study of peridynamics and finite element method for practical modeling of cracks in topology optimization. *Symmetry*, *13*(8), 1407.

Li, P., Wu, Y., & Yvonnet, J. (2021). A simp-phase field topology optimization framework to maximize quasi-brittle fracture resistance of 2d and 3d composites. *Theoretical and Applied Fracture Mechanics*, *114*, 102919.

Magnus, J. R., & Neudecker, H. (2019). *Matrix differential calculus with applications in statistics and econometrics*. John Wiley & Sons.

Miehe, C., Hofacker, M., & Welschinger, F. (2010). A phase field model for rate-independent crack propagation: Robust algorithmic implementation based on operator splits. *Computer Methods in Applied Mechanics and Engineering*, *199*(45-48), 2765–2778.

Nagai, M., Ikeda, T., & Miyazaki, N. (2013). Stress intensity factor analyses of three-dimensional interface cracks using tetrahedral finite elements. *Computational Mechanics*, *51*(5), 603–615.

Nakamura, T., & Parks, D. (1989). Antisymmetrical 3-d stress field near the crack front of a thin elastic plate. *International Journal of Solids and Structures*, *25*(12), 1411–1426.

Payen, D. J., & Bathe, K.-J. (2012). A stress improvement procedure. *Computers & structures*, *112*, 311–326.

Rice, J. R. (1968). A path independent integral and the approximate analysis of strain concentration by notches and cracks. *Journal of Applied Mechanics*, *35(2)*, 379–386.

Russ, J. B., & Waisman, H. (2019). Topology optimization for brittle fracture resistance. *Computer Methods in Applied Mechanics and Engineering*, *347*, 238–263.

Senhora, F. V., Giraldo-Londono, O., Menezes, I. F., & Paulino, G. H. (2020). Topology optimization with local stress constraints: A stress aggregation-free approach. *Structural and Multidisciplinary Optimization*, *62*(4), 1639–1668.

Sethian, J. A., & Wiegmann, A. (2000). Structural boundary design via level set and immersed interface methods. *Journal of computational physics*, *163*(2), 489–528.

Sharma, R., Zhang, J., Langelaar, M., van Keulen, F., & Aragón, A. M. (2018). An improved stress recovery technique for low-order 3d finite elements. *International Journal for Numerical Methods in Engineering*, *114*(1), 88–103.

Silva, M., Geubelle, P. H., & Tortorelli, D. A. (2011). Energy release rate approximation for small surface-breaking cracks using the topological derivative. *Journal of the Mechanics and Physics of Solids*, *59*(5), 925–939.

Soghrati, S., Aragón, A. M., Armando Duarte, C., & Geubelle, P. H. (2012). An interface-enriched generalized fem for problems with discontinuous gradient fields. *International Journal for Numerical Methods in Engineering*, *89*(8), 991–1008.

Sohouli, A., Kefal, A., Abdelhamid, A., Yildiz, M., & Suleman, A. (2020). Continuous density-based topology optimization of cracked structures using peridynamics. *Structural and Multidisciplinary Optimization*, *62*, 2375–2389.

Sun, X., Fang, X., & Guan, L. (2009). *Mechanics of materials*. Beijing: Higher Education Press.

van den Boom, S. J., Zhang, J., van Keulen, F., & Aragón, A. M. (2019). A stable interface-enriched formulation for immersed domains with strong enforcement of essential boundary conditions. *International Journal for Numerical Methods in Engineering*, *120*(10), 1163–1183.

van den Boom, S. J., Zhang, J., van Keulen, F., & Aragón, A. M. (2021). An interface-enriched generalized finite element method for level set-based topology optimization. *Structural and Multidisciplinary Optimization*, *63*(1), 1–20.

Vukelić, G., & Vizentin, G. (2017). Common case studies of marine structural failures. *Failure Analysis and Prevention. InTech*, 135–151.

Wang, C., & Qian, X. (2018). Heaviside projection–based aggregation in stress-constrained topology optimization. *International Journal for Numerical Methods in Engineering*, *115*(7), 849–871.

Wang, H., Liu, J., & Wen, G. (2022). An efficient multi-resolution topology optimization scheme for stiffness maximization and stress minimization. *Engineering Optimization*, *54*(1), 40–60.

Wang, S., & Wang, M. Y. (2006). Radial basis functions and level set method for structural topology optimization. *International journal for numerical methods in engineering*, *65*(12), 2060–2090.

Wendland, H. (1995). Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree. *Advances in computational Mathematics*, *4*(1), 389–396.

Wu, C., Fang, J., Zhou, S., Zhang, Z., Sun, G., Steven, G. P., & Li, Q. (2020). Level-set topology optimization for maximizing fracture resistance of brittle materials using phase-field fracture model. *International Journal for Numerical Methods in Engineering*, *121*(13), 2929–2945.

Wu, C., Fang, J., Zhou, S., Zhang, Z., Sun, G., Steven, G. P., & Li, Q. (2021). A path-dependent level set topology optimization with fracture criterion. *Computers & Structures*, *249*, 106515.

Xia, L., Da, D., & Yvonnet, J. (2018). Topology optimization for maximizing the fracture resistance of quasi-brittle composites. *Computer Methods in Applied Mechanics and Engineering*, *332*, 234–254.

Yang, R., & Chen, C. (1996). Stress-based topology optimization. *Structural optimization*, *12*(2), 98–105.

Zaccone, G. (2015). *Python parallel programming cookbook*. Birmingham: Packt Publishing Ltd.

Zhang, J., van den Boom, S. J., van Keulen, F., & Aragón, A. M. (2019). A stable discontinuity-enriched finite element method for 3-d problems containing weak and strong discontinuities. *Computer Methods in Applied Mechanics and Engineering*, *355*, 1097–1123.

Zhang, J., van Keulen, F., & Aragón, A. M. (2022). On tailoring fracture resistance of brittle structures: A level set interface-enriched topology optimization approach. *Computer Methods in Applied Mechanics and Engineering*, *388*, 114–189.

Zhang, W. (2016). Technical problem identification for the failures of the liberty ships. *Challenges*, *7*(2), 20.