Advanced high performance computing methods for the solution of crack propagation and material design problems using the extended Finite Element method (XFEM)

By
Serafeim Bakalakos

School of Civil Engineering
Institute of Structural Analysis and Antiseismic Research

National Technical University of Athens

Supervisors: Prof. Manolis Papadrakakis and Prof. Vissarion Papadopoulos

A thesis submitted for the degree of
Doctor of Philosophy

June, 2022

To my parents.

Acknowledgments

Undertaking this PhD has been a truly life-changing experience for me and required the support and guidance that I received from many people.

First and foremost, I want to thank my parents for their constant support and unconditional love throughout these years and my life in general.

This thesis would not have been possible without the guidance, support and motivation of my supervisors Professor Manolis Papadrakakis and Professor Vissarion Papadopoulos. Thank you for your patience, your continuous belief in me and for being so generous with your time and intellect.

I would also like to express my gratitude to the rest of my thesis committee, professors Konstantinos Spiliopoulos and Nikolaos Lagaros for their insightful comments and being ever available for consultation.

Moreover, this thesis would be a lot poorer without the help of my friends and colleagues Yannis Kalogeris, George Stavroulakis and Manolis Georgioudakis, without our stimulating discussions that bore many inspirations and without the sleepless nights we were working together before deadlines.

Finally, I gratefully acknowledge the funding received towards my Ph.D from the Special Account for Research Funding (E.L.K.E.) of National Technical University of Athens (N.T.U.A.) scholarship. Funding was also provided by participation in the project "Optimal multiscale design of innovative materials for heat exchange applications, (HEAT-68/1286)", from the European Regional Development Fund and Greek National Funds through the operational program Competitiveness, Entrepreneurship and Innovation, under the call Research-Create-Innovate.

Ευχαριστίες

Η εκπόνηση αυτής της διδακροτικής διατριβής ήταν εμπειρία καθοριστικής σημασίας για μένα και χρειάστηκε τη στήριξη και την καθοδήγηση πολλών ανθρώπων.

Πρώταρχικά θέλω να ευχαριστήσω τους γονείς μου για τη συνεχή στήριξη και αγάπη τους κατά τη διάρκεια αυτών των χρόνων και της ζωής μου γενικότερα.

Η διατριβή αυτή θα ήταν αδύνατη χωρίς την καθοδήγηση και παρότρυνση από τους επιβλέποντες καθηγητές μου, κ. Μανόλη Παπαδρακάκη και κ. Βησσαρίωνα Παπαδόπουλο. Σας ευχαριστώ για την υπομονή σας, τη συνεχή εμπιστοσύνη σε μένα και τη γενναιοδωρία με το χρόνο και τις γνώσεις σας.

Θα ήθελα επίσης να εκφράσω την ευγνωμοσύνη μου στα άλλα μέλη της συμβουλευτικής επιτροπής, καθηγητές κ. Κωνσταντίνο Σπηλιόπουλο και κ. Νικόλαο Λαγαρό, για τα εύστοχα σχόλια και τη διαθεσιμότητα τους για συμβουλές.

Επιπλέον, αυτή η διατριβή θα ήταν πολύ φτωχότερη χωρίς τη βοήθεια των συνεργατών και φίλων μου Ιωάννη Καλογερή, Γεώργιο Σταυρουλάκη και Μανώλη Γεωργιουδάκη, χωρίς τις συζητήσεις μας που οδήγησαν σε πολλές εμπνεύσεις και χωρίς τις άυπνες νύχτες που εργαζόμασταν μαζί πριν διάφορες διορίες.

Τέλος είμαι ευγνώμον για την ευγενική χορηγεία του Ειδικού Λογαριασμού Κονδυλίων Έρευνας (Ε.Λ.Κ.Ε) του Εθνικού Μετσοβίου Πολυτεχνείου (Ε.Μ.Π.) με τη μορφή υποτροφίας για την εκπόνηση της διδακτορικής διατριβής μου. Περεταίρω οικονομική ενίσχυση έλαβα από συμμετοχή στο πρόγραμμα 'Βέλτιστος σχεδιασμός σε πολλαπλές κλίμακες καινοτόμων υλικών για εφαρμογές μετάδοσης θερμότητας (HEAT-68/1286)' υπό τη δράση: Επιχειρησιακό Πρόγραμμα Ανταγωνιστικότητα-Επιχειρηματικότητα-Καινοτομία (ΕΠΑνΕΚ) με συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης.

Advanced high performance computing methods for the solution of crack propagation and material design problems using the extended Finite Element method (XFEM)

Abstract

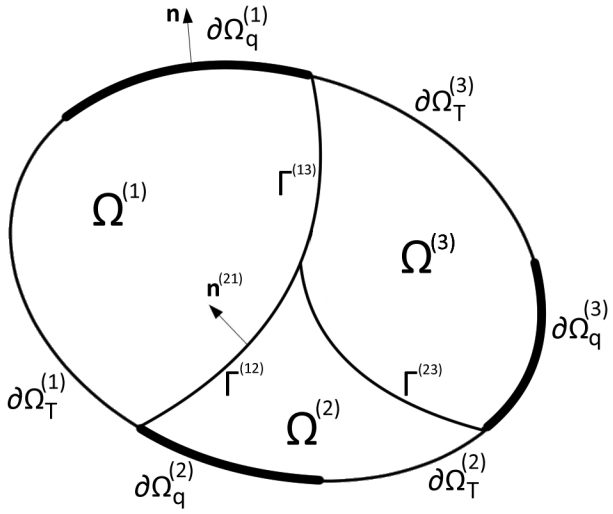
The need for advanced high-performance materials in the industry led to the development of various innovative solutions over the years, designed to possess application-specific properties, such as improved thermal conductivity. To model heat transfer in composite materials, their complex micro-structure, as well as the thermal resistance at the interfaces between materials must be taken into account. The standard finite element treatment requires very fine meshes to conform to the complex geometry of these interfaces. This thesis proposes an eXtended Finite Element Method (XFEM) formulation that captures the temperature jump by enriching the polynomial approximation around the material interfaces with appropriate discontinuous functions. Specifically, a new XFEM enrichment scheme is developed to address the issue of multiple-phase junctions, namely areas where multiple interfaces with different resistance properties intersect. In addition, a double-mesh LSM technique is developed for describing the geometry of material interfaces. A very fine mesh is employed by a Level Set Method (LSM) to represent complex interface geometries with high accuracy, whereas XFEM uses a mesh that does not conform to the material interfaces, but is instead a coarser version of the LSM mesh, to reduce the computational cost of the analysis. The combined numerical model is first validated against existing results from the literature on polycrystalline materials. Then, it is applied for heat conduction analysis of polymers reinforced with carbon-nanotubes. The unknown thermal resistance between these materials is inferred by calibrating the numerically predicted effective conductivity to corresponding experimental measurements. The proposed XFEM model can be straightforwardly extended to other similar problem types, such us elasticity or electrical conduction.

XFEM is also an attractive choice for modeling crack propagation, by enriching the polynomial displacement field of FEM with specialized non-smooth functions, without the need of remeshing in the vicinity of the crack at each propagation step. However, this enrichment causes the stiffness matrix to become strongly ill-conditioned, rendering the convergence of iterative solvers very slow. On the other hand, direct solvers are inefficient in 3D problems, due to the increased bandwidth of the system matrix. In this thesis, two domain decomposition solvers, namely FETI-DP and P-FETI-DP, are proposed for solving the linear systems resulting from XFEM crack propagation analysis in large-scale 3D problems. By modifying the coarse problem of both solvers, any singularities caused by the crack propagation are avoided and the XFEM-related ill-conditioning is completely eliminated, ensuring the scalability of FETI-DP and P-FETI-DP as the number of subdomains is increased. Finally, an efficient implementation in high performance computing systems, specifically computer clusters is developed, by altering the original FETI-DP and P-FETI-DP equations to minimize communication and computation bottlenecks in distributed memory environments.

Κεφάλαιο 0

Ελληνική περίληψη

Διάδοση θερμότητας σε σύνθετα υλικά



Σχήμα 0.1: Μετάδοση θερμότητας σε φορέα που αποτελείται από πολλαπλές φάσεις υλικού

Έστω ότι το σώμα Ω αποτελείται από n_p φάσεις υλικού $\Omega^{(i)}$, οι οποίες χωρίζονται από n_b διεπιφάνειες $\Gamma^{(ij)} \equiv \Gamma^{(ji)}$. Το σχήμα 0.1 δείχνει ένα παράδειγμα με τρεις φάσεις. Το εξωτερικό σύνορο $\partial\Omega$ του φορέα έχει κάθετο διάνυσμα $\boldsymbol{n}$ και αποτελείται από τα εξωτερικά σύνορα των επιμέρους φάσεων υλικού $\partial\Omega = \bigcup_{i=1}^{n_p} \partial\Omega^{(i)}$. Καθένα από αυτά χωρίζεται στα συμπληρωματικά μέρη $\partial\Omega_T^{(i)}$ και $\partial\Omega_q^{(i)}$, έτσι ώστε $\partial\Omega^{(i)} = \partial\Omega_T^{(i)} \cup \partial\Omega_q^{(i)}$. Συνοριακές συνθήκες Dirichlet, Neumann επιβάλλονται αντίστοιχα στα $\partial\Omega_T^{(i)}$ και $\partial\Omega_q^{(i)}$

$$\begin{aligned} T &= \bar{T} \quad \text{ον } \partial\Omega_T^{(i)}, \\ \boldsymbol{q}\cdot\boldsymbol{n} &= -\bar{q}_n \quad \text{ον } \partial\Omega_q^{(i)}. \end{aligned} \tag{1}$$

όπου $T = T(\boldsymbol{x})$ είναι το (βαθμωτό) πεδίο θερμοκρασίας και $\boldsymbol{q} = \boldsymbol{q}(\boldsymbol{x})$ είναι το (διανυσματικό) πεδίο ροής θερμότητας. Στη γενική περίπτωση, η θερμική αγωγιμότητα $\boldsymbol{k}^{(i)}$ της φάσης $\Omega^{(i)}$ είναι συμμετρικός τανυστής δευτέρας τάξης. Ο νόμος *Fourier* μεταξύ της θερμοκρασίας και ροής θερμότητας στο εσωτερικό κάθε φάσης $\Omega^{(i)}$ είναι

$$\boldsymbol{q}(\boldsymbol{x}) = -\boldsymbol{k}^{(i)}(\boldsymbol{x})\cdot\nabla T(\boldsymbol{x}), \quad \boldsymbol{x}\in\Omega^{(i)}, \quad i = 1,\dots n_p \tag{2}$$

Για μια δεδομένη πηγή θερμότητας $r(\boldsymbol{x})$, η εξίσωση της θερμοκρασίας στο εσωτερικό του Ω, σε σταθερή κατάσταση, είναι η εξίσωση *Poisson*

$$\nabla\cdot\boldsymbol{q}(\boldsymbol{x}) = r(\boldsymbol{x}) \tag{3}$$

όπου $\nabla\cdot\boldsymbol{q}(\boldsymbol{x})$ είναι η απόκλιση του πεδίου ροής θερμότητας. Σε αυτό το ετερογενές υλικό, κάθε διεπιφάνεια $\Gamma^{(ij)}$ παρουσιάζει διεπιφανειακή θερμική αντίσταση Kapitza $\alpha^{(ij)}$, ή ισοδύναμα διεπιφανεικαή αγωγιμότητα $k^{(ij)}$, που ορίζεται ως το αντίστροφο της αντίστασης, δηλαδή $k^{(ij)} = \frac{1}{\alpha^{(ij)}}$. Επομένως, η θερμική συμπεριφορά χαρακτηρίζεται από άλμα στο πεδίο θερμοκράσίας εγκάρσια σε κάθε διεπιφάνεια μεταξύ υλικών

$$[\![T]\!]^{(ij)} = -\alpha^{(ij)}\boldsymbol{q}^{(i)}\cdot\boldsymbol{n}^{(ij)} \quad \text{ον } \Gamma^{(ij)} \tag{4}$$

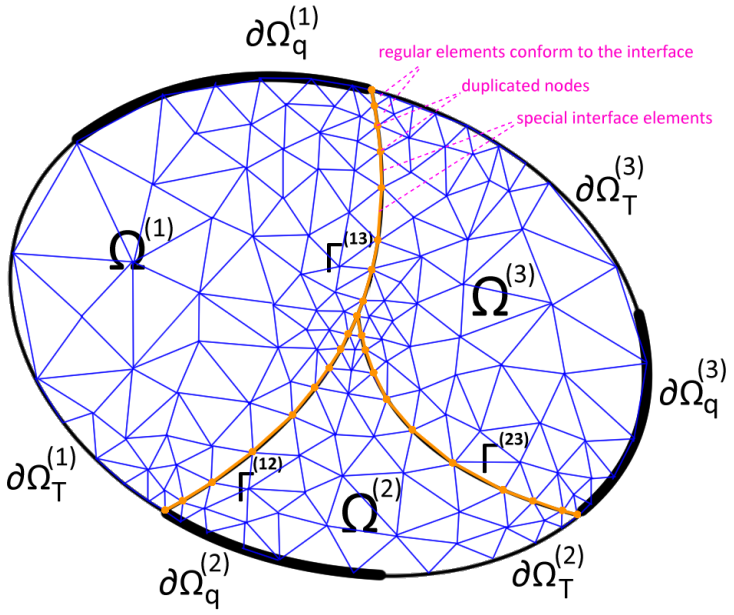
όπου ο τελεστής $[\![\cdot]\!]^{(ij)} = (\cdot)^{(j)} - (\cdot)^{(i)}$ υποδεικνύει το άλμα εγκάρσια στη διεπιφάνεια $\Gamma^{(ij)}$. Το μοναδιαίο διάνυσμα $\boldsymbol{n}^{(ij)}$ κάθετο στη $\Gamma^{(ij)} \equiv \Gamma^{(ji)}$ είναι προσανατολιμσένο από τη φάση $\Omega^{(i)}$ προς τη $\Omega^{(j)}$ και ισχύει ότι

$$\boldsymbol{n}^{(ij)} = -\boldsymbol{n}^{(ji)} \tag{5}$$

Επιπλέον, το πεδίο ροής θερμότητας είναι συνεχές εγκάρσια σε κάθε διεπιφάνεια $\Gamma^{(ij)}$

$$\boldsymbol{q}^{(i)}\cdot\boldsymbol{n}^{(ij)} = \boldsymbol{q}^{(j)}\cdot\boldsymbol{n}^{(ji)} \quad \text{ον } \Gamma^{(ij)} \tag{6}$$

όπου $\boldsymbol{q}^{(i)}$ και $\boldsymbol{q}^{(j)}$ είναι οι τιμές του πεδίου ροής θερμότητας στις διαφορετικές πλευρές της διεπιφάνειας $\Gamma^{(ij)}$. Το παραπάνω πρόβλημα συνοριακών τιμών μπορεί να λυθεί με τη συμβατική μέθοδο πεπερασμένων στοιχείων (ΜΠΣ), αλλά απαιτούνται ορισμένες τροποποιήσεις. Πρώτον, ένα πλέγμα μπεπερασμένων στοιχείων, που να ακολουθεί τη γεωμετρία των διεπιφανειών μεταξύ των φάσεων υλικού $\Gamma^{(ij)}$, πρέπει να δημιουργηθεί, όπως φαίνεται στο σχήμα 0.2. Δεύτερον, οι κόμβοι που βρίσκονται πάνω στις διεπιφάνειες υλικού πρέπει να αναπαραχθούν, ώστε να υπάρχει ένας διαφορετικός κόμβος για κάθε φάση υλικού $\Omega^{(i)}$ και να μπορεί να προσομοιωθεί το άλμα θερμοκρασίας. Τέλος, ειδικά πεπερασμένα στοιχεία μεταξύ των πολλαπλών ταυτιζόμενων κόμβων πρέπει να χρησιμοποιηθούν για την προσομοίωση της διεπιφανειακής αντίστασης στις εξισώσεις (4) και (6). Ωστόσο, αυτή η προσέγγιση με ΜΠΣ είναι μη αποδοτική ή ακόμα και

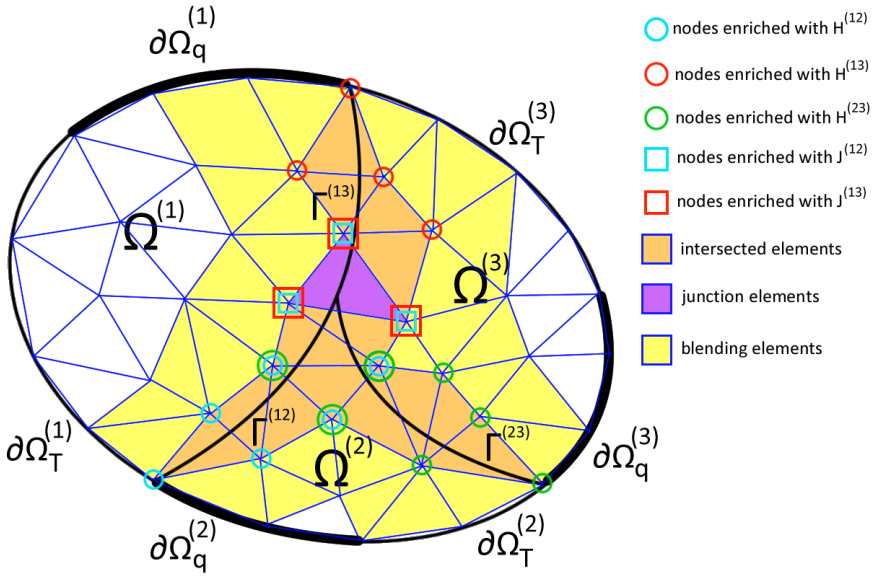


Σχήμα 0.2: Προσομοίωση μετάδοσης θερμότητας σε σύνθετα υλικά με τη συμβατική μέθοδο πεπερασμένων στοιχείων.

αδύνατη σε τρισδιάστατα προβλήματα μεγάλης κλίμακας. Σε αυτές τις περιπτώσεις, ένας πολύ μεγάλος αριθμός από μικρά πεπερασμένα στοιχεία απαιτείται γύρω από τις διεπιφάνειες, ειδικά στις περιοχές με απότομες στροφές. Αυτή η πυκνή διακριτοποίηση αυξάνει έντονα το υπολογιστικό κόστος της ΜΠΣ και κυρίως της επίλυσης του παραγόμενου γραμμικού συστήματος.

Για την αντιμετώπιση αυτού του προβλήματος, στην παρούσα διατριβή προτείνεται μια πρωτότυπη προσέγγιση, η οποία βασίζεται στην επεκταμένη μέθοδο πεπερασμένων στοιχείων (ΕΜΠΣ) για την προσομοίωση μετάδοσης θερμότητας σε σύνθετα υλικά. Στην ΕΜΠΣ, τα πεπερασμένα στοιχεία μπορούν να τέμνονται από μία ή περισσότερες διεπιφάνειες υλικού, όπως φαίνεται στοο σχήμα 0.3, αντί να χρειάζεται να προσαρμόζονται στη γεωμετρία των διεπιφανειών. Προκειμένου να προσομοιωθεί το άλμα στο πεδίο θερμοκρασίας, ειδικές ασυνεχείς συναρτήσεις βάσης εισάγονται στους κόμβους γύρω από τις διεπιφάνειες υλικού. Τα στοιχεία που τέμνονται από τις διεπιφάνειες καλούνται εμπλουτισμένα στοιχεία. Οι ασυνεχείς συναρτήσεις βάσης εισάγονται στους κόμβους των εμπλουτισμένων στοιχείων, που ονομάζονται εμπλουτισμένοι κόμβοι, σε αντίθεση με τους υπόλοιπους συμβατικούς κόμβους. Τα στοιχεία χωρίς εμπλουτισμένους κόμβους ονομάζονται συμβατικά στοιχεία και συμπεριφέρονται όπως στη ΜΠΣ. Επιπλέον, τα μεικτά στοιχεία δεν τέμνονται από διεπιφάνειες υλικού, αλλά μοιράζονται έναν ή περισσότερους κόμβους με εμπλουτισμένα στοιχεία. Γενικώς, το πεδίο θερμοκρασίας δεν μπορεί να αναπαραχθεί με μεγάλη ακρίβεια εντός των μεικτών στοιχείων. Ωστόσο, η προτεινόμενη μεθοδολογία ΕΜΠΣ αποφεύγει το πρόβλημα μεικτών στοιχείων, χρησιμοποιώντας κατάλληλες συναρτήσεις εμπλουτισμού. Έτσι σύνθετες γεωμετρίες μπορούν να αναπαρασταθούν εύκολα και με ακρίβεια με τη μέθοδο ισοϋψών καμπυλών, ενώ ένα απλό και αραιό πλέγμα, που δεν ακολου-

θεί αυτές τις γεωμετρίες, μπορεί να χρησιμοποιηθεί για να μειώσει το υπολογιστικό κόστος της ανάλυσης.



Σχήμα 0.3: Συναρτήσεις εμπλουτισμού Heaviside και συμβολής όταν τέμνονται διεπιφάνειες υλικού.

Όπως η ΜΠΣ, η ΕΜΠΣ εφαρμόζεται πάνω στην ασθενή μορφή του προβλήματος συνοριακών τιμών. Έστω ο συναρτησιακός χώρος όλων των επιτρεπτών πεδίων θερμοκρασίας (δοκιμαστικές συναρτήσεις)

$$\mathbb{D} = \{T : T = \bar{T} \text{ ον } \partial\Omega_T,\ T \text{ ασυνεχής σε } \Gamma^{(ij)},\ \forall (i,j) \in M^\Gamma\} \tag{7}$$

Επίσης ο συναρτησιακός χώρος συναρτήσεων στάθμισης είναι

$$\mathbb{W} = \{\delta T : \delta T = 0 \text{ ον } \partial\Omega_T,\ \delta T \text{ ασυνεχής σε } \Gamma^{(ij)},\ \forall (i,j) \in M^\Gamma\} \tag{8}$$

Η ασθενής μορφή του προβλήματος συνοριακών τιμών τίθεται ως εξής: 'Βρες δοκιμαστική συνάρτηση $T \in \mathbb{D}$, έτσι ώστε για όλες τις συναρτήσεις στάθμισης $\delta T \in \mathbb{W}$ να ισχύει η ακόλουθη ολοκληρωτική εξίσωση:'

$$\sum_{(i,j)\in M^\Gamma} \int_{\Gamma^{(ij)}} [\![\delta T]\!]^{(ij)} \frac{1}{\alpha^{(ij)}} [\![T]\!]^{(ij)}\ d\Gamma + \int_\Omega \nabla\delta T \cdot \boldsymbol{k} \cdot \nabla T d\Omega = \int_\Omega \delta T\ r\ d\Omega + \int_{\partial\Omega_q} \delta T\ \bar{q}_n\ d\Gamma \tag{9}$$

Στη γενική περίπτωση πολλαπλών φάσεων υλικού, μία συνάρτηση εμπλουτισμού $Heaviside$ $H^{(pq)}$ ανά διεπιφάνεια $\Gamma^{(pq)}$ απαιτείται για τη μοντελοποίηση του άλματος:

$$H^{(pq)}(\boldsymbol{x}) = \begin{cases} -1, & \boldsymbol{x} \in \Omega^{(p)} \\ +1, & \boldsymbol{x} \in \Omega^{(q)} \end{cases} \tag{10}$$

Ωστόσο, εντός κάποιων στοιχείων τέμνονται οι διεπιφάνειες μεταξύ τριών ή περισσοτέρων φάσεων υλικού, όπως φαίνεται στο σχήμα 0.3. Για να αναπαραχθεί το ασυνεχές πεδίο θερμοκρασίας εντός αυτών των στοιχείων, στην παρούσα διατριβή προτείνεται η χρήση συναρτήσεων συμβολής. Οι κόμβοι ενός στοιχείου, που περιέχει σημεία συμβολής $n_J \geq 3$ διεπιφανειών, εμπλουτίζονται με $n_J - 1$ συναρτήσεις συμβολής, αντί για συναρτήσεις $Heaviside$. Η συνάρτηση συμβολής $J^{(rs)}(\boldsymbol{x})$ για τη διεπιφάνεια $\Gamma^{(rs)}$ μεταξύ των φάσεων $\Omega^{(r)}, \Omega^{(s)}$, η οποία τέμνει 2 ή περισσότερες άλλες διεπιφάνειες, είναι

$$J^{(rs)}(\boldsymbol{x}) = \begin{cases} -1, & \boldsymbol{x} \in \Omega^{(r)} \\ +1, & \boldsymbol{x} \in \Omega^{(s)} \\ 0, & \boldsymbol{x} \in \Omega - \left(\Omega^{(r)} \cup \Omega^{(s)}\right) \end{cases} \tag{11}$$

Είναι βολικό να αναφερόμαστε σε κάθε εμπλουτισμό $Heaviside$ ως $H^b(\boldsymbol{x}) = H^{(pq)}(\boldsymbol{x})$, όπου $b = 1, \cdots n_b$ είναι ένας ακέραιος που αντιστοιχεί στο ζεύγος (p, q). Ομοίως, αν συνολικά υπάρχουν n_c συναρτήσεις συμβολής, καθεμία από αυτές θα ονομάζεται $J^c(\boldsymbol{x}) = J^{(rs)}(\boldsymbol{x})$, όπου $c = 1, \cdots n_c$ είναι ένας ακέραιος που αντιστοιχεί στο ζεύγος (r, s). Η προσέγγιση του πεδίου θερμοκρασίας στην ΕΜΠΣ εκφράζεται ως

$$\begin{aligned} T^h(\boldsymbol{x}) = & \sum_{k \in M} N_k(\boldsymbol{x}) T_k \\ & + \sum_{b=1}^{n_b} \left(\sum_{k \in M_H^b} N_k(\boldsymbol{x}) \left(H^b(\boldsymbol{x}) - H^b(\boldsymbol{x}_k) \right) \widetilde{T}_k^b \right) \\ & + \sum_{c=1}^{n_c} \left(\sum_{k \in M_J^c} N_k(\boldsymbol{x}) \left(J^c(\boldsymbol{x}) - J^c(\boldsymbol{x}_k) \right) \widehat{T}_k^c \right) \end{aligned} \tag{12}$$

όπου $N_k(\boldsymbol{x})$ είναι οι συναρτήσεις σχήματος της ΜΠΣ και T_k είναι επικόμβιες θερμοκρασίες, δηλαδή συμβατικοί βαθμοί ελευθερίας (β.ε.). Αντίθετα, $H^b(\boldsymbol{x})$ / $J^c(\boldsymbol{x})$ είναι καθεμία από τις n_b / n_c συναρτήσεις εμπλουτισμού $Heaviside$ / συμβολής, M_H^b / M_J^c είναι το σύνολο των κόμβων που εμπλουτίζονται με $H^b(\boldsymbol{x})$ / $J^c(\boldsymbol{x})$ και $\widetilde{T}_k^b$ / $\widehat{T}_k^c$ είναι οι αντίστοιχοι εμπλουτισμένοι β.ε.

Το άλμα του πεδίου θερμοκρασίας μπορεί να προσεγγιστεί χρησιμοποιώντας τους εμπλουτισμένους β.ε. Αν τα σημεία $\boldsymbol{x}^{(i)} \equiv \boldsymbol{x}^{(j)} \equiv \boldsymbol{x}$, ταυτίζονται αλλά βρίσκονται σε διαφορετικές πλευρές της $\Gamma^{(ij)}$, τότε το άλμα είναι

$$\left[\!\left[T^h(\boldsymbol{x}) \right]\!\right]^{(ij)} = \sum_{b=1}^{n_b} \left(\sum_{k \in M_H^b} N_k(\boldsymbol{x}) \left[\!\left[H^b \right]\!\right]^{(ij)} \widetilde{T}_k^b \right) + \sum_{c=1}^{n_c} \left(\sum_{k \in M_J^c} N_k(\boldsymbol{x}) \left[\!\left[J^c \right]\!\right]^{(ij)} \widehat{T}_k^c \right) \tag{13}$$

όπου $\left[\!\left[H^b\right]\!\right]^{(ij)}$ και $\left[\!\left[J^c\right]\!\right]^{(ij)}$ είναι 0, εκτός αν οι εμπλουτισμοί $H^b = H^{(pq)}$ και $J^c = J^{(rs)}$ έχουν εισαχθεί για τη μοντελοποίηση του άλματος εγκάρσια στη $\Gamma^{(ij)}$, δηλαδή $(pq) = (rs) = (ij)$. Γενικώς

$$\begin{aligned} \left[\!\left[H^{(b)}\right]\!\right]^{(ij)}(\boldsymbol{x}) = \left[\!\left[H^{(pq)}\right]\!\right]^{(ij)}(\boldsymbol{x}) = H^{(pq)}\left(\boldsymbol{x}^{(j)}\right) - H^{(pq)}\left(\boldsymbol{x}^{(i)}\right) = 0,\ -2,\ \text{ορ}\ +2 \\ \left[\!\left[J^{(c)}\right]\!\right]^{(ij)}(\boldsymbol{x}) = \left[\!\left[J^{(rs)}\right]\!\right]^{(ij)}(\boldsymbol{x}) = J^{(rs)}\left(\boldsymbol{x}^{(j)}\right) - J^{(rs)}\left(\boldsymbol{x}^{(i)}\right) = 0,\ -2,\ +2,\ -1\ \text{ορ}\ +1 \end{aligned} \tag{14}$$

Αντίθετα με άλλες στρατηγικές εμπλουτισμού, οι προτεινόμενοι εμπλουτισμοί *Heaviside* και συμβολής δεν προκαλούν σφάλματα ακρίβειας στα μεικτά στοιχεία, αφού

$$\begin{aligned} H^{(pq)}(\boldsymbol{x}) - H^{(pq)}(\boldsymbol{x}_k) = 0 \\ J^{(rs)}(\boldsymbol{x}) - J^{(rs)}(\boldsymbol{x}_k) = 0 \end{aligned} \tag{15}$$

και το εμπλουτισμένο μέρος του πεδίου προσέγγισης απαλείφεται

$$T^h(\boldsymbol{x}) = \sum_{k \in M} N_k(\boldsymbol{x})\, T_k \tag{16}$$

Αντικαθιστώντας το πεδίο προσέγγισης της εξίσωσης (12) στην ασθενή μορφή της εξίσωσης (9), προκύπτει το γραμμικό σύστημα

$$(\boldsymbol{K}_\Omega + \boldsymbol{K}_\Gamma) \cdot \boldsymbol{d} = \boldsymbol{f} \tag{17}$$

όπου $\boldsymbol{d}$ είναι οι άγνωστες θερμοκρασίες στους συμβατικούς και εμπλουτισμένους β.ε., $\boldsymbol{K}_\Omega$ είναι το χωρικό μητρώο αγωγιμότητας

$$\boldsymbol{K}_\Omega = \int_\Omega \boldsymbol{B}^T(\boldsymbol{x})\, \boldsymbol{k}(\boldsymbol{x})\, \boldsymbol{B}(\boldsymbol{x})\, d\Omega \tag{18}$$

$\boldsymbol{K}_\Gamma$ είναι το διεπιφανειακό μητρώο αγωγιμότητας

$$\boldsymbol{K}_\Gamma = \sum_{(i,j) \in M^\Gamma} \int_{\Gamma_e^{(ij)}} \frac{1}{\alpha^{(ij)}} \left(\overline{\boldsymbol{N}}^{(ij)}(\boldsymbol{x})\right)^T \cdot \overline{\boldsymbol{N}}^{(ij)}(\boldsymbol{x})\, d\Gamma \tag{19}$$

και $\boldsymbol{f}$ είναι τα εξωτερικά θερμικά φορτία

$$\boldsymbol{f} = \int_{\partial\Omega_q} \boldsymbol{N}^T \bar{q}_v d\Gamma + \int_\Omega \boldsymbol{N}^T r d\Omega \tag{20}$$

Ο πίνακας συναρτήσεων βάσης $\boldsymbol{N}$ για τους κόμβους k ενός στοιχείου είναι

$$\begin{aligned} \boldsymbol{N}(\boldsymbol{x}) &= \left[\begin{array}{cc} \boldsymbol{N}^{std}(\boldsymbol{x}) & \boldsymbol{N}^{enr}(\boldsymbol{x}) \end{array}\right] \\ \boldsymbol{N}^{std}(\boldsymbol{x}) &= \left[\begin{array}{ccc} \cdots & N_k(\boldsymbol{x}) & \cdots \end{array}\right] \\ \boldsymbol{N}^{enr}(\boldsymbol{x}) &= \left[\begin{array}{ccc} \cdots & N_k(\boldsymbol{x})\left(\Psi^a(\boldsymbol{x}) - \Psi^a(\boldsymbol{x}_k)\right) & \cdots \end{array}\right] \end{aligned} \tag{21}$$

όπου $\Psi^a(\boldsymbol{x})$ αναπαριστά οποιαδήποτε συνάρτηση εμπλουτισμού $Heaviside$ $H^b(\boldsymbol{x})$ ή συμβολής $J^c(\boldsymbol{x})$. Οι αντιστοιχες παράγωγοι των συναρτήσεων βάσης βρίσκονται στον πίνακα $\boldsymbol{B}$

$$
\begin{aligned}
\boldsymbol{B}(\boldsymbol{x}) &= \left[\begin{array}{cc} \boldsymbol{B}^{std}(\boldsymbol{x}) & \boldsymbol{B}^{enr}(\boldsymbol{x}) \end{array} \right] \\
\boldsymbol{B}^{std}(\boldsymbol{x}) &= \left[\begin{array}{ccc} & \dfrac{\partial N_k(\boldsymbol{x})}{\partial x} & \\ \cdots & \dfrac{\partial N_k(\boldsymbol{x})}{\partial y} & \cdots \\ & \dfrac{\partial N_k(\boldsymbol{x})}{\partial z} & \end{array} \right] \\
\boldsymbol{B}^{enr}(\boldsymbol{x}) &= \left[\begin{array}{ccc} & \dfrac{\partial \left(N_k(\boldsymbol{x}) \left(\Psi^a(\boldsymbol{x}) - \Psi^a(\boldsymbol{x}_k) \right) \right)}{\partial x} & \\ \cdots & \dfrac{\partial \left(N_k(\boldsymbol{x}) \left(\Psi^a(\boldsymbol{x}) - \Psi^a(\boldsymbol{x}_k) \right) \right)}{\partial y} & \cdots \\ & \dfrac{\partial \left(N_k(\boldsymbol{x}) \left(\Psi^a(\boldsymbol{x}) - \Psi^a(\boldsymbol{x}_k) \right) \right)}{\partial z} & \end{array} \right]
\end{aligned}
\tag{22}
$$

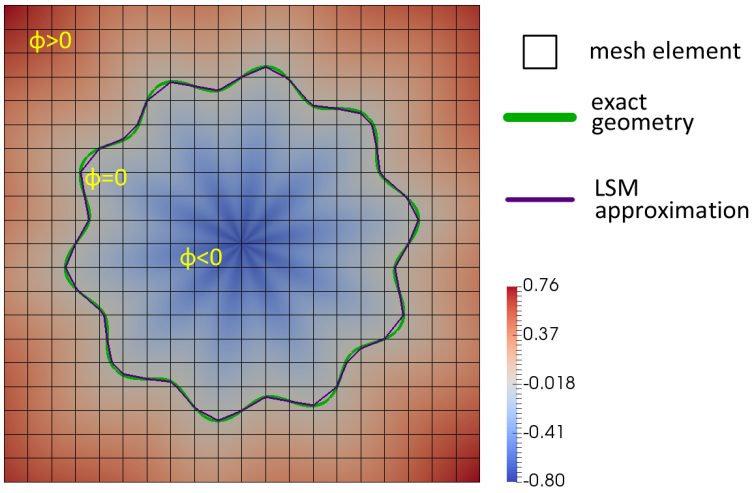
και $\overline{\boldsymbol{N}}^{(ij)}$ είναι πίνακας που χρησιμοποιείται για να παρεμβάλει το άλμα του πεδίου θερμοκρασίας εγκάρσια στην διεπιφάνεια $\Gamma^{(ij)}$

$$
\begin{aligned}
\overline{\boldsymbol{N}}^{(ij)}(\boldsymbol{x}) &= \left[\begin{array}{cc} \overline{\boldsymbol{N}}_{std} & \overline{\boldsymbol{N}}^{(ij)}_{enr}(\boldsymbol{x}) \end{array} \right] \\
\overline{\boldsymbol{N}}_{std} &= \boldsymbol{0} \\
\overline{\boldsymbol{N}}^{(ij)}_{enr}(\boldsymbol{x}) &= \left[\begin{array}{ccc} \cdots & N_k(\boldsymbol{x}) \left[\!\left[\Psi^a(\boldsymbol{x}) \right]\!\right]^{(ij)} & \cdots \end{array} \right]
\end{aligned}
\tag{23}
$$

Αναπαράσταση διεπιφανειών υλικού

Προκειμένου να αναπαρασταθεί η γεωμετρία των διεπιφανειών υλικού, χρησιμοποιείται η μέθοδος ισουψών καμπυλών ($Level\ Set\ Method$ - LSM). Στη LSM, ορίζεται η συνάρτηση προσημασμένης απόστασης $\phi(\boldsymbol{x})$ από ένα σημείο $\boldsymbol{x}$ προς την καμπύλη ή επιφάνεια, όπως φαίνεται στο σχήμα 0.4. Η καμπύλη/επιφάνεια περιγράφεται έμμεσα ως μηδενική ισοϋψής της συνάρτησης απόστασης. Η συνάρτηση απόστασης υπολογίζεται και αποθηκεύεται στους κόμβους $\boldsymbol{x}_k$ του πλέγματος. Έπειτα για οποιοδήποτε άλλο σημείο, χρησιμοποιούνται οι πολυωνυμικές συναρτήσεις σχήματος της ΜΠΣ:

$$
\phi(\boldsymbol{x}(\boldsymbol{\xi})) = \sum_{k=1}^{n_{nodes}} N_k(\boldsymbol{\xi}) \phi_k \tag{24}
$$



Σχήμα 0.4: Συνάρτηση προσημασμένης απόστασης στη LSM. Υπολογίζεται στους κόμβους και παρεμβάλεται εντός των στοιχείων.

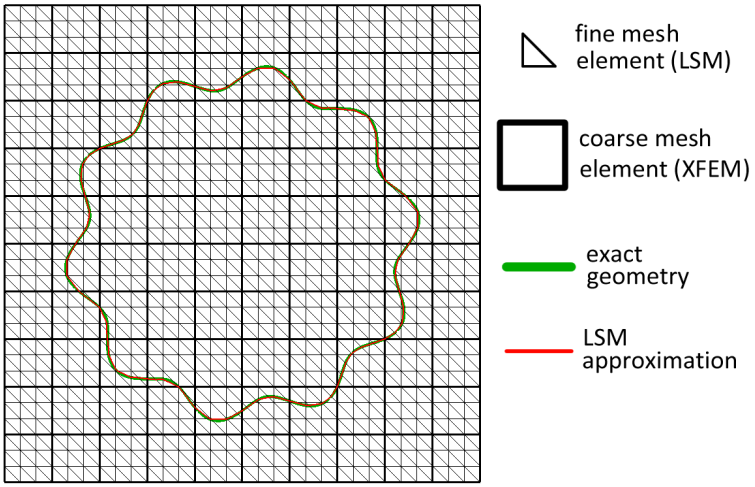
Η LSM συνεργάζεται με την ΕΜΠΣ, καθώς χρησιμοποιεί το ίδιο πλέγμα πεπερασμένων στοιχείων για να αναπαραστήσει τις ασυνέχειες και να εκτελέσει τις γεωμετρικές λειτουργίες που χρειάζεται η ΕΜΠΣ. Ωστόσο, για να περιγραφούν γεωμετρίες με υψηλή καμπυλότητα ή απότομες στροφές, απαιτείται ένα πολύ πυκνό πλέγμα, το οποίο αυξάνει ταχύτατα τις απαιτήσεις μνήμης και χρόνου για την ανάλυση με ΕΜΠΣ. Σε αυτή τη διατριβή αναπτύχθηκε μια προσέγγιση LSM με διπλό πλέγμα, η οποία χρησιμοποιεί ένα αραιό πλέγμα για αποδοτική ανάλυση ΕΜΠΣ και ένα πυκνό πλέγμα για ακριβή γεωμετρική περιγραφή με την LSM. Αυτά τα 2 πλέγμα ταυτίζονται σε συγκεκριμένους κόμβους, όπως φαίνεται στο σχήμα 0.5, ώστε να επιτρέπεται η συνεργασία ΕΜΠΣ-LSM.

Το αραιό πλέγμα αποτελείται από τετραπλευρικά στοιχεία 4 κόμβων σε δισδιάστατα προβλήμα ή εξαεδρικά στοιχεία 8 κόμβων σε τρισδιάστατα προβλήμα, ενώ το πυκνό πλέγμα αποτελείται από τριγωνικά ή τετραεδρικά στοιχεία αντίστοιχα. Η απεικόνιση μεταξύ των συστημάτων συντεταγμένων ενός στοιχείου αραιού πλέγματος και ενός στοιχείου πυκνού πλέγματος γίνεται χρησιμοποιώντας ένα βοηθητικό σύστημα συντεταγμένων, όπως φαίνεται στο σχήμα 0.6.

Η γεωμετρική αναπαράσταση μίας διεπιφάνειας υλικού με τη μέθοδο LSM μπορεί να χρησιμοποιηθεί για να βρεθεί η τομή των στοιχείων με τη διεπιφάνεια. Έστω ότι $\boldsymbol{r}_{P1}$ και $\boldsymbol{r}_{P2}$ είναι οι συντεταγμένες των κόμβων μιας πλευράς ενός στοιχείου του πυκνού πλέγματος. Τότε η πλευρά (P_1P_2) τέμνεται απο τη διεπιφάνεια αν $\phi_{P1} \cdot \phi_{P2} \leq 0$ και το σημείο τομής $\boldsymbol{r}_O$ είναι

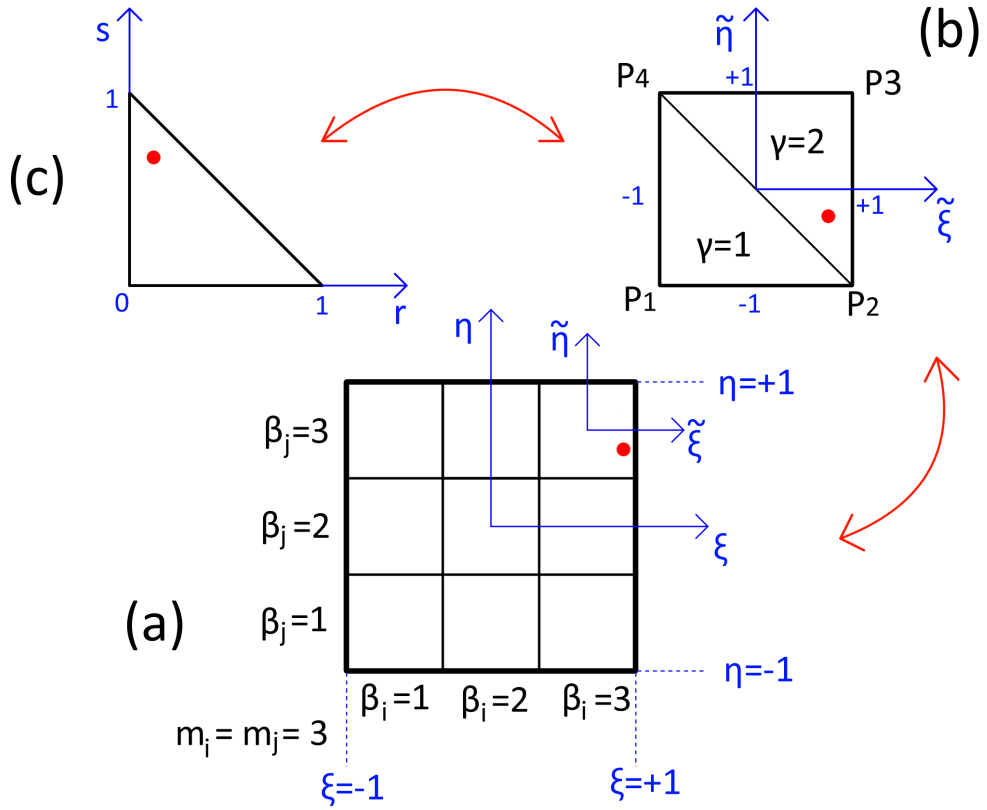
$$\boldsymbol{r}_O = \boldsymbol{r}_{P1} + \frac{0 - \phi_{P1}}{\phi_{P2} - \phi_{P1}} (\boldsymbol{r}_{P2} - \boldsymbol{r}_{P1}) \tag{25}$$

Ο προσδιορισμός αυτών των τομών είναι απαραίτητος για την ΕΜΠΣ, αφού το επιφανειακό ολοκλήρωμα της εξίσωσης (19) υπολογίζεται πάνω στα αποκοπτόμενα τμήματα (γραμμές σε δισδιάστατα προβλήματα, τρίγωνα σε τρισδιάστατα) κάθε στοιχείου. Επιπροσθέτως, για το



Σχήμα 0.5: Αραιό πλέγμα για ΕΜΠΣ και πυκνό για LSM

χωρικό ολοκλήρωμα της εξίσωσης (18), κάθε τεμνόμενο στοιχείο χωρίζεται σε υποστοιχεία (τρίγωνα σε δισδιάστατα προβλήματα, τετράεδρα σε τρισδιάστατα) που ακολουθούν τη γεωμετρία των αποκοπτόμενων τμημάτων, τα οποία προκύπτουν από την προτεινόμενη μέθοδο LSM με διπλό πλέγμα.



Σχήμα 0.6: Τοπικό σύστημα συντεταγμένων στοιχείων: α) Στοιχείο αραιού πλέγματος . β) Βοηθητικό σύστημα συντεταγμένων. γ) Στοιχείο πυκνού πλέγματος.

Διάδοση ρωγμών με την ΕΜΠΣ

Επιπλέον, μελετάται η ΕΜΠΣ για ψαθυρή διάδοση ρωγμών υπό την υπόθεση γραμμικής ελαστικής θραυστομηχανικής, όπου το μέγεθος της πλαστικής ζώνης είναι τόσο μικρό, ώστε να ενσωματωθεί σε μία ελαστική ζώνη γύρω από το μέτωπο της ρωγμής. Έστω ο φορέας Ω που περιέχει μία ρωγμή Γ_d, όπως φαίνεται στο σχήμα 0.7. Dirichlet και Neumann συνοριακές συνθήκες επιβάλλονται στα εξωτερικά σύνορα Γ_u και Γ_t, αντίστοιχα, ενώ στην επιφάνεια της ρωγμής δεν υπάρχει ελκυστής:

$$\begin{aligned} \boldsymbol{u} &= \widetilde{\boldsymbol{u}} \quad && \text{ον } \Gamma_u \\ \boldsymbol{\sigma}\cdot\boldsymbol{n}_\Gamma &= \widetilde{\boldsymbol{t}} \quad && \text{ον } \Gamma_t \\ \boldsymbol{\sigma}\cdot\boldsymbol{n}_{\Gamma_d} &= \boldsymbol{0} \quad && \text{ον } \Gamma_d \end{aligned} \tag{26}$$

όπου $\boldsymbol{u}$ είναι το πεδίο μετατοπίσεων, $\boldsymbol{\sigma}$ ο τανυστής τάσεων *Cauchy*, $\widetilde{\boldsymbol{u}}$ οι επιβαλλόμενες μετατοπίσεις και $\widetilde{\boldsymbol{t}}$ ο επιβαλλόμενος ελκυστής. Αν $\boldsymbol{\epsilon}(\boldsymbol{u})$ είναι ο τανυστής παραμορφώσεων, $\boldsymbol{C}$ ο καταστατικός τανυστής και $\boldsymbol{b}$ οι επιβαλλόμενες χωρικές δυνάμεις, τότε η εξισώση ισορροπίας

και ο καταστατικός νόμος είναι αντίστοιχα

$$\begin{aligned} \nabla : \boldsymbol{\sigma} + \boldsymbol{b} &= \mathbf{0} \\ \boldsymbol{\sigma} &= \boldsymbol{C} : \boldsymbol{\epsilon}(\boldsymbol{u}) \end{aligned} \tag{27}$$

Η ασθενής μορφή του προβλήματος τίθεται ως εξής: Βρες δοκιμαστική συνάρτηση $\boldsymbol{u}$ που ανήκει στο συναρτησιακό χώρο

$$\mathbb{U} = \{\boldsymbol{v} \in \mathbb{H} : \boldsymbol{v} = \widetilde{\boldsymbol{u}} \text{ στη } \Gamma_u \text{ , } \boldsymbol{v} \text{ ασυνεχής στη } \Gamma_d\} \tag{28}$$

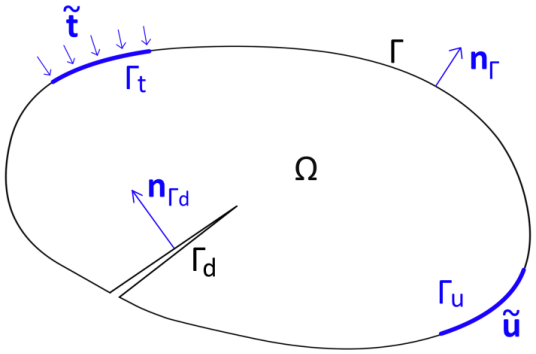
έτσι ώστε

$$\int_\Omega \boldsymbol{\epsilon}(\boldsymbol{w}) : \boldsymbol{C} : \boldsymbol{\epsilon}(\boldsymbol{u}) d\Omega = \int_\Omega \boldsymbol{w} \cdot \boldsymbol{b} \; d\Omega + \int_{\Gamma_t} \boldsymbol{w} \cdot \widetilde{\boldsymbol{t}} \; d\Gamma \tag{29}$$

για όλες τις συναρτήσεις στάθμισης $\boldsymbol{w}$ που ανήκουν στο χώρο

$$\mathbb{U}_0 = \{\boldsymbol{v} \in \mathbb{H} : \boldsymbol{v} = \mathbf{0} \text{ στη } \Gamma_u \text{ , } \boldsymbol{v} \text{ ασυνεχής στη } \Gamma_d\} \tag{30}$$

$\mathbb{H}$ είναι H^1 χώρος Hilbert συναρτήσεων που είναι ομαλές στο Ω, αλλά ασυνεχείς εγκάρσια στη Γ_d.



Σχήμα 0.7: Φορέας με ρωγμή

Για τη προσομοίωση του ασυνεχούς πεδίου μετατοπίσεων $\boldsymbol{u}$, η ΕΜΠΣ εμπλουτίζει το πολυωνυμικό προσεγγιστικό πεδίο της συμβατικής ΜΠΣ με μη συνεχείς συναρτήσεις βάσης. Το πλέγμα πεπερασμένων στοιχείων είναι ανεξάρτητο από τη γεωμετρία της ρωγμής και δεν την ακολουθεί. Αντίθετα κάποια στοιχεία τέμνονται από την επιφάνεια ή το μέτωπο της ρωγμής, όπως φαίνεται στο σχήμα 0.8β′. Οι κόμβοι των στοιχείων που τέμνονται από το μέτωπο της ρωγμής, εμπλουτίζονται με 4 ασυμπτωτικές συναρτήσεις αιχμής-ρωγμής $F_m(\boldsymbol{x})$, οι οποίες εξάγονται από τη θεωρία γραμμικής ελαστικής θραυστομηχανικής

$$\{F_m(\boldsymbol{x})\}_{m=1}^4 = \{F_m(r,\theta)\}_{m=1}^4 = \left\{ \sqrt{r}\sin(\frac{\theta}{2}); \sqrt{r}\cos(\frac{\theta}{2}); \sqrt{r}\sin(\frac{\theta}{2})\sin(\theta); \sqrt{r}\cos(\frac{\theta}{2})\sin(\theta) \right\} \tag{31}$$

όπου $(r, \theta) = (r(\boldsymbol{x}), \theta(\boldsymbol{x}))$ είναι οι συντεταγμένες ενός σημείου σε πολικό σύστημα που ορίζεται στο μέτωπο της ρωγμής, όπως φαίνεται στο σχήμα 0.8α΄. Επιπροσθέτως, οι κόμβοι των στοιχείων που τέμνονται από την επιφάνεια της ρωγμής, αλλά όχι το μέτωπο, εμπλουτίζονται με τη συνάρτηση $Heaviside$ $H(\boldsymbol{x})$

$$H(\boldsymbol{x}) = H(\phi(\boldsymbol{x})) = \begin{cases} +1, & \phi(\boldsymbol{x}) \geq 0 \\ -1, & \phi(\boldsymbol{x} < 0 \end{cases} \tag{32}$$

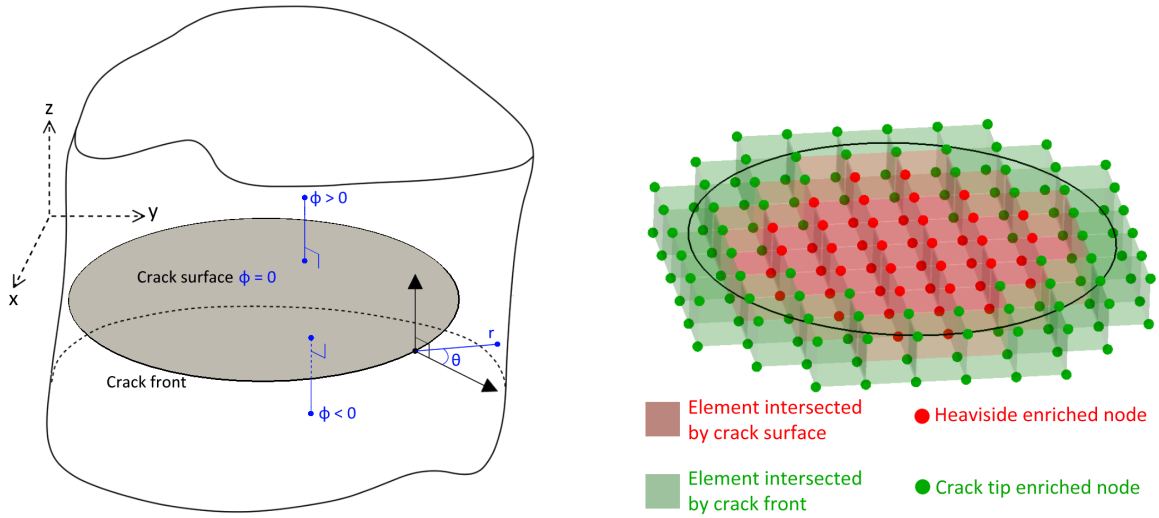
όπου $\phi(\boldsymbol{x})$ είναι η προσημασμένη απόσταση από ένα σημείου $\boldsymbol{x}$ ως την επιφάνεια της ρωγμής, όπως φαίνεται στο σχήμα 0.8α΄. Ο εμπλουτισμός της ΕΜΠΣ επιβάλλεται τοπικά γύρω από τη ρωγμή, ενώ τα υπόλοιπα στοιχεία και οι υπόλοιποι κόμβοι του πλέγματος δεν αλληλεπιδρούν με τη ρωγμή. Έστω M, M^H και M^T τα σύνολα όλων των κόμβων που δεν είναι εμπλουτισμένοι, είναι εμπλουτισμένοι με τη συνάρτηση $Heaviside$ ή είναι εμπλουτισμένοι με τις συναρτήσεις αιχμής-ρωγμής, αντίστοιχα. Τότε το εμπλουτισμένο προσεγγιστικό πεδίο, που χρησιμοποιείται στην ΕΜΠΣ, είναι

$$\begin{aligned} \boldsymbol{u}^h(\boldsymbol{x}) = & \sum_{i \in M} N_i(\boldsymbol{x}) \boldsymbol{u}_i \\ & + \sum_{j \in M^H} N_j(\boldsymbol{x}) \left(H(\boldsymbol{x}) - H(\boldsymbol{x}_j) \right) \boldsymbol{a}_j \\ & + \sum_{k \in M^T} N_k(\boldsymbol{x}) \Big(\sum_{m=1}^{4} \left(F_m(\boldsymbol{x}) - F_m(\boldsymbol{x}_k) \right) \boldsymbol{b}_k^m \Big) \end{aligned} \tag{33}$$

όπου $\boldsymbol{u}_i$ είναι οι συμβατικοί β.ε., που εκφράζουν επικόμβιες μετατοπίσεις, ενώ $\boldsymbol{a}_j$ και $\boldsymbol{b}_k^m$ είναι εμπλουτισμένοι β.ε., οι οποίοι εισάγονται από την ΕΜΠΣ στους κόμβους που είναι εμπλουτισμένοι με συναρτήσεις $Heaviside$ και αιχμής-ρωγμής, αντίστοιχα. Όλες οι $N_i(\boldsymbol{x})$, $N_j(\boldsymbol{x})$, $N_k(\boldsymbol{x})$ είναι πολυωνυμικές συναρτήσεις σχήματος, ίδιες με αυτές που χρησιμοποιούνται στη συμβατική ΜΠΣ. Το πρώτο άθροισμα στο δεξί μέλος της εξίσωσης (33) αντιστοιχεί στο προσέγγισστικό πεδίο της συμβατικής ΜΠΣ. Τα άλλα δύο αθροίσματα περιέχουν εμπλουτισμένες συναρτήσεις βάσης, που επιτρέπουν στο προσεγγιστικό πεδίο να μοντελοποιεί (α) το άλμα του πεδίου μετατοπίσεων κάθετα στην επιφάνεια της ρωγμής (εμπλουτισμός $Heaviside$) και (β) τον απειρισμό των πεδίων παραμορφώσεων και τάσεων στο μέτωπο της ρωγμής (εμπλουτισμός αιχμής-ρωγμής).

Χρησιμοποιώντας την προσέγγιση του πεδίου μετατοπίσεων από την εξίσωση (33), η ασθενής μορφή της εξίσωσης (29) καταλήγει σε γραμμικό σύστημα με αγνώστους τις επικόμβιες μετατοπίσεις $\boldsymbol{u}$

$$\begin{aligned} & \boldsymbol{K}\ \boldsymbol{u} = \boldsymbol{f} \\ & \boldsymbol{K} = \int_\Omega \boldsymbol{B}^T\ \boldsymbol{C}\ \boldsymbol{B}\ d\Omega \\ & \boldsymbol{f} = \int_\Omega \boldsymbol{N}^T\ \boldsymbol{b}\ d\Omega + \int_{\Gamma_t} \boldsymbol{N}^T\ \widetilde{\boldsymbol{t}}\ d\Gamma \end{aligned} \tag{34}$$



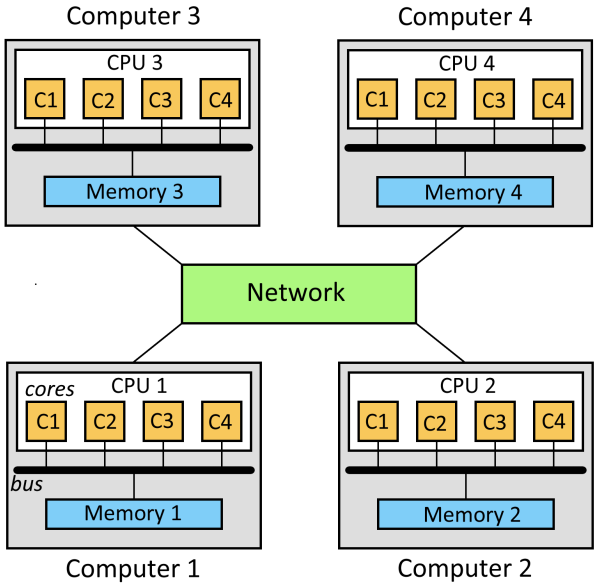
(α′) (β′)

Σχήμα 0.8: Ρωγμή σε τρισδιάστατο φορέα. (α) Προσημασμένες αποστάσεις ϕ από την επιφάνεια της ρωγμής και πολικές συντεταγμένες (r, θ) γύρω από το μέτωπο. (β) Εμπλουτισμένοι κόμβοι και στοιχεία που τέμνονται από τη ρωγμή.

όπου $\boldsymbol{N}$, $\boldsymbol{B}$ είναι πίνακες που περιέχουν τις συναρτήσεις βάσης και τις παραγώγους του, αντίστοιχα. Προκειμένου να αναπαρασταθεί η γεωμετρία της ρωγμής, χρησιμοποιείται η ρητή-έμμεση υβριδική μέθοδος που προτάθηκε από Fries and Baydoun (2012). Αντίθετα με πλήρως έμμεσες περιγραφές, αυτή η μέθοδος μπορεί να ανανεώνει εύκολα τη γεωμετρία της ρωγμής σε τρισδιάστατα προβλήματα, επειδή περιγράφει τη ρωφμή ως ένα πλέγμα τριγωνικών στοιχείων. Ταυτόχρονα, συνεργάζεται με την ΕΜΠΣ, αφού χρησιμοποιεί το ίδιο πλέγμα και προσημασμένες αποστάσεις, οι οποίες αποθηκεύονται στους κόμβους και μετά παρεμβάλονται στο εσωτερικό των στοιχείων, για να υπολογιστούν τα $\phi(\boldsymbol{x})$, $r(\boldsymbol{x})$ και $\theta(\boldsymbol{x})$ σε οποιοδήποτε σημείο. Αφού περιγραφεί η γεωμετρία της ρωγμής με αυτή τη μέθοδο, εκτελείται ανάλυση ΕΜΠΣ για να υπολογιστούν τα πεδία μετατοπίσεων, παραμορφώσεων και τάσεων. Έπειτα η ρωγμή διαδίδεται ανανεώνοντας τις θέσεις των σημείων που ορίζουν το μέτωπο της ρωγμής και προσθέτοντας νέα τρίγωνα.

Μέθοδοι υποφορέων σε υπολογιστικά συστήματα υψηλών επιδόσεων

Η παρούσα διατριβή εστιάζει σε συμπλέγματα υπολογιστών, δηλαδή περιβάλλοντα που αποτελούνται από πολλαπλούς υπολογιστές, καθένας από τους οποίους διαθέτει τους δικούς του επεξεργαστές και μνήμη και επικοινωνεί με τους υπόλοιπους μέσω τοπικού δικτύου (LAN), ώστε να επιλυθεί απο κοινού ένα υπολογιστικό πρόβλημα. Τα συμπλέγματα υπολογιστών τυπικά



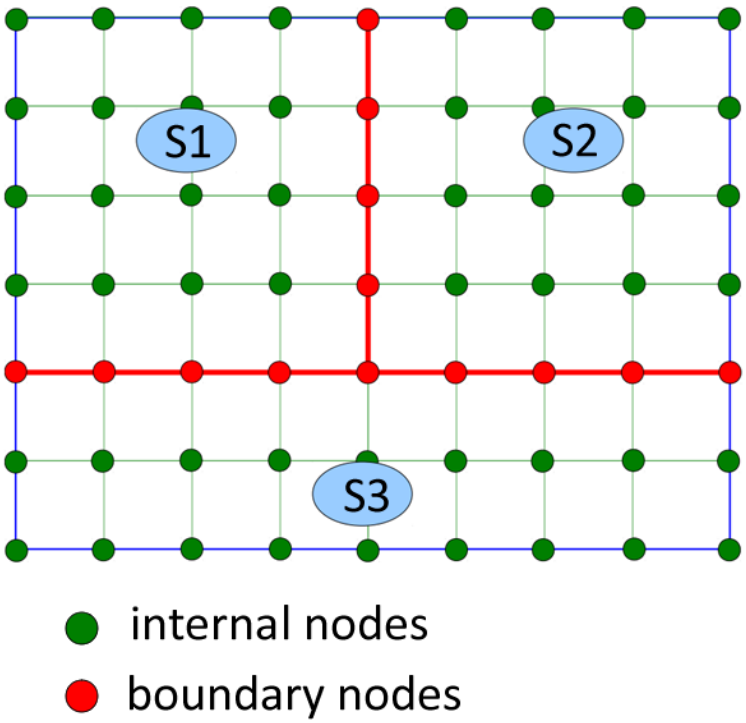
Σχήμα 0.9: Σύμπλεγμα υπολογιστών

χρησιμοποιούν υβριδική κατανεμημένη μνήμη, όπου κάθε υπολογιστής έχει τη δική του μνήμη, αλλά αυτή μοιράζεται μεταξύ των επεξεργαστών του, όπως φαίνεται στο σχήμα 0.9. Επομένως, κάθε υπολογιστής είναι μηχάνημα κοινής μνήμης, αλλά το σύστημα όλων των δικυτωμένων υπολογιστών διαθέτει κατανεμημένη μνήμη. Σε αυτό το υβριδικό σύστημα η επικοινωνία μεταξύ επεξεργστών του ίδιου υπολογιστή είναι πολύ γρηγορότερη από ότι μεταξύ επεξεργαστών που ανήκουν σε διαφορετικούς υπολογιστές. Συνολικά τα συμπλέγματα υπολογιστών έχουν τα ακόλουθα πλεονεκτήματα:

- Απόδοση. Τα προγράμματα εκτελούνται παράλληλα.
- Κλιμακωσιμότητα. Η υπολογιστική ισχύς και διαθέσιμη μνήμη μπορούν να αυξάνονται διαρκώς, προσθέτοντας νέους υπολογιστούς που έχουν τη δική τους μνήμη και επεξεργαστές.
- Χαμηλό οικονομικό κόστος. Το ίδιο επιθυμητό επίπεδο υπολογιστικής ισχύος μπορεί να επιτευχθεί πολύ πιο φθηνά με ένα σύμπλεγμα συνηθισμένων υπολογιστών, από ότι με έναν μόνο υπολογιστή υψηλών προδιαγραφών.
- Αξιοπιστία. Η αστοχία ή συντήρηση ενός ή περισσοτέρων υπολογιστών δεν απαγορεύει τη λειτουργία του υπόλοιπου συστήματος, αλλά απλά μειώνει προσωρινά την απόδοσή του.

Σε αυτή τη διατριβή, προτείνονται δύο μέθοδοι υποφορέων, οι FETI-DP και P-FETI-DP, για την επίλυση του γραμμικού συστήματος της εξίσωσης (34). Αυτοί οι επιλύτες υψηλή κλιμακωσιμότητα και μπορούν να υλοποιηθούν αποδοτικά σε περιβάλλοντα υψηλών επιδόσεων, όπως

συμπλέγματα υπολογιστών. Ως μέθοδοι υποφορέων, διαιρούν τον καθολικό φορέα σε πολλαπλούς υποφορείς, τους οποίους επεξεργάζονται ανεξάρτητα και παράλληλα. Οι β.ε. χωρίζονται σε συνοριακούς β.ε., που αντιστοιχούν σε κόμβους στο σύνορο μεταξύ δύο ή περισσοτέρων υποφορέων, και εσωτερικούς β.ε.που αντιστοιχούν σε κόμβους που ανήκουν σε ένα μόνο υποφορέα, όπως φαίνεται στο σχήμα 0.10.



Σχήμα 0.10: Εσωτερικοί και συνοριακοί κόμβοι υποφορέων.

P-FETI-DP

Στη μέθοδο P-FETI-DP ο πίνακας δυσκαμψίας, το διάνυσμα μετατοπίσεων και το διάνυσμα δυνάμεων χωρίζονται σε μέρη που αντιστοιχούν στους εσωτερικούς (δείκτης i) και συνοριακούς (δείκτης b) β.ε.:

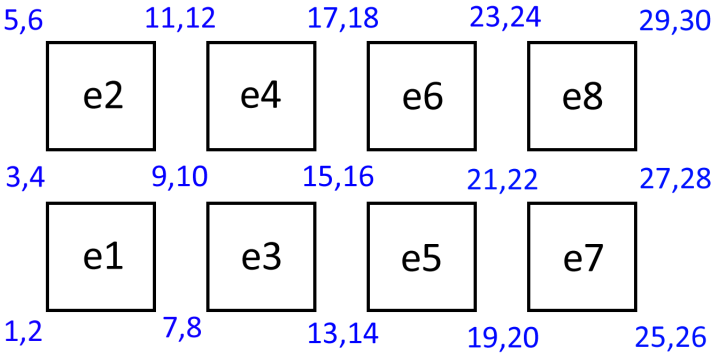
$$\boldsymbol{K}^s = \begin{bmatrix} \boldsymbol{K}^s_{ii} & \boldsymbol{K}^s_{ib} \\ (\boldsymbol{K}^s_{ib})^T & \boldsymbol{K}^s_{bb} \end{bmatrix} \quad \boldsymbol{u}^s = \begin{bmatrix} \boldsymbol{u}^s_i \\ \boldsymbol{u}^s_b \end{bmatrix} \quad \boldsymbol{f}^s = \begin{bmatrix} \boldsymbol{f}^s_i \\ \boldsymbol{f}^s_b \end{bmatrix} \tag{35}$$

Μετά από στατική συμπύκνωση των εσωτερικών β.ε., το συμπλήρωμα Schur $\boldsymbol{S}^s_{bb}$ για τον $\boldsymbol{K}^s_{ii}$ κάθε υποφορέα και το αντίστοιχο διάνυσμα δυνάμεων $\boldsymbol{z}^s_b$ είναι

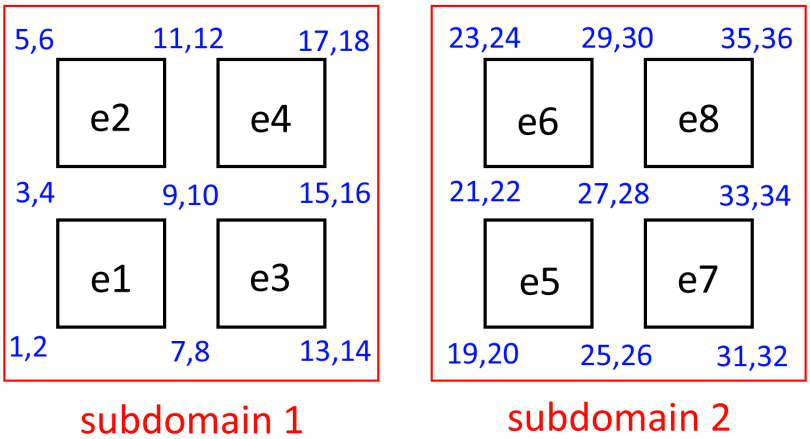
$$\boldsymbol{S}^s_{bb} = \boldsymbol{K}^s_{bb} - (\boldsymbol{K}^s_{ib})^T (\boldsymbol{K}^s_{ii})^{-1} \boldsymbol{K}^s_{ib} \tag{36}$$

$$\boldsymbol{z}_b^s = \boldsymbol{f}_b^s - (\boldsymbol{K}_{ib}^s)^T (\boldsymbol{K}_{ii}^s)^{-1} \boldsymbol{f}_b^s \tag{37}$$

Ο επεκταμένος φορέας ορίζεται ως μία δομή που περιέχει όλους τους β.ε., αλλά κάθε συνοριακός β.ε. εμφανίζεται πολλές φορές, συγκεκριμένα μία φορά για κάθε υποφορέα όπου ανήκει, όπως φαίνεται στο σχήμα 0.11. Αντίθετα ο καθολικός φορέας περιέχει μία φορά κάθε β.ε. του μοντέλου.



(α′) Καθολικός φορέας. Κάθε β.ε. εμφανίζεται μία φορά.



(β′) Επεκταμένος φορέας. Κάθε β.ε. εμφανίζεται μία φορά για κάθε υποφορέα όπου ανήκει.Εδώ ταυτίζονται οι εξής β.ε.: $13 \equiv 19$, $14 \equiv 20$, $15 \equiv 21$, $16 \equiv 22$, $17 \equiv 23$, $18 \equiv 24$

Σχήμα 0.11

Οι πίνακες και τα διανύσματα του επεκταμένου φορέα αποτελούνται από τους αντίστοιχους πίνακες και διανύσματα των υποφορέων. Π.χ.

$$\boldsymbol{u}_b^e = \begin{bmatrix} \boldsymbol{u}_b^1 \\ \vdots \\ \boldsymbol{u}_b^{n_s} \end{bmatrix} \quad \hat{\boldsymbol{z}}_b^e = \begin{bmatrix} \boldsymbol{z}_b^1 \\ \vdots \\ \boldsymbol{z}_b^{n_s} \end{bmatrix} \quad \boldsymbol{K}_{bb}^e = \begin{bmatrix} \boldsymbol{K}_{bb}^1 & & \\ & \ddots & \\ & & \boldsymbol{K}_{bb}^{n_s} \end{bmatrix} \quad \boldsymbol{K}_{bi}^e = \begin{bmatrix} \boldsymbol{K}_{bi}^1 & & \\ & \ddots & \\ & & \boldsymbol{K}_{bi}^{n_s} \end{bmatrix} \tag{38}$$

$$(\boldsymbol{K}_{ii}^e)^{-1} = \begin{bmatrix} (\boldsymbol{K}_{ii}^1)^{-1} & & \\ & \ddots & \\ & & (\boldsymbol{K}_{ii}^{n_s})^{-1} \end{bmatrix} \quad \boldsymbol{S}_{bb}^e = \begin{bmatrix} \boldsymbol{S}_{bb}^1 & & \\ & \ddots & \\ & & \boldsymbol{S}_{bb}^{n_s} \end{bmatrix} \tag{39}$$

Στην συνηθισμένη διατύπωση της P-FETI-DP, οι μετατοπίσεις, που αντιστοιχούν σε όλους τους συνοριακούς β.ε. του καθολικού φορέα, συγκεντρώνονται σε ένα διάνυσμα $\boldsymbol{u}_b$ μήκους n_b. Η αντιστοίχηση μεταξύ $\boldsymbol{u}_b$ και $\boldsymbol{u}_b^s$ γίνεται με δυαδικούς πίνακες $\boldsymbol{L}_b^s$, οι οποίοι έχουν στοιχεία 0, 1 και διαστάσεις διμενσιονς $(n_b^s \times n_b)$, όπου n_b^s και n_b είναι ο αριθμός των συνοριακών β.ε. του υποφορέα s και του καθολικού φορέα, αντίστοιχα:

$$\boldsymbol{u}_b^s = \boldsymbol{L}_b^s \boldsymbol{u}_b \tag{40}$$

Η επικοινωνία μεταξύ των υποφορέων πραγματοποιείται με πράξεις απεικόνισης-συμπύκνωσης των διανυσμάτων δυνάμεων που αντιστοιχούν στους υποφορείς. Π.χ. για τα διανύσματα $\boldsymbol{y}_b^s = \boldsymbol{S}_{bb}^s \cdot \boldsymbol{u}_b^s$:

$$\boldsymbol{y}_b = \sum_{s=1}^{n_s} (\boldsymbol{L}_b^s)^T \boldsymbol{y}_b^s \tag{41}$$

Ωστόσο, σε αυτή τη διατριβή προτείνεται μια εναλλακτική διατύπωση που αποφεύγει καθολικά διανύσματα και πίνακες. Η επικοινωνία γίνεται απευθείας μεταξύ των υποφορέων, αντί να χρησιμοποιούνται καθολικά διανύσματα. Για το σκοπό αυτό, οι πίνακες απεικόνισης $\boldsymbol{L}_b^s$, μεταξύ υποφορέα-καθολικού φορέα, αντικαθίστανται με πίνακες απεικόνισης $\boldsymbol{M}_b^{st}$, μεταξύ υποφορέα-υποφορέα. Για κάθε ζεύγος υποφορέων (s,t), ένας δυαδικός πίνακας χωρίς πρόσημο, δηλαδή πίνακς με τιμές 0, 1, $\boldsymbol{M}_b^{st}$ $(n_b^s \times n_b^t)$ ορίζεται, ο οποίος απεικονίζει τους συνοριακούς β.ε. του υποφορέα t στους συνοριακούς β.ε. του s. Δύο υποφορείς ορίζονται ως γειτονικοί, αν έχουν κοινούς συνοριακούς κόμβους και άρα β.ε. Αν δύο υποφορείς s,t δεν είναι γειτονικοί, τότε δεν έχουν κοινούς β.ε. και οι αντίστοιχοι πίνακες απεικόνισης είναι $\boldsymbol{M}_b^{st} = 0$, $\boldsymbol{M}_b^{ts} = 0$. Από την άλλη, για τον ίδιο υποφορέα s: $\boldsymbol{M}_b^{ss} = I$. Οι πράξεις απεικόνισης-συμπύκνωσης από διανύσματα υποφορέων σε καθολικό διάνυσμα εκτελούνται με κατανεμημένο τρόπο. Για το διάνυσμα $\hat{\boldsymbol{y}}_b^s$ ενός δεδομένου υποφορέα s, η πρόσθεση των κοινών στοιχείων με άλλους υποφορείς μπορεί να γίνει ως

$$\boldsymbol{y}_b^s = \sum_{\substack{t=1,\cdots n_s \\ t \neq s}} \boldsymbol{M}_b^{st} \hat{\boldsymbol{y}}_b^s \tag{42}$$

Ο αντίστοιχος πίνακς επεκταμένου φορέα $\boldsymbol{M}_b^e$ $(n_b^e \times n_b^e)$ ορίζεται ως

$$\boldsymbol{M}_b^e = \begin{bmatrix} \boldsymbol{M}_b^{11} & \boldsymbol{M}_b^{12} & \cdots & \boldsymbol{M}_b^{1n_s} \\ \boldsymbol{M}_b^{21} & \boldsymbol{M}_b^{22} & \cdots & \boldsymbol{M}_b^{2n_s} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{M}_b^{n_s1} & \boldsymbol{M}_b^{n_s2} & \cdots & \boldsymbol{M}_b^{n_sn_s} \end{bmatrix} \tag{43}$$

και η πρόσθεση όλων των διανυσμάτων του επεκταμένου φορέα γίνεται ως

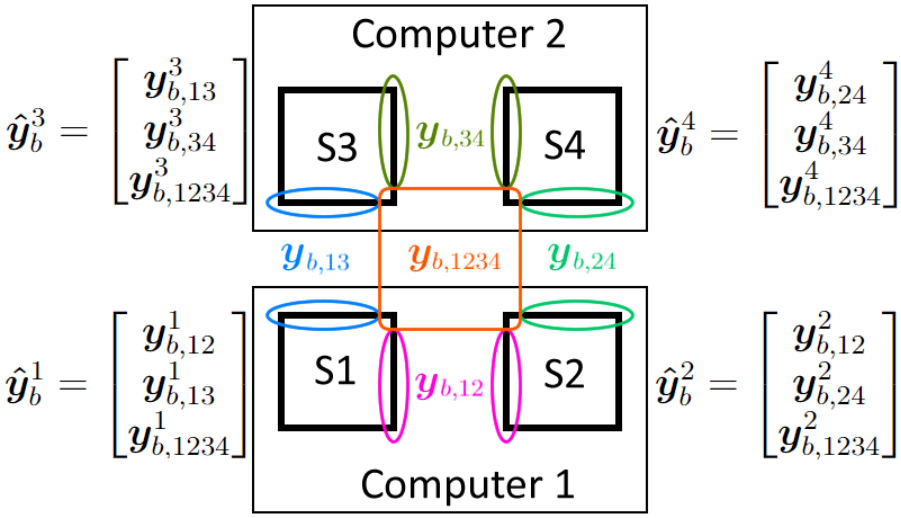
$$\boldsymbol{y}_b^e = \begin{bmatrix} \boldsymbol{y}_b^1 \\ \vdots \\ \boldsymbol{y}_b^{n_s} \end{bmatrix} = \boldsymbol{M}_b^e \hat{\boldsymbol{y}}_b^e \tag{44}$$

όπου $\boldsymbol{y}_b^e$ $(n_b^e \times 1)$ είναι ένα διάνυσμα δυνάμεων για τον επεκταμένο φορέα, το οποίο περιέχει πολλαπλές εμφανίσεις των ίδιων στοιχείων με το καθολικό διάνυσμα $\boldsymbol{y}_b$ $(n_b \times 1)$. Σύμφωνα με αυτή τη διατύπωση, το γραμμικό σύστημα που εκφράζει το συνοριακό πρόβλημα της P-FETI-DP είναι

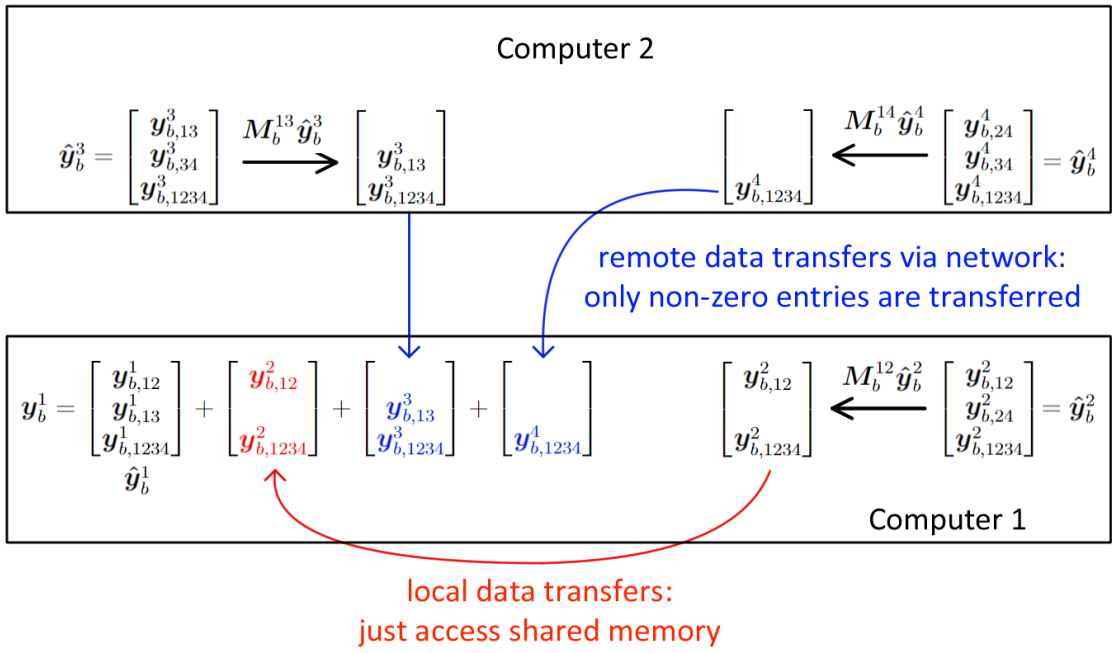
$$\boldsymbol{M}_b^e \boldsymbol{S}_{bb}^e \boldsymbol{x}_b^e = \boldsymbol{M}_b^e \hat{\boldsymbol{z}}_b^e \tag{45}$$

Αυτό το γραμμικό σύστημα είναι συμμετρικό, θετικά ορισμένο και λύνεται με επαναληπτικό αλγόριθμο όπως η μέθοδος Προσταθεροποιημένων Συζυγών Κλίσεων (ΠΣΚ). Η προτεινόμενη διατύπωση μπορεί να υλοποιηθεί αποδοτικά σε συστήματα κατανεμημένης μνήμης όπως συμπλέγματα υπολογιστών. Όλα τα διανύσματα και οι πίνακες των υποφορέων αποθηκεύονται μόνο στους υπολογιστές που ανατίθενται για τους εκάστοτε υποφορείς. Γειτονικού υποφορείς που ανήκουν στον ίδιο υπολογιστή ανταλλάσσουν στοιχεία διανυσμάτων με αμελητέο κόστος, αφού αυτά τα δεδομένα βρίσκονται στον ίδιο χώρο μνήμης. Γειτονικοί υποφορείς που ανήκουν σε διαφορετικούς υποφορείς ανταλλάσσουν στοιχεία διανυσμάτων μέσω δικτύου. Αυτή η επικοινωνία περιορίζεται μόνο στα στοιχεία κοινών β.ε. μεταξύ υποφορέων και κατανέμεται ομοιόμορφα εντός τουδικτύου, χωρίς μεταφορές μνήμης σε ένα κεντρικό σημείο που θα προκαλούσαν συμφόρηση. Επιπλέον, αντί να εκτελούνται καθολικές πράξεις σε ένα υποφορέα, ενώ οι υπόλοιποι είναι αδρανής, η προτεινόμενη διατύπωση κατανέμει τις πράξεις ομοιόμορφα σε όλους τους υπολογιστές. Ένα παράδειγμα δίνεται στο σχήμα 0.12. Χρησιμοποιώντας τους πίνακες $\boldsymbol{M}_b^{1s}$ απομονώνονται τα στοιχεία σε συνοριακούς β.ε. που είναι κοινοί μεταξύ του υποφορέα 1 και κάθε άλλου υποφορέα s. Έπειτα μεταφέρονται μόνο αυτά τα κοινά στοιχεία στο χώρο μνήμης που βρίσκεται ο υποφορέας 1 και τελικά προστίθενται στο διάνυσμα του υποφορέα 1. Παράλληλα με τον υποφορέα 1, εκτελούνται τα ίδια βήματα για όλους τους άλλους υποφορείς.

Οι μέθοδοι FETI-DP, P-FETI-DP χρησιμοποιούν το ίδιο αραιό πρόβλημα, το οποίο είναι ένα πολύ μικρότερο βοηθητικό σύστημα που επιταχύνει τη σύγκλιση, συζεύγοντας τους υπολογισμούς των υποφορέων και διαδίδοντας ομοιόμορφα το σφάλμα σε κάθε επανάληψη ΠΣΚ. Για τη δημιουργία του αραιού προβλήματος ορίζονται οι γωνιακοί κόμβοι, οι οποίοι αποτελούν υποσύνολο των συνοριακών κόμβων και βρίσκονται στην αρχή και το τέλος κάθε πλευράς κάθε υποφορέα, όπως φαίνεται στο σχήμα 0.13. Οι γωνιακοί β.ε., που αντιστοιχούν σε αυτούς τους γωνιακούς κόμβους, συμβολίζονται με το δείκτη c. Οι υπόλοιποι β.ε. κάθε υποφορέα



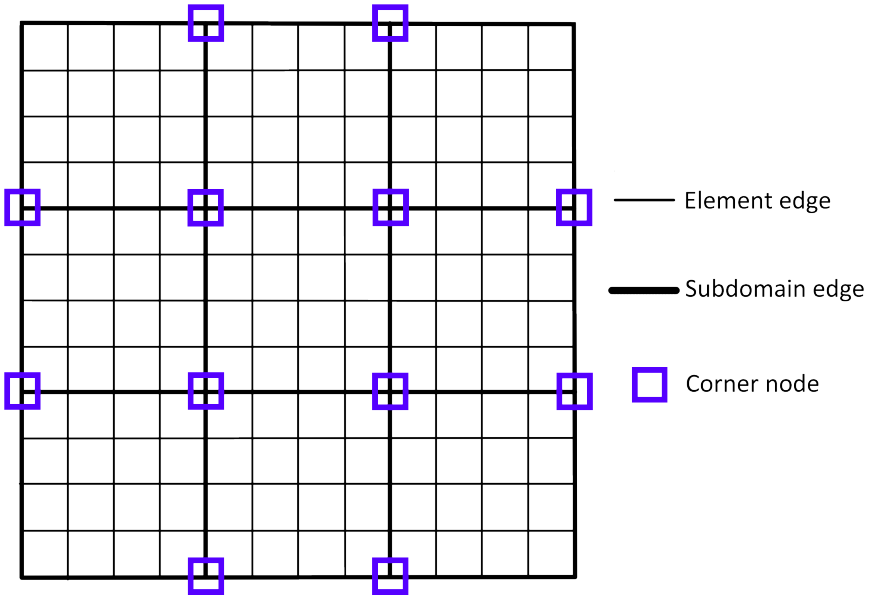
(α′)



(β′)

Σχήμα 0.12: Κατανεμημένη μορφή των πράξεων απεικόνισης-συμπύκνωσης, στην περίπτωση γειτονικών υποφορέων και μετατοπίσεων στους κοινούς συνοριακούς β.ε.

συμβολίζονται με το δείκτη r. Ο πίνακας δυσκαμψίας $\boldsymbol{K}^s$, το διάνυσμα μετατοπίσεων $\boldsymbol{u}^s$ και το



Σχήμα 0.13: Ορισμός γωνιακών κόμβων.

διάνυσμα δυνάμεων $\boldsymbol{f}^s$ κάθε υποφορέα s χωρίζονται ως εξής:

$$\boldsymbol{K}^s = \begin{bmatrix} \boldsymbol{K}^s_{rr} & \boldsymbol{K}^s_{rc} \\ (\boldsymbol{K}^s_{rc})^T & \boldsymbol{K}^s_{cc} \end{bmatrix} \quad \boldsymbol{u}^s = \begin{bmatrix} \boldsymbol{u}^s_r \\ \boldsymbol{u}^s_c \end{bmatrix} \quad \boldsymbol{f}^s = \begin{bmatrix} \boldsymbol{f}^s_r \\ \boldsymbol{f}^s_c \end{bmatrix} \tag{46}$$

Οι μετατοπίσεις, που αντιστοιχούν σε όλους τους γωνιακούς β.ε. τους καθολικού φορέα, συγκεντρώνονται στο διάνυσμα $\boldsymbol{u}_c$ με μήκος n_c. Η αντιστοίχηση μεταξύ $\boldsymbol{u}_c$ και $\boldsymbol{u}^s_c$ γίνεται με δυαδικούς πίνακες $\boldsymbol{L}^s_c$ που έχουν στοιχεία 0, 1 και διαστάσεις $(n^s_c \times n_c)$, όπου n^s_c και n_c είναι το πλήθος των γωνιακών β.ε. του υποφορέα s και του καθολικού φορέα, αντίστοιχα.

$$\boldsymbol{u}^s_c = \boldsymbol{L}^s_c \boldsymbol{u}_c \tag{47}$$

Μετά από στατική συμπύκνωση των υπόλοιπων β.ε., το συμπλήρωμα Schur $\boldsymbol{S}^s_{cc}$ για τον $\boldsymbol{K}^s_{rr}$ κάθε υποφορέα και το αντίστοιχο διάνυσμα δυνάμεων $\boldsymbol{z}^s_c$ είναι

$$\boldsymbol{S}^s_{cc} = \boldsymbol{K}^s_{cc} - (\boldsymbol{K}^s_{rc})^T (\boldsymbol{K}^s_{rr})^{-1} \boldsymbol{K}^s_{rc} \tag{48}$$

$$\boldsymbol{z}^s_c = \boldsymbol{f}^s_c - (\boldsymbol{K}^s_{rc})^T (\boldsymbol{K}^s_{rr})^{-1} \boldsymbol{f}^s_r \tag{49}$$

Έπειτα αυτοί οι πίνακες και διανύσματα υποφορέων προστίθενται για να παραχθούν ο καθολικός γωνιακός πίνακας δυσκαμψίας $\boldsymbol{S}_{cc}$ και το αντίστοιχο καθολικό διάνυσμα δυνάμεων $\boldsymbol{z}_c$

$$\boldsymbol{S}_{cc} = \sum_{s=1}^{n_s} (\boldsymbol{L}^s_c)^T \boldsymbol{S}^s_{cc} \boldsymbol{L}^s_c = \sum_{s=1}^{n_s} (\boldsymbol{L}^s_c)^T \left(\boldsymbol{K}^s_{cc} - (\boldsymbol{K}^s_{rc})^T (\boldsymbol{K}^s_{rr})^{-1} \boldsymbol{K}^s_{rc} \right) \boldsymbol{L}^s_c \tag{50}$$

$$\boldsymbol{z}_c = \sum_{s=1}^{n_s} (\boldsymbol{L}_c^s)^T \boldsymbol{z}_c^s = \sum_{s=1}^{n_s} (\boldsymbol{L}_c^s)^T \left(\boldsymbol{f}_c^s - (\boldsymbol{K}_{rc}^s)^T (\boldsymbol{K}_{rr}^s)^{-1} \boldsymbol{f}_r^s \right) \tag{51}$$

Με κατάλληλη επιλογή των γωνιακών κόμβων, ο πίνακας $\boldsymbol{K}_{rr}^s$ είναι αντιστρέψιμος. Το αραιό πρόβλημα ορίζεται ως το ακόλουθο γραμμικό σύστημα

$$\boldsymbol{S}_{cc} \cdot \boldsymbol{x}_c = \boldsymbol{y}_c \tag{52}$$

το οποίο λύνεται χρησιμοποιώντας άμεσο αλγόριθμο, όπως είναι η παραγοντοποίηση Cholesky, αφού είναι πολύ μικρότερο και πρέπει να λύνεται μία φορά σε κάθε επανάληψη ΠΣΚ του συνοριακού προβλήματος. Προκειμένου να μειωθούν οι επαναλήψεις ΠΣΚ που απαιτούνται για το συνοριακό πρόβλημα, η P-FETI-DP χρησιμοποιεί τον προσταθεροποιητή

$$\begin{aligned}
(\widetilde{\boldsymbol{A}}_{bb}^e)^{-1} &= \boldsymbol{A}_{bb}^{r,e} + \boldsymbol{A}_{bc}^e \boldsymbol{A}_{cc}^e \boldsymbol{A}_{cb}^e \\
\boldsymbol{A}_{bb}^{r,e} &= \boldsymbol{M}_b^e \boldsymbol{W}_b^e \left(\boldsymbol{N}_{r,b}^e \right)^T \left(\boldsymbol{K}_{rr}^e \right)^{-1} \boldsymbol{N}_{r,b}^e \boldsymbol{W}_b^e \\
\boldsymbol{A}_{cb}^e &= \Big(- \boldsymbol{K}_{cr}^e \left(\boldsymbol{K}_{rr}^e \right)^{-1} \boldsymbol{N}_{r,b}^e + \boldsymbol{N}_{c,b}^e \Big) \boldsymbol{W}_b^e \\
\boldsymbol{A}_{bc}^e &= \left(\boldsymbol{N}_{c,b}^e \right)^T - \boldsymbol{M}_b^e \boldsymbol{W}_b^e \left(\boldsymbol{N}_{r,b}^e \right)^T \left(\boldsymbol{K}_{rr}^e \right)^{-1} \boldsymbol{K}_{rc}^e \\
\boldsymbol{A}_{cc}^e &= \boldsymbol{L}_c^e \boldsymbol{S}_{cc}^{-1} \left(\boldsymbol{L}_c^e \right)^T
\end{aligned} \tag{53}$$

όπου

- $\boldsymbol{W}_b^s$ είναι ο αντίστροφος διαγώνιου πίνακα, τα στοιχεία του οποίου είναι οι πολλαπλότητες των συνοριακών β.ε. του υποφορέα s.
- $\boldsymbol{N}_{r,b}^s$ είναι δυαδικός πίνακας(0, 1 ως στοιχεία) με διαστάσεις $(n_r^s \times n_b^s)$ που απεικονίζει τους συνοριακούς β.ε. ενός υποφορέα στους υπόλοιπους β.ε. του ίδιου υποφορέα.
- $\boldsymbol{N}_{c,b}^s$ είναι δυαδικός πίνακας(0, 1 ως στοιχεία) με διαστάσεις $(n_c^s \times n_b^s)$ που απεικονίζει τους συνοριακούς β.ε. ενός υποφορέα στους γωνιακούς β.ε. του ίδιου υποφορέα.
- Ο πολλαπλασιασμός του αντιστρόφου $\boldsymbol{S}_{cc}^{-1}$ με ένα διάνυσμα είναι ισοδύναμος με επίλυση του αραιού προβλήματος.

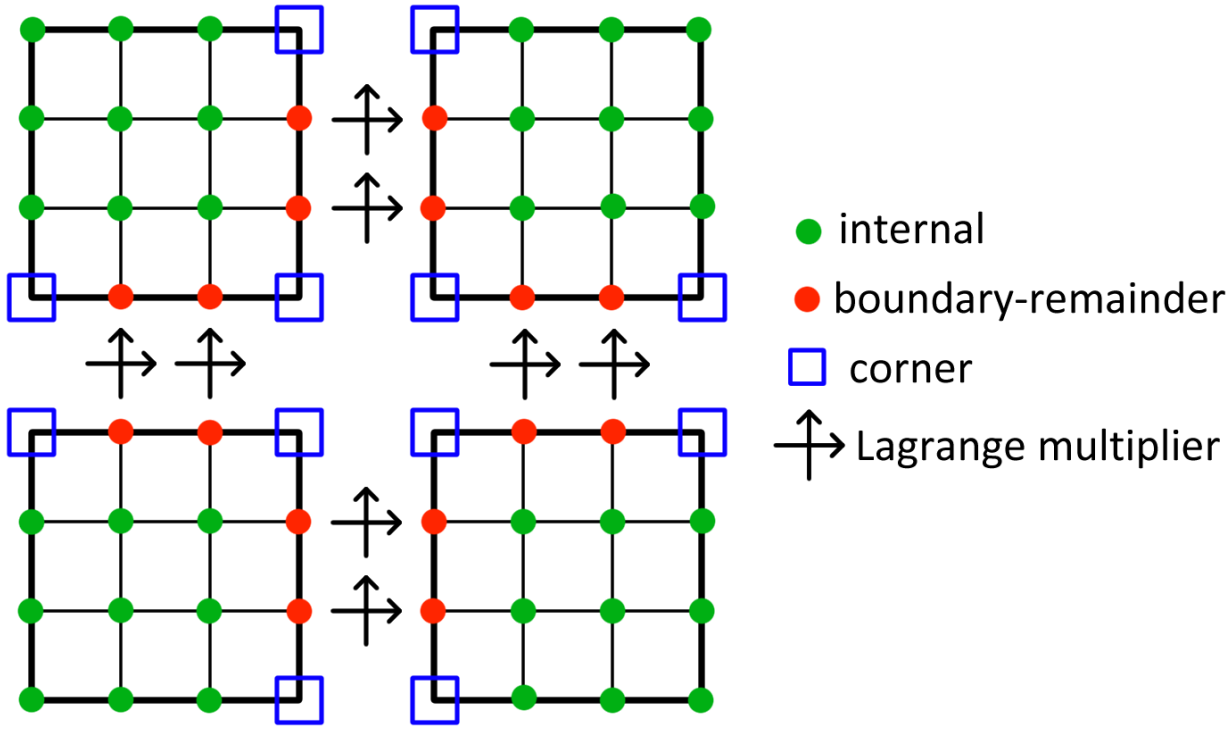
FETI-DP

Στη μέθοδο FETI-DP, οι υπόλοιποι β.ε. χωρίζονται περεταίρω σε εσωτερικούς β.ε., οι οποίοι αφορούν κόμβους που ανήκουν σε ένα μόνο υποφορέα και συμβολίζονται με το δείκτη i, και συνοριακούς-υπόλοιπους β.ε., οι οποίοι συμβολίζονται με το δείκτη b_r και αντιστοιχούν σε συνοριακούς κόμβους που δεν είναι γωνιακοί:

$$\boldsymbol{K}_{rr}^s = \begin{bmatrix} \boldsymbol{K}_{ii}^s & \boldsymbol{K}_{ib_r}^s \\ (\boldsymbol{K}_{ib_r}^s)^T & \boldsymbol{K}_{b_r b_r}^s \end{bmatrix} \quad \boldsymbol{u}_r^s = \begin{bmatrix} \boldsymbol{u}_i^s \\ \boldsymbol{u}_{b_r}^s \end{bmatrix} \quad \boldsymbol{f}_r^s = \begin{bmatrix} \boldsymbol{f}_i^s \\ \boldsymbol{f}_{b_r}^s \end{bmatrix} \tag{54}$$

Η συνέχεια μετατοπίσεων μεταξύ των αποσυνδεδεμένων υποφορέων αποκαθίσταται με την εφαρμογή συνθηκών συμβιβαστότητας για εμφανίσεις του ίδιου συνοριακού β.ε. σε διαφορετικούς υποφορείς. Αυτές οι εξισώσεις συγκεντρώνονται σε προσημασμένους δυαδικούς πίνακες $\boldsymbol{B}_r^s$, που έχουν στοιχεία με τιμές 0, 1, -1 και διαστάσεις $(n_\lambda \times n_r^s)$, όπου n_λ είναι το πλήθος εξισώσεων συμβιβαστότητας για τον καθολικό φορέα και n_r^s ο αριθμός των υπόλοιπων β.ε. του υποφορέα s:

$$\sum_{s=1}^{n_s} \boldsymbol{B}_r^s \boldsymbol{u}_r^s = \boldsymbol{0} \tag{55}$$



Σχήμα 0.14: Πολλαπλασιαστές Langrange εφαρμοζόμενοι στους συνοριακούς-υπόλοιπους β.ε. των υποφορέων.

Για να λυθούν οι καθολικές εξισώσεις ισορροπίας $\boldsymbol{K}\boldsymbol{u} = \boldsymbol{f}$ παρουσία αυτών των περιορισμών, εφαρμόζονται πολλαπλασιαστές Langrange $\boldsymbol{\lambda}$ στους συνοριακούς-υπόλοιπους β.ε., ώστε να επιβληθεί συμβιβαστότητα μετατοπίσεων, όπως φαίνεται στο σχήμα 0.14. Πρέπει να σημειωθεί ότι στους γωνιακούς β.ε. δεν εφαρμόζονται πολλαπλασιαστές Langrange. Ακολούθως, οι εξισώσεις ισορροπίας γραφονται ως

$$\boldsymbol{K}_{rr}^s \boldsymbol{u}_r^s + \boldsymbol{K}_{rc}^s \boldsymbol{L}_c^s \boldsymbol{u}_c + (\boldsymbol{B}_r^s)^T \boldsymbol{\lambda} = \boldsymbol{f}_r^s \tag{56}$$

$$\sum_{s=1}^{n_s}(\boldsymbol{L}_c^s)^T(\boldsymbol{K}_{rc}^s)^T\boldsymbol{u}_r^s+\sum_{s=1}^{n_s}(\boldsymbol{L}_c^s)^T(\boldsymbol{K}_{cc}^s)^T\boldsymbol{L}_c^s\boldsymbol{u}_c^s=\sum_{s=1}^{n_s}(\boldsymbol{L}_c^s)^T\boldsymbol{f}_c^s \tag{57}$$

Με κατάλληλη επιλογή των γωνιακών β.ε., ο πίνακας $\boldsymbol{K}_{rr}^s$ είναι αντιστρέψιμος και η εξίσωση (56) γραφεται ως:

$$\boldsymbol{u}_r^s=(\boldsymbol{K}_{rr}^s)^{-1}\left(\boldsymbol{f}_r^s-(\boldsymbol{B}_r^s)^T\boldsymbol{\lambda}-\boldsymbol{K}_{rc}^s\boldsymbol{L}_c^s\boldsymbol{u}_c\right) \tag{58}$$

Σύμφωνα με την κατανεμημένη διατύπωση που προτείνεται στην παρούσα διατριβή, κάθε πίνακας $\boldsymbol{B}_r^s$ $(n_\lambda\times n_r^s)$ αντικαθίσταται με έναν άλλο προσημασμένο δυαδικό πίνακα $\boldsymbol{C}_r^s$ $(n_\lambda^s\times n_r^s)$, ο οποίος απεικονίζει τους υπόλοιπους β.ε. του υποφορέα s στους πολλαπλασιαστές Langrange του υποφορέα s, αντί να απεικονίζει στους καθολικούς πολλαπλασιαστές Langrange, όπως ο πίνακας $\boldsymbol{B}_r^s$. Επιπροσθέτως, παρόμοια με την P-FETI-DP, δυαδικοί μη προσημασμένοι πίνακες $\boldsymbol{M}_\lambda^{st}$ χρησιμοποιούνται για να απεικονίσουν τους πολλαπλασιαστές Langrange του υποφορέα t στους πολλαπλασιαστές Langrange του υποφορέα s. Για τον επεκταμένο φορέα:

$$\boldsymbol{M}_\lambda^e=\begin{bmatrix}\boldsymbol{M}_\lambda^{11} & \boldsymbol{M}_\lambda^{12} & \cdots & \boldsymbol{M}_\lambda^{1n_s}\\ \boldsymbol{M}_\lambda^{21} & \boldsymbol{M}_\lambda^{22} & \cdots & \boldsymbol{M}_\lambda^{2n_s}\\ \vdots & \vdots & \ddots & \vdots\\ \boldsymbol{M}_\lambda^{n_s1} & \boldsymbol{M}_\lambda^{n_s2} & \cdots & \boldsymbol{M}_\lambda^{n_sn_s}\end{bmatrix} \tag{59}$$

Συνδυάζοντας τις εξισώσεις (58, 55, 57) και χρησιμοποιώντας τους δυαδικούς πίνακες απεικόνισης $\boldsymbol{C}_r^e$ και $\boldsymbol{M}_\lambda^e$, καταλήγουμε σε

$$\left(\boldsymbol{F}_{Irr}^e+\boldsymbol{F}_{Irc}^e\boldsymbol{A}_{cc}^e\boldsymbol{F}_{Icr}^e\right)\boldsymbol{\lambda}^e=\boldsymbol{d}_r^e-\boldsymbol{F}_{Irc}^e\boldsymbol{A}_{cc}^e\hat{\boldsymbol{z}}_c^e \tag{60}$$

όπου

$$\begin{aligned}\boldsymbol{F}_{Irr}^e&=\boldsymbol{M}_\lambda^e\boldsymbol{C}_r^e(\boldsymbol{K}_{rr}^e)^{-1}(\boldsymbol{C}_r^e)^T\\ \boldsymbol{F}_{Irc}^e&=\boldsymbol{M}_\lambda^e\boldsymbol{C}_r^e(\boldsymbol{K}_{rr}^e)^{-1}\boldsymbol{K}_{rc}^e\\ \boldsymbol{F}_{Icr}^e&=\boldsymbol{K}_{cr}^e(\boldsymbol{K}_{rr}^e)^{-1}(\boldsymbol{C}_r^e)^T\\ \boldsymbol{A}_{cc}^e&=\boldsymbol{L}_c^e\boldsymbol{S}_{cc}^{-1}(\boldsymbol{L}_c^e)^T\\ \boldsymbol{d}_r^e&=\boldsymbol{M}_\lambda^e\boldsymbol{C}_r^e(\boldsymbol{K}_{rr}^e)^{-1}\boldsymbol{f}_r^e\end{aligned} \tag{61}$$

Το γραμμικό σύστημα της εξίσωσης (60)είναι το συνοριακό πρόβλημα της FETI-DP και λύνεται με τη μέθοδο ΠΣΚ. Μπορούμε να παρατηρήσουμε ότι ο πίνακας $\boldsymbol{A}_{cc}^e$ περιέχει το αραιό πρόβλημα της FETI-DP. Αφού λυθεί το συνοριακό πρόβλημα και βρεθούν οι τιμές των πολλαπλασιαστών Lagrange $\boldsymbol{\lambda}^e$, οι μετατοπίσεις στους γωνιακούς και υπόλοιπους β.ε. υπολογίζονται ως

$$\boldsymbol{u}_c^e=\boldsymbol{A}_{cc}^e\left(\hat{\boldsymbol{z}}_c^e+\boldsymbol{F}_{Icr}^e\boldsymbol{\lambda}^e\right) \tag{62}$$

$$\boldsymbol{u}_r^e=(\boldsymbol{K}_{rr}^e)^{-1}\left(\boldsymbol{f}_r^e-(\boldsymbol{C}_r^e)^T\boldsymbol{\lambda}^e-\boldsymbol{K}_{rc}^e\boldsymbol{u}_c^e\right) \tag{63}$$

Προκειμένου να ελαττωθούν οι επαναλήψεις ΠΣΚ που απαιτούνται για την επίλυση του συνοριακού προβλήματος, η FETI-DP χρησιμοποιεί προσταθεροποιητή με την ακόλουθη γενική μορφή

$$(\widetilde{\boldsymbol{F}}^e_{Irr})^{-1} = \boldsymbol{M}^e_\lambda \boldsymbol{C}^e_{b_r} \boldsymbol{W}^e_{b_r} \widetilde{\boldsymbol{S}}^e_{b_r b_r} \boldsymbol{W}^e_{b_r} (\boldsymbol{C}^e_{b_r})^T \tag{64}$$

όπου $\boldsymbol{W}^s_{b_r}$ είναι ο αντίστροφος διαγωνίου πίνακα, του οποίου τα στοιχεία ταυτίζονται με τις πολλαπλότητες των συνοριακών-υπολοίπων β.ε., και $\boldsymbol{C}^s_{b_r}$ είναι οι στήλες του $\boldsymbol{C}^s_r$ που αντιστοιχούν στους συνοριακούς-υπόλοιπους β.ε. Ανάλογα με τον ορισμό του $\widetilde{\boldsymbol{S}}^e_{b_r b_r}$, μπορούν να εξαχθούν οι ακόλουθοι προσταθεροποιητές:

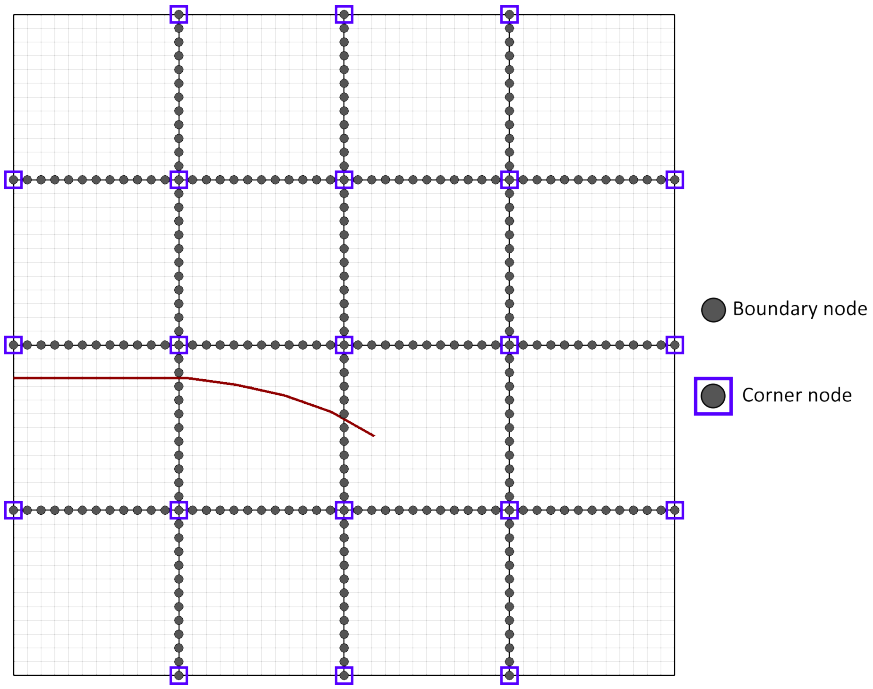
$$\widetilde{\boldsymbol{S}}^e_{b_r b_r} = \begin{cases} \boldsymbol{K}^e_{b_r b_r} - (\boldsymbol{K}^e_{ib_r})^T (\boldsymbol{K}^e_{ii})^{-1} \boldsymbol{K}^e_{ib_r} & \text{προσταθεροποιητής Dirichlet} \\ \boldsymbol{K}^e_{b_r b_r} & \text{προσταθεροποιητής lumped} \end{cases} \tag{65}$$

Ο προσταθεροποιητής Dirichlet χρησιμοποιεί το πλήρες συμπλήρωμα Schur των εσωτερικών β.ε., επομένως έχει μεγαλύτερο υπολογιστικό κόστος, αλλά οδηγεί σε λιγότερες επαναλήψεις ΠΣΚ, σε σύγκριση με τον προσταθεροποιητή lumped.

Μέθοδοι υποφορέων για διάδοση ρωγμών με ΕΜΠΣ

Οι μέθοδοι FETI-DP, P-FETI-DP αναπτύχθηκαν για προβλήματα μηχανικής σε συνεχές μέσο. Ωστόσο, στη θραυστομηχανική μία ή περισσότερες ρωγμές διαδίδονται σε έναν ασυνεχή φορέα και κάποιοι υποφορείς μπορεί να τέμνονται εξ΄ ολοκλήρου από αυτές, οπότε οι αντίστοιχοι πίνακες $\boldsymbol{K}^s_{rr}$ γίνονται μη αντιστρέψιμοι. Ένα παράδειγμα φαίνεται στο σχήμα 0.15, όπου μία ρωγμή διαδίδεται σε δισδιάστατο σώμα και αλληλεπιδρά με τρεις υποφορείς, από τους οποίους οι δύο τεμνονται πλήρως από αυτήν.

Ο εμπλουτισμός της ΕΜΠΣ μοντελοποιεί το άλμα του πεδίου μετατοπίσεων στη ρωγμή, εισάγοντας ασυνεχείς συναρτήσεις βάσης και αντίστοιχους εμπλουτισμένου β.ε. Όταν η ρωγμή τέμνει πλήρως έναν υποφορέα s, οι γραμμές και στήλες του K^s_{rr}, που αντιστοιχούν στους εμπλουτισμένους β.ε., γίνονται γραμμικά εξαρτημένες. Σε αυτήν την περίπτωση, ο υποφορέας διαιρείται ουσιαστικά σε δύο επιπλέοντα τμήματα/μηχανισμούς, τα οποία κινούνται ανεξάρτητα το ένα από το άλλο, όπως φαίνεται στο σχήμα 0.16. Για να λυθεί το πρόβλημα των μη αντιστρέψιμων πινάκων $\boldsymbol{K}^s_{rr}$, προτείνεται η ακόλουθη διαδικασία. Πρώτον, προσδιορίζονται οι γραμμικά εξαρτημένες γραμμές και στήλες του $\boldsymbol{K}^s_{rr}$, εντοπίζοντας τους αντίστοιχους β.ε. Έστω $\mathbb{M}_b$ το σύνολο των συνοριακών β.ε., δηλαδή των β.ε. που ανήκουν σε δύο ή περισσότερους υποφορείς. Οι Heaviside εμπλουτισμένοι β.ε. $\boldsymbol{a}_j$ της εξίσωσης (33) ανήκουν στο σύνολο $\mathbb{M}_H$, ενώ οι εμπλουτισμένοι β.ε. $\boldsymbol{b}^1_k$, που εισάγονται για την πρώτη συνάρτηση αιχμής-ρωγμής F_1 στην εξίσωση (31), ανήκουν στο σύνολο $\mathbb{M}_{T^1}$. Αν ένας υποφορέας τέμνεται πλήρως από μία ρωγμή, τότε οι β.ε. που ανήκουν στο σύνολο $\mathbb{M}_b \cap (\mathbb{M}_H \cup \mathbb{M}_{T^1})$ είναι υπεύθυνοι για τη μοντελοποίηση του άλματος του πεδίου μετατοπίσεων και για την ανάπτυξη εσωτερικών μηχανισμών στον υποφορέα. Αν αυτοί οι β.ε. γίνουν γωνιακοί β.ε., τότε αφαιρούνται οι γραμμικά εξαρτημένες



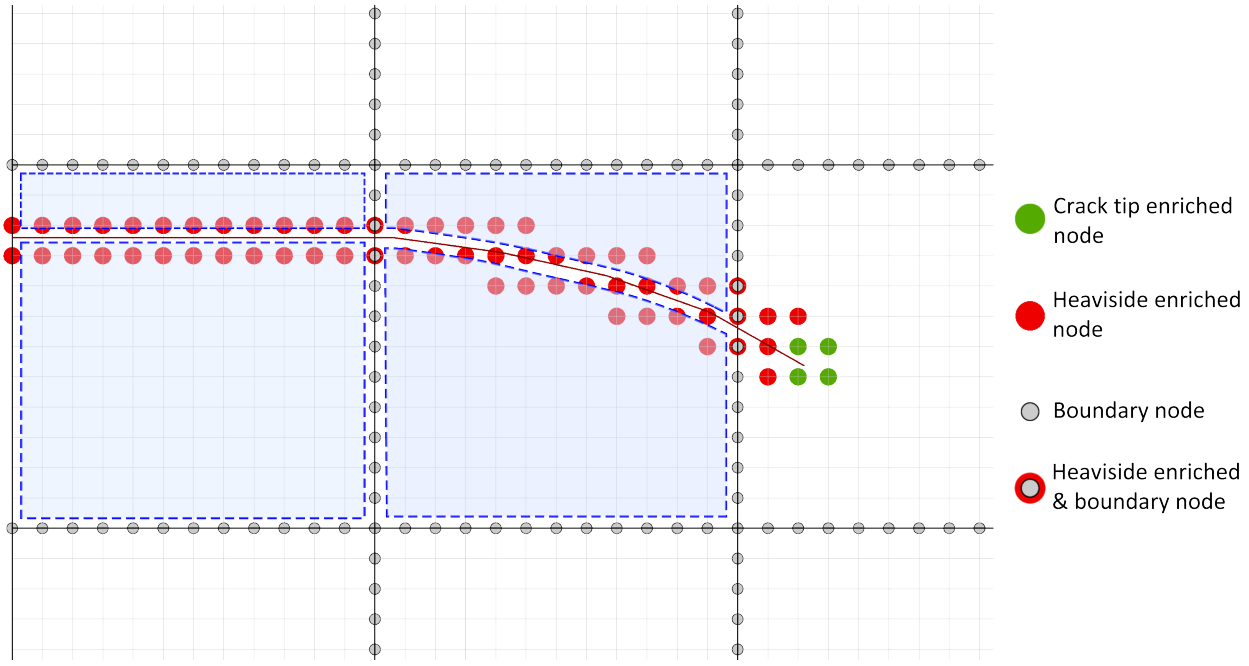
Σχήμα 0.15: Συνοριακοί και γωνιακοί κόμβοι των συμβατικών μεθόδων FETI-DP, P-FETI-DP.

γραμμές/στήλες από τα $\boldsymbol{K}_{rr}^{s}$ και επαναφέρεται η αντιστρεψιμότητα. Αν $\mathbb{M}_{c,std}$ είναι το σύνολο των συμβατικών β.ε., τότε η προτεινόμενη αλλαγή συνίσταται στον καθορισμό του συνόλου

$$\mathbb{M}_c = \mathbb{M}_{c,std} \cup (\mathbb{M}_b \cap \mathbb{M}_H) \cup (\mathbb{M}_b \cap \mathbb{M}_{T^1}) \tag{66}$$

και τη χρήση του στους επιλύτες FETI-DP, P-FETI-DP ως σύνολο γωνιακών κόμβων, όπως φαίνεται στο σχήμα 0.17, αντί για το σύνολο $\mathbb{M}_{c,std}$. Αυτοί οι συνοριακοί β.ε. μεταφέρονται στο μητρώο δυσκαμψίας $\boldsymbol{K}_{cc}^{s}$ του υποφορέα και τελικά στο καθολικό μητρώο δυσκαμψίας $\boldsymbol{S}_{cc}$. Με την προτεινόμενη μετατροπή, οι υποφορείς μπορούν να τέμνονται αυθαίρετα από ρωγμές και να επιλέγονται με κριτήριο την ελαχιστοποίηση των απαιτήσεων μνήμης και τους υπολογιστικού χρόνου.

Μία άλλη δυσκολία στις αναλύσεις διάδοσης ρωγμών με ΕΜΠΣ είναι η κακή κατάσταση των πινάκων δυσκαμψίας, λόγω της σημαντικής διαφοράς μεταξύ των τιμών που αντιστοιχούν στους εμπλουτισμένους από συναρτήσεις αιχμής-ρωγμής β.ε. και των τιμών που αντιστοιχούν στους συμβατικούς και εμπλουτισμένους από Heaviside β.ε. Στο πλάισιο των FETI-DP, P-FETI-DP, η κατάσταση των πινάκων επιβραδύνει την επαναληπτική επίλυση των συνοριακών προβλημάτων και οι συμβατικοί προσταθεροποιητές των παραπάνω επιλυτών δεν επαρκούν για να μειωθούν οι επαναλήψεις. Επομένως προτείνεται η εξής μετατροπή για να απαλειφεί η κακή κατάσταση. Έστω $\mathbb{M}_{T^m}$, $m = 1, \cdots 4$ το σύνολο των $\boldsymbol{b}_k^m$ β.ε. που εισάγονται για κάθες συνάρτηση αιχμής-ρωγμής της εξίσωσης (31). Οι όροι που προκαλούν κακή κατάσταση του συνοριακού

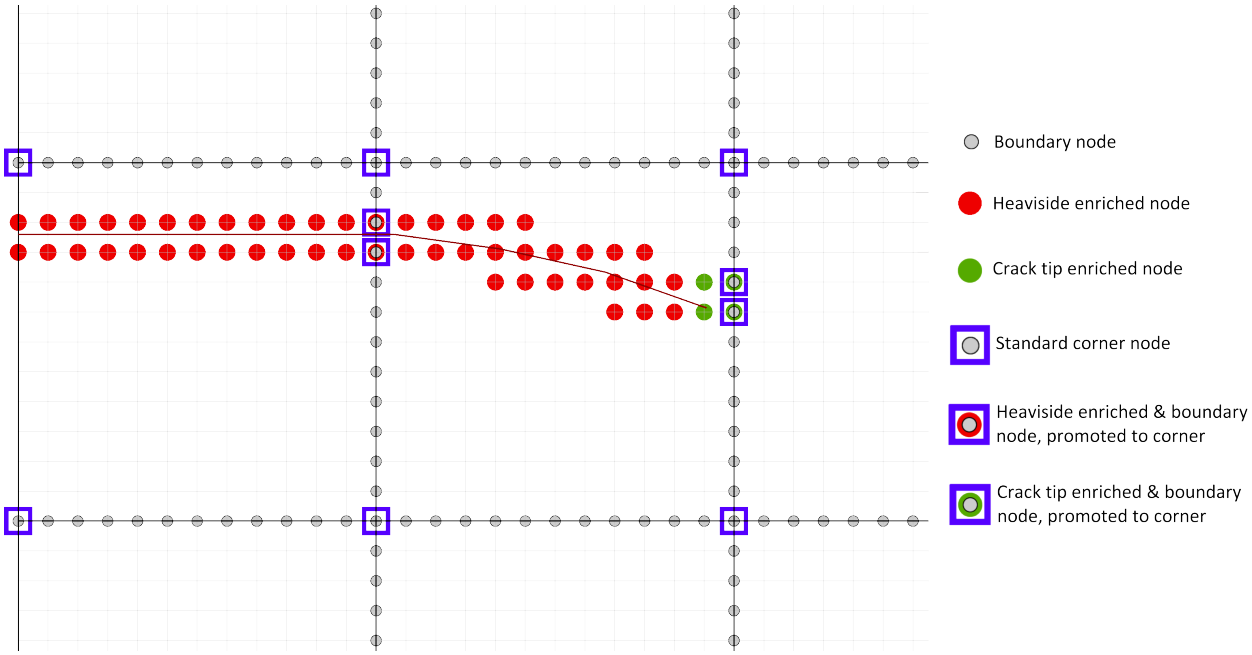


Σχήμα 0.16: Επιπλέοντα τμήματα των υποφορέων που τέμνονται πλήρως από τη ρωγμή.

προβλήματος αντιστοιχούν στους β.ε. που ανήκουν στο σύνολο $\mathbb{M}_b \cap \mathbb{M}_{T^m}$, $m = 1, \cdots 4$. Η προτεινόμενη τεχνική αντιμετωπίζει αυτούς τους β.ε. ως γωνιακούς, όπως φαίνεται στο σχήμα 0.17, βελτιώνοντας έτσι την ικανότητα του αραιού προβλήματος της εξίσωσης (52), να διανέμει το σφάλμα μεταξύ των υποφορέων σε κάθε επανάληψη της ΠΣΚ. Έτσι ο προσταθεροποιητής της P-FETI-DP μπορεί να αντιμετωπίσει την κακή κατάσταση λόγω εμπλουτισμού ΕΜΠΣ, αφού περιέχει το αραιό πρόβλημα. Παρότι ο προσταθεροποιητής της FETI-DP δεν σχετίζεται με γωνιακούς β.ε. και δεν επηρεάζεται από αυτή την μετατροπή, ο δείκτης κατάστασης του πίνακα του συνοριακού προβλήματος $\boldsymbol{F}_{Irr} + \boldsymbol{F}_{Irc}\boldsymbol{S}_{cc}^{-1}\boldsymbol{F}_{Icr}$ βελτιώνεται, επειδή α) το αραιό πρόβλημα συμπεριλαμβάνεται απευθείας σε αυτόν τον πίνακα και β) οι προβληματικοί β.ε. του συνόλου $\mathbb{M}_b \cap \mathbb{M}_{T^m}$, $m = 1, \cdots 4$ αφαιρούνται από αυτόν. Λαμβάνοντας υπόψη της μετατροπή για να αποφεύγονται μη αντιστρέψιμοι $\boldsymbol{K}_{rr}^s$ πίνακες, οι προτεινόμενοι FETI-DP, P-FETI-DP επιλύτες χρησιμοποιούν το σύνολο

$$\mathbb{M}_c = \mathbb{M}_{c,std} \cup (\mathbb{M}_b \cap \mathbb{M}_H) \bigcup_{m=1}^{4} (\mathbb{M}_b \cap \mathbb{M}_{T^m}) \tag{67}$$

ως γωνιακούς β.ε. Χωρίς αυτή τη μετατροπή, οι επίμαχες μεθοδοι υποφορέων παρουσιάζουν σημαντική αύξηση στις επαναλήψεις που απαιτούνται για σύγκλιση. Στα αριθμητικά παραδείγματα παρατηρείται αύξηση ως και 245%. Χρησιμοποιώντας την εξίσωση (67) για ορισμό των γωνιακών β.ε., εξαφανίζεται πλήρως η κακή κατάσταση λόγω ΕΜΠΣ και οι προτεινόμενοι FETI-DP, P-FETI-DP επιλυτών σταματούν να είναι ευαίσθητοι στην θέση και των αριθμό των εμπλου-



Σχήμα 0.17: Οι συνοριακοί κόμβοι που εμπλουτίζονται με συναρτήσεις Heaviside και αιχμής-ρωγμής, μετατρέπονται σε γωνιακούς, για να αποφευχθούν μη αντιστρέψιμοι πίνακες $\boldsymbol{K}_{rr}^{s}$ στις FETI-DP, P-FETI-DP.

τισμένων κόμβων. Επιπλέον, αποφεύγοντας την αύξηση των επαναλήψεων, αποκαθίσταται η κλιμακωσιμότητα των FETI-DP, P-FETI-DP, δηλαδή οι επανλήψεις μειώνονται όσο αυξάνονται οι υποφορείς. Αυτό επιτρέπει την υλοποίηση των προτεινόμενων επιλυτών σε περιβάλλοντα κατανεμημένης μνήμης, όπου η υπολογιστική ισχύς και η διαθέσιμη μνήμη μπορούν να αυξάνονται αυθαίρετα, απλά προσθέτοντας νέους υπολογιστές. Η ικανότητα να ανατίθενται πολλοί υποφορείς σε όλους τους διαθέσιμους επεξεργαστές, χωρίς να αυξάνονται οι επαναλήψεις για σύγκλιση, είναι απαραίτητη για την εκμετάλλευση των συστημάτων κατανεμημένης μνήμης και καθιστά τους προτεινόμενους επιλύτες πολύ ελκυστικούς για την επίλυση προβλημάτων μεγάλης κλίμακας.

Στην περίπτωση ψαθυρής διάδοσης ρωγμών με ΕΜΠΣ, όπου λίγα μόνο στοιχεία των πινάκων δυσκαμψίας αλλάζουν από το ένα βήμα στο επόμενο, είναι δυνατή περεταίρω βελτίωση απόδοσης είναι. Αυτά τα στοιχεί αντιστοιχούν σε εμπλουτισμένους β.ε. κοντά στο μέτωπο της ρωγμής, συγκεκριμένα β.ε. Heaviside και αιχμής-ρωγμής που εισάγονται στο τρέχον βήμα διάδοσης, καθώς και β.ε. αιχμής-ρωγμής που είχαν εισαχθεί στο προηγούμενο βήμα, αλλά στο τρέχον αφαιρούνται. Στους προτεινόμενους επιλύτες FETI-DP, P-FETI-DP σολερς, πολλοί υποφορείς δεν αλληλεπιδρούν με το μέτωπο της ρωγμής σε κάθε βήμα διάδοσης. Στην περίπτωση αυτή, όλοι οι αντίστοιχοι πίνακες, διανύσματα και λοιπά δεδομένα των υποφορέων παραμένουν ίδια με το τελευταίο βήμα που άλλαξαν και μπορούν να επαναχρησιμοποιηθούν. Επίσης, προτείνεται εδώ μία τεχνική επαναρχικοποίησης για τα συνοριακά προβλήματα των FETI-DP, P-FETI-DP,

προκειμένουν να μειωθούν οι επαναλήψεις. Συγκεκριμένα, κατά την επίλυση του συνοριακού προβλήματος της P-FETI-DP, χρειάζεται μια αρχική εκτίμηση του διανύσματος λύσης $\widetilde{\boldsymbol{u}}_b$ για την πρώτη επανάληψη ΠΣΚ. Στο πρώτο βήμα διάδοσης, ως αρχική εκτίμηση χρησιμοποιείται το μηδενικό διάνυσμα $\widetilde{\boldsymbol{u}}_b^{t=0} = \mathbf{0}$. Έστω $\boldsymbol{u}_b^t$ η λύση τους συνοριακού προβλήματος κατά το βήμα t

$$\boldsymbol{u}_b^t = \left[\begin{array}{ccc} \boldsymbol{u}_{std}^t & \boldsymbol{u}_H^t & \boldsymbol{u}_T^t \end{array} \right] \tag{68}$$

όπου $\boldsymbol{u}_{std}^t$ είναι οι μετατοπίσεις στους συμβατικούς β.ε., $\boldsymbol{u}_H^t$ στους Heaviside συνοριακούς β.ε. και $\boldsymbol{u}_T^t$ στους συνοριακούς β.ε. αιχμής-ρωγμής. Τότε οι μετατοπίσεις στους συμβατικούς και Heaviside β.ε. μπορούν να χρησιμοποιηθούν ως αρχική εκτίμηση της λύσης του επόμενου βήματος διάδοσης

$$\widetilde{\boldsymbol{u}}_b^{t+1} = \left[\begin{array}{cccc} \widetilde{\boldsymbol{u}}_{std}^{t+1} & \widetilde{\boldsymbol{u}}_{H1}^{t+1} & \widetilde{\boldsymbol{u}}_{H2}^{t+1} & \widetilde{\boldsymbol{u}}_T^{t+1} \end{array} \right] = \left[\begin{array}{cccc} \boldsymbol{u}_{std}^t & \boldsymbol{u}_H^t & \mathbf{0} & \mathbf{0} \end{array} \right] \tag{69}$$

όπου $\widetilde{\boldsymbol{u}}_{H1}^{t+1}$ αντιστοιχούν στους Heaviside συνοριακούς β.ε. που υπάρχουν κατά τα βήματα t και $t+1$, ενώ $\widetilde{\boldsymbol{u}}_{H2}^{t+1}$ αντιστοιχούν στους Heaviside συνοριακούς β.ε. που προστίθενται κατά το βήμα $t+1$. Αντίθετα, τα $\widetilde{\boldsymbol{u}}_{H2}^{t+1}$ και $\widetilde{\boldsymbol{u}}_T^{t+1}$ αντιστοιχούν σε β.ε. που δεν είναι παρόντες κατά το βήμα t, οπότε η αρχική εκτίμηση για αυτούς είναι $\mathbf{0}$. Παρόμοια τεχνική επαναρχικοποίησης προτείνεται για την FETI-DP, όπου το συνοριακό πρόβλημα εκφράζεται σε όρους πολλαπλασιαστών Lagrange, που εφαρμόζονται στου συνοριακούς-υπόλοιπους β.ε., αντί για όλους τους συνοριακούς β.ε. στην P-FETI-DP. Χησιμοποιώντας την εξίσωση (67), όλοι οι εμπλουτισμένοι συνοριακοί-υπόλοιποι β.ε. μετατρέπονται σε γωνιακούς, επομένως οι αντίστοιχοι πολλαπλασιαστές Lagrange αφαιρούνται. Αυτό διευκολύνει την τεχνική επαναρχικοποίησης, αφού εξασφαλίζει ότι το συνοριακό πρόβλημα της FETI-DP εμπλέκει μόνο συμβατικούς συνοριακούς-υπόλοιπους β.ε., οι οποίει είναι ίδιοι σε όλα τα βήματα διάδοσης της ρωγμής. Έτσι το συνολικό διάνυσμα λύσης κατά ένα βήμα διάδοσης μπορεί να επαναχρησιμοποιηθεί ως αρχική εκτίμηση για το επόμενο βήμα

$$\widetilde{\boldsymbol{\lambda}}^{t+1} = \boldsymbol{\lambda}^t \tag{70}$$

Στα αριθμητικά παραδείγματα της παρούσας διατριβής, η τεχνική επαναρχικοποίησης μπορεί να ελαττώσειτις επαναλήψεις ΠΣΚ κατά 40% για τη P-FETI-DP και 37% για τη FETI-DP. Σε συνδυασμό με την επαναχρησιμοποίηση πινάκων και δεδομένων των υποφορέων από προηγούμενα βήματα, ο χρόνος που απαιτείται για επίλυση μπορεί να μειωθεί κατά 50% και για τις δύο μεθόδους.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

1.1 XFEM for modeling heat transfer in nanocomposite materials

The development of polymer materials with high thermal conductivity is of great interest to many industrial sectors, including chemical, construction, electronics, automotive, aerospace and energy industries. As most polymers are thermal insulators, it is necessary to add thermally conductive fillers in order to improve their thermal properties. Towards this direction, carbon-based nanomaterials such as graphene (G), graphite (GRF) and carbon nanotubes (CNTs), stand out as ideal candidates for inclusions in reinforced composites (RCs) due to their extraordinary thermal conductivity. In particular, CNTs are expected by the scientific community to have the highest thermal conductivity amongst conductive materials (Che et al., 2017; H. Chen et al., 2016), reaching thermal conductivity values between 2000-6000 $Wm^{-1}K^{-1}$, which is higher than diamond, graphite and carbon-fibers (Berber et al., 2000; Hussain et al., 2017; W.-b. Zhang et al., 2015).

Based on the above, several authors postulated that CNTs can make a polymer nanocomposite thermally conductive (Ajorloo et al., 2019; Liao et al., 2015; G. Zhang et al., 2010), provided that these are properly aligned. However, experimental investigations showed that by adding CNTs to polymers, the increase in the effective conductivity of the composite is significantly below theoretical expectations (Gojny et al., 2006; Konstantopoulos et al., 2021; Moisala et al., 2006; Xie, 2007; Yunsheng et al., 2006). This discrepancy between theoretical predictions and experimental measurements can be explained by the interface thermal conductivity or conductance (Chalopin et al., 2009; Marcos-Gómez et al., 2010; Yvonnet et al., 2011), which is a phenomenological parameter used for modeling imperfect interfaces. The impact of imperfect interfaces in the effective conductivity of CNT reinforced composites, has also been investigated in several simulations using finite element methods (Kamiński & Ostrowski, 2021) or atomistic approaches (Carlborg et al., 2008; Kumar & Murthy, 2009; Saha & Shi, 2007; Zhong & Lukes, 2006), which reported very low conductance (equivalently, high resistivity) values, mostly attributed to the phonon scattering mechanism arising at the

interfaces between CNTs and the surrounding polymer matrices (Marconnet et al., 2013).

This 'in silico' characterization of heterogeneous composite materials is becoming more and more accessible nowadays, with the increase in computational resources. Simulation-based material design can efficiently replace the experimental procedure and, thus, minimize the required time and cost for the development of new materials with desirable target properties. On the downside, accurately modeling the behavior of such materials is a very challenging computational mechanics problem due to the complexity of the physical phenomena arising in multiple scales (nano-, micro-, macroscale) and the uncertainties in the material parameters and microstructural geometry.

When working with composite materials, effective properties are usually pursued by homogenizing their physical properties in some volume average quantities. Based on a known topology of the composite's microstructure and the physical-mechanical properties of its components, effective properties are usually extracted either analytically, using formulas, or numerically, using modern computational tools. Analytical solutions are mostly limited to simplified cases with ideal geometries, while numerical methods can handle more complex and realistic cases. A key concept in the numerical analysis of composite materials is the representative volume element (RVE), which is the smallest volume over which a measurement can be made that will yield a value representative of the whole. Typically, the behavior of RVEs is estimated using the finite element method (FEM) and through the process of homogenization (Geers et al., 2010; Miehe & Koch, 2002), the macroscopic properties of the composite material are obtained. This generic approach is applicable to all types of material characterization, thermal (Wu et al., 2013), mechanical (Papadopoulos & Impraimakis, 2017) and electrical (Seidel & Puydupin-Jamin, 2011).

In the framework of FEM, the process of generating RVEs of multi-phase heterogeneous materials relies on the use of advanced mesh generators to design the mesh around the different phases. In this case, the finite element mesh is generated such that the element boundaries have to conform to the geometry of the interface between the phases. This approach is often time consuming and difficult to improve. More importantly, it leads to an excessive number of elements, since such conforming meshes need significant refinement to accurately discretize the inclusion geometries, especially the more complex ones, which introduces additional mesh sensitivity issues and leads to a substantial increase in the computing cost of the FEM analysis. In addition, the zones, where two or more phases are in contact, require appropriate surface elements to capture their interaction. For instance, cohesive zones (Papadopoulos et al., 2017; Savvas & Papadopoulos, 2014), coherent interfaces with surface energy effects (Yvonnet et al., 2008) and equivalent eigenstrains for a coating layer at the interfaces (Benvenuti, 2014) are introduced for mechanical interaction. Similarly, conductive heat transfer across the boundaries of the phases exhibiting Kapitza thermal resistance is modeled in Yvonnet et al. (2011). To address these limitations of FEM, several automatic mesh generation techniques have been developed (Golias & Dutton, 1997; Talischi et al., 2012; Y. Zhang et al., 2010), but they could still face significant challenges for complex microstructural topologies.

The first half of this dissertation focuses on simulating conductive heat transfer in het-

erogenous multi-phase materials with complex microstructural topologies, using the extended finite element method (XFEM). XFEM was originally developed by Belytschko and Black (1999) and Daux et al. (2000) in an effort to remedy the meshing issues posed by FEM in crack propagation analysis, but has also been successfully applied to problems involving material interfaces (Beese et al., 2018; Moës et al., 2003; Sukumar et al., 2001). Moreover, XFEM is particularly suitable for problems requiring repeated simulations for different realizations of the RVE's microstructure, since there is no need to generate a new finite element mesh at each simulation (Savvas et al., 2014). In the applications investigated in this dissertation, the interactions between different material phases must be modelled, as well as the existence of interfacial resistance at their boundaries, which produces discontinuities in the temperature field. To this end, an XFEM formulation is developed in two publications: Bakalakos et al. (2020) and Bakalakos et al. (2022), with a novel enrichment strategy and an efficient implicit representation of the complex microstucture of a composite material's RVE. Even though the proposed method is demonstrated in heat conduction problems, it can be straightforwardly extended to other similar type problems, such us electrical conduction or elasticity problems with cohesive interfaces.

The proposed methodology offers a series of advantages over traditional FEM modeling. First, it does not require advanced mesh generators and complex refined meshes, since it employs a simple and coarse, thus cost-efficient, mesh for XFEM, while the complex geometry of the material interfaces is represented with the Level Set Method (LSM) on a second mesh that is much finer and related to the coarse XFEM mesh. LSM is an implicit method to describe curves and geometries, which was developed in Osher and Sethian (1988) and first used in conjuction with XFEM in Stolarska et al. (2001) and Sukumar et al. (2001). The double-mesh approach adopted in this dissertation allows high accuracy in the representation of inclusions with arbitrarily complex 2D and 3D geometries, without increasing the computing cost of XFEM analysis. Furthermore, the discontinuities in the temperature field are captured via appropriate enrichment functions used in the XFEM approximation, instead of using specialized surface boundary elements, and different interfacial resistances can be assigned to all phase surfaces that are in contact. Similar approaches to treat boundary interactions in the context of XFEM can be found in Yvonnet et al. (2011), where the Kapitza thermal resistance at a simple interface between two materials was modeled. Also, Bansal et al. (2019) investigates the case of multiple inclusions which interact with the matrix but not with each other. Extending previous approaches, in this research, a novel XFEM enrichment scheme is developed for the case of junctions, that is, areas where multiple boundaries with different interface resistance intersect.

Moreover, another property of CNTs is that they act as heterogeneous nucleating agents for polymer crystallizing along the interface. This induces the formation of a transcrystalline layer that surrounds the CNT in a process known as CNT-induced polymer crystallization (S. Zhang et al., 2008). This layer has improved thermal properties compared to the amorphous polymer, which affects the overall thermal conductivity of the composite. On the other hand, its formation is associated with great uncertainties regarding its shape around the individual CNTs. Therefore, modeling heat transmission between CNTs and surrounding matrix by

employing imperfect interfaces with equivalent conductance that takes into account this phenomenon is an alternative, reasonably reliable approach.

Due to the aforementioned complexities, interfacial thermal conductance cannot be accurately predicted through experimental setups or numerical models based on methods of molecular dynamics. In this dissertation, the conductance of the interface between CNTS and the polymer matrix is treated as a model parameter and its value is inferred based on experimental measurements of other directly measurable quantities. The general premise behind model parameter identification is to calibrate the parameters of a detailed numerical model so as to agree with experimental measurements. This approach has been extensively used for the purposes of material characterization to identify the values of parameters that are not directly observable (Bogdanor et al., 2015; Bogdanor et al., 2013; Pyrialakos et al., 2021; Savvas et al., 2020). For instance, in Pyrialakos et al. (2021) the mechanical interactions properties between polymers and CNTs at the microscopic level were obtained, based on deformation measurements of macroscopic structures comprised of the CNT-reinforced polymer. In this dissertation, the focus is on modeling heat transfer in CNT-reinforced polyethylene (PE), and the unknown value of the interfacial conductance is calibrated using the experimental results on the effective conductivity of the reinforced composites provided by Konstantopoulos et al. (2021). Once the value of the interface conductance is inferred, the data-informed model is then employed for the investigation of optimal CNT configurations in the parent material that will provide upper bounds on the effective thermal conductivity CNT-reinforced polymers can achieve.

1.2 XFEM for crack propagation analysis in high performance environments

The extended finite element method used so far was actually originally developed by Belytschko and Black (1999) and Moës et al. (1999) for analyzing brittle crack propagation in 2D problems. In this context, the polynomial approximation space of XFEM is enriched with problem specific functions, in order to accurately model the discontinuous displacement field and the singular strain/stress field around a crack. These enriched basis functions correspond to additional degrees of freedom (DOFs), which are introduced around a crack and are called enriched DOFs to differentiate from the standard DOFs that express nodal displacements. As elaborated in Moës et al. (1999), the mesh is independent from the crack geometry and does not have to conform to it. Thus, simple structured or unstructured meshes can be used without the need of remeshing, whenever the crack grows, and mapping the displacement field between the old and new meshes, which would result in lower accuracy.

Ever since, XFEM has become one of the most popular methods to simulate fracture phenomena, with brittle as well as cohesive (Moës & Belytschko, 2002) or plastic (Elguedj et al., 2006) material behavior and even thermo-mechanical coupling (Duflot, 2008). Branching and intersecting cracks can also be modeled with the addition of specialized enrichment

functions Daux et al. (2000). Apart from crack propagation, with the selection of appropriate, problem-specific enrichment functions, XFEM has been extended to a wide range of applications, such as modeling composite materials, as discussed in the previous paragraphs, fluid-structure interaction (Gerstenberger & Wall, 2008), contact problems (Khoei et al., 2009), topology optimization (Abdi et al., 2017; Villanueva & Maute, 2014), where it can produce efficient structures without requiring very fine meshes, and probabilistic shape optimization (Georgioudakis et al., 2017).

XFEM is almost always coupled with the Level Set Method (LSM) for representing the geometry of discontinuities, such as cracks or material interfaces. LSM was originally proposed by Osher and Sethian (1988) in order to implicitly describe moving curves or surfaces as the zero contour of a signed distance function (distance of points to the curve/surface), evaluated over a fixed mesh. This approach can be naturally combined with XFEM, since it uses the same mesh to determine which elements and nodes interact with the discontinuities, as well as providing a very efficient way to determine on which side of the discontinuity an arbitrary point lies. The first combination of XFEM and LSM was developed in Stolarska and Chopp (2003) for simple 2D cracks, but has since been extended to 3D problems (Gravouil et al., 2002). A number of improvements have emerged, such as the fast marching method (Chopp & Sukumar, 2003) and the vector level set (Ventura et al., 2003). Last but not least, Fries and Baydoun (2012) propose a hybrid implicit-explicit approach, to combine both the straightforward update of an explicit crack description and the compatibility of an implicit description with XFEM, while treating both 2D and 3D cracks in a consistent manner.

Despite its general success, XFEM has certain shortcomings, especially with regards to the solution of the resulting linear systems of equations. The local enrichment scheme of XFEM introduces additional DOFs only around a crack, which avoids redundantly increasing the size of the resulting linear system. However, the enriched basis functions, particularly those introduced to model the singular strain/stress field around the crack front, cause the stiffness matrix to become very ill-conditioned. Therefore, iterative solvers exhibit slow convergence for crack propagation problems. Although direct solvers are not affected as much from this ill-conditioning, they are inefficient in 3D problems, where the bandwidth of the stiffness matrix is much higher, resulting in dramatically increased time and memory required for factorizing it. In order to address the need for solvers that can efficiently solve linear systems resulting from XFEM, various specialized solvers have been developed.

Bechet et al. (2005) developed a preconditioning scheme, based on the Cholesky decomposition of certain node-level submatrices of the global stiffness matrix. This preconditioner can then be used to improve the convergence rate of standard iterative solvers and is independent from the type of enrichment function used. Similarly, in Lang et al. (2014), a low-cost geometric preconditioner constructed from the nodal basis functions is proposed to eliminate the ill-conditioning caused by the Heaviside enrichment in problems with material interfaces. As far as direct solvers are concerned, Pais et al. (2012) implemented an exact reanalysis Cholesky solver which updates the factorized matrix at each crack propagation step, instead of rebuilding and refactorizing at the global level. Another reanalysis-type algorithm is featured in Feng and Han (2019), where the transfer operations of a geometric multigrid solver

are established at the beginning of the analysis and then reused at each crack propagation step. The method presented in Gravouil et al. (2008) was also based on geometric multigrid, but added finer mesh patches around small cracks. Moreover, the algebraic multigrid solver of Hiriyur et al. (2012) modifies the sparsity pattern of the prolongator operator to prevent interpolation across cracks. In contrast, Gerstenberger and Tuminaro (2013) introduced a simple modification of algebraic multigrid, in order to use black-box AMG software.

Domain decomposition methods (DDM) are widely considered as the most computationally efficient solvers for large-scale problems, particularly in parallel computing architectures. The first DDM for XFEM crack propagation was proposed in Wyart et al. (2008) with the aim of reusing existing FEM software for problems with small isolated cracks, rather than developing a high performance solver for demanding problems with cracks propagating throughout the whole domain. This technique was based on the FETI method (Farhat & Roux, 1991), using only two subdomains, a large uncracked subdomain, assigned to a general purpose FEM software and a smaller subdomain, located around a crack which is modelled with XFEM. A more performance-oriented approach was proposed by Menk and Bordas (2011), where the domain was separated into one subdomain containing all standard DOFs and multiple subdomains containing the enriched ones. Cholesky factorization was applied to stiffness matrices of enriched subdomains and QR factorization to matrices connecting them with the large monolithic subdomain with standard DOFs. The resulting DD matrix was used as a preconditioner, which was effective at reducing the number of iterations in 2D problems, but did not scale well, since the convergence rate decreased when increasing the number of subdomains.

Furthermore, Waisman and Berger-Vergiat (2013) implemented a multiplicative Schwarz domain decomposition preconditioner to accelerate the convergence of a generalized minimum residual solver in 2D problems. The domain was partitioned into one uncracked subdomain, which was treated with an algebraic multigrid approach, and many smaller subdomains defined around cracks, which were concurrently solved with direct methods. A similar approach for 2D crack propagation problems was presented by X. Chen and Cai (2022), where the preconditioner was based on an additive Schwarz DD solver. This method used LU factorization for subdomains with enriched DOFs and incomplete LU for subdomains with standard DOFs. Even though the cracked subdomains were still dependent on the locations of cracks, multiple uncracked subdomains could be used. Neither Waisman and Berger-Vergiat (2013) nor X. Chen and Cai (2022) scaled well with the number of subdomains, since by increasing the subdomains, an increase on the required iterations was observed.

In this dissertation, the modification of two well-established domain decomposition solvers, namely the FETI-DP (Farhat et al., 2000) and P-FETI-DP (Fragakis & Papadrakakis, 2003), is proposed in order to solve the linear systems resulting from 3D XFEM crack propagation analysis. Instead of decomposing the domain into cracked and uncracked subdomains, the domain is partitioned into an arbitrary number of load-balanced subdomains, independently from the location of cracks, while treating both standard and enriched DOFs consistently. The customizable coarse problem of FETI-DP and P-FETI-DP enables the introduction of

XFEM-specific modifications that eliminate the ill-conditioning caused by the enrichment functions, as well as any singularities that the cracks may induce as they propagate throughout subdomains. The reduced bandwidth of subdomain-level stiffness matrices, afforded by the flexible partitioning, results in a drastic decrease of computing time and memory requirements for solving large-scale 3D crack propagation problems. Furthermore, the subdomain-level matrices and the linear system solution of one crack propagation step are reused, in order to reduce the computing effort and iterations of the next step. In contrast to all previous DDM implementations for XFEM crack propagation, the solvers proposed in this work exhibit excellent numerical scalability, when increasing the number of subdomains.

This scalability is essential for efficiently implementing them in modern high performance computing environments, where the available memory and processing power can be arbitrarily increased by including additional multicore CPUs and GPUs in distributed memory systems. In order to take advantage of these computer clusters, this dissertation modifies the equations of the original FETI-DP and P-FETI-DP methods, altering the communication pattern between subdomains. Specifically, algebraic operations, involving matrices and vectors that contain terms from all subdomains, are replaced with equivalent operations between neighboring subdomains only. As a result, memory transfers are minimized and distributed evenly across the network of processors, while global operations, which require a centralized processor to gather and process data from all subdomains, thus becoming a possible bottleneck, are avoided.

1.3 Outline

Besides this introductory chapter, the rest of this dissertation is organized in 6 chapters, outlined as follows:

Chapter 2 presents the proposed XFEM methodology for simulating conductive heat transfer in composites with complex geometries. After presenting the boundary value problem of heat transfer, the procedure of traditional FEM is explained, along with its limitations. Then XFEM is used to solve this problem, with simple non-conforming meshes, while taking into account the thermal resistance of interfaces between different material phases. The novel enrichment strategy to capture the discontinuous temperature field near junctions of 3 or more material phases is elaborated and the final algebraic equations of XFEM are extracted. Special attention has been given to the numerical integration in this discontinuous medium, as well as along the material interfaces. Finally, an LSM variation that uses a dedicated mesh for geometric operations is developed, in order to achieve the desired accuracy in representing material interfaces, without increasing the computational cost of XFEM.

Chapter 3 presents a series of numerical applications, where the methodology of chapter 2 is employed. Initially, the method of computational homogenization is presented to extract the macroscopic conductivity from a micro-scale RVE. Then, the proposed XFEM-based numerical model is validated against FEM in a synthetic benchmark and against existing results from the literature on heat conduction in multigrain materials. Specifically, poly-

crystalline silicene is studied, a material consisting of many silicene crystallites (grains) of varying size and orientation. Subsequently the proposed formulation is applied to the more demanding case of estimating the effective conductivity of a polymeric material reinforced with carbon nanotubes. The effect of the thermal conductance of interfaces between different materials on the effective thermal conductivity of the material is investigated in 2D and 3D problems. Moreover, the unknown values of the parameters of the numerical model are calibrated using experimental measurements in polyethylene-CNT nanocomposites and optimal CNT configurations that maximize thermal conductivity are sought.

Chapter 4 reviews crack propagation analysis in the framework of XFEM. The strong and weak form of the boundary value problem are posed and the problem-specific Heaviside and crack-tip enrichment functions, used in the XFEM formulation, are described. A hybrid explicit-implicit method for representing 2D and 3D cracks, which was proposed in Fries and Baydoun (2012), is explained and then some modifications are introduced, in order to make it more robust. Finally, two alternative ways to predict the crack propagation path, after the XFEM solution of each step, are examined.

Chapter 5 is concerned with algorithms used to solve the systems of algebraic equations resulting from FEM and XFEM analysis. First, well-established solvers developed for FEM are reviewed. These belong to three categories: direct solvers, iterative solvers and domain decomposition methods (DDM). Then DMM solvers, based on FETI-DP and P-FETI-DP, are proposed for the linear systems resulting from crack propagation analysis in the framework of XFEM. The problem-specific difficulties, namely singular matrices in subdomains intersected by cracks and severe ill-conditioning due to the XFEM enrichment, are eliminated by introducing appropriate modifications to the coarse problem of FETI-DP and P-FETI-DP. Furthermore, optimizations with respect to the computing cost and convergence rate are proposed, by reusing computations and solutions of one propagation step to improve the performance of the solvers in the next steps. Moreover, an efficient implementation of the proposed FETI-DP and P-FETI-DP solvers is developed for computer clusters, where multiple computers, each with its own processors and memory, cooperate over a network to solve a common problem. Specifically, the original equations of both solvers are altered to reduce data transfers between computers, which host subsets of the total subdomains, and avoid possible bottlenecks, such as global operations.

Chapter 6 investigates the performance of the proposed solvers in 3D crack propagation applications and compares them to optimized direct and iterative solvers, as well as a solver developed specifically for XFEM crack propagation. These comparisons are performed using a single computer with a multicore CPU and fully parallel execution of each solver. The results justify that the proposed solvers are significantly more efficient in all cases and that they scale well as the number of subdomains is increased. Finally, the performance and parallel scalability of the proposed FETI-DP and P-FETI-DP solvers, when executed on a computer cluster, are investigated.

Chapter 7 presents a summary of the contributions of this research. Lastly, Appendix A reviews some of the basics of FEM, needed to fully comprehend the proposed XFEM methodology of chapter 2.

Chapter 2

XFEM for composites

2.1 Heat transfer analysis with FEM

Before investigating how to model the thermal behaviour of composite materials, this section reviews the basic principles of heat transfer analysis with the standard Finite Element Method (FEM), when there is only one homogeneous material.

2.1.1 The boundary value problem

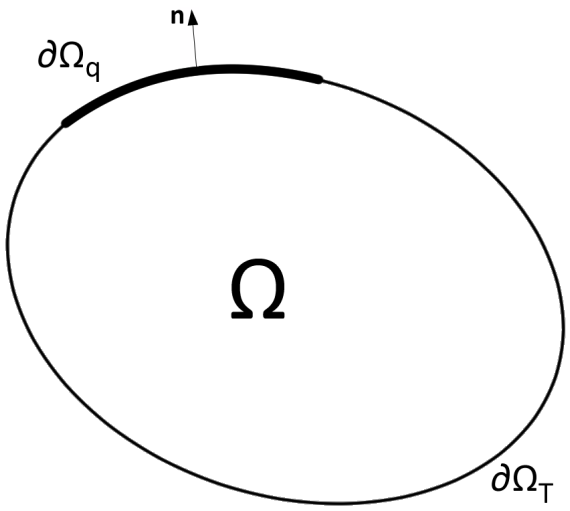


Figure 2.1: Heat transfer in domain with a single material phase

Let Ω be the domain of a body composed of a single material and $\partial\Omega$ its boundary, as illustrated in figure 2.1. The external boundary $\partial\Omega$ has an outward normal vector $\boldsymbol{n}$ and is

divided into complementary parts $\partial\Omega_T$ and $\partial\Omega_q$, such that $\partial\Omega = \partial\Omega_T \cup \partial\Omega_q$. Dirichlet and Neumann boundary conditions are applied respectively on $\partial\Omega_T$ and $\partial\Omega_q$

$$\begin{aligned} T &= \bar{T} \quad \text{on } \partial\Omega_T, \\ \boldsymbol{q}\cdot\boldsymbol{n} &= -\bar{q}_n \quad \text{on } \partial\Omega_q. \end{aligned} \tag{2.1}$$

where $T = T(\boldsymbol{x})$ is the (scalar) temperature field and $\boldsymbol{q} = \boldsymbol{q}(\boldsymbol{x})$ is the heat flux vector field, $\bar{T}$ is the temperature prescribed at the boundary $\partial\Omega_T$ and is the surface heat flux prescribed at the boundary $\partial\Omega_q$. In the general case of an anisotropic material, the thermal conductivity $\boldsymbol{k}$ is a second order tensor with 4 components in 2D problems

$$\boldsymbol{k} = \begin{bmatrix} k_{xx} & k_{xy} \\ k_{xy} & k_{yy} \end{bmatrix} \tag{2.2}$$

and 9 components in 3D problems

$$\boldsymbol{k} = \begin{bmatrix} k_{xx} & k_{xy} & k_{xz} \\ k_{xy} & k_{yy} & k_{yz} \\ k_{xz} & k_{yz} & k_{zz} \end{bmatrix} \tag{2.3}$$

According to the reciprocity relation derived from the Onsagar's principle of thermodynamics of irreversible processes (Ozisik, 1993), the conductivity tensor is symmetric, thus $k_{xy} = k_{yx}$, $k_{xz} = k_{zx}$, $k_{yz} = k_{zy}$. The constitutive relation between the temperature and heat flux in the interior of Ω, which is also called *Fourier's law*, is defined as

$$\boldsymbol{q}(\boldsymbol{x}) = -\boldsymbol{k}(\boldsymbol{x})\cdot\nabla T(\boldsymbol{x}) \tag{2.4}$$

For a given heat source $r(\boldsymbol{x})$, the steady-state equation governing the temperature field in the interior of Ω is the *Poisson's equation*, namely the following elliptic partial differential equation

$$\nabla\cdot\boldsymbol{q}(\boldsymbol{x}) = r(\boldsymbol{x}) \tag{2.5}$$

where $\nabla\cdot\boldsymbol{q}(\boldsymbol{x})$ is the divergence of the heat flux field. By combining the Poisson equation (2.5) with the boundary conditions of equation (2.1) and constitutive relation of equation (2.4), the boundary value problem (BVP) of steady-state heat transfer can be posed as: "Find a function $T(\boldsymbol{x})$ for the temperature field, so that the following equations are satisfied:".

$$\begin{cases} \nabla\cdot\boldsymbol{q} = r \quad \text{in } \Omega \\ \boldsymbol{q} = -\boldsymbol{k}\cdot\nabla T \\ T = \bar{T} \quad \text{on } \partial\Omega_T \\ \boldsymbol{q}\cdot\boldsymbol{n} = -\bar{q}_n \quad \text{on } \partial\Omega_q \end{cases} \tag{2.6}$$

or equivalently

$$\begin{cases} \boldsymbol{k}\cdot\nabla^2 T + r = 0 \quad \text{in } \Omega \\ T = \bar{T} \quad \text{on } \partial\Omega_T \\ \boldsymbol{q}\cdot\boldsymbol{n} = -\bar{q}_n \quad \text{on } \partial\Omega_q \end{cases} \tag{2.7}$$

2.1.2 Weak form

The system in equation 2.7 is known as the *strong form* of the BVP and it requires the solution scalar field to be at least two times continuously differentiable, namely $T \in C^2(\Omega)$. This condition can be relaxed by using a variational formulation of the BVP, referred to as the *weak form*, which conveniently incorporates the boundary conditions in the differential equation. Let the function space of admissible temperature fields (trial function space) be

$$\mathbb{D} = \{T : T = \bar{T} \text{ on } \partial\Omega_T\} \tag{2.8}$$

Also define the function space of weighting functions (test function space) as

$$\mathbb{W} = \{\delta T : \delta T = 0 \text{ on } \partial\Omega_T\} \tag{2.9}$$

The weak form of the BVP is then posed as: "Find a trial function $T \in \mathbb{D}$, such that for all test functions $\delta T \in \mathbb{W}$ the following integral equation holds:"

$$\int_\Omega \nabla\delta T \cdot \boldsymbol{k} \cdot \nabla T d\Omega = \int_\Omega \delta T \; r \; d\Omega + \int_{\partial\Omega_q} \delta T \; \bar{q}_n \; d\Gamma = 0 \tag{2.10}$$

The weak form can be derived from the strong form. Assume that the solution is one of the trial functions $T \in \mathbb{D}$. By multiplying both sides of the Poisson equation (2.5) with an arbitrary test function $\delta T \in \mathbb{W}$ and integrating over Ω we obtain

$$\nabla \cdot \boldsymbol{q} = r \iff \int_\Omega \delta T \; \nabla \cdot \boldsymbol{q} \; d\Omega = \int_\Omega \delta T \; r \; d\Omega \tag{2.11}$$

Applying the product rule of differentiation

$$\int_\Omega \nabla \cdot (\delta T \; \boldsymbol{q}) d\Omega - \int_\Omega \nabla\delta T \cdot \boldsymbol{q} d\Omega = \int_\Omega \delta T \; r \; d\Omega \tag{2.12}$$

At this point, let's state the Divergence theorem: Given a continuous domain Ω, with external boundary $\partial\Omega$, outwards normal vector $\boldsymbol{n}$ on the boundary and a continuous vector function $\boldsymbol{F}$, the integration of its divergence over the domain is equivalent to the integration of the function itself over the boundary

$$\int_\Omega \nabla \cdot \boldsymbol{F} \; d\Omega = \int_{\partial\Omega} \boldsymbol{F} \cdot \boldsymbol{n} \; d\Gamma \tag{2.13}$$

By applying the divergence theorem to the first integral of equation (2.12)

$$\int_\Omega \nabla \cdot (\delta T \; \boldsymbol{q}) d\Omega = \int_{\partial\Omega} \delta T \; \boldsymbol{q} \cdot \boldsymbol{n} \; d\Gamma \tag{2.14}$$

By decomposing the integral over $\partial\Omega$ into sub-integrals over $\partial\Omega_T$ and $\partial\Omega_q$ and then imposing the Neumann boundary conditions of equation (2.1), the previous equation becomes

$$\int_\Omega \nabla \cdot (\delta T \ \boldsymbol{q}) d\Omega = \cancel{\int_{\partial\Omega_T} \delta T \ \boldsymbol{0} \cdot \boldsymbol{n} d\Gamma} + \int_{\partial\Omega_q} \delta T(-\bar{q}_n) d\Gamma \tag{2.15}$$

where we have used that $\boldsymbol{q}(\boldsymbol{x}) = 0$ on $\partial\Omega_T$. Substituting back into equation (2.12)

$$0 - \int_{\partial\Omega_q} \delta T \ \bar{q}_n \ d\Gamma - \int_\Omega \nabla \delta T \cdot \boldsymbol{q} d\Omega = \int_\Omega \delta T \ r \ d\Omega \tag{2.16}$$

Finally, using the constitutive relation of equation (2.4), the weak form is obtained

$$\int_\Omega \nabla \delta T \cdot \boldsymbol{k} \cdot \nabla T d\Omega = \int_\Omega \delta T \ r \ d\Omega + \int_{\partial\Omega_q} \delta T \ \bar{q}_n \ d\Gamma = 0$$

2.1.3 FEM approximation

In practice, obtaining an exact solution $T(\boldsymbol{x})$ of equation (2.10) is not feasible in most applications of interest, since the function spaces $\mathbb{D}$ and $\mathbb{W}$ are infinite-dimensional. Therefore FEM seeks approximate solutions $T^h(\boldsymbol{x})$ that belong to finite-dimensional spaces, using the Galerkin approximation

$$\mathbb{D}^h = \{T \in H^1(\Omega) : T = \bar{T} \text{ on } \partial\Omega_T\} \subseteq \mathbb{D} \tag{2.17}$$

The test functions are treated similarly

$$\mathbb{W}^h = \{\delta T \in H^1(\Omega) : \delta T = 0 \text{ on } \partial\Omega_T\} \subseteq \mathbb{W} \tag{2.18}$$

where $H^1(\Omega)$ are Hilbert spaces, that is they contain smooth functions over Ω that are square integrable themselves and have square integrable derivatives of order = 1. Then the problem reduces to finding the best approximation $T^h \in \mathbb{D}^h$ such that equation (2.10) is satisfied for all test functions $\delta T \in \mathbb{W}^h$. Because $\mathbb{D}^h$ and $\mathbb{W}^h$ are finite-dimensional spaces, they are spanned by a finite number of basis functions $\{\mathcal{N}_1, \mathcal{N}_2, \cdots\}$. Therefore, the Galerkin approximation consists of searching for an approximate trial function $T^h \in \mathbb{D}^h$, which can be expressed as a linear combination of these basis functions, as can the test functions $\delta T^h \in \mathbb{W}^h$

$$\begin{aligned} T^h(\boldsymbol{x}) &= a_1 \mathcal{N}_1(\boldsymbol{x}) + a_2 \mathcal{N}_2(\boldsymbol{x}) + \cdots \\ \delta T^h(\boldsymbol{x}) &= b_1 \mathcal{N}_1(\boldsymbol{x}) + b_2 \mathcal{N}_2(\boldsymbol{x}) + \cdots \end{aligned} \tag{2.19}$$

The basis functions $\{\mathcal{N}_1, \mathcal{N}_2, \cdots\}$ are chosen according to the problem and the Galerkin method used. Then, finding the approximate solution T^h reduces to identifying the coefficients $\{a_1, a_2, \cdots\}$ that minimize the error $\left\|T^h - T\right\|$. The coefficients $\{b_1, b_2, \cdots\}$ are not

calculated, since they are different for each test function δT^h and the weak form equation (2.10) must hold for all of them. Taking into account these approximations, the *finite-dimensional weak form* (also called the *Galerkin weak form*) will be used in the remainder of this section

$$\int_\Omega \nabla \delta T^h \cdot \boldsymbol{k} \cdot \nabla T^h d\Omega = \int_\Omega \delta T^h \; r \; d\Omega + \int_{\partial\Omega_q} \delta T^h \; \bar{q}_n \; d\Gamma = 0 \tag{2.20}$$

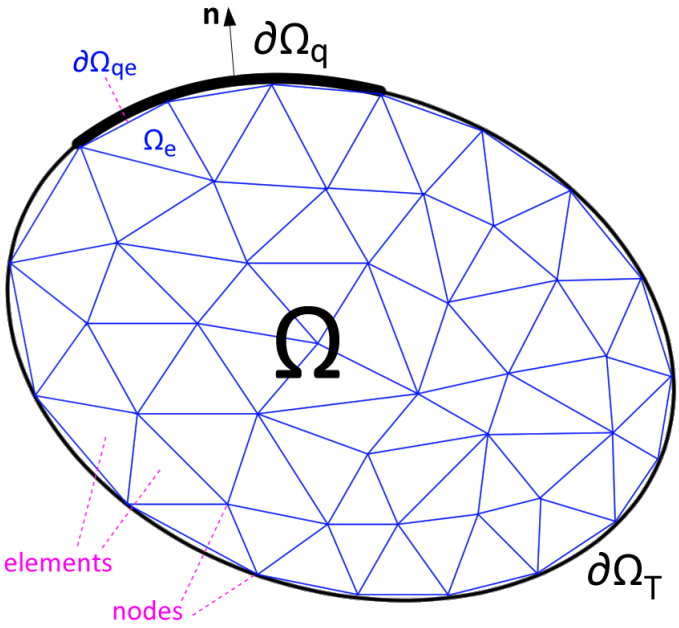


Figure 2.2: An unstructured finite element mesh, consisting of triangular elements.

In FEM, the domain Ω is decomposed into a number of non-overlapping patches Ω_e called *finite elements*, such as the triangles illustrated in figure 2.2. The vertices of these elements are called *nodes* and the set of all nodes of the domain will be referred to as M. If the number of finite elements is n_e, then the domain Ω and its external boundary $\partial\Omega_q$ are decomposed into

$$\begin{aligned} \Omega &= \bigcup_{e=1}^{n_e} \Omega_e \\ \partial\Omega_q &= \bigcup_{e=1}^{n_e} \partial\Omega_{qe} \end{aligned} \tag{2.21}$$

where $\partial\Omega_{qe}$ is the part of $\partial\Omega_q$ that coincides with an edge (2D) or face (3D) of element e, as depicted in figure 2.2. If an element has fewer than two (2D problems) or three (3D

problems) nodes that lie on the boundary $\partial\Omega_q$, then $\partial\Omega_{qe} = 0$. $\partial\Omega_T$ can be partitioned similarly, but is of no interest for now. The integrals of the finite-dimensional weak form equation (2.20) can also be decomposed into

$$\sum_{e=1}^{n_e} \int_{\Omega_e} \nabla \delta T^h \cdot \boldsymbol{k} \cdot \nabla T^h d\Omega = \sum_{e=1}^{n_e} \int_{\Omega_e} \delta T^h \; r \; d\Omega + \sum_{e=1}^{n_e} \int_{\partial\Omega_{qe}} \delta T^h \; \bar{q}_n \; d\Gamma \tag{2.22}$$

FEM is a Galerkin method that employs piecewise polynomials $N_k(\boldsymbol{x})$ as basis functions, which are defined for each node of each element. These are commonly called *shape functions* and are usually Lagrange polynomials that degenerate to 0 outside their corresponding element. The trial and test functions can now be expressed with respect to the basis functions

$$T^h(\boldsymbol{x}) = \sum_{k \in M} N_k(\boldsymbol{x}) \, T_k \tag{2.23}$$

$$\delta T^h(\boldsymbol{x}) = \sum_{k \in M} N_k(\boldsymbol{x}) \, \delta T_k \tag{2.24}$$

where N_k is the polynomial basis function of node k and T_k, δT_k are the unknown coefficients defined for the basis function of node k. Such a coefficient is referred to as a *degree of freedom* (DOF) and represents the temperature at node k. Equation (2.23) calculates the temperature field at any point of the domain, using the basis functions and DOFs defined at all nodes. Of course the basis functions are 0 outside the element they are defined in. Inside each element Ω_e

$$\begin{aligned} T^h(\boldsymbol{x}) &= \sum_{k \in M_e} N_k(\boldsymbol{x}) \, T_k \\ \delta T^h(\boldsymbol{x}) &= \sum_{k \in M_e} N_k(\boldsymbol{x}) \, \delta T_k \end{aligned} \tag{2.25}$$

where M_e is the set of nodes of element e. It is more convenient to work with the matrix-vector form of the above equations

$$\begin{aligned} T^h(\boldsymbol{x}) &= \boldsymbol{N}(\boldsymbol{x}) \cdot \boldsymbol{d}_e \\ \delta T^h(\boldsymbol{x}) &= \boldsymbol{N}(\boldsymbol{x}) \cdot \boldsymbol{\delta d}_e \end{aligned} \tag{2.26}$$

where $\boldsymbol{N}(\boldsymbol{x})$ is a row vector containing the shape functions of all nodes of element e and $\boldsymbol{d}_e$, $\boldsymbol{\delta d}_e$ are vectors containing the DOFs of the element:

$$\boldsymbol{N}(\boldsymbol{x}) = \begin{bmatrix} \cdots & N_k(\boldsymbol{x}) & \cdots \end{bmatrix} \quad \boldsymbol{d}_e = \begin{bmatrix} \vdots \\ T_k \\ \vdots \end{bmatrix} \quad \boldsymbol{\delta d}_e = \begin{bmatrix} \vdots \\ \delta T_k \\ \vdots \end{bmatrix} \tag{2.27}$$

Also consider that all DOFs of the domain, namely the values of the trial and test solution at all nodes, are gathered in the *global* vectors $\boldsymbol{d}$ and $\boldsymbol{\delta d}$. To extract the element vector from the corresponding global vector we use

$$\begin{aligned} \boldsymbol{d}_e &= \boldsymbol{P}_e \cdot \boldsymbol{d} \\ \boldsymbol{\delta d}_e &= \boldsymbol{P}_e \cdot \boldsymbol{\delta d} \end{aligned} \tag{2.28}$$

where $\boldsymbol{P}_e$ is a boolean matrix, namely it contains only 0 or 1 entries, that correlates each element DOF (row of $\boldsymbol{P}_e$) to one exactly global DOF (column of $\boldsymbol{P}_e$). Therefore, the approximation of the temperature field can be rewritten as

$$\begin{aligned} T^h(\boldsymbol{x}) &= \boldsymbol{N}(\boldsymbol{x}) \cdot \boldsymbol{d}_e = \boldsymbol{N}(\boldsymbol{x}) \cdot \boldsymbol{P}_e \cdot \boldsymbol{d} \\ \delta T^h(\boldsymbol{x}) &= \boldsymbol{N}(\boldsymbol{x}) \cdot \boldsymbol{\delta d}_e = \boldsymbol{N}(\boldsymbol{x}) \cdot \boldsymbol{P}_e \cdot \boldsymbol{\delta d} \end{aligned} \tag{2.29}$$

The derivatives of the discretized trial and test function in matrix-vector form are

$$\begin{aligned} \nabla T^h(\boldsymbol{x}) &= \boldsymbol{B}(\boldsymbol{x}) \cdot \boldsymbol{d}_e = \boldsymbol{B}(\boldsymbol{x}) \cdot \boldsymbol{P}_e \cdot \boldsymbol{d} \\ \nabla \delta T^h(\boldsymbol{x}) &= \boldsymbol{B}(\boldsymbol{x}) \cdot \boldsymbol{\delta d}_e = \boldsymbol{B}(\boldsymbol{x}) \cdot \boldsymbol{P}_e \cdot \boldsymbol{\delta d} \end{aligned} \tag{2.30}$$

where matrix $\boldsymbol{B}(\boldsymbol{x})$ contains the derivatives of the shape functions for each node

$$\boldsymbol{B}(\boldsymbol{x}) = \begin{bmatrix} & \dfrac{\partial N_k(\boldsymbol{x})}{\partial x} & \\ \cdots & \dfrac{\partial N_k(\boldsymbol{x})}{\partial y} & \cdots \\ & \dfrac{\partial N_k(\boldsymbol{x})}{\partial z} & \end{bmatrix} \tag{2.31}$$

In matrix-vector form, the tensor multiplication $\nabla \delta T^h \cdot \boldsymbol{k} \cdot \nabla T^h$ of equation (2.22) is written as a row vector $\times$ a matrix $\times$ a column vector: $\left(\nabla \delta T^h\right)^T \cdot \boldsymbol{k} \cdot \nabla T^h$. Also δT^h is a scalar, thus $\delta T^h = \left(\delta T^h\right)^T$. Therefore, equation (2.22) is written as

$$\sum_{e=1}^{n_e} \int_{\Omega_e} \left(\nabla \delta T^h\right)^T \cdot \boldsymbol{k} \cdot \nabla T^h d\Omega = \sum_{e=1}^{n_e} \int_{\Omega_e} \left(\delta T^h\right)^T r \; d\Omega + \sum_{e=1}^{n_e} \int_{\partial \Omega_{qe}} \left(\delta T^h\right)^T \bar{q}_n \; d\Gamma \tag{2.32}$$

Combining all equations above, we obtain the matrix-vector form of the finite-dimensional weak form used in FEM

$$\begin{aligned}
&\sum_{e=1}^{n_e} \int_{\Omega_e} \left(\boldsymbol{B}\left(\boldsymbol{x}\right) \cdot \boldsymbol{P}_e \cdot \boldsymbol{\delta d}\right)^T \cdot \boldsymbol{k}\left(\boldsymbol{x}\right) \cdot \left(\boldsymbol{B}\left(\boldsymbol{x}\right) \cdot \boldsymbol{P}_e \cdot \boldsymbol{d}\right) d\Omega \\
=&\sum_{e=1}^{n_e} \int_{\Omega_e} \left(\boldsymbol{N}\left(\boldsymbol{x}\right) \cdot \boldsymbol{P}_e \cdot \boldsymbol{\delta d}\right)^T r\left(\boldsymbol{x}\right) d\Omega \\
+&\sum_{e=1}^{n_e} \int_{\partial\Omega_{qe}} \left(\boldsymbol{N}\left(\boldsymbol{x}\right) \cdot \boldsymbol{P}_e \cdot \boldsymbol{\delta d}\right)^T \bar{q}_n\left(\boldsymbol{x}\right) d\Gamma
\end{aligned} \tag{2.33}$$

Pulling the constants out of the corresponding integrals and summations:

$$\begin{aligned}
&\boldsymbol{\delta d}^T \cdot \left(\sum_{e=1}^{n_e} \boldsymbol{P}_e^T \cdot \left(\int_{\Omega_e} \boldsymbol{B}^T\left(\boldsymbol{x}\right) \cdot \boldsymbol{k}\left(\boldsymbol{x}\right) \cdot \boldsymbol{B}\left(\boldsymbol{x}\right) d\Omega\right) \cdot \boldsymbol{P}_e\right) \cdot \boldsymbol{d} \\
=&\boldsymbol{\delta d}^T \cdot \sum_{e=1}^{n_e} \boldsymbol{P}_e^T \cdot \left(\int_{\Omega_e} \boldsymbol{N}^T\left(\boldsymbol{x}\right) r\left(\boldsymbol{x}\right) d\Omega\right) \\
+&\boldsymbol{\delta d}^T \cdot \sum_{e=1}^{n_e} \boldsymbol{P}_e^T \cdot \left(\int_{\partial\Omega_{qe}} \boldsymbol{N}^T\left(\boldsymbol{x}\right) \bar{q}_n\left(\boldsymbol{x}\right) d\Gamma\right)
\end{aligned} \tag{2.34}$$

At this point, we should define the element-level conductivity matrix

$$\boldsymbol{K}_e = \int_{\Omega_e} \boldsymbol{B}^T\left(\boldsymbol{x}\right) \cdot \boldsymbol{k}\left(\boldsymbol{x}\right) \cdot \boldsymbol{B}\left(\boldsymbol{x}\right) d\Omega \tag{2.35}$$

and the element-level thermal load vectors

$$\begin{aligned}
\boldsymbol{f}_{re} &= \int_{\Omega_e} \boldsymbol{N}^T\left(\boldsymbol{x}\right) r\left(\boldsymbol{x}\right) d\Omega \\
\boldsymbol{f}_{qe} &= \int_{\partial\Omega_{qe}} \boldsymbol{N}^T\left(\boldsymbol{x}\right) \bar{q}_n\left(\boldsymbol{x}\right) d\Gamma
\end{aligned} \tag{2.36}$$

By assembling all element-level conductivity matrices $\boldsymbol{K}_e$, we obtain the global conductivity matrix $\boldsymbol{K}$, which expresses the thermal conductivity of the whole discretized domain

$$\boldsymbol{K} = \sum_{e=1}^{n_e} \boldsymbol{P}_e^T \cdot \boldsymbol{K}_e \cdot \boldsymbol{P}_e = \sum_{e=1}^{n_e} \boldsymbol{P}_e^T \cdot \left(\int_{\Omega_e} \boldsymbol{B}^T(\boldsymbol{x}) \cdot \boldsymbol{k}(\boldsymbol{x}) \cdot \boldsymbol{B}(\boldsymbol{x}) \, d\Omega \right) \cdot \boldsymbol{P}_e \tag{2.37}$$

Similarly we assemble all element-level vectors $\boldsymbol{f}_{re}$, $\boldsymbol{f}_{qe}$ into a global vector $\boldsymbol{f}$ that represents the thermal loads due to the heat source $r(\boldsymbol{x})$ and the surface heat flux $\bar{q}_n(\boldsymbol{x})$

$$\boldsymbol{f} = \sum_{e=1}^{n_e} \boldsymbol{P}_e^T \cdot (\boldsymbol{f}_{re} + \boldsymbol{f}_{qe}) = \sum_{e=1}^{n_e} \boldsymbol{P}_e^T \cdot \left(\int_{\Omega_e} \boldsymbol{N}^T(\boldsymbol{x}) \, r(\boldsymbol{x}) + \int_{\partial\Omega_{qe}} \boldsymbol{N}^T(\boldsymbol{x}) \, \bar{q}_n(\boldsymbol{x}) \right) \tag{2.38}$$

Finally, equation (2.34) is rewritten as a linear system:

$$\boldsymbol{K} \cdot \boldsymbol{d} = \boldsymbol{f} \tag{2.39}$$

To take the Dirichlet boundary conditions into account, the prescribed nodal temperatures $T_k = \bar{T}$ are removed from the vector $\boldsymbol{d}$, which will then contain only unknown nodal temperatures. The corresponding rows and columns are also removed from the matrix $\boldsymbol{K}$ and vector $\boldsymbol{f}$. As a result, the domain is adequately supported and the linear system of equation (2.39) is positive definite. Therefore, there is a unique solution for the nodal temperatures $\boldsymbol{d}$. Solving this linear system is an interesting topic that will be explained in detail in later chapters. Once the nodal temperatures $\boldsymbol{d}$ have been calculated, the temperature and heat flux fields at any point in the domain can be approximated using equations (2.23) and (2.4).

2.2 Multi-phase heat transfer analysis with XFEM

This section describes how to model heat transfer in composites consisting of multiple material phases, the interfaces between which exhibit thermal resistance. A novel approach based on the Extended Finite Element Method (XFEM) is proposed and compared with standard FEM.

2.2.1 The boundary value problem

Let a body Ω be divided into n_p material phases $\Omega^{(1)}$, $\Omega^{(2)}$, ... $\Omega^{(n_p)}$, separated by n_b interfaces $\Gamma^{(1)}$, $\Gamma^{(2)}$, ... $\Gamma^{(n_b)}$. Each interface $\Gamma^{(b)}$ will be written as $\Gamma^{(ij)} \equiv \Gamma^{(ji)}$ to signify that it separates the solid phases $\Omega^{(i)}$, $\Omega^{(j)}$. An example with three phases is illustrated in figure 2.3. The external boundary $\partial\Omega$ of the whole domain has an outward normal vector $\boldsymbol{n}$ and consists of the external boundaries of individual phases $\partial\Omega = \bigcup_{i=1}^{n_p} \partial\Omega^{(i)}$. Each of

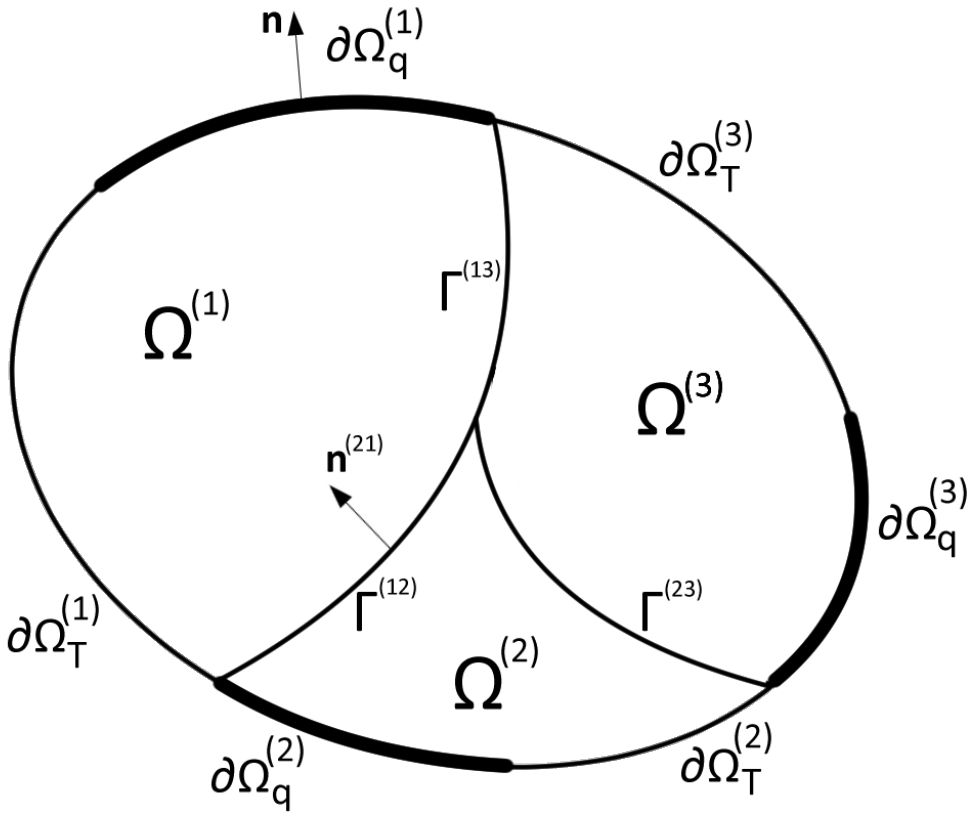


Figure 2.3: Heat transfer in domain with multiple material phases

these boundaries is further divided into complementary parts $\partial\Omega_T^{(i)}$ and $\partial\Omega_q^{(i)}$, such that $\partial\Omega^{(i)} = \partial\Omega_T^{(i)} \cup \partial\Omega_q^{(i)}$. It is possible that a phase $\Omega^{(i)}$ is entirely internal, namely it does not have any common points with the external boundary $\partial\Omega$ and is bounded only by material interfaces $\Gamma^{(ij)}$. In this case, $\partial\Omega^{(i)} = \partial\Omega_T^{(i)} = \partial\Omega_q^{(i)} = \emptyset$. Otherwise, Dirichlet and Neumann boundary conditions are applied respectively on $\partial\Omega_T^{(i)}$ and $\partial\Omega_q^{(i)}$

$$\begin{aligned} T &= \bar{T} \quad \text{on } \partial\Omega_T^{(i)}, \\ \boldsymbol{q}\cdot\boldsymbol{n} &= -\bar{q}_n \quad \text{on } \partial\Omega_q^{(i)}. \end{aligned} \tag{2.40}$$

where $T = T(\boldsymbol{x})$ is the (scalar) temperature field and $\boldsymbol{q} = \boldsymbol{q}(\boldsymbol{x})$ is the heat flux vector field. In the general case of anisotropic materials, the thermal conductivity $\boldsymbol{k}^{(i)}$ of phase $\Omega^{(i)}$ is a symmetric second order tensor, defined similarly to equations (2.2) and (2.3). The constitutive relation between the temperature and heat flux in the interior of each phase $\Omega^{(i)}$, which is also called *Fourier's law*, is defined as

$$\boldsymbol{q}(\boldsymbol{x}) = -\boldsymbol{k}^{(i)}(\boldsymbol{x})\cdot\nabla T(\boldsymbol{x}), \quad \boldsymbol{x}\in\Omega^{(i)}, \quad i = 1,\dots n_p \tag{2.41}$$

For a given heat source $r(\boldsymbol{x})$, the steady-state equation governing the temperature field in the interior of Ω is the *Poisson's equation*, namely the following elliptic partial differential equation

$$\nabla \cdot \boldsymbol{q}(\boldsymbol{x}) = r(\boldsymbol{x}) \tag{2.42}$$

where $\nabla \cdot \boldsymbol{q}(\boldsymbol{x})$ is the divergence of the heat flux field. In this heterogeneous material, each interface $\Gamma^{(ij)}$ exhibits Kapitza (interfacial) thermal resistance $\alpha^{(ij)}$, or equivalently interfacial conductance $k^{(ij)}$ which is the reciprocal of the resistance, that is, $k^{(ij)} = \frac{1}{\alpha^{(ij)}}$. Therefore, the thermal behavior on $\Gamma^{(ij)}$ is characterized by a jump in the temperature field across each material interface

$$[\![T]\!]^{(ij)} = -\alpha^{(ij)} \boldsymbol{q}^{(i)} \cdot \boldsymbol{n}^{(ij)} \quad \text{on } \Gamma^{(ij)} \tag{2.43}$$

where $[\![\cdot]\!]^{(ij)} = (\cdot)^{(j)} - (\cdot)^{(i)}$ is an operator denoting the jump across the material interface $\Gamma^{(ij)}$. The unit vector $\boldsymbol{n}^{(ij)}$ normal to $\Gamma^{(ij)} \equiv \Gamma^{(ji)}$ is directed from $\Omega^{(i)}$ into $\Omega^{(j)}$ and it holds that

$$\boldsymbol{n}^{(ij)} = -\boldsymbol{n}^{(ji)} \tag{2.44}$$

Meanwhile, the heat flux field is continuous across each interface $\Gamma^{(ij)}$

$$\boldsymbol{q}^{(i)} \cdot \boldsymbol{n}^{(ij)} = \boldsymbol{q}^{(j)} \cdot \boldsymbol{n}^{(ji)} \quad \text{on } \Gamma^{(ij)} \tag{2.45}$$

where $\boldsymbol{q}^{(i)}$ and $\boldsymbol{q}^{(j)}$ are the values of the heat flux field on each side of the interface $\Gamma^{(ij)}$. By combining the Poisson equation (2.42) with the boundary conditions of equations (2.40) and (2.45) and the constitutive relations of equations (2.41) and (2.43), the boundary value problem (BVP) of steady-state heat transfer can be posed as: "Find a function $T(\boldsymbol{x})$ for the temperature field, so that the following equations are satisfied:"

$$\begin{cases} \nabla \cdot \boldsymbol{q} = r \quad \text{in } \Omega \\ \boldsymbol{q} = -\boldsymbol{k}^{(i)} \cdot \nabla T, \quad \text{in each } \Omega^{(i)}, \quad i = 1, \dots n_p \\ [\![T]\!]^{(ij)} = -\alpha^{(ij)} \boldsymbol{q}^{(i)} \cdot \boldsymbol{n}^{(ij)} \quad \text{on each } \Gamma^{(ij)} \\ \boldsymbol{q}^{(i)} \cdot \boldsymbol{n}^{(ij)} = \boldsymbol{q}^{(j)} \cdot \boldsymbol{n}^{(ji)} \quad \text{on each } \Gamma^{(ij)} \\ T = \bar{T} \quad \text{on } \partial\Omega_T \\ \boldsymbol{q} \cdot \boldsymbol{n} = -\bar{q}_n \quad \text{on } \partial\Omega_q \end{cases} \tag{2.46}$$

2.2.2 FEM vs XFEM modeling

The BVP of equation (2.46) can be solved with the standard FEM, but the following modifications are required. First of all, a mesh that conforms to the interfaces between the material phases $\Gamma^{(ij)}$ must be generated, as depicted in figure 2.4. Second, the nodes that lie on these material interfaces must be duplicated, so that different nodes are defined for each

material phase $\Omega^{(i)}$. This is necessary, in order to model the temperature jump of equation (2.43). These duplicate nodes are used to define DOFs, namely nodal temperatures, that are different on each side of the interfaces. Finally, special finite elements that connect the duplicate nodes and lie exactly on the interface (see figure 2.4) must be defined. These interface elements differ from the usual elements used for calculating the weak form integrals in the bulk of the domain. Instead they are responsible for modeling the interface behavior described in equations (2.43) and (2.45).

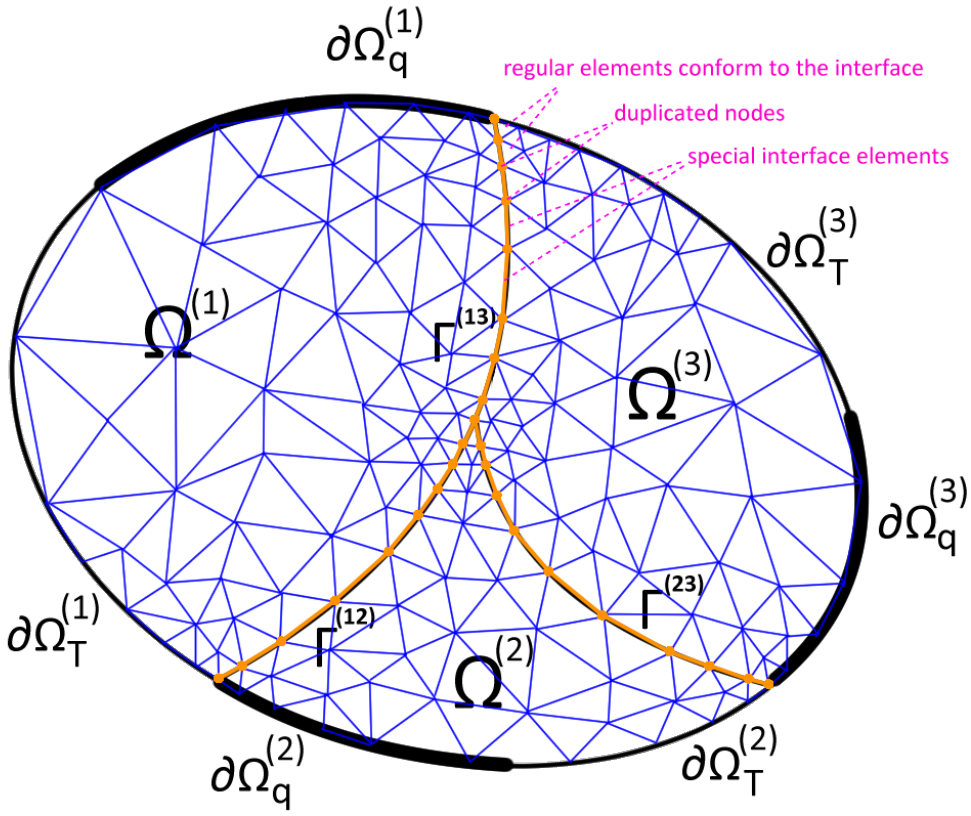


Figure 2.4: Modeling heat transfer in composite materials with FEM.

Unfortunately, this FEM approach has a number of drawbacks. A suitable mesh generator is required, in order to create a mesh that conforms to material interfaces. Afterwards, the generated conforming mesh needs to be modified. The nodes on the material interface must be located and then duplicated. Furthermore, special interface elements must be added to the mesh based on these nodes. While these operations are certainly demanding and increase the complexity of the analysis, the main problem lies in trying to produce a mesh that conforms to complex geometries of multiple material interfaces. In these cases, a large number of very small elements is required in the vicinity of the interfaces, particularly near sharp turns and kinks. This fine discretization increases the computational cost of the whole FEM procedure

and especially the solution of the linear system of equation (2.39). Moreover, duplicate nodes and their corresponding extra DOFs are added exactly where the conforming mesh is very fine, further increasing the size of the linear system, as well as the memory and time needed for its solution. As more material phases are modeled, the overall FEM simulation rapidly becomes more cumbersome and inefficient or even impossible in large scale 3D problems. Finally, in problems where the material interfaces evolve and move throughout the analysis, the conforming mesh must be regenerated at each iteration and all mesh-related operations listed previously need to be repeated. In addition, the temperature and heat flux fields need to be mapped between the old and new meshes, which introduces errors.

To overcome the aforementioned limitations, in this dissertation a novel approach based on the Extended Finite Element Method (XFEM) is proposed for modeling heat transfer in composite materials. In XFEM, the elements can be intersected by one or more material interfaces, instead of having to conform to their geometry. In order to model the discontinuous temperature field, non-smooth basis functions are introduced at nodes near the material interfaces. figure 2.5 illustrates a non-conforming XFEM mesh that interacts with three material interfaces. The elements that are intersected by the interfaces are called *enriched elements*. The non-smooth basis functions are introduced in the nodes of enriched elements, which will be called *enriched nodes*. All other nodes of the domain will be called *standard nodes*. All elements that do not have any enriched nodes are called *standard elements* and behave identically to FEM elements. Finally, elements that are not intersected by the material interfaces, but share one or more enriched nodes with the enriched elements, are called *blending elements*. In general, the temperature field is not perfectly reproduced in these blending elements, which leads to reduced accuracy. However, the XFEM formulation developed here avoids all blending-related problems, by using appropriate enrichment functions.

The proposed XFEM approach provides a convenient framework to model heat transfer in composites with multiple material phases, by avoiding any dependencies of the mesh from the geometry of the material interfaces. Complex geometries can be represented easily with explicit or level set methods, while the mesh used for the analysis can be arbitrarily intersected by the material interfaces, rather than having to conform to them. Therefore, there is no need for specialized mesh generators and problems with evolving material interfaces pose no additional difficulties. Moreover, the density of the finite element mesh can be selected such that the desired accuracy is met without needlessly increasing the computational cost, instead of employing very fine meshes only to represent the complex geometries. In Yvonnet et al. (2011), XFEM was used to model the Kapitza thermal resistance between different materials. Unfortunately, that approach was only applicable for domains consisting of only two materials and cannot be used in more interesting problems, such as composites with multiple inclusions that interact with each other and the surrounding matrix material. To remedy this, a more general XFEM formulation, which allows modeling steady state heat transfer in composites consisting of an arbitrary number of material phases, is proposed in Bakalakos et al. (2020), Bakalakos et al. (2022) and elaborated here.

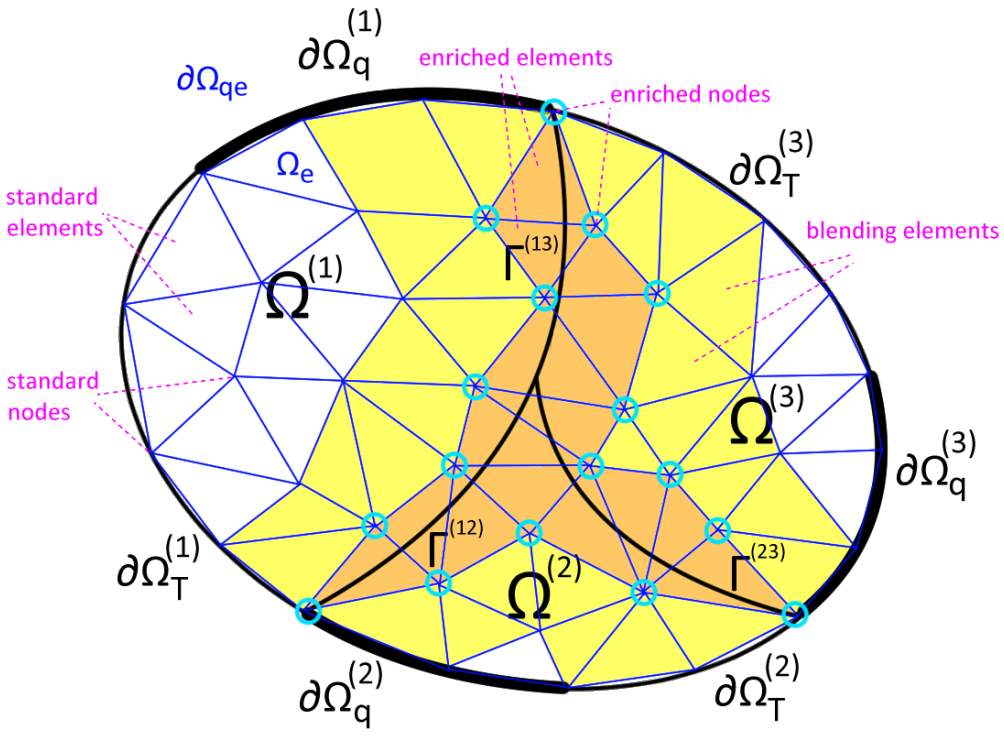


Figure 2.5: Modeling heat transfer in composite materials with XFEM.

2.2.3 Discontinuous divergence theorem

In this section, we will formulate the divergence theorem for a vector function $\boldsymbol{F}$ that is discontinuous in the whole domain Ω. Particularly, $\boldsymbol{F}$ is continuous in each material phase $\Omega^{(i)}$, $i = 1, \cdots n_p$, but discontinuous across each material interface $\Gamma^{(ij)}$. The value of $\boldsymbol{F}$ at points that lie exactly on an interface $\Gamma^{(ij)}$ is either $\boldsymbol{F}^{(i)}$ or $\boldsymbol{F}^{(j)}$ depending on which side we examine $\boldsymbol{F}$ from ($\boldsymbol{F}^{(i)} \neq \boldsymbol{F}^{(j)}$). Let M^{Γ} be the set of all unordered pairs (i, j), for which there is a material interface $\Gamma^{(ij)}$. For a given phase p let $M_{\Gamma}^{(p)}$ be the set of all unordered pairs (p, j), for which there is a material interface $\Gamma^{(pj)} \equiv \Gamma^{(jp)}$. Obviously $M_{\Gamma}^{(p)} \subset M^{\Gamma}$. As depicted in figure 2.3, each phase $\Omega^{(i)}$ is bounded by its corresponding material interfaces and external boundary $\partial\Omega^{(i)}$. We define $\partial\Omega_{+}^{(i)}$ to be the total boundary of phase $\Omega^{(i)}$

$$\partial\Omega_{+}^{(i)} = \partial\Omega^{(i)} \bigcup_{(i,j)\in M_{\Gamma}^{(i)}} \Gamma^{(ij)} \tag{2.47}$$

The outwards normal vector $\boldsymbol{n}^{(i)}$ defined on $\partial\Omega_{+}^{(i)}$ coincides either with the outwards normal vector $\boldsymbol{n}$ defined on the external boundary $\partial\Omega$ of the whole domain Ω or with the outwards normal vector $\boldsymbol{n}^{(ij)}$ defined on the material interfaces bounding $\Omega^{(i)}$.

$$\boldsymbol{n}^{(i)} = \begin{cases} \boldsymbol{n} & \text{on } \partial\Omega^{(i)} \\ \boldsymbol{n}^{(ij)} & \text{on } \Gamma^{(ij)} \end{cases} \tag{2.48}$$

Note that $\boldsymbol{n}^{(ij)}$ is oriented from phase $\Omega^{(i)}$ towards $\Omega^{(j)}$. Similarly, for phase $\Omega^{(j)}$ the outwards normal vector on the same material interface would be $\boldsymbol{n}^{(ji)} = -\boldsymbol{n}^{(ij)}$. At this point we can apply the divergence theorem for each material phase $\Omega^{(i)}$ separately, since the vector function $\boldsymbol{F}$ is continuous in its interior

$$\int_{\Omega^{(i)}} \nabla \cdot \boldsymbol{F} \; d\Omega = \int_{\partial\Omega_{+}^{(i)}} \boldsymbol{F} \cdot \boldsymbol{n}^{(i)} \; d\Gamma \tag{2.49}$$

By decomposing the surface integral according to equations (2.47) and (2.48)

$$\int_{\Omega^{(i)}} \nabla \cdot \boldsymbol{F} \; d\Omega = \int_{\partial\Omega^{(i)}} \boldsymbol{F} \cdot \boldsymbol{n} \; d\Gamma + \sum_{(i,j)\in M_{\Gamma}^{(i)}} \int_{\Gamma^{(ij)}} \boldsymbol{F}^{(i)} \cdot \boldsymbol{n}^{(ij)} \; d\Gamma \tag{2.50}$$

If we apply equation (2.50) for all phases $i = 1, \cdots n_p$ and then sum

$$\begin{aligned} \sum_{i=1}^{n_p} \int_{\Omega^{(i)}} \nabla \cdot \boldsymbol{F} \; d\Omega = & \sum_{i=1}^{n_p} \int_{\partial\Omega^{(i)}} \boldsymbol{F} \cdot \boldsymbol{n} \; d\Gamma \\ & + \sum_{(i,j)\in M^{\Gamma}} \left(\int_{\Gamma^{(ij)}} \boldsymbol{F}^{(i)} \cdot \boldsymbol{n}^{(ij)} \; d\Gamma + \int_{\Gamma^{(ji)}} \boldsymbol{F}^{(j)} \cdot \boldsymbol{n}^{(ji)} \; d\Gamma \right) \end{aligned} \tag{2.51}$$

The first two sums can be calculated by merging the integrals over Ω and $\partial\Omega$ respectively

$$\begin{aligned} \sum_{i=1}^{n_p} \int_{\Omega^{(i)}} \nabla \cdot \boldsymbol{F} \; d\Omega &= \int_{\Omega} \nabla \cdot \boldsymbol{F} \; d\Omega \\ \sum_{i=1}^{n_p} \int_{\partial\Omega^{(i)}} \boldsymbol{F} \cdot \boldsymbol{n} \; d\Gamma &= \int_{\partial\Omega} \boldsymbol{F} \cdot \boldsymbol{n} \; d\Gamma \end{aligned} \tag{2.52}$$

Since $\Gamma^{(ji)}$ coincides with $\Gamma^{(ij)}$, but is oriented opposite to it, the third sum of equation (2.51) becomes

$$\begin{aligned}
&\sum_{(i,j)\in M^\Gamma} \left(\int_{\Gamma^{(ij)}} \boldsymbol{F}^{(i)} \cdot \boldsymbol{n}^{(ij)} \, d\Gamma + \int_{\Gamma^{(ji)}} \boldsymbol{F}^{(j)} \cdot \boldsymbol{n}^{(ji)} \, d\Gamma \right) \\
&= \sum_{(i,j)\in M^\Gamma} \left(\int_{\Gamma^{(ij)}} \boldsymbol{F}^{(i)} \cdot \boldsymbol{n}^{(ij)} \, d\Gamma - \int_{\Gamma^{(ij)}} \boldsymbol{F}^{(j)} \cdot \boldsymbol{n}^{(ji)} \, d\Gamma \right) \\
&= \sum_{(i,j)\in M^\Gamma} \int_{\Gamma^{(ij)}} \boldsymbol{F}^{(i)} \cdot \boldsymbol{n}^{(ij)} - \boldsymbol{F}^{(j)} \cdot \boldsymbol{n}^{(ji)} \, d\Gamma
\end{aligned} \tag{2.53}$$

Substituting back into equation (2.51), the divergence theorem for this discontinuous vector function $\boldsymbol{F}$ is obtained

$$\int_\Omega \nabla \cdot \boldsymbol{F} \, d\Omega = \int_{\partial\Omega} \boldsymbol{F} \cdot \boldsymbol{n} \, d\Gamma + \sum_{(i,j)\in M^\Gamma} \int_{\Gamma^{(ij)}} \boldsymbol{F}^{(i)} \cdot \boldsymbol{n}^{(ij)} - \boldsymbol{F}^{(j)} \cdot \boldsymbol{n}^{(ji)} \, d\Gamma \tag{2.54}$$

2.2.4 Weak form

The system in equation 2.46 is known as the *strong form* of the BVP and it requires the solution scalar field to be at least two times continuously differentiable inside each material phase, namely $T \in C^2(\Omega^{(i)})$. This condition can be relaxed by using a variational formulation of the BVP, referred to as the *weak form*, which conveniently incorporates the boundary conditions in the differential equation. Let the function space of admissible temperature fields (trial function space) be

$$\mathbb{D} = \{T : T = \bar{T} \text{ on } \partial\Omega_T,\ T \text{ discontinuous on } \Gamma^{(ij)},\ \forall (i,j) \in M^\Gamma\} \tag{2.55}$$

Also define the function space of weighting functions (test function space) as

$$\mathbb{W} = \{\delta T : \delta T = 0 \text{ on } \partial\Omega_T,\ \delta T \text{ discontinuous on } \Gamma^{(ij)},\ \forall (i,j) \in M^\Gamma\} \tag{2.56}$$

The weak form of the BVP is then posed as: "Find a trial function $T \in \mathbb{D}$, such that for all test functions $\delta T \in \mathbb{W}$ the following integral equation holds:"

$$\sum_{(i,j)\in M^\Gamma} \int_{\Gamma^{(ij)}} [\![\delta T]\!]^{(ij)} \frac{1}{\alpha^{(ij)}} [\![T]\!]^{(ij)} \, d\Gamma + \int_\Omega \nabla \delta T \cdot \boldsymbol{k} \cdot \nabla T d\Omega = \int_\Omega \delta T \ r \ d\Omega + \int_{\partial\Omega_q} \delta T \ \bar{q}_n \ d\Gamma \tag{2.57}$$

The weak form can be derived from the strong form. Assume that the solution is one of the trial functions $T \in \mathbb{D}$. By multiplying both sides of the Poisson equation (2.42) with an arbitrary test function $\delta T \in \mathbb{W}$ and integrating over Ω we obtain

$$\nabla \cdot \boldsymbol{q} = r \iff \int_\Omega \delta T \ \nabla \cdot \boldsymbol{q} \ d\Omega = \int_\Omega \delta T \ r \ d\Omega \tag{2.58}$$

Applying the product rule of differentiation

$$\int_\Omega \nabla \cdot (\delta T \ \boldsymbol{q}) d\Omega - \int_\Omega \nabla \delta T \cdot \boldsymbol{q} d\Omega = \int_\Omega \delta T \ r \ d\Omega \tag{2.59}$$

By applying the divergence theorem of equation (2.54) for the discontinuous vector function $\boldsymbol{F} = \delta T \ \boldsymbol{q}$, the first integral of the previous equation becomes

$$\begin{aligned} \int_\Omega \nabla \cdot (\delta T \ \boldsymbol{q}) d\Omega = & \int_{\partial\Omega} \delta T \ \boldsymbol{q} \cdot \boldsymbol{n} \ d\Gamma \\ & + \sum_{(i,j) \in M^\Gamma} \int_{\Gamma^{(ij)}} \delta T^{(i)} \ \boldsymbol{q}^{(i)} \cdot \boldsymbol{n}^{(ij)} - \delta T^{(j)} \ \boldsymbol{q}^{(j)} \cdot \boldsymbol{n}^{(ji)} \ d\Gamma \end{aligned} \tag{2.60}$$

We decompose the integral over $\partial\Omega$ into sub-integrals over $\partial\Omega_T$ and $\partial\Omega_q$ and then impose the Neumann boundary conditions of equation (2.40)

$$\int_{\partial\Omega} \delta T \ \boldsymbol{q} \cdot \boldsymbol{n} \ d\Gamma = \cancel{\int_{\partial\Omega_T} \delta T \ \boldsymbol{0} \cdot \boldsymbol{n} d\Gamma} + \int_{\partial\Omega_q} \delta T (-\bar{q}_n) = - \int_{\partial\Omega_q} \delta T \ \bar{q}_n \ d\Gamma \tag{2.61}$$

where we have used that $\boldsymbol{q}(\boldsymbol{x}) = 0$ on $\partial\Omega_T$. Meanwhile, by taking into account the continuity of the heat flux field across each interface $\Gamma^{(ij)}$ (see equation (2.45)), the last integrals of equation (2.60) can be rewritten

$$\begin{aligned} & \sum_{(i,j) \in M^\Gamma} \int_{\Gamma^{(ij)}} \delta T^{(i)} \ \boldsymbol{q}^{(i)} \cdot \boldsymbol{n}^{(ij)} - \delta T^{(j)} \ \boldsymbol{q}^{(j)} \cdot \boldsymbol{n}^{(ji)} \ d\Gamma \\ = & \sum_{(i,j) \in M^\Gamma} \int_{\Gamma^{(ij)}} \delta T^{(i)} \ \boldsymbol{q}^{(i)} \cdot \boldsymbol{n}^{(ij)} - \delta T^{(j)} \ \boldsymbol{q}^{(i)} \cdot \boldsymbol{n}^{(ij)} \ d\Gamma \\ = & \sum_{(i,j) \in M^\Gamma} \int_{\Gamma^{(ij)}} \left(\delta T^{(i)} - \delta T^{(j)} \right) \boldsymbol{q}^{(i)} \cdot \boldsymbol{n}^{(ij)} \ d\Gamma \\ = & \sum_{(i,j) \in M^\Gamma} \int_{\Gamma^{(ij)}} - [\![\delta T]\!]^{(ij)} \ \boldsymbol{q}^{(i)} \cdot \boldsymbol{n}^{(ij)} \ d\Gamma \end{aligned} \tag{2.62}$$

which can be further simplified by using the interfacial constitutive relation of equation (2.43)

$$\begin{aligned}\sum_{(i,j)\in M^\Gamma} \int_{\Gamma^{(ij)}} \delta T^{(i)}\ \boldsymbol{q}^{(i)}\cdot\boldsymbol{n}^{(ij)} - \delta T^{(j)}\ \boldsymbol{q}^{(j)}\cdot\boldsymbol{n}^{(ji)}\ d\Gamma \\ = \sum_{(i,j)\in M^\Gamma} \int_{\Gamma^{(ij)}} [\![\delta T]\!]^{(ij)}\ \frac{1}{\alpha^{(ij)}}\ [\![T]\!]^{(ij)}\ d\Gamma\end{aligned} \tag{2.63}$$

Substituting equation (2.61) and (2.63) into equation (2.60) produces

$$\int_\Omega \nabla\cdot(\delta T\ \boldsymbol{q})d\Omega = -\int_{\partial\Omega_q} \delta T\ \bar{q}_n\ d\Gamma + \sum_{(i,j)\in M^\Gamma} \int_{\Gamma^{(ij)}} [\![\delta T]\!]^{(ij)}\ \frac{1}{\alpha^{(ij)}}\ [\![T]\!]^{(ij)}\ d\Gamma \tag{2.64}$$

which can then be substituted back into equation (2.59)

$$\begin{aligned}-\int_{\partial\Omega_q} \delta T\ \bar{q}_n\ d\Gamma + \sum_{(i,j)\in M^\Gamma} \int_{\Gamma^{(ij)}} [\![\delta T]\!]^{(ij)}\ \frac{1}{\alpha^{(ij)}}\ [\![T]\!]^{(ij)}\ d\Gamma \\ -\int_\Omega \nabla\delta T\cdot\boldsymbol{q}d\Omega = \int_\Omega \delta T\ r\ d\Omega\end{aligned} \tag{2.65}$$

Finally, using the bulk constitutive relation of equation (2.41), the weak form is obtained

$$\sum_{(i,j)\in M^\Gamma} \int_{\Gamma^{(ij)}} [\![\delta T]\!]^{(ij)}\ \frac{1}{\alpha^{(ij)}}\ [\![T]\!]^{(ij)}\ d\Gamma + \int_\Omega \nabla\delta T\cdot\boldsymbol{k}\cdot\nabla T d\Omega = \int_\Omega \delta T\ r\ d\Omega + \int_{\partial\Omega_q} \delta T\ \bar{q}_n\ d\Gamma$$

2.2.4.1 Finite-dimensional weak form

Finding the exact solution $T(\boldsymbol{x})$ of equation (2.57) is not feasible in general, since the function spaces $\mathbb{D}$ and $\mathbb{W}$ are infinite-dimensional. Similarly to standard FEM, XFEM seeks approximate solutions $T^h(\boldsymbol{x})$ that belong to finite-dimensional spaces, using the Galerkin approximation

$$\mathbb{D}^h = \{T\in\widetilde{H}^1(\Omega) : T = \bar{T} \text{ on } \partial\Omega_T\}\subseteq\mathbb{D} \tag{2.66}$$

The test functions are treated similarly

$$\mathbb{W}^h = \{\delta T\in\widetilde{H}^1(\Omega) : \delta T = 0 \text{ on } \partial\Omega_T\}\subseteq\mathbb{W} \tag{2.67}$$

where $\widetilde{H}^1(\Omega)$ are function spaces consisting of piece-wise Hilbert spaces $H^1(\Omega^{(i)})$. Specifically, they contain functions that:

- are smooth and square integrable and have have square integrable derivatives of $order = 1$, inside each material phase $\Omega^{(i)}$, $i = 1, \cdots n_p$
- are discontinuous across each material interface $\Gamma^{(ij)}$, $\forall (i,j) \in M^{\Gamma}$

Then the problem reduces to finding the best approximation $T^h \in \mathbb{D}^h$ such that equation (2.57) is satisfied for all test functions $\delta T \in \mathbb{W}^h$. Since $\mathbb{D}^h$ and $\mathbb{W}^h$ are finite-dimensional spaces, they are spanned by a finite number of basis functions $\{\mathcal{N}_1, \mathcal{N}_2, \cdots\}$. Therefore, the Galerkin approximation consists of searching for an approximate trial function $T^h \in \mathbb{D}^h$ that can be expressed as a linear combination of these basis functions, as can the test function $\delta T^h \in \mathbb{W}^h$:

$$\begin{aligned} T^h(\boldsymbol{x}) &= a_1 \mathcal{N}_1(\boldsymbol{x}) + a_2 \mathcal{N}_2(\boldsymbol{x}) + \cdots \\ \delta T^h(\boldsymbol{x}) &= b_1 \mathcal{N}_1(\boldsymbol{x}) + b_2 \mathcal{N}_2(\boldsymbol{x}) + \cdots \end{aligned} \tag{2.68}$$

XFEM chooses appropriate basis functions $\{\mathcal{N}_1, \mathcal{N}_2, \cdots\}$ that can capture the discontinuous trial and test functions. Then, finding the approximate solution T^h reduces to identifying the coefficients $\{a_1, a_2, \cdots\}$ that minimize the error $\left\| T^h - T \right\|$. The coefficients $\{b_1, b_2, \cdots\}$ are not calculated, since they are different for each test function δT^h and the weak form equation (2.57) must hold for all of them. Taking into account these approximations, the finite-dimensional weak form will be used in the remainder of this chapter

$$\begin{aligned} \sum_{(i,j) \in M^{\Gamma}} \int_{\Gamma^{(ij)}} \left[\!\left[\delta T^h \right]\!\right]^{(ij)} \frac{1}{\alpha^{(ij)}} \left[\!\left[T^h \right]\!\right]^{(ij)} \, d\Gamma + \int_{\Omega} \nabla \delta T^h \cdot \boldsymbol{k} \cdot \nabla T^h d\Omega \\ = \int_{\Omega} \delta T^h \; r \; d\Omega + \int_{\partial \Omega_q} \delta T^h \; \bar{q}_n \; d\Gamma \end{aligned} \tag{2.69}$$

Similarly to standard FEM, XFEM decomposes the domain Ω into a number of non-overlapping finite elements Ω_e. Unlike FEM, the finite elements of XFEM can be intersected by the material interfaces $\Gamma^{(ij)}$, as illustrated in figure 2.5. If the number of finite elements is n_e, then the domain Ω and its external boundary $\partial \Omega_q$ are decomposed into

$$\begin{aligned} \Omega &= \bigcup_{e=1}^{n_e} \Omega_e \\ \partial \Omega_q &= \bigcup_{e=1}^{n_e} \partial \Omega_{qe} \end{aligned} \tag{2.70}$$

where $\partial \Omega_{qe}$ is the part of $\partial \Omega_q$ that coincides with an edge (2D) or face (3D) of element e, as depicted in figure 2.5. If an element has fewer than two (2D problems) or three (3D

problems) nodes that lie on the boundary $\partial\Omega_q$, then $\partial\Omega_{qe} = 0$. $\partial\Omega_T$ can be partitioned similarly, but is of no interest for now. Similarly the material interfaces $\Gamma^{(ij)}$ can be divided into

$$\partial\Gamma^{(ij)} = \bigcup_{e=1}^{n_e} \partial\Gamma_e^{(ij)} \ , \ \forall (i,j) \in M^\Gamma \tag{2.71}$$

where $\Gamma_e^{(ij)}$ is the part of the material interface $\Gamma^{(ij)}$ that lies inside element e and on its edges (2D problems) or faces (3D problems). If element e is not intersected by or does not conform to $\Gamma^{(ij)}$, then obviously $\Gamma_e^{(ij)} = 0$. The integrals of the finite-dimensional weak form equation (2.69) can also be decomposed into

$$\begin{aligned} \sum_{e=1}^{n_e} \left(\sum_{(i,j)\in M^\Gamma} \int_{\Gamma_e^{(ij)}} \left[\!\left[\delta T^h \right]\!\right]^{(ij)} \frac{1}{\alpha^{(ij)}} \left[\!\left[T^h \right]\!\right]^{(ij)} \, d\Gamma \right) + \sum_{e=1}^{n_e} \int_{\Omega_e} \nabla \delta T^h \cdot \boldsymbol{k} \cdot \nabla T^h d\Omega \\ = \sum_{e=1}^{n_e} \int_{\Omega} \delta T^h \ r \ d\Omega + \sum_{e=1}^{n_e} \int_{\partial\Omega_q} \delta T^h \ \bar{q}_n \ d\Gamma \end{aligned} \tag{2.72}$$

2.2.5 XFEM enrichment

In order to capture the jumps of the temperature field across material interfaces, XFEM enriches the approximation field used in FEM (see equation (2.23)) with discontinuous functions.

2.2.5.1 Simple case with only 2 materials

Consider the simplest case depicted, where there are only two material phases $\Omega^{(1)}$, $\Omega^{(2)}$ and only one material interface Γ between them, as illustrated in figure 2.6. Let M^H be the set of nodes of all elements intersected by the interface Γ. XFEM enriches the basis functions defined at all nodes in M^H with a discontinuous function, typically the *Heaviside* function (also called *step* function)

$$H(\boldsymbol{x}) = \begin{cases} -1, & \boldsymbol{x} \in \Omega^{(1)} \\ +1, & \boldsymbol{x} \in \Omega^{(2)} \end{cases} \tag{2.73}$$

The approximate temperature field used in XFEM will then be

$$T^h(\boldsymbol{x}) = \underbrace{\sum_{k\in M} N_k(\boldsymbol{x}) \, T_k}_{T^{\text{std}}} + \underbrace{\sum_{k\in M^H} N_k(\boldsymbol{x}) \left(H(\boldsymbol{x}) - H(\boldsymbol{x}_k) \right) \widetilde{T}_k}_{T^{\text{enr}}} \tag{2.74}$$

where $\boldsymbol{x}_k$ are the coordinates of node k, $N_k(\boldsymbol{x})$ are the same Lagrange polynomial shape functions used in FEM and T_k are nodal temperatures. It can be observed that the first part

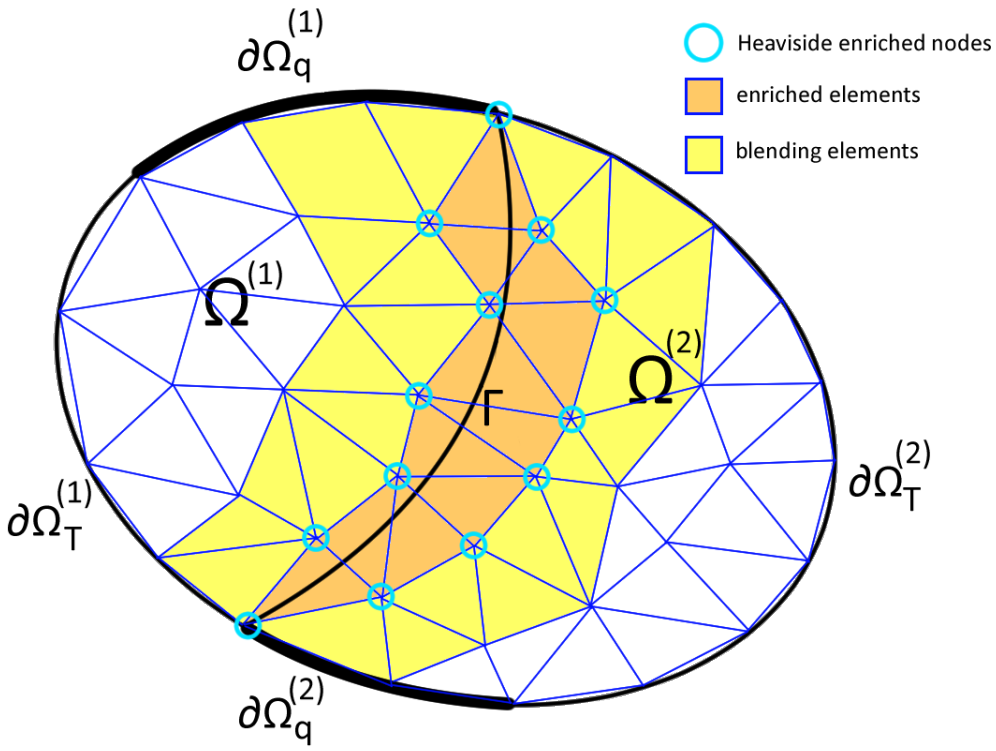


Figure 2.6: Heaviside enrichment for a single material interface.

T^{std} is the approximation field used in FEM (see equation (2.23)), while the second part T^{enr} is introduced by XFEM to model the discontinuous behaviour. Furthermore, $\widetilde{T}_k$ is an extra DOF introduced to node k for the enriched basis function $N_k(\boldsymbol{x})(H(\boldsymbol{x}) - H(\boldsymbol{x}_k))$. These extra DOFs will be referred to as *enriched DOFs* in the following, to differentiate them from T_k, which will be called *standard DOFs*. Unlike T_k, enriched DOFs do not represent nodal temperatures. However, enriched DOFs introduced due to Heaviside enrichment can be used to interpolate the temperature field jump on Γ:

$$
\begin{aligned}
\left[\!\left[T^h(\boldsymbol{x})\right]\!\right]^{(1,2)} &= T^h\left(\boldsymbol{x}^{(2)}\right) - T^h\left(\boldsymbol{x}^{(1)}\right) \\
&= \cancel{\sum_{k\in M} N_k\left(\boldsymbol{x}^{(2)}\right)\widetilde{T}_k} + \sum_{k\in M^H} N_k\left(\boldsymbol{x}^{(2)}\right)\left(H\left(\boldsymbol{x}^{(2)}\right) - H\left(\boldsymbol{x}_k\right)\right)\widetilde{T}_k \\
&- \cancel{\sum_{k\in M} N_k\left(\boldsymbol{x}^{(1)}\right)\widetilde{T}_k} - \sum_{k\in M^H} N_k\left(\boldsymbol{x}^{(1)}\right)\left(H\left(\boldsymbol{x}^{(1)}\right) - H\left(\boldsymbol{x}_k\right)\right)\widetilde{T}_k \\
&= \sum_{k\in M^H} N_k(\boldsymbol{x})\left(+1 - H\left(\boldsymbol{x}_k\right)\right)\widetilde{T}_k - \sum_{k\in M^H} N_k(\boldsymbol{x})\left(-1 - H\left(\boldsymbol{x}_k\right)\right)\widetilde{T}_k \\
&= \sum_{k\in M^H} N_k(\boldsymbol{x})\left(+1 - \cancel{H\left(\boldsymbol{x}_k\right)} + 1 + \cancel{H\left(\boldsymbol{x}_k\right)}\right)\widetilde{T}_k \\
&= 2\sum_{k\in M^H} N_k(\boldsymbol{x})\,\widetilde{T}_k
\end{aligned}
\tag{2.75}
$$

where $\boldsymbol{x}^{(1)}$ coincides with $\boldsymbol{x}^{(2)}$, but lie on different sides of Γ and $N_k\left(\boldsymbol{x}^{(2)}\right) = N_k\left(\boldsymbol{x}^{(1)}\right) = N_k(\boldsymbol{x})$, since N_k is continuous.

2.2.5.2 General case

In the general case of multiple material phases, one Heaviside enrichment $H^{(pq)}$ per interface $\Gamma^{(pq)}$ is needed to model the temperature jump across it:

$$
H^{(pq)}(\boldsymbol{x}) = \begin{cases} -1, & \boldsymbol{x} \in \Omega^{(p)} \\ +1, & \boldsymbol{x} \in \Omega^{(q)} \end{cases} \tag{2.76}
$$

However, in some elements the interfaces of three or more material phases will intersect, as illustrated in figure 2.7. Enriching the nodes of the element that contains this point with one Heaviside function for each interface cannot reproduce the discontinuous temperature field. Instead, in this thesis a *junction enrichment* function is proposed to model the discontinuous displacement field in the case of these *junction points*. The nodes of an element containing a junction point (or line in 3D) created by $n_J \geq 3$ interfaces are enriched with $n_J - 1$ junction functions, one for each intersecting interface except for the last (Bakalakos et al., 2020). In addition, these nodes are not enriched with Heaviside functions. The junction function $J^{(rs)}(\boldsymbol{x})$ used for the interface $\Gamma^{(rs)}$ between the phases $\Omega^{(r)}, \Omega^{(s)}$, which intersects 2 or more other interfaces, is

$$
J^{(rs)}(\boldsymbol{x}) = \begin{cases} -1, & \boldsymbol{x} \in \Omega^{(r)} \\ +1, & \boldsymbol{x} \in \Omega^{(s)} \\ 0, & \boldsymbol{x} \in \Omega - \left(\Omega^{(r)} \cup \Omega^{(s)}\right) \end{cases} \tag{2.77}
$$

In figure 2.7 it can be observed that, a node can be *i*) enriched with junction functions, *ii*) enriched with one or more Heaviside functions corresponding to different material interfaces

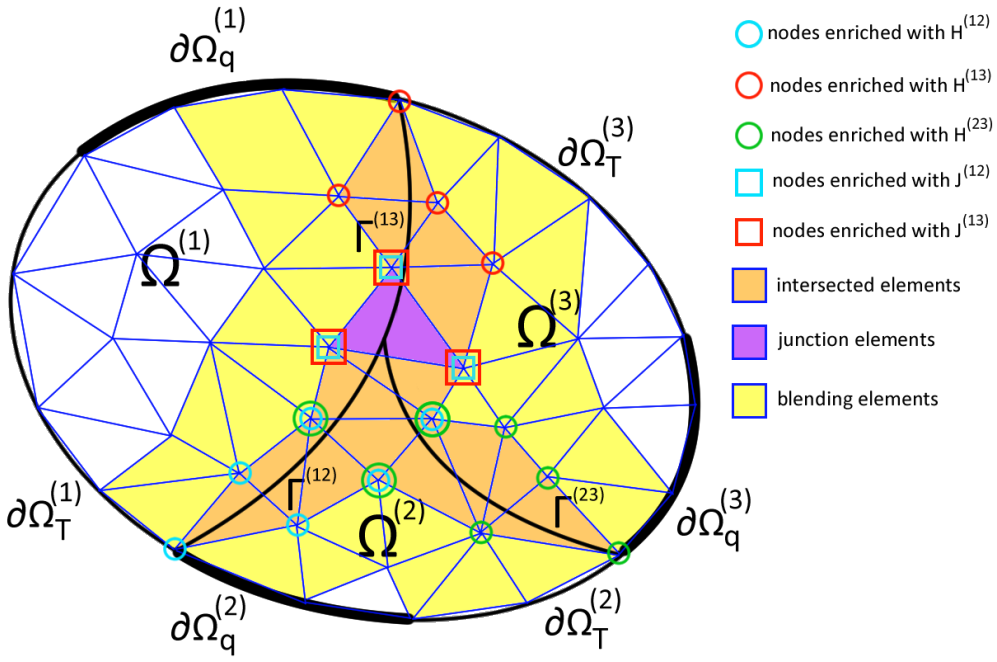


Figure 2.7: Heaviside and junction enrichments when material interfaces intersect.

or *iii*) not enriched at all. Similarly, an element can *i*) contain junctions, *ii*) be intersected by one or more material interfaces, *iii*) be far away from any interface or *iv*) have common nodes with elements in categories i) and ii), in which case it is called a *blending* element. It is more convenient to refer to each Heaviside enrichment as $H^b(\boldsymbol{x}) = H^{(pq)}(\boldsymbol{x})$, where $b = 1, \cdots n_b$ is an integer corresponding to the pair (p, q). Similarly if there are n_c junction enrichments applied in total, with $n_c < n_b$ in general, each one can be referred to as $J^c(\boldsymbol{x}) = J^{(rs)}(\boldsymbol{x})$, where $c = 1, \cdots n_c$ is an integer corresponding to the pair (r, s). The complete XFEM approximation of the temperature field can now be expressed as

$$\begin{aligned} T^h(\boldsymbol{x}) = & \sum_{k \in M} N_k(\boldsymbol{x}) T_k \\ & + \sum_{b=1}^{n_b} \left(\sum_{k \in M_H^b} N_k(\boldsymbol{x}) \left(H^b(\boldsymbol{x}) - H^b(\boldsymbol{x}_k) \right) \widetilde{T}_k^b \right) \\ & + \sum_{c=1}^{n_c} \left(\sum_{k \in M_J^c} N_k(\boldsymbol{x}) \left(J^c(\boldsymbol{x}) - J^c(\boldsymbol{x}_k) \right) \widehat{T}_k^c \right) \end{aligned} \tag{2.78}$$

where

- $N_k(\boldsymbol{x})$ are the Lagrange polynomial shape functions, also used in FEM

- T_k are the standard DOFs representing nodal temperatures.
- $H^b(\boldsymbol{x})$ is any of the n_b Heaviside enrichments applied and M_H^b is the set of nodes enriched with $H^b(\boldsymbol{x})$
- $J^c(\boldsymbol{x})$ is any of the n_c junction enrichments applied and M_J^c is the set of nodes enriched with $J^c(\boldsymbol{x})$
- $\widetilde{T}_k^b$ is an enriched DOF introduced at node k due to the enrichment $H^b(\boldsymbol{x})$
- $\widehat{T}_k^c$ is an enriched DOF introduced at node k due to the enrichment $J^c(\boldsymbol{x})$

Equation (2.78) can be written more concisely by grouping all Heaviside and junction enrichment functions and representing them collectively as $\Psi^a(\boldsymbol{x})$

$$T^h(\boldsymbol{x}) = \sum_{k\in M} N_k(\boldsymbol{x})\, T_k + \sum_{a=1}^{n_a}\left(\sum_{k\in M_\Psi^a} N_k(\boldsymbol{x})\left(\Psi^a(\boldsymbol{x}) - \Psi^a(\boldsymbol{x}_k)\right)\overline{T}_k^a\right) \tag{2.79}$$

where

- $n_a = n_b + n_c$ is the total number of Heaviside and junction enrichment functions
- $\Psi^a(\boldsymbol{x})$, $a = 1, 2\cdots, n_a$ is any of the Heaviside or junction enrichment functions
- M_Ψ^a is the set of all nodes enriched with the enrichment $\Psi^a(\boldsymbol{x})$
- $\overline{T}_k^a$ is the enriched DOF introduced at node k due to the enrichment $\Psi^a(\boldsymbol{x})$

2.2.5.3 Temperature field jump

The jump of the temperature field can be approximated using the enriched DOFs. Let $\boldsymbol{x}$ be a point on the material interface $\Gamma^{(ij)}$ separating $\Omega^{(i)}$ and $\Omega^{(i)}$. Then the points $\boldsymbol{x}^{(i)} \equiv \boldsymbol{x}^{(j)} \equiv \boldsymbol{x}$, lie on different sides of $\Gamma^{(ij)}$, namely $\Omega^{(i)}$ and $\Omega^{(j)}$, respectively. The shape functions N_k are continuous thus

$$N_k\left(\boldsymbol{x}^{(i)}\right) = N_k\left(\boldsymbol{x}^{(j)}\right) = N_k(\boldsymbol{x}) \tag{2.80}$$

However, some of the enrichment functions $\Psi^a(\boldsymbol{x})$ are introduced to model the temperature jump across this material interface and are discontinuous on $\Gamma^{(ij)}$, namely $[\![\Psi^a(\boldsymbol{x})]\!]^{(ij)} \neq 0$. Meanwhile, the rest are introduced to model the temperature jump across another material interface and are continuous on $\Gamma^{(ij)}$, namely $[\![\Psi^a(\boldsymbol{x})]\!]^{(ij)} = 0$. For example, for any Heaviside enrichment function $H^{(pq)}(\boldsymbol{x})$

$$[\![H^{(pq)}]\!]^{(ij)}(\boldsymbol{x}) = H^{(pq)}\left(\boldsymbol{x}^{(j)}\right) - H^{(pq)}\left(\boldsymbol{x}^{(i)}\right) = 0,\ -2,\ \text{or}\ +2 \tag{2.81}$$

depending on whether one or both of the phases $\Omega^{(p)}$ and $\Omega^{(q)}$ coincide with $\Omega^{(i)}$ and $\Omega^{(j)}$. Similarly, for any junction enrichment function $J^{(rs)}(\boldsymbol{x})$

$$\left[\!\left[J^{(rs)}\right]\!\right]^{(ij)}(\boldsymbol{x}) = J^{(rs)}\left(\boldsymbol{x}^{(j)}\right) - J^{(rs)}\left(\boldsymbol{x}^{(i)}\right) = 0,\ -2,\ +2,\ -1 \text{ or } +1 \tag{2.82}$$

depending on whether one or both of the phases $\Omega^{(r)}$ and $\Omega^{(s)}$ coincide with $\Omega^{(i)}$ and $\Omega^{(j)}$. The temperature field jump across $\Gamma^{(ij)}$ can be approximated as

$$\begin{aligned}
\left[\!\left[T^h(\boldsymbol{x})\right]\!\right]^{(ij)} =& T^h\left(\boldsymbol{x}^{(j)}\right) - T^h\left(\boldsymbol{x}^{(i)}\right) \\
=& \sum_{k\in M} N_k\left(\boldsymbol{x}^{(j)}\right) T_k + \sum_{a=1}^{n_a}\left(\sum_{k\in M_\Psi^a} N_k\left(\boldsymbol{x}^{(j)}\right)\left(\Psi^a\left(\boldsymbol{x}^{(j)}\right) - \Psi^a(\boldsymbol{x}_k)\right)\overline{T}_k^a\right) \\
& - \sum_{k\in M} N_k\left(\boldsymbol{x}^{(i)}\right) T_k - \sum_{a=1}^{n_a}\left(\sum_{k\in M_\Psi^a} N_k\left(\boldsymbol{x}^{(i)}\right)\left(\Psi^a\left(\boldsymbol{x}^{(i)}\right) - \Psi^a(\boldsymbol{x}_k)\right)\overline{T}_k^a\right) \\
=& \cancel{\sum_{k\in M} N_k(\boldsymbol{x})\, T_k} + \sum_{a=1}^{n_a}\left(\sum_{k\in M_\Psi^a} N_k(\boldsymbol{x})\left(\Psi^a\left(\boldsymbol{x}^{(j)}\right) - \Psi^a(\boldsymbol{x}_k)\right)\overline{T}_k^a\right) \\
& - \cancel{\sum_{k\in M} N_k(\boldsymbol{x})\, T_k} - \sum_{a=1}^{n_a}\left(\sum_{k\in M_\Psi^a} N_k(\boldsymbol{x})\left(\Psi^a\left(\boldsymbol{x}^{(i)}\right) - \Psi^a(\boldsymbol{x}_k)\right)\overline{T}_k^a\right) \\
=& \sum_{a=1}^{n_a}\left(\sum_{k\in M_\Psi^a} N_k(\boldsymbol{x})\left(\Psi^a\left(\boldsymbol{x}^{(j)}\right) - \cancel{\Psi^a(\boldsymbol{x}_k)} - \Psi^a\left(\boldsymbol{x}^{(i)}\right) + \cancel{\Psi^a(\boldsymbol{x}_k)}\right)\overline{T}_k^a\right)
\end{aligned} \tag{2.83}$$

therefore the approximation of the temperature field jump across the interface $\Gamma^{(ij)}$ can be calculated with respect to all enriched DOFs as

$$\left[\!\left[T^h(\boldsymbol{x})\right]\!\right]^{(ij)} = \sum_{a=1}^{n_a}\left(\sum_{k\in M_\Psi^a} N_k(\boldsymbol{x})\left[\!\left[\Psi^a\right]\!\right]^{(ij)}\overline{T}_k^a\right) \tag{2.84}$$

Regardless of whether $\Psi^a(\boldsymbol{x})$ is a Heaviside or junction enrichment, $\left[\!\left[\Psi^a(\boldsymbol{x})\right]\!\right]^{(ij)} = 0$ means that the approximation of the temperature field jump $\left[\!\left[T^h(\boldsymbol{x})\right]\!\right]^{(ij)}$ across the interface $\Gamma^{(ij)}$ does not depend on the enriched DOFs $\overline{T}_k^a$ introduced for this enrichment $\Psi^a(\boldsymbol{x})$. In fact, the jump across each material interface $\Gamma^{(ij)}$ depends only on those enrichments introduced specifically for $\Gamma^{(ij)}$ and their corresponding DOFs.

2.2.5.4 Blending elements

If an element is not intersected and does not include a junction, but is located next to another element that does, then only some of its nodes will be enriched. Figure 2.7 depicts examples of such blending elements, which form a layer between standard (non-enriched) elements and fully enriched ones. The polynomial shape function used in FEM and XFEM have the partition of unity property, namely any function $f(\boldsymbol{x})$ can be recovered inside an element e

$$\sum_{k \in M_e} N_k(\boldsymbol{x}) f(\boldsymbol{x}_k) = f(\boldsymbol{x}) \tag{2.85}$$

where M_e is the set of all nodes of the element e. Because blending elements have only some of their nodes enriched, the partition of unity property is violated inside them and an enrichment function $\Psi^a(\boldsymbol{x})$ cannot be recovered

$$\sum_{k \in M_e^{\Psi^a}} N_k(\boldsymbol{x}) \Psi^a(\boldsymbol{x}_k) \neq \sum_{k \in M_e} N_k(\boldsymbol{x}) \Psi^a(\boldsymbol{x}_k) = \Psi^a(\boldsymbol{x}) \tag{2.86}$$

where $M_e^{\Psi^a} \subset M_e$ are only those nodes of element e that are enriched with $\Psi^a(\boldsymbol{x})$. The incomplete interpolation inside blending elements generally introduces errors into the approximation field of equation (2.79). However, the Heaviside and junction enrichments of the proposed formulation do not encounter this is problem. A blending element is not intersected by a material interface $\Gamma^{(ij)}$, thus all of its points lie on the same side of $\Gamma^{(ij)}$, which means that the enrichment $\Psi^a(\boldsymbol{x})$ that was introduced for the discontinuity $\Gamma^{(ij)}$ has the same value in all points of the blending element, including its nodes:

$$\Psi^a(\boldsymbol{x}) = \left\{ \begin{array}{c} H^{(pq)}(\boldsymbol{x}) = H^{(pq)}(\boldsymbol{x}_k) \\ \text{or} \\ J^{(rs)}(\boldsymbol{x}) = J^{(rs)}(\boldsymbol{x}_k) \end{array} \right\} = \Psi^a(\boldsymbol{x}_k) \tag{2.87}$$

As a result, no errors are introduced to the approximate temperature field, since its enriched part vanishes inside blending elements

$$\begin{aligned} T^h(\boldsymbol{x}) &= \sum_{k \in M_e} N_k(\boldsymbol{x}) T_k + \sum_{a=1}^{n_a} \left(\sum_{k \in M_e^{\Psi^a}} N_k(\boldsymbol{x}) \cancel{\left(\Psi^a(\boldsymbol{x}) - \Psi^a(\boldsymbol{x}_k) \right)} \overline{T}_k^a \right) \\ &= \sum_{k \in M_e} N_k(\boldsymbol{x}) T_k \end{aligned} \tag{2.88}$$

Similarly, the approximate field evaluated at a node $\boldsymbol{x}_l$ is equal to the standard DOF T_l, since the standard part is

$$T^{std}(\boldsymbol{x}_l) = \sum_{k \in M} N_k(\boldsymbol{x}_l) T_k = \sum_{\substack{k \in M \\ k \neq l}} N_k(\boldsymbol{x}_l) T_k + N_l(\boldsymbol{x}_l) T_l = 0 + 1 \; T_l = T_l \tag{2.89}$$

because

$$N_k(\boldsymbol{x}_l) = \begin{cases} 1 & , k = l \\ 0 & , k \neq l \end{cases} \tag{2.90}$$

and due to the use of a shifted enrichment $\Psi^a(\boldsymbol{x}) - \Psi^a(\boldsymbol{x}_k)$, the enriched part degenerates to 0

$$\begin{aligned} T^{enr}(\boldsymbol{x}_l) &= \sum_{a=1}^{n_a} \left(\sum_{k \in M_\Psi^a} N_k(\boldsymbol{x}_l) \left(\Psi^a(\boldsymbol{x}_l) - \Psi^a(\boldsymbol{x}_k) \right) \overline{T}_k^a \right) \\ &= \sum_{a=1}^{n_a} \left(\sum_{\substack{k \in M_\Psi^a \\ k \neq l}} \cancel{N_k(\boldsymbol{x}_l)} \left(\Psi^a(\boldsymbol{x}_l) - \Psi^a(\boldsymbol{x}_k) \right) \overline{T}_k^a + N_l(\boldsymbol{x}_l) \cancel{\left(\Psi^a(\boldsymbol{x}_l) - \Psi^a(\boldsymbol{x}_l) \right)} \overline{T}_l^a \right) \\ &= 0 \end{aligned} \tag{2.91}$$

As a result, applying Dirichlet boundary conditions $T(\boldsymbol{x}) = \bar{T}$ to node $\boldsymbol{x}_l$ is done similarly to FEM, namely by setting the corresponding standard DOF T_j equal to the prescribed temperature $\bar{T}$, even if nodes on $\partial\Omega_T$ are enriched:

$$\bar{T} = T^h(\boldsymbol{x}_l) = T^{std}(\boldsymbol{x}_l) + T^{enr}(\boldsymbol{x}_l) = T_l \tag{2.92}$$

2.2.6 XFEM discretized equations

Similarly to FEM, the approximation of equation (2.79) is also used for the test function

$$\delta T^h(\boldsymbol{x}) = \sum_{k \in M} N_k(\boldsymbol{x}) \delta T_k + \sum_{a=1}^{n_a} \left(\sum_{k \in M_\Psi^a} N_k(\boldsymbol{x}) \left(\Psi^a(\boldsymbol{x}) - \Psi^a(\boldsymbol{x}_k) \right) \overline{\delta T}_k^a \right) \tag{2.93}$$

where δT_k and $\overline{\delta T}_k^a$ are standard and enriched DOFs for the test function and different from T_k and $\overline{T}_k^a$, in general. It is more convenient to work with the matrix-vector form of the approximate temperature field

$$\begin{aligned} T^h &= \boldsymbol{N}(\boldsymbol{x}) \cdot \boldsymbol{d}_e \\ \delta T^h &= \boldsymbol{N}(\boldsymbol{x}) \cdot \boldsymbol{\delta d}_e \end{aligned} \tag{2.94}$$

where $\boldsymbol{N}(\boldsymbol{x})$ is a matrix (row vector actually) consisting of the standard (submatrix $\boldsymbol{N}^{std}(\boldsymbol{x})$) and enriched (submatrix $\boldsymbol{N}^{enr}(\boldsymbol{x})$) basis functions corresponding to each each enrichment $\Psi^a(\boldsymbol{x})$ and each node k of element e:

$$\begin{aligned} \boldsymbol{N}(\boldsymbol{x}) &= \left[\begin{array}{cc} \boldsymbol{N}^{std}(\boldsymbol{x}) & \boldsymbol{N}^{enr}(\boldsymbol{x}) \end{array} \right] \\ \boldsymbol{N}^{std}(\boldsymbol{x}) &= \left[\begin{array}{ccc} \cdots & N_k(\boldsymbol{x}) & \cdots \end{array} \right] \\ \boldsymbol{N}^{enr}(\boldsymbol{x}) &= \left[\begin{array}{ccc} \cdots & N_k(\boldsymbol{x})\left(\Psi^a(\boldsymbol{x}) - \Psi^a(\boldsymbol{x}_k)\right) & \cdots \end{array} \right] \end{aligned} \tag{2.95}$$

and $\boldsymbol{d}_e$, $\boldsymbol{\delta d}_e$ are vectors containing the standard (subvector $\boldsymbol{d}_e^{std}$) and enriched (subvector $\boldsymbol{d}_e^{enr}$) DOFs introduced for each enrichment Ψ^a to each node k of the element e:

$$\begin{aligned} \boldsymbol{d}_e &= \left[\begin{array}{c} \boldsymbol{d}_e^{std} \\ \boldsymbol{d}_e^{enr} \end{array} \right] \quad \boldsymbol{d}_e^{std} = \left[\begin{array}{c} \vdots \\ T_k \\ \vdots \end{array} \right] \quad \boldsymbol{d}_e^{enr} = \left[\begin{array}{c} \vdots \\ \overline{T}_k^a \\ \vdots \end{array} \right] \\ \boldsymbol{\delta d}_e &= \left[\begin{array}{c} \boldsymbol{\delta d}_e^{std} \\ \boldsymbol{\delta d}_e^{enr} \end{array} \right] \quad \boldsymbol{\delta d}_e^{std} = \left[\begin{array}{c} \vdots \\ \delta T_k \\ \vdots \end{array} \right] \quad \boldsymbol{\delta d}_e^{enr} = \left[\begin{array}{c} \vdots \\ \overline{\delta T}_k^a \\ \vdots \end{array} \right] \end{aligned} \tag{2.96}$$

Also consider that all standard and enriched DOFs of the domain are gathered in the *global* vectors $\boldsymbol{d}$ and $\boldsymbol{\delta d}$ for the trial and test functions, respectively. To extract the element vector from the corresponding global vector the matrix $\boldsymbol{P}_e$ is used

$$\begin{aligned} \boldsymbol{d}_e &= \boldsymbol{P}_e \cdot \boldsymbol{d} \\ \boldsymbol{\delta d}_e &= \boldsymbol{P}_e \cdot \boldsymbol{\delta d} \end{aligned} \tag{2.97}$$

where $\boldsymbol{P}_e$ is a boolean matrix, namely it contains only 0 or 1 entries, that correlates each element DOF (row of $\boldsymbol{P}_e$) to one exactly global DOF (column of $\boldsymbol{P}_e$). Therefore, the approximation of the temperature field can be rewritten as

$$\begin{aligned} T^h(\boldsymbol{x}) &= \boldsymbol{N}(\boldsymbol{x}) \cdot \boldsymbol{d}_e = \boldsymbol{N}(\boldsymbol{x}) \cdot \boldsymbol{P}_e \cdot \boldsymbol{d} \\ \delta T^h(\boldsymbol{x}) &= \boldsymbol{N}(\boldsymbol{x}) \cdot \boldsymbol{\delta d}_e = \boldsymbol{N}(\boldsymbol{x}) \cdot \boldsymbol{P}_e \cdot \boldsymbol{\delta d} \end{aligned} \tag{2.98}$$

The derivatives of the discretized trial and test function in matrix-vector form are

$$\begin{aligned} \nabla T^h(\boldsymbol{x}) &= \boldsymbol{B}(\boldsymbol{x}) \cdot \boldsymbol{d}_e = \boldsymbol{B}(\boldsymbol{x}) \cdot \boldsymbol{P}_e \cdot \boldsymbol{d} \\ \nabla \delta T^h(\boldsymbol{x}) &= \boldsymbol{B}(\boldsymbol{x}) \cdot \boldsymbol{\delta d}_e = \boldsymbol{B}(\boldsymbol{x}) \cdot \boldsymbol{P}_e \cdot \boldsymbol{\delta d} \end{aligned} \tag{2.99}$$

where matrix $\boldsymbol{B}(\boldsymbol{x})$ contains the derivatives of the standard (submatrix $\boldsymbol{B}^{std}(\boldsymbol{x})$) and enriched (submatrix $\boldsymbol{B}^{enr}(\boldsymbol{x})$) basis functions corresponding to each each enrichment $\Psi^a(\boldsymbol{x})$ and each node k of element e:

$$
\begin{aligned}
\boldsymbol{B}(\boldsymbol{x}) &= \left[\begin{array}{cc} \boldsymbol{B}^{std}(\boldsymbol{x}) & \boldsymbol{B}^{enr}(\boldsymbol{x}) \end{array} \right] \\
\boldsymbol{B}^{std}(\boldsymbol{x}) &= \left[\begin{array}{ccc} & \dfrac{\partial N_k(\boldsymbol{x})}{\partial x} & \\ \cdots & \dfrac{\partial N_k(\boldsymbol{x})}{\partial y} & \cdots \\ & \dfrac{\partial N_k(\boldsymbol{x})}{\partial z} & \end{array} \right] \\
\boldsymbol{B}^{enr}(\boldsymbol{x}) &= \left[\begin{array}{ccc} & \dfrac{\partial \left(N_k(\boldsymbol{x}) \left(\Psi^a(\boldsymbol{x}) - \Psi^a(\boldsymbol{x}_k) \right) \right)}{\partial x} & \\ \cdots & \dfrac{\partial \left(N_k(\boldsymbol{x}) \left(\Psi^a(\boldsymbol{x}) - \Psi^a(\boldsymbol{x}_k) \right) \right)}{\partial y} & \cdots \\ & \dfrac{\partial \left(N_k(\boldsymbol{x}) \left(\Psi^a(\boldsymbol{x}) - \Psi^a(\boldsymbol{x}_k) \right) \right)}{\partial z} & \end{array} \right]
\end{aligned}
\tag{2.100}
$$

Since both Heaviside and junction enrichment functions are piece-wise constant, their partial derivatives in each material phase $\Omega^{(i)}$ are

$$
\frac{\partial \Psi^a(\boldsymbol{x})}{\partial x} = \frac{\partial \Psi^a(\boldsymbol{x})}{\partial y} = \frac{\partial \Psi^a(\boldsymbol{x})}{\partial z} = 0 \tag{2.101}
$$

and the calculation of $\boldsymbol{B}^{enr}(\boldsymbol{x})$ can be simplified to

$$
\boldsymbol{B}^{enr}(\boldsymbol{x}) = \left[\begin{array}{ccc} & \dfrac{\partial N_k(\boldsymbol{x})}{\partial x} \left(\Psi^a(\boldsymbol{x}) - \Psi^a(\boldsymbol{x}_k) \right) & \\ \cdots & \dfrac{\partial N_k(\boldsymbol{x})}{\partial y} \left(\Psi^a(\boldsymbol{x}) - \Psi^a(\boldsymbol{x}_k) \right) & \cdots \\ & \dfrac{\partial N_k(\boldsymbol{x})}{\partial z} \left(\Psi^a(\boldsymbol{x}) - \Psi^a(\boldsymbol{x}_k) \right) & \end{array} \right] \tag{2.102}
$$

Furthermore, the matrix-vector form of the temperature field jump across a material interface $\Gamma^{(ij)}$ is

$$
\left[\!\left[T^h(\boldsymbol{x}) \right]\!\right]^{(ij)} = \overline{\boldsymbol{N}}^{(ij)}(\boldsymbol{x}) \cdot \boldsymbol{d}_e = \left[\begin{array}{cc} \overline{\boldsymbol{N}}_{std} & \overline{\boldsymbol{N}}^{(ij)}_{enr}(\boldsymbol{x}) \end{array} \right] \cdot \left[\begin{array}{c} \boldsymbol{\delta d}^{std}_e \\ \boldsymbol{\delta d}^{enr}_e \end{array} \right] \tag{2.103}
$$

where $\overline{\boldsymbol{N}}^{(ij)}(\boldsymbol{x})$, $\overline{\boldsymbol{N}}_{std}$ and $\overline{\boldsymbol{N}}^{(ij)}_{enr}(\boldsymbol{x})$ are row vectors:

$$\begin{aligned}
&\overline{\boldsymbol{N}}^{(ij)}(\boldsymbol{x}) = \left[\begin{array}{cc} \overline{\boldsymbol{N}}_{std} & \overline{\boldsymbol{N}}^{(ij)}_{enr}(\boldsymbol{x}) \end{array}\right] \\
&\overline{\boldsymbol{N}}_{std} = \boldsymbol{0} \\
&\overline{\boldsymbol{N}}^{(ij)}_{enr}(\boldsymbol{x}) = \left[\begin{array}{ccc} \cdots & N_k(\boldsymbol{x}) \left[\!\left[\Psi^a(\boldsymbol{x})\right]\!\right]^{(ij)} & \cdots \end{array}\right]
\end{aligned} \tag{2.104}$$

As discussed in section 2.2.5.3, given an interface $\Gamma^{(ij)}$, only those enrichments that were introduced specifically to model the temperature jump across $\Gamma^{(ij)}$ will have $\left[\!\left[\Psi^a(\boldsymbol{x})\right]\!\right]^{(ij)} \neq 0$. Therefore, most entries of $\overline{\boldsymbol{N}}^{(ij)}_{enr}(\boldsymbol{x})$ will be 0, since they correspond to other enrichment functions, which were introduced for material interfaces other than $\Gamma^{(ij)}$. Similarly, for the test function δT:

$$\left[\!\left[\delta T^h(\boldsymbol{x})\right]\!\right]^{(ij)} = \overline{\boldsymbol{N}}^{(ij)}(\boldsymbol{x}) \cdot \boldsymbol{\delta d}_e \tag{2.105}$$

and with respect to global DOF vectors

$$\begin{aligned}
\left[\!\left[T^h(\boldsymbol{x})\right]\!\right]^{(ij)} &= \overline{\boldsymbol{N}}^{(ij)}(\boldsymbol{x}) \cdot \boldsymbol{P}_e \cdot \boldsymbol{\delta d} \\
\left[\!\left[\delta T^h(\boldsymbol{x})\right]\!\right]^{(ij)} &= \overline{\boldsymbol{N}}^{(ij)}(\boldsymbol{x}) \cdot \boldsymbol{P}_e \cdot \boldsymbol{\delta d}
\end{aligned} \tag{2.106}$$

In matrix-vector form, the tensor multiplication $\nabla \delta T^h \cdot \boldsymbol{k} \cdot \nabla T^h$ of equation (2.72) is written as a row vector $\times$ a matrix $\times$ a column vector: $\left(\nabla \delta T^h\right)^T \cdot \boldsymbol{k} \cdot \nabla T^h$. Moreover, δT^h and $\left[\!\left[\delta T^h\right]\!\right]^{(ij)}$ are scalar quantities, thus $\delta T^h = \left(\delta T^h\right)^T$ and $\left[\!\left[\delta T^h\right]\!\right]^{(ij)} = \left(\left[\!\left[\delta T^h\right]\!\right]^{(ij)}\right)^T$. Therefore, equation (2.72) is rewritten as

$$\begin{aligned}
\sum_{e=1}^{n_e} \left(\sum_{(i,j)\in M^\Gamma} \int_{\Gamma_e^{(ij)}} \left(\left[\!\left[\delta T^h\right]\!\right]^{(ij)}\right)^T \frac{1}{\alpha^{(ij)}} \left[\!\left[T^h\right]\!\right]^{(ij)} \, d\Gamma \right) + \sum_{e=1}^{n_e} \int_{\Omega_e} \left(\nabla \delta T^h\right)^T \cdot \boldsymbol{k} \cdot \nabla T^h d\Omega \\
= \sum_{e=1}^{n_e} \int_{\Omega_e} \left(\delta T^h\right)^T r \, d\Omega + \sum_{e=1}^{n_e} \int_{\partial\Omega_{qe}} \left(\delta T^h\right)^T \bar{q}_n \, d\Gamma
\end{aligned} \tag{2.107}$$

By combining all equations above, the matrix-vector form of the finite-dimensional weak

form used in XFEM is obtained

$$\begin{aligned}
&\sum_{e=1}^{n_e}\left(\sum_{(i,j)\in M^{\Gamma}}\int_{\Gamma_e^{(ij)}}\left(\overline{\boldsymbol{N}}^{(ij)}(\boldsymbol{x})\cdot\boldsymbol{P}_e\cdot\boldsymbol{\delta d}\right)^T\frac{1}{\alpha^{(ij)}}\left(\overline{\boldsymbol{N}}^{(ij)}(\boldsymbol{x})\cdot\boldsymbol{P}_e\cdot\boldsymbol{d}\right)d\Gamma\right)\\
&+\sum_{e=1}^{n_e}\int_{\Omega_e}\left(\boldsymbol{B}(\boldsymbol{x})\cdot\boldsymbol{P}_e\cdot\boldsymbol{\delta d}\right)^T\cdot\boldsymbol{k}(\boldsymbol{x})\cdot\left(\boldsymbol{B}(\boldsymbol{x})\cdot\boldsymbol{P}_e\cdot\boldsymbol{d}\right)d\Omega\\
=&\sum_{e=1}^{n_e}\int_{\Omega_e}\left(\boldsymbol{N}(\boldsymbol{x})\cdot\boldsymbol{P}_e\cdot\boldsymbol{\delta d}\right)^T r(\boldsymbol{x})\,d\Omega\\
&+\sum_{e=1}^{n_e}\int_{\partial\Omega_{qe}}\left(\boldsymbol{N}(\boldsymbol{x})\cdot\boldsymbol{P}_e\cdot\boldsymbol{\delta d}\right)^T\bar{q}_n(\boldsymbol{x})\,d\Gamma
\end{aligned}\tag{2.108}$$

Pulling the constants out of the corresponding integrals and summations:

$$\begin{aligned}
&\boldsymbol{\delta d}^T\cdot\left(\sum_{e=1}^{n_e}\boldsymbol{P}_e^T\cdot\left(\sum_{(i,j)\in M^{\Gamma}}\int_{\Gamma_e^{(ij)}}\frac{1}{\alpha^{(ij)}}\left(\overline{\boldsymbol{N}}^{(ij)}(\boldsymbol{x})\right)^T\cdot\overline{\boldsymbol{N}}^{(ij)}(\boldsymbol{x})\,d\Gamma\right)\cdot\boldsymbol{P}_e\right)\cdot\boldsymbol{d}\\
&+\boldsymbol{\delta d}^T\cdot\left(\sum_{e=1}^{n_e}\boldsymbol{P}_e^T\cdot\left(\int_{\Omega_e}\boldsymbol{B}^T(\boldsymbol{x})\cdot\boldsymbol{k}(\boldsymbol{x})\cdot\boldsymbol{B}(\boldsymbol{x})\,d\Omega\right)\cdot\boldsymbol{P}_e\right)\cdot\boldsymbol{d}\\
=&\boldsymbol{\delta d}^T\cdot\sum_{e=1}^{n_e}\boldsymbol{P}_e^T\cdot\left(\int_{\Omega_e}\boldsymbol{N}^T(\boldsymbol{x})\,r(\boldsymbol{x})\,d\Omega\right)\\
&+\boldsymbol{\delta d}^T\cdot\sum_{e=1}^{n_e}\boldsymbol{P}_e^T\cdot\left(\int_{\partial\Omega_{qe}}\boldsymbol{N}^T(\boldsymbol{x})\,\bar{q}_n(\boldsymbol{x})\,d\Gamma\right)
\end{aligned}\tag{2.109}$$

At this point, the element-level conductivity matrices and thermal load vectors need to be defined. The *bulk conductivity matrix* of an element e is

$$\boldsymbol{K}_e=\int_{\Omega_e}\boldsymbol{B}^T(\boldsymbol{x})\cdot\boldsymbol{k}(\boldsymbol{x})\cdot\boldsymbol{B}(\boldsymbol{x})\,d\Omega\tag{2.110}$$

whereas the *interface conductivity matrix* of the element is

$$\overline{\boldsymbol{K}}_e = \sum_{(i,j)\in M^\Gamma} \int_{\Gamma_e^{(ij)}} \frac{1}{\alpha^{(ij)}} \left(\overline{\boldsymbol{N}}^{(ij)}(\boldsymbol{x})\right)^T \cdot \overline{\boldsymbol{N}}^{(ij)}(\boldsymbol{x})\, d\Gamma \tag{2.111}$$

and finally the element-level thermal load vectors are

$$\begin{aligned} \boldsymbol{f}_{re} &= \int_{\Omega_e} \boldsymbol{N}^T(\boldsymbol{x})\, r(\boldsymbol{x})\, d\Omega \\ \boldsymbol{f}_{qe} &= \int_{\partial\Omega_{qe}} \boldsymbol{N}^T(\boldsymbol{x})\, \bar{q}_n(\boldsymbol{x})\, d\Gamma \end{aligned} \tag{2.112}$$

Equations (2.110, 2.112) are similar to standard FEM. The interface conductivity matrix of equation (2.110) represents the element-level conductivity due to the interfacial conductance $k^{(ij)}$ which is the reciprocal of the resistance, that is, $k^{(ij)} = \frac{1}{\alpha^{(ij)}}$. By assembling all element-level conductivity matrices $\boldsymbol{K}_e$ and $\overline{\boldsymbol{K}}_e$, the global conductivity matrix $\boldsymbol{K}$ is obtained, which expresses the thermal conductivity of the whole discretized domain:

$$\begin{aligned} \boldsymbol{K} =& \sum_{e=1}^{n_e} \boldsymbol{P}_e^T \cdot \left(\boldsymbol{K}_e + \overline{\boldsymbol{K}}_e\right) \cdot \boldsymbol{P}_e \\ =& \sum_{e=1}^{n_e} \boldsymbol{P}_e^T \cdot \left(\int_{\Omega_e} \boldsymbol{B}^T(\boldsymbol{x}) \cdot \boldsymbol{k}(\boldsymbol{x}) \cdot \boldsymbol{B}(\boldsymbol{x})\, d\Omega \right. \\ & \left. + \sum_{(i,j)\in M^\Gamma} \int_{\Gamma_e^{(ij)}} \frac{1}{\alpha^{(ij)}} \left(\overline{\boldsymbol{N}}^{(ij)}(\boldsymbol{x})\right)^T \cdot \overline{\boldsymbol{N}}^{(ij)}(\boldsymbol{x})\, d\Gamma \right) \cdot \boldsymbol{P}_e \end{aligned} \tag{2.113}$$

Similarly, all element-level vectors $\boldsymbol{f}_{re}$, $\boldsymbol{f}_{qe}$ are assembled into a global vector $\boldsymbol{f}$ that represents the thermal loads due to the heat source $r(\boldsymbol{x})$ and the surface heat flux $\bar{q}_n(\boldsymbol{x})$

$$\boldsymbol{f} = \sum_{e=1}^{n_e} \boldsymbol{P}_e^T \cdot (\boldsymbol{f}_{re} + \boldsymbol{f}_{qe}) = \sum_{e=1}^{n_e} \boldsymbol{P}_e^T \cdot \left(\int_{\Omega_e} \boldsymbol{N}^T(\boldsymbol{x})\, r(\boldsymbol{x})\, d\Omega + \int_{\partial\Omega_{qe}} \boldsymbol{N}^T(\boldsymbol{x})\, \bar{q}_n(\boldsymbol{x})\, d\Gamma \right) \tag{2.114}$$

Finally, equation (2.109) is rewritten as a linear system:

$$\boldsymbol{K} \cdot \boldsymbol{d} = \boldsymbol{f} \tag{2.115}$$

As explained in section 2.2.5.4, the Dirichlet boundary conditions can be considered by applying the prescribed temperatures to the standard DOFs of the corresponding nodes l

that lie on $\partial\Omega_T$, namely $T_l = \bar{T}$. These standard DOFs are removed from the vector $\boldsymbol{d}$, which will then contain only unknown standard and enriched DOFs. The corresponding rows and columns are also removed from the matrix $\boldsymbol{K}$ and vector $\boldsymbol{f}$. As a result, the domain is adequately supported and the linear system of equation (2.115) is positive definite. Therefore, there is a unique solution for the standard and enriched DOFs in $\boldsymbol{d}$. Solving this linear system is an interesting topic that will be explained in detail in later chapters. Once the the standard and enriched DOFs in $\boldsymbol{d}$ have been calculated, the temperature and heat flux fields at any point in the domain can be approximated using equations (2.79) and (2.41), while equation (2.84) is used for the jump of the temperature field.

2.2.6.1 Example

Consider the example illustrated in figure 2.8 with 3 material phases $\Omega^{(1)}$, $\Omega^{(2)}$, $\Omega^{(3)}$. The Heaviside enrichment functions used to model the temperature field jump across the material interfaces $\Gamma^{(12)}, \Gamma^{(13)}, \Gamma^{(23)}$ are respectively

$$\Psi^1(\boldsymbol{x}) = H^{(12)}(\boldsymbol{x}) = \begin{cases} -1, & \boldsymbol{x} \in \Omega^{(1)} \\ +1, & \boldsymbol{x} \in \Omega^{(2)} \end{cases}$$

$$\Psi^2(\boldsymbol{x}) = H^{(13)}(\boldsymbol{x}) = \begin{cases} -1, & \boldsymbol{x} \in \Omega^{(1)} \\ +1, & \boldsymbol{x} \in \Omega^{(3)} \end{cases}$$

$$\Psi^3(\boldsymbol{x}) = H^{(23)}(\boldsymbol{x}) = \begin{cases} -1, & \boldsymbol{x} \in \Omega^{(2)} \\ +1, & \boldsymbol{x} \in \Omega^{(3)} \end{cases}$$

The nodes of elements intersected by one or more of the interfaces are enriched with one or more of the above functions. However, the element that connects nodes 6-7-11-10 contains the junction point of these three interfaces, therefore its nodes are only enriched with $3 - 1 = 2$ junction enrichments:

$$\Psi^4(\boldsymbol{x}) = J^{(12)}(\boldsymbol{x}) = \begin{cases} -1, & \boldsymbol{x} \in \Omega^{(1)} \\ +1, & \boldsymbol{x} \in \Omega^{(2)} \\ 0, & \boldsymbol{x} \in \Omega^{(3)} \end{cases}$$

$$\Psi^5(\boldsymbol{x}) = J^{(13)}(\boldsymbol{x}) = \begin{cases} -1, & \boldsymbol{x} \in \Omega^{(1)} \\ +1, & \boldsymbol{x} \in \Omega^{(3)} \\ 0, & \boldsymbol{x} \in \Omega^{(2)} \end{cases}$$

It should be noted that a junction enriched node is not enriched with Heaviside functions associated with the interfaces that form the junction. In this example, $H^{(12)}(\boldsymbol{x})$, $H^{(13)}(\boldsymbol{x})$ and $H^{(14)}(\boldsymbol{x})$ are not applied to nodes enriched with $J^{(12)}(\boldsymbol{x})$ and $J^{(13)}(\boldsymbol{x})$. All other nodes of the mesh are not enriched. The vector of standard and enriched DOFs of the element that connects nodes 10-11-15-14 is

$$\overline{\boldsymbol{T}} = \begin{bmatrix} T_{10} & T_{11} & T_{15} & T_{14} & \overline{T}^4_{10} & \overline{T}^5_{10} & \overline{T}^4_{11} & \overline{T}^5_{11} & \overline{T}^2_{15} & \overline{T}^3_{15} & \overline{T}^2_{14} & \overline{T}^3_{14} \end{bmatrix}^T$$

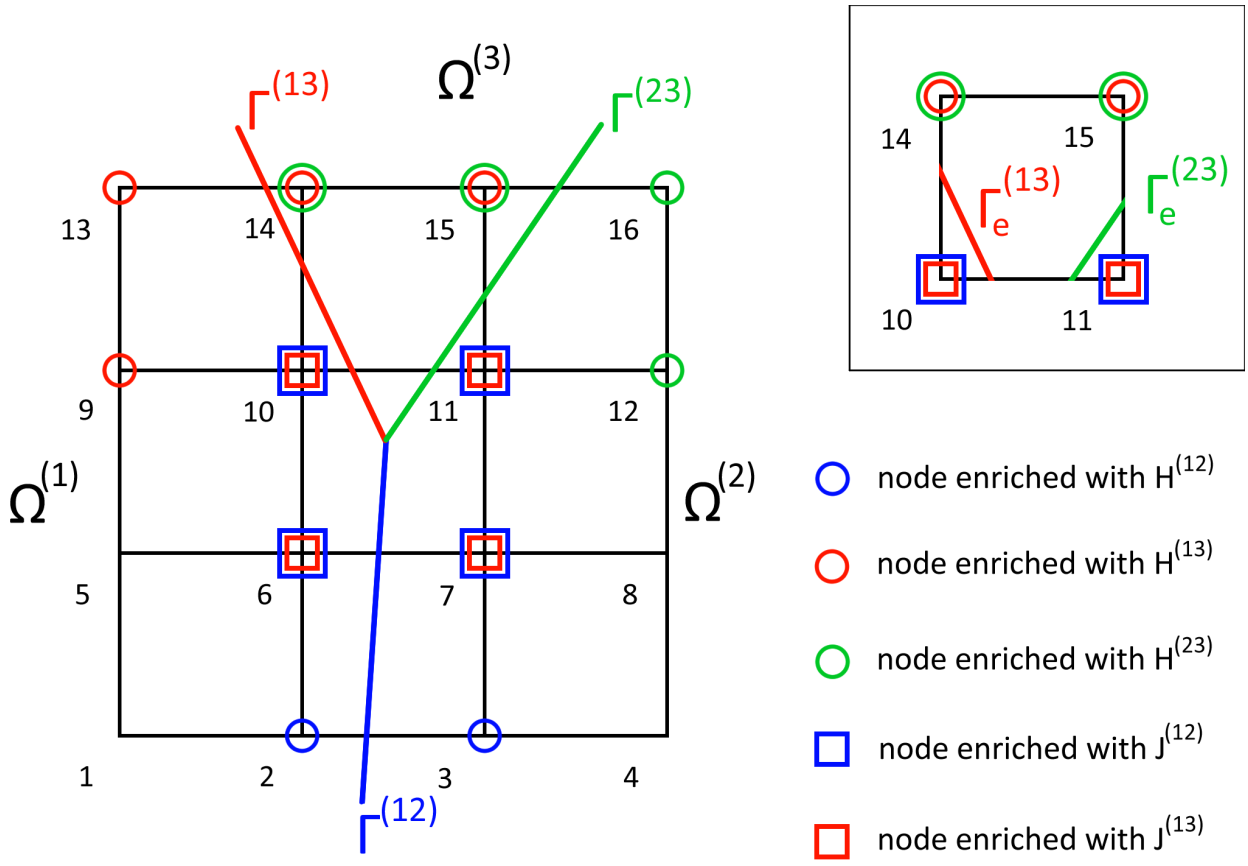


Figure 2.8: Step and junction enrichment of various nodes.

The row vector $\overline{\boldsymbol{N}}^{(ij)}(\boldsymbol{x})$ of equation (2.103) used for approximating the temperature field jump across a material interface in the same element 10-11-15-14, e.g. $\overline{\boldsymbol{N}}^{(13)}(\boldsymbol{x})$ for $\Gamma^{(13)}$ is

$$\overline{\boldsymbol{N}}^{(13)}(\boldsymbol{x}) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ N_{10}(\boldsymbol{x}) \cdot [\![\Psi^4(\boldsymbol{x})]\!]^{(13)} \\ N_{10}(\boldsymbol{x}) \cdot [\![\Psi^5(\boldsymbol{x})]\!]^{(13)} \\ N_{11}(\boldsymbol{x}) \cdot [\![\Psi^4(\boldsymbol{x})]\!]^{(13)} \\ N_{11}(\boldsymbol{x}) \cdot [\![\Psi^5(\boldsymbol{x})]\!]^{(13)} \\ N_{15}(\boldsymbol{x}) \cdot [\![\Psi^2(\boldsymbol{x})]\!]^{(13)} \\ N_{15}(\boldsymbol{x}) \cdot [\![\Psi^3(\boldsymbol{x})]\!]^{(13)} \\ N_{14}(\boldsymbol{x}) \cdot [\![\Psi^2(\boldsymbol{x})]\!]^{(13)} \\ N_{14}(\boldsymbol{x}) \cdot [\![\Psi^3(\boldsymbol{x})]\!]^{(13)} \end{bmatrix}^T = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ N_{10}(\boldsymbol{x}) \cdot (+1) \\ N_{10}(\boldsymbol{x}) \cdot (+2) \\ N_{11}(\boldsymbol{x}) \cdot (+1) \\ N_{11}(\boldsymbol{x}) \cdot (+2) \\ N_{15}(\boldsymbol{x}) \cdot (+2) \\ N_{15}(\boldsymbol{x}) \cdot (\pm 0) \\ N_{14}(\boldsymbol{x}) \cdot (+2) \\ N_{14}(\boldsymbol{x}) \cdot (\pm 0) \end{bmatrix}^T$$

where enrichment $\Psi^3(\boldsymbol{x}) = H^{(23)}(\boldsymbol{x})$ does not affect the jump across $\Gamma^{(13)}$, thus entries corresponding to it, as well as to standard DOFs, are equal to 0. Similarly, for the temperature field jump across $\Gamma^{(23)}$ in the same element 10-11-15-14:

$$\overline{\boldsymbol{N}}^{(23)}(\boldsymbol{x}) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ N_{10}(\boldsymbol{x}) \cdot [\![\Psi^4(\boldsymbol{x})]\!]^{(23)} \\ N_{10}(\boldsymbol{x}) \cdot [\![\Psi^5(\boldsymbol{x})]\!]^{(23)} \\ N_{11}(\boldsymbol{x}) \cdot [\![\Psi^4(\boldsymbol{x})]\!]^{(23)} \\ N_{11}(\boldsymbol{x}) \cdot [\![\Psi^5(\boldsymbol{x})]\!]^{(23)} \\ N_{15}(\boldsymbol{x}) \cdot [\![\Psi^2(\boldsymbol{x})]\!]^{(23)} \\ N_{15}(\boldsymbol{x}) \cdot [\![\Psi^3(\boldsymbol{x})]\!]^{(23)} \\ N_{14}(\boldsymbol{x}) \cdot [\![\Psi^2(\boldsymbol{x})]\!]^{(23)} \\ N_{14}(\boldsymbol{x}) \cdot [\![\Psi^3(\boldsymbol{x})]\!]^{(23)} \end{bmatrix}^T = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ N_{10}(\boldsymbol{x}) \cdot (-1) \\ N_{10}(\boldsymbol{x}) \cdot (+1) \\ N_{11}(\boldsymbol{x}) \cdot (-1) \\ N_{11}(\boldsymbol{x}) \cdot (+1) \\ N_{15}(\boldsymbol{x}) \cdot (\pm 0) \\ N_{15}(\boldsymbol{x}) \cdot (+2) \\ N_{14}(\boldsymbol{x}) \cdot (\pm 0) \\ N_{14}(\boldsymbol{x}) \cdot (+2) \end{bmatrix}^T$$

where enrichment $\Psi^2(\boldsymbol{x}) = H^{(13)}(\boldsymbol{x})$ does not affect the jump across $\Gamma^{(23)}$, thus entries corresponding to it, as well as to standard DOFs, are equal to 0. Furthermore, the $\boldsymbol{B}^{std}(\boldsymbol{x})$ and $\boldsymbol{B}^{enr}(\boldsymbol{x})$ matrices of element 10-11-15-14 are

$$\boldsymbol{B}^{std}(\boldsymbol{x}) = \begin{bmatrix} \frac{\partial N_1(\boldsymbol{x})}{\partial x} & \frac{\partial N_2(\boldsymbol{x})}{\partial x} & \frac{\partial N_3(\boldsymbol{x})}{\partial x} & \frac{\partial N_3(\boldsymbol{x})}{\partial x} \\ \frac{\partial N_1(\boldsymbol{x})}{\partial y} & \frac{\partial N_2(\boldsymbol{x})}{\partial y} & \frac{\partial N_3(\boldsymbol{x})}{\partial y} & \frac{\partial N_3(\boldsymbol{x})}{\partial y} \end{bmatrix}$$

$$\boldsymbol{B}^{enr}(\boldsymbol{x}) = \begin{bmatrix} \boldsymbol{B}_{10}^{enr}(\boldsymbol{x}) & \boldsymbol{B}_{11}^{enr}(\boldsymbol{x}) & \boldsymbol{B}_{15}^{enr}(\boldsymbol{x}) & \boldsymbol{B}_{14}^{enr}(\boldsymbol{x}) \end{bmatrix}$$

where the submatrices of $\boldsymbol{B}_k^{enr}(\boldsymbol{x})$ that correspond to each node are

$$\begin{aligned}
\boldsymbol{B}_{10}^{enr}(\boldsymbol{x}) &= \begin{bmatrix} \frac{\partial N_1(\boldsymbol{x})}{\partial x}\cdot\left(\Psi^4(\boldsymbol{x})-\Psi^4(\boldsymbol{x}_{10})\right) & \frac{\partial N_1(\boldsymbol{x})}{\partial x}\cdot\left(\Psi^5(\boldsymbol{x})-\Psi^5(\boldsymbol{x}_{10})\right) \\ \frac{\partial N_1(\boldsymbol{x})}{\partial y}\cdot\left(\Psi^4(\boldsymbol{x})-\Psi^4(\boldsymbol{x}_{10})\right) & \frac{\partial N_1(\boldsymbol{x})}{\partial y}\cdot\left(\Psi^5(\boldsymbol{x})-\Psi^5(\boldsymbol{x}_{10})\right) \end{bmatrix} \\
&= \begin{bmatrix} \frac{\partial N_1(\boldsymbol{x})}{\partial x}\cdot\left(\Psi^4(\boldsymbol{x})+1\right) & \frac{\partial N_1(\boldsymbol{x})}{\partial x}\cdot\left(\Psi^5(\boldsymbol{x})+1\right) \\ \frac{\partial N_1(\boldsymbol{x})}{\partial y}\cdot\left(\Psi^4(\boldsymbol{x})+1\right) & \frac{\partial N_1(\boldsymbol{x})}{\partial y}\cdot\left(\Psi^5(\boldsymbol{x})+1\right) \end{bmatrix}
\end{aligned}$$

$$\begin{aligned}
\boldsymbol{B}_{11}^{enr}(\boldsymbol{x}) &= \begin{bmatrix} \frac{\partial N_2(\boldsymbol{x})}{\partial x}\cdot\left(\Psi^4(\boldsymbol{x})-\Psi^4(\boldsymbol{x}_{11})\right) & \frac{\partial N_2(\boldsymbol{x})}{\partial x}\cdot\left(\Psi^5(\boldsymbol{x})-\Psi^5(\boldsymbol{x}_{11})\right) \\ \frac{\partial N_2(\boldsymbol{x})}{\partial y}\cdot\left(\Psi^4(\boldsymbol{x})-\Psi^4(\boldsymbol{x}_{11})\right) & \frac{\partial N_2(\boldsymbol{x})}{\partial y}\cdot\left(\Psi^5(\boldsymbol{x})-\Psi^5(\boldsymbol{x}_{11})\right) \end{bmatrix} \\
&= \begin{bmatrix} \frac{\partial N_2(\boldsymbol{x})}{\partial x}\cdot\left(\Psi^4(\boldsymbol{x})-1\right) & \frac{\partial N_2(\boldsymbol{x})}{\partial x}\cdot\Psi^5(\boldsymbol{x}) \\ \frac{\partial N_2(\boldsymbol{x})}{\partial y}\cdot\left(\Psi^4(\boldsymbol{x})-1\right) & \frac{\partial N_2(\boldsymbol{x})}{\partial y}\cdot\Psi^5(\boldsymbol{x}) \end{bmatrix}
\end{aligned}$$

$$\begin{aligned}
\boldsymbol{B}_{15}^{enr}(\boldsymbol{x}) &= \begin{bmatrix} \frac{\partial N_3(\boldsymbol{x})}{\partial x}\cdot\left(\Psi^2(\boldsymbol{x})-\Psi^2(\boldsymbol{x}_{15})\right) & \frac{\partial N_3(\boldsymbol{x})}{\partial x}\cdot\left(\Psi^3(\boldsymbol{x})-\Psi^3(\boldsymbol{x}_{15})\right) \\ \frac{\partial N_3(\boldsymbol{x})}{\partial y}\cdot\left(\Psi^2(\boldsymbol{x})-\Psi^2(\boldsymbol{x}_{15})\right) & \frac{\partial N_3(\boldsymbol{x})}{\partial y}\cdot\left(\Psi^3(\boldsymbol{x})-\Psi^3(\boldsymbol{x}_{15})\right) \end{bmatrix} \\
&= \begin{bmatrix} \frac{\partial N_3(\boldsymbol{x})}{\partial x}\cdot\left(\Psi^2(\boldsymbol{x})-1\right) & \frac{\partial N_3(\boldsymbol{x})}{\partial x}\cdot\left(\Psi^3(\boldsymbol{x})-1\right) \\ \frac{\partial N_3(\boldsymbol{x})}{\partial y}\cdot\left(\Psi^2(\boldsymbol{x})-1\right) & \frac{\partial N_3(\boldsymbol{x})}{\partial y}\cdot\left(\Psi^3(\boldsymbol{x})-1\right) \end{bmatrix}
\end{aligned}$$

$$\begin{aligned}
\boldsymbol{B}_{14}^{enr}(\boldsymbol{x}) &= \begin{bmatrix} \frac{\partial N_4(\boldsymbol{x})}{\partial x}\cdot\left(\Psi^2(\boldsymbol{x})-\Psi^2(\boldsymbol{x}_{14})\right) & \frac{\partial N_4(\boldsymbol{x})}{\partial x}\cdot\left(\Psi^3(\boldsymbol{x})-\Psi^3(\boldsymbol{x}_{14})\right) \\ \frac{\partial N_4(\boldsymbol{x})}{\partial y}\cdot\left(\Psi^2(\boldsymbol{x})-\Psi^2(\boldsymbol{x}_{14})\right) & \frac{\partial N_4(\boldsymbol{x})}{\partial y}\cdot\left(\Psi^3(\boldsymbol{x})-\Psi^3(\boldsymbol{x}_{14})\right) \end{bmatrix} \\
&= \begin{bmatrix} \frac{\partial N_4(\boldsymbol{x})}{\partial x}\cdot\left(\Psi^2(\boldsymbol{x})-1\right) & \frac{\partial N_4(\boldsymbol{x})}{\partial x}\cdot\left(\Psi^3(\boldsymbol{x})-1\right) \\ \frac{\partial N_4(\boldsymbol{x})}{\partial y}\cdot\left(\Psi^2(\boldsymbol{x})-1\right) & \frac{\partial N_4(\boldsymbol{x})}{\partial y}\cdot\left(\Psi^3(\boldsymbol{x})-1\right) \end{bmatrix}
\end{aligned}$$

where $N_1(\boldsymbol{x})$, $N_2(\boldsymbol{x})$, $N_3(\boldsymbol{x})$, $N_4(\boldsymbol{x})$ are the shape functions corresponding to the nodes of each quadrilateral element in counter-clockwise order and the enrichment functions evaluated at nodes are

$$\begin{aligned}
\Psi^2(\boldsymbol{x}_{14}) &= H^{(13)}(\boldsymbol{x}_{14}) = +1 \\
\Psi^2(\boldsymbol{x}_{15}) &= H^{(13)}(\boldsymbol{x}_{15}) = +1 \\
\Psi^3(\boldsymbol{x}_{14}) &= H^{(23)}(\boldsymbol{x}_{14}) = +1 \\
\Psi^3(\boldsymbol{x}_{15}) &= H^{(23)}(\boldsymbol{x}_{15}) = +1 \\
\Psi^4(\boldsymbol{x}_{10}) &= J^{(12)}(\boldsymbol{x}_{10}) = -1 \\
\Psi^4(\boldsymbol{x}_{11}) &= J^{(12)}(\boldsymbol{x}_{11}) = +1 \\
\Psi^5(\boldsymbol{x}_{10}) &= J^{(13)}(\boldsymbol{x}_{10}) = -1 \\
\Psi^5(\boldsymbol{x}_{11}) &= J^{(13)}(\boldsymbol{x}_{11}) = 0
\end{aligned}$$

Lastly, in order to calculate the element-level interface conductivity matrix of equation (2.111), integration over the part of each interface that lies inside the element, is needed. E.g. for the element with nodes 10-11-15-14, two contour integrals are needed, over $\Gamma_e^{(13)}$ and $\Gamma_e^{(23)}$, as illustrated in figure 2.8.

2.2.7 XFEM integration

In traditional FEM, the integrands that appear in the matrix-vector weak form are created by multiplying and dividing the polynomial shape functions and their derivatives and thus they are polynomials themselves. These integrals can be calculated by using numerical integration. Specifically, Gaussian-Legendre quadrature is used, which guarantees exact integration of polynomials with the minimum number of required integration points. Since it is also naturally coupled with the isoparametric element formulation, it has been established as the de facto integration rule in FEM.

However, XFEM uses discontinuous enriched basis functions, which means that the integrands that appear in Eqs. (2.110-2.112) are not polynomials in general. Using Gauss-Legendre quadrature introduces a substantial loss in accuracy, even if the number of integration points is increased. As a result, alternative integration rules, that are consistent with the enrichment functions and the geometry of discontinuities, must be developed. Specifically for the present formulation, the Heaviside and junction enrichment functions are piece-wise constant, therefore the basis functions, their derivatives and generally all matrices and vectors in the integrals of interest are piece-wise polynomials. Observing figure 2.7, the following cases are possible:

- *Standard elements* that have no enriched nodes. In this case, regular Gauss-Legendre quadrature can be used, since standard elements have no enriched nodes and their basis functions are polynomials.
- *Blending elements* that are neither intersected nor contain junctions, but have some enriched nodes. As explained in section 2.2.5.4, the enriched part of the temperature field approximation defined in the present XFEM formulation, vanishes outside elements that are intersected. Consequently, all discontinuous functions vanish in blending elements and Gauss-Legendre quadrature can be used for the remaining polynomials, similarly to standard elements.
- *Enriched elements* that are intersected by one or more material interfaces. Some of these may also contain junctions. In this case, which will be elaborated in this section, piece-wise polynomial functions need to be integrated over regions with different material properties, specifically different conductivity tensors.

2.2.7.1 Bulk integrals

Integrating over intersected elements is achieved by partitioning an intersected element into multiple subcells, namely sub-triangles in 2D or sub-tetrahedra in 3D, which conform to the material interface, as depicted in Figs. 2.9 and 2.10, respectively. Each of these subcells belongs to only one material phase, thus the conductivity tesor is constant inside the subcell. Note that although a conforming submesh is indeed used inside each intersected element, the global finite element mesh continues not to conform to the material interfaces, thus retaining all advantages of XFEM discussed in section 2.2.2. Although any element type can be used in the present XFEM formulation, this dissertation focuses on isoparametric elements, which are reviewed in Appendix A. Using this approach, all shape functions N_k are defined as functions $N_k(\boldsymbol{\xi})$ of the coordinates ξ, η, ζ of the natural system of the element. The element-level *bulk* conductivity matrix of equation (2.110) is calculated by integrating in the natural system

$$\begin{aligned} \boldsymbol{K}_e &= \int_{\widetilde{\Omega}_e} \boldsymbol{B}^T(\boldsymbol{\xi}) \cdot \boldsymbol{k}(\boldsymbol{\xi}) \cdot \boldsymbol{B}(\boldsymbol{\xi})\ t \det\left(\boldsymbol{J}_{NG}(\boldsymbol{\xi})\right) d\xi d\eta \quad \text{in 2D} \\ \boldsymbol{K}_e &= \int_{\widetilde{\Omega}_e} \boldsymbol{B}^T(\boldsymbol{\xi}) \cdot \boldsymbol{k}(\boldsymbol{\xi}) \cdot \boldsymbol{B}(\boldsymbol{\xi}) \det\left(\boldsymbol{J}_{NG}(\boldsymbol{\xi})\right) d\xi d\eta d\zeta \quad \text{in 3D} \end{aligned} \tag{2.116}$$

where $\widetilde{\Omega}_e$ is the surface (in 2D problems) or space (in 3D problems) occupied by the isoparametric element in its natural coordinate system, t is the thickness of the domain in 2D problems, $\boldsymbol{k}(\boldsymbol{\xi}) = \boldsymbol{k}(\boldsymbol{x}(\boldsymbol{\xi}))$ is the conductivity tensor at the point $\boldsymbol{x} = \boldsymbol{x}(\boldsymbol{\xi})$, $\det(\cdot)$ is the matrix determinant operator and $\boldsymbol{J}_{NG}(\boldsymbol{\xi})$ is the Jacobian matrix of the isoparametric mapping, as defined in Eqs. (A.2, A.3). The matrix $\boldsymbol{B}(\boldsymbol{\xi}) = \boldsymbol{B}(\boldsymbol{x}(\boldsymbol{\xi}))$ contains the derivatives of basis functions with respect to the global coordinates x, y, z, expressed as functions of the natural coordinates ξ, η, ζ:

$$\boldsymbol{B}(\boldsymbol{x}(\boldsymbol{\xi})) = \boldsymbol{B}(\boldsymbol{\xi}) = \left(\boldsymbol{J}_{NG}(\boldsymbol{\xi})\right)^{-1} \cdot \widetilde{\boldsymbol{B}}(\boldsymbol{\xi}) \tag{2.117}$$

where $\widetilde{\boldsymbol{B}}(\boldsymbol{\xi})$ contains the derivatives of basis functions with respect to the natural coordinates ξ, η, ζ, expressed as functions of the natural coordinates:

$$\widetilde{\boldsymbol{B}}(\boldsymbol{\xi}) = \left[\begin{array}{cc} \widetilde{\boldsymbol{B}}^{std}(\boldsymbol{\xi}) & \widetilde{\boldsymbol{B}}^{enr}(\boldsymbol{\xi}) \end{array} \right]$$

$$\widetilde{\boldsymbol{B}}^{std}(\boldsymbol{\xi}) = \left[\begin{array}{ccc} & \dfrac{\partial N_k(\boldsymbol{\xi})}{\partial \xi} & \\ \cdots & \dfrac{\partial N_k(\boldsymbol{\xi})}{\partial \eta} & \cdots \\ & \dfrac{\partial N_k(\boldsymbol{\xi})}{\partial \zeta} & \end{array} \right] \tag{2.118}$$

$$\widetilde{\boldsymbol{B}}^{enr}(\boldsymbol{\xi}) = \left[\begin{array}{ccc} & \dfrac{\partial N_k(\boldsymbol{\xi})}{\partial \xi}\left(\Psi^a(\boldsymbol{x}(\boldsymbol{\xi})) - \Psi^a(\boldsymbol{x}_k)\right) & \\ \cdots & \dfrac{\partial N_k(\boldsymbol{\xi})}{\partial \eta}\left(\Psi^a(\boldsymbol{x}(\boldsymbol{\xi})) - \Psi^a(\boldsymbol{x}_k)\right) & \cdots \\ & \dfrac{\partial N_k(\boldsymbol{\xi})}{\partial \zeta}\left(\Psi^a(\boldsymbol{x}(\boldsymbol{\xi})) - \Psi^a(\boldsymbol{x}_k)\right) & \end{array} \right]$$

2.2.7.1.1 2D problems

The integral of 2.116 needs to be calculated in each sub-triangle, where the integrand functions are continuous. As figure 2.9 illustrates, an auxiliary coordinate system (r, s) is defined for each sub-triangle, which is mapped to the natural coordinate system (ξ, η), using Lagrange polynomial shape functions

$$\boldsymbol{\xi} = \boldsymbol{\xi}(\boldsymbol{r}) = \sum_{k=1}^{3} N_k(\boldsymbol{r})\, \boldsymbol{\xi}_k \tag{2.119}$$

where $k = 1, \cdots 3$ are the nodes of the sub-triangle, $\boldsymbol{\xi}_k$ are their natural coordinates and $N_k(\boldsymbol{r})$ are the shape functions corresponding to them, which are defined in equation (A.8). The Jacobian matrix of the mapping from auxiliary to natural system $\boldsymbol{J}_{AN}(\boldsymbol{r})$ is

$$\boldsymbol{J}_{AN}(\boldsymbol{r}) = \begin{bmatrix} \dfrac{\partial \xi}{\partial r} & \dfrac{\partial \eta}{\partial r} \\ \dfrac{\partial \xi}{\partial s} & \dfrac{\partial \eta}{\partial s} \end{bmatrix} = \begin{bmatrix} \xi_2 - \xi_1 & \eta_2 - \eta_1 \\ \xi_3 - \xi_1 & \eta_3 - \eta_1 \end{bmatrix} \tag{2.120}$$

Similarly to equation (A.11), the determinant of this Jacobian matrix is constant and related to the area of the sub-triangle in the natural coordinate system:

$$\det(\boldsymbol{J}_{AN}) = \|(\boldsymbol{\xi}_2 - \boldsymbol{\xi}_1) \times (\boldsymbol{\xi}_3 - \boldsymbol{\xi}_1)\| = 2\, A_{tri} \tag{2.121}$$

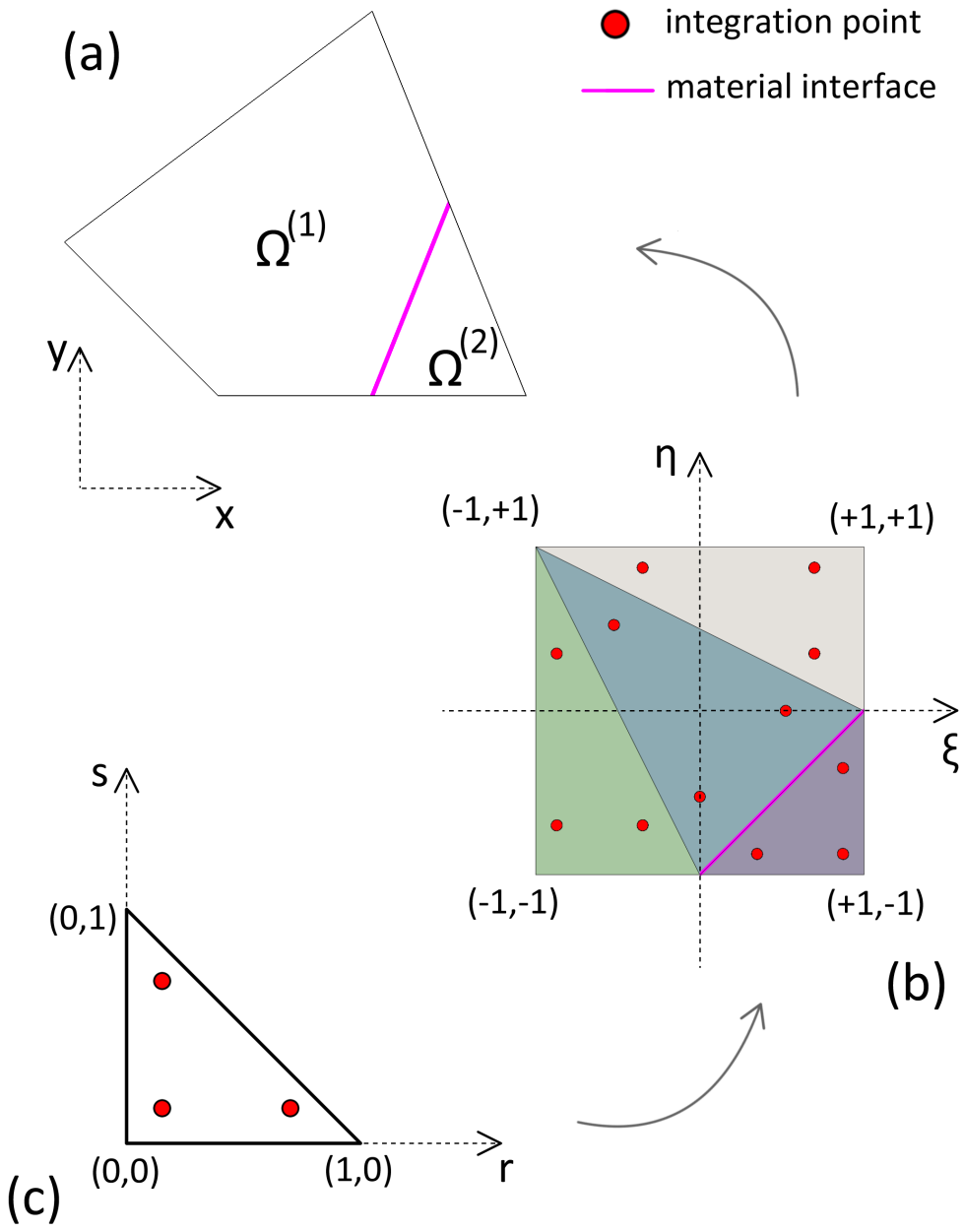


Figure 2.9: Integration in an intersected 2D element. a) Global coordinate system, b) Natural coordinate system of the element, c) Auxiliary coordinate system for each sub-triangle.

The integral of 2.116 can now be converted to the auxiliary system:

$$\boldsymbol{K}_e = \int_0^{1-r} \int_0^{1} \boldsymbol{B}^T\left(\boldsymbol{\xi}\left(\boldsymbol{r}\right)\right) \cdot \boldsymbol{k}\left(\boldsymbol{\xi}\left(\boldsymbol{r}\right)\right) \cdot \boldsymbol{B}\left(\boldsymbol{\xi}\left(\boldsymbol{r}\right)\right)\ t\ \det\left(\boldsymbol{J}_{NG}\left(\boldsymbol{\xi}\left(\boldsymbol{r}\right)\right)\right) \det\left(\boldsymbol{J}_{AN}\right) drds \tag{2.122}$$

which can be calculated using numerical integration:

$$\boldsymbol{K}_e = \sum_{p=1}^{n_{GP}} \boldsymbol{B}^T\left(\boldsymbol{\xi}\left(\boldsymbol{r}_p\right)\right) \cdot \boldsymbol{k}\left(\boldsymbol{\xi}\left(\boldsymbol{r}_p\right)\right) \cdot \boldsymbol{B}\left(\boldsymbol{\xi}\left(\boldsymbol{r}_p\right)\right)\ t\ \det\left(\boldsymbol{J}_{NG}\left(\boldsymbol{\xi}\left(\boldsymbol{r}_p\right)\right)\right) \det\left(\boldsymbol{J}_{AN}\right) w_p \tag{2.123}$$

where $p = 1, \cdots n_{GP}$ is the set of integration points from all sub-triangles, $\boldsymbol{r}_p$ are the coordinates of each integration point in the auxiliary system, w_p is its corresponding weight coefficient and $\boldsymbol{\xi}\left(\boldsymbol{r}_p\right)$ is calculated using equation (2.119). In each sub-triangle, a subset of the integration points is generated using Gauss-Legendre quadrature, therefore table A.5 is used for $\boldsymbol{r}_p$ and w_p.

2.2.7.1.2 3D problems

Similarly to the 2D case, an auxiliary coordinate system (r, s, t) is defined for each sub-tetrahedron, as illustrated in figure 2.10. Mapping from the auxiliary to the natural coordinate system (ξ, η, ζ) is done using Lagrange polynomial shape functions:

$$\boldsymbol{\xi} = \boldsymbol{\xi}\left(\boldsymbol{r}\right) = \sum_{k=1}^{4} N_k\left(\boldsymbol{r}\right) \boldsymbol{\xi}_k \tag{2.124}$$

where $k = 1, \cdots 4$ are the nodes of the sub-tetrahedron, $\boldsymbol{\xi}_k$ are their natural coordinates and $N_k\left(\boldsymbol{r}\right)$ are the shape functions corresponding to them, which are defined in equation (A.13). The Jacobian matrix of the mapping from auxiliary to natural system $\boldsymbol{J}_{AN}\left(\boldsymbol{r}\right)$ is

$$\boldsymbol{J}_{AN}\left(\boldsymbol{r}\right) = \begin{bmatrix} \dfrac{\partial\xi}{\partial r} & \dfrac{\partial\eta}{\partial r} & \dfrac{\partial\zeta}{\partial r} \\ \dfrac{\partial\xi}{\partial s} & \dfrac{\partial\eta}{\partial s} & \dfrac{\partial\zeta}{\partial s} \\ \dfrac{\partial\xi}{\partial t} & \dfrac{\partial\eta}{\partial t} & \dfrac{\partial\zeta}{\partial t} \end{bmatrix} = \begin{bmatrix} \xi_2 - \xi_1 & \eta_2 - \eta_1 & \zeta_2 - \zeta_1 \\ \xi_3 - \xi_1 & \eta_3 - \eta_1 & \zeta_3 - \zeta_1 \\ \xi_4 - \xi_1 & \eta_4 - \eta_1 & \zeta_4 - \zeta_1 \end{bmatrix} \tag{2.125}$$

Similarly to equation (A.16), the determinant of this Jacobian matrix is constant and related to the volume of the sub-tetrahedron in the natural coordinate system:

$$\det\left(\boldsymbol{J}_{AN}\right) = \left|\left(\boldsymbol{\xi}_4 - \boldsymbol{\xi}_1\right) \cdot \left(\left(\boldsymbol{\xi}_2 - \boldsymbol{\xi}_1\right) \times \left(\boldsymbol{\xi}_3 - \boldsymbol{\xi}_1\right)\right)\right| = 6\ V_{tet} \tag{2.126}$$

The integral of 2.116 can now be converted to the auxiliary system:

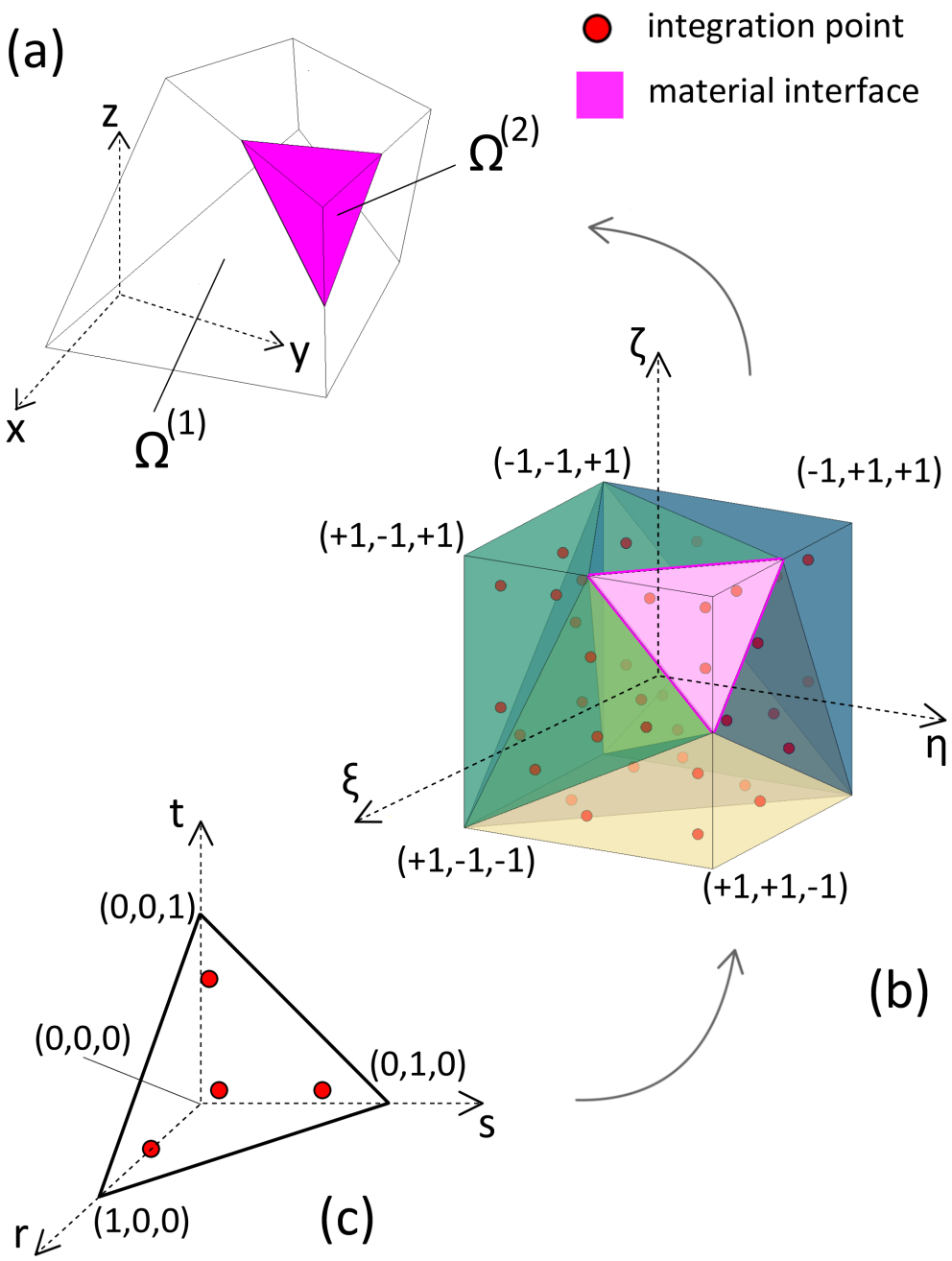


Figure 2.10: Integration in an intersected 3D element. a) Global coordinate system, b) Natural coordinate system of the element, c) Auxiliary coordinate system for each sub-tetrahedron.

$$\boldsymbol{K}_e = \int_0^{1-r-s} \int_0^{1-r} \int_0^{1} \boldsymbol{B}^T(\boldsymbol{\xi}(\boldsymbol{r})) \cdot \boldsymbol{k}(\boldsymbol{\xi}(\boldsymbol{r})) \cdot \boldsymbol{B}(\boldsymbol{\xi}(\boldsymbol{r})) \det(\boldsymbol{J}_{NG}(\boldsymbol{\xi}(\boldsymbol{r}))) \det(\boldsymbol{J}_{AN}) \, drdsdt \quad (2.127)$$

which can be calculated using numerical integration:

$$\boldsymbol{K}_e = \sum_{p=1}^{n_{GP}} \boldsymbol{B}^T\left(\boldsymbol{\xi}\left(\boldsymbol{r}_p\right)\right) \cdot \boldsymbol{k}\left(\boldsymbol{\xi}\left(\boldsymbol{r}_p\right)\right) \cdot \boldsymbol{B}\left(\boldsymbol{\xi}\left(\boldsymbol{r}_p\right)\right)\ t \det\left(\boldsymbol{J}_{NG}\left(\boldsymbol{\xi}\left(\boldsymbol{r}_p\right)\right)\right) \det\left(\boldsymbol{J}_{AN}\right) w_p \tag{2.128}$$

where $p = 1, \cdots n_{GP}$ is the set of integration points from all sub-tetrahedra, $\boldsymbol{r}_p$ are the coordinates of each integration point in the auxiliary system, w_p is its corresponding weight coefficient and $\boldsymbol{\xi}\left(\boldsymbol{r}_p\right)$ is calculated using equation (2.124). In each sub-tetrahedron, a subset of the integration points is generated using Gauss-Legendre quadrature, therefore table A.7 is used for $\boldsymbol{r}_p$ and w_p.

2.2.7.2 Interface integrals

To obtain the partition of an element into subcells, first the part $\Gamma_e^{(ij)}$ of each material interface $\Gamma^{(ij)}$ that lies inside the element e needs to be determined. In this dissertation, $\Gamma_e^{(ij)}$ is approximated by simpler geometries, such as line segments in 2D or triangles in 3D, as illustrated in Figs. 2.11, 2.12. The geometric operations required to obtain these *interface segments* are elaborated in section 2.3. Apart from the bulk integration over the domain of each element, there is also need to integrate over the surface of $\Gamma_e^{(ij)}$, in order to calculate the element-level interface conductivity matrix of equation (2.111). This integration will be performed by integrating over each interface segment $\Gamma_{e,s}^{(ij)}$, thus equation (2.111) can be written as

$$\overline{\boldsymbol{K}}_e = \sum_{(i,j)\in M^\Gamma} \sum_{\beta=1}^{n_{e,\beta}^{(ij)}} \int_{\Gamma_{e,\beta}^{(ij)}} \frac{1}{\alpha^{(ij)}} \left(\overline{\boldsymbol{N}}^{(ij)}\left(\boldsymbol{x}\right)\right)^T \cdot \overline{\boldsymbol{N}}^{(ij)}\left(\boldsymbol{x}\right) d\Gamma \tag{2.129}$$

where $n_{e,\beta}^{(ij)}$ is the number of interface segments $\Gamma_{e,\beta}^{(ij)}$ (line segments in 2D or triangles in 3D) that comprise the interface $\Gamma_e^{(ij)}$ inside element e. The matrices $\overline{\boldsymbol{N}}^{(ij)}\left(\boldsymbol{x}\right) = \overline{\boldsymbol{N}}^{(ij)}\left(\boldsymbol{x}\left(\boldsymbol{\xi}\right)\right)$ are calculated using the shape functions of the isoparametric element, thus they are functions of the natural coordinates $\boldsymbol{\xi}$:

$$\begin{aligned}
&\overline{\boldsymbol{N}}^{(ij)}\left(\boldsymbol{x}\left(\boldsymbol{\xi}\right)\right) = \overline{\boldsymbol{N}}^{(ij)}\left(\boldsymbol{\xi}\right) = \left[\begin{array}{cc} \overline{\boldsymbol{N}}_{std} & \overline{\boldsymbol{N}}_{enr}^{(ij)}\left(\boldsymbol{\xi}\right) \end{array}\right] \\
&\overline{\boldsymbol{N}}_{std} = \boldsymbol{0} \\
&\overline{\boldsymbol{N}}_{enr}^{(ij)}\left(\boldsymbol{\xi}\right) = \left[\begin{array}{ccc} \cdots & N_k\left(\boldsymbol{\xi}\right)\left[\!\left[\Psi^a\left(\boldsymbol{x}\left(\boldsymbol{\xi}\right)\right)\right]\!\right]^{(ij)} & \cdots \end{array}\right]
\end{aligned} \tag{2.130}$$

where $N_k\left(\boldsymbol{\xi}\right)$ is the shape function corresponding to node k of the isoparametric element. Thus, equation (2.129) is rewritten as

$$\overline{\boldsymbol{K}}_e = \sum_{(i,j)\in M^\Gamma} \sum_{\beta=1}^{n_{e,\beta}^{(ij)}} \int_{\Gamma_{e,\beta}^{(ij)}} \frac{1}{\alpha^{(ij)}} \left(\overline{\boldsymbol{N}}^{(ij)}(\boldsymbol{\xi})\right)^T \cdot \overline{\boldsymbol{N}}^{(ij)}(\boldsymbol{\xi})\, d\Gamma \tag{2.131}$$

By grouping all interface segments together the previous equation can be simplified to

$$\overline{\boldsymbol{K}}_e = \sum_{s=1}^{n_{e,\gamma}} \int_{\Gamma_{e,\gamma}} \frac{1}{\alpha^{(\gamma)}} \left(\overline{\boldsymbol{N}}^{(\gamma)}(\boldsymbol{\xi})\right)^T \cdot \overline{\boldsymbol{N}}^{(\gamma)}(\boldsymbol{\xi})\, d\Gamma \tag{2.132}$$

where $n_{e,\gamma}$ is the total number of intersection segments inside element e and each $\Gamma_{e,\gamma}$ corresponds to one interface segment $\Gamma_{e,\beta}^{(ij)}$ of the element. Since γ corresponds to a unique triple (i,j,β), $\overline{\boldsymbol{N}}^{(\gamma)}(\boldsymbol{\xi})$ corresponds to $\overline{\boldsymbol{N}}^{(ij)}(\boldsymbol{\xi})$ and $\alpha^{(\gamma)}$ to $\alpha^{(ij)}$.

2.2.7.2.1 2D problems

Similarly to the bulk integrals, an auxiliary coordinate system is used for the intersection segments. In 2D problems an 1D auxiliary system with a single coordinate (r) is defined for each line segment, as illustrated in figure 2.11. Using the Lagrange polynomial shape functions, mappings are defined from the auxiliary system to the natural (mapping AN) and global (mapping AG) ones:

$$\begin{aligned} \boldsymbol{\xi} = \boldsymbol{\xi}(r) &= \sum_{k=1}^{2} N_k(r)\boldsymbol{\xi}_k \\ \boldsymbol{x} = \boldsymbol{x}(r) &= \sum_{k=1}^{2} N_k(r)\boldsymbol{x}_k \end{aligned} \tag{2.133}$$

where $k = 1, 2$ are the nodes of the interface (line) segment $\Gamma_{e,\gamma}$, $\boldsymbol{x}_k$ and $\boldsymbol{\xi}_k$ are their global and natural coordinates, respectively, and $N_k(\boldsymbol{r})$ are the polynomial shape functions corresponding to them. These Lagrange polynomials and their derivatives are

$$\begin{aligned} N_1(r) &= \frac{1-r}{2} \quad N_2(r) = \frac{1+r}{2} \\ \frac{dN_1(r)}{dr} &= -\frac{1}{2} \quad \frac{dN_2(r)}{dr} = +\frac{1}{2} \end{aligned} \tag{2.134}$$

In order to calculate the integral of any function $f(\boldsymbol{x}) = f(\boldsymbol{x}(r))$, the auxiliary system is used

$$\int_\Gamma f(\boldsymbol{x})\, d\Gamma = \int_{-1}^{+1} f(\boldsymbol{x}(r)) \underbrace{\left\| \frac{d\boldsymbol{x}(r)}{dr} \right\|}_{J_{AG}} dr \tag{2.135}$$

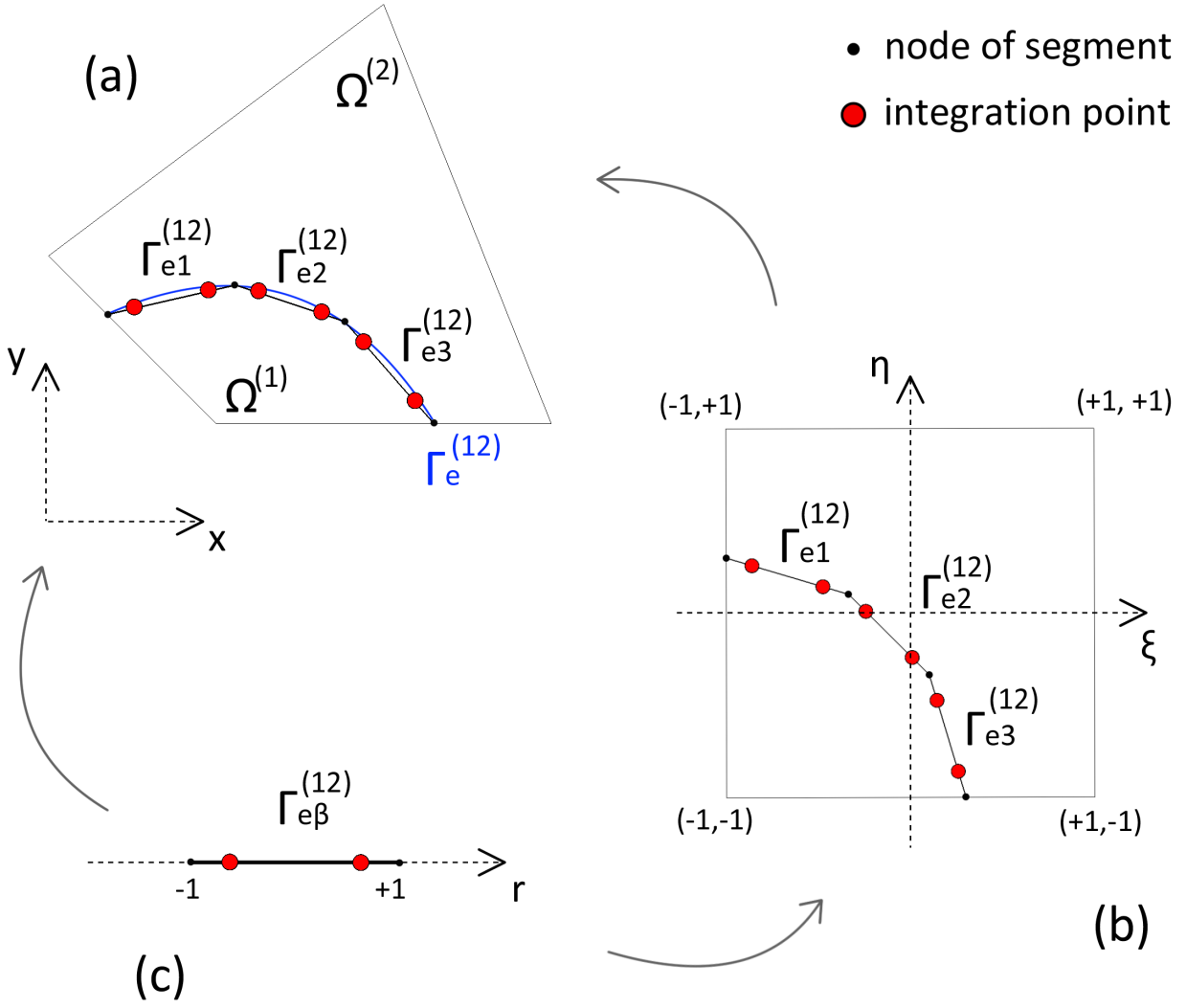


Figure 2.11: Integration over a 2D material interface in an intersected element. a) Global coordinate system, b) Natural coordinate system of the element, c) Auxiliary coordinate system for each line segment of the material interface.

where $J_{AG} = J_{AG}(r) = \left\| \frac{d\boldsymbol{x}(r)}{dr} \right\|$ is the local length distortion factor of the mapping (AG). The total derivative with respect to the auxiliary coordinate r of the global coordinates, which constitute a vector function $\boldsymbol{x}(r)$, is

$$\frac{d\boldsymbol{x}(r)}{dr} = \begin{bmatrix} \dfrac{dx(r)}{dr} \\ \dfrac{dy(r)}{dr} \end{bmatrix} = \begin{bmatrix} \dfrac{dN_1(r)}{dr}x_1 + \dfrac{dN_2(r)}{dr}x_2 \\ \dfrac{dN_1(r)}{dr}y_1 + \dfrac{dN_2(r)}{dr}y_2 \end{bmatrix} = \frac{1}{2}\begin{bmatrix} x_2 - x_1 \\ \\ y_2 - y_1 \end{bmatrix} \tag{2.136}$$

thus the distortion factor is equal to

$$J_{AG} = \left\| \frac{d\boldsymbol{x}(r)}{dr} \right\| = \sqrt{\left(\frac{dx(r)}{dr}\right)^2 + \left(\frac{dy(r)}{dr}\right)^2} = \frac{1}{2}\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} = \frac{L_{seg}}{2} \tag{2.137}$$

where L_{seg} is the length of the intersection segment $(\boldsymbol{x}_1,\ \boldsymbol{x}_2)$ in the global cartesian system. Note that J_{AG} is constant and independent from the auxiliary coordinate r. The integral of equation (2.132) can be converted to

$$\overline{\boldsymbol{K}}_e = \sum_{s=1}^{n_{e,\gamma}} \int_{-1}^{+1} \frac{1}{\alpha^{(\gamma)}} \left(\overline{\boldsymbol{N}}^{(\gamma)}\left(\boldsymbol{\xi}(r)\right)\right)^T \cdot \overline{\boldsymbol{N}}^{(\gamma)}\left(\boldsymbol{\xi}(r)\right)\ t\ J_{AG}\ dr \tag{2.138}$$

where t is the thickness of the domain. By using numerical integration, the above equation becomes

$$\overline{\boldsymbol{K}}_e = \sum_{s=1}^{n_{e,\gamma}} \sum_{p=1}^{n_{GP}^{(\gamma)}} \frac{1}{\alpha^{(\gamma)}} \left(\overline{\boldsymbol{N}}^{(\gamma)}\left(\boldsymbol{\xi}(r_p)\right)\right)^T \cdot \overline{\boldsymbol{N}}^{(\gamma)}\left(\boldsymbol{\xi}(r_p)\right)\ t\ J_{AG}\ w_p \tag{2.139}$$

where $p = 1, \cdots n_{GP}^{(\gamma)}$ are the integration points used for the intersection line segment $\Gamma_{e,\gamma}$, r_p and w_p are the coordinate and weight coefficient for Gauss-Legendre quadrature, as listed in A.3 and $\boldsymbol{\xi}(r_p)$ is calculated using equation (2.133).

2.2.7.2.2 3D problems

On the other hand, in 3D problems a 2D auxiliary coordinate system $(r,\ s)$ is used for each intersection segment, which are now triangles, as illustrated in figure 2.12. Using the Lagrange polynomial shape functions, mappings are defined from the auxiliary system to the natural (mapping AN) and global (mapping AG) ones:

$$\begin{aligned} \boldsymbol{\xi} = \boldsymbol{\xi}(r,s) &= \sum_{k=1}^{3} N_k(r,s)\boldsymbol{\xi}_k \\ \boldsymbol{x} = \boldsymbol{x}(r,s) &= \sum_{k=1}^{3} N_k(r,s)\boldsymbol{x}_k \end{aligned} \tag{2.140}$$

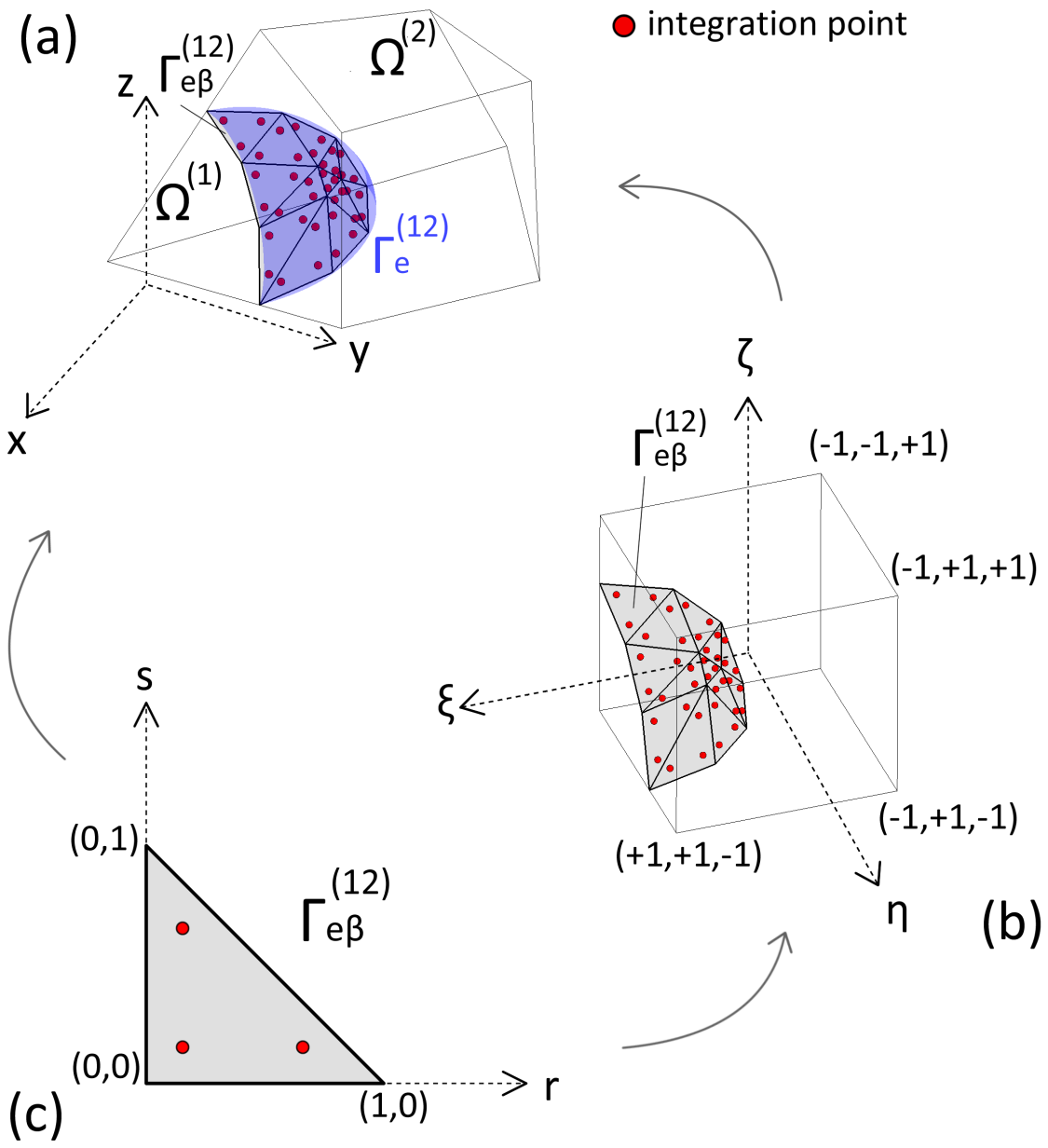


Figure 2.12: Integration over a 3D material interface in an intersected element. a) Global coordinate system, b) Natural coordinate system of the element, c) Auxiliary coordinate system for each triangular segment of the material interface.

where $k = 1, \cdots 3$ are the nodes of the (triangular) interface segment $\Gamma_{e,\gamma}$, $\boldsymbol{x}_k$ and $\boldsymbol{\xi}_k$ are their global and natural coordinates, respectively, and $N_k(\boldsymbol{r})$ are the polynomial shape functions corresponding to them. These Lagrange polynomials and their derivatives are

$$\begin{array}{lll} N_1(r,s) = 1-r-s & N_2(r,s) = r & N_3(r,s) = s \\ \dfrac{\partial N_1(r,s)}{\partial r} = -1 & \dfrac{\partial N_2(r,s)}{\partial r} = +1 & \dfrac{\partial N_3(r,s)}{\partial r} = 0 \\ \dfrac{\partial N_1(r,s)}{\partial s} = -1 & \dfrac{\partial N_2(r,s)}{\partial s} = 0 & \dfrac{\partial N_3(r,s)}{\partial s} = +1 \end{array} \tag{2.141}$$

In order to calculate the integral of any function $f(\boldsymbol{x}) = f(\boldsymbol{x}(r,s))$, the auxiliary system is used

$$\int_\Gamma f(\boldsymbol{x})\, d\Gamma = \int_0^{1-r}\int_0^1 f(\boldsymbol{x}(r,s)) \underbrace{\left\| \frac{\partial \boldsymbol{x}(r,s)}{\partial r} \times \frac{\partial \boldsymbol{x}(r,s)}{\partial s} \right\|}_{J_{AG}} dr ds \tag{2.142}$$

where $J_{AG} = J_{AG}(r,s) = \left\| \dfrac{\partial \boldsymbol{x}(r,s)}{\partial r} \times \dfrac{\partial \boldsymbol{x}(r,s)}{\partial s} \right\|$ is the local area distortion factor of the mapping (AG). The partial derivatives with respect to the auxiliary coordinates (r,s) of the global coordinates, which constitute a vector function $\boldsymbol{x}(r,s)$ are

$$\begin{aligned} \frac{\partial \boldsymbol{x}(r,s)}{\partial r} &= \begin{bmatrix} \dfrac{\partial x(r,s)}{\partial r} \\ \dfrac{\partial y(r,s)}{\partial r} \\ \dfrac{\partial z(r,s)}{\partial r} \end{bmatrix} = \begin{bmatrix} \displaystyle\sum_{k=1}^{3} \frac{\partial N_k(r,s)}{\partial r} x_k \\ \displaystyle\sum_{k=1}^{3} \frac{\partial N_k(r,s)}{\partial r} y_k \\ \displaystyle\sum_{k=1}^{3} \frac{\partial N_k(r,s)}{\partial r} z_k \end{bmatrix} = \begin{bmatrix} x_2 - x_1 \\ y_2 - y_1 \\ z_2 - z_1 \end{bmatrix} = \boldsymbol{x}_2 - \boldsymbol{x}_1 \\ \frac{\partial \boldsymbol{x}(r,s)}{\partial s} &= \begin{bmatrix} \dfrac{\partial x(r,s)}{\partial s} \\ \dfrac{\partial y(r,s)}{\partial s} \\ \dfrac{\partial z(r,s)}{\partial s} \end{bmatrix} = \begin{bmatrix} \displaystyle\sum_{k=1}^{3} \frac{\partial N_k(r,s)}{\partial s} x_k \\ \displaystyle\sum_{k=1}^{3} \frac{\partial N_k(r,s)}{\partial s} y_k \\ \displaystyle\sum_{k=1}^{3} \frac{\partial N_k(r,s)}{\partial s} z_k \end{bmatrix} = \begin{bmatrix} x_3 - x_1 \\ y_3 - y_1 \\ z_3 - z_1 \end{bmatrix} = \boldsymbol{x}_3 - \boldsymbol{x}_1 \end{aligned} \tag{2.143}$$

thus the distortion factor is equal to

$$J_{AG} = \left\| \frac{\partial \boldsymbol{x}(r,s)}{\partial r} \times \frac{\partial \boldsymbol{x}(r,s)}{\partial s} \right\| = \|(\boldsymbol{x}_2 - \boldsymbol{x}_1) \times (\boldsymbol{x}_3 - \boldsymbol{x}_1)\| = 2\ A_{tri} \tag{2.144}$$

where A_{tri} is the area of the triangular intersection segment $(\boldsymbol{x}_1,\ \boldsymbol{x}_2,\ \boldsymbol{x}_3)$ in the global cartesian system. Note that J_{AG} is constant and independent from the auxiliary coordinates r, s. The integral of equation (2.132) can be converted to

$$\overline{\boldsymbol{K}}_e = \sum_{s=1}^{n_{e,\gamma}} \int_0^{1-r} \int_0^1 \frac{1}{\alpha^{(\gamma)}} \left(\overline{\boldsymbol{N}}^{(\gamma)} \left(\boldsymbol{\xi}(r,s) \right) \right)^T \cdot \overline{\boldsymbol{N}}^{(\gamma)} \left(\boldsymbol{\xi}(r,s) \right)\ J_{AG}\ drds \tag{2.145}$$

By using numerical integration, the above equation becomes

$$\overline{\boldsymbol{K}}_e = \sum_{s=1}^{n_{e,\gamma}} \sum_{p=1}^{n_{GP}^{(\gamma)}} \frac{1}{\alpha^{(\gamma)}} \left(\overline{\boldsymbol{N}}^{(\gamma)} \left(\boldsymbol{\xi}(r_p,s_p) \right) \right)^T \cdot \overline{\boldsymbol{N}}^{(\gamma)} \left(\boldsymbol{\xi}(r_p,s_p) \right)\ J_{AG}\ w_p \tag{2.146}$$

where $p = 1, \cdots n_{GP}^{(\gamma)}$ are the integration points used for the triangular intersection segment $\Gamma_{e,\gamma}$, (r_p, s_p) and w_p are the coordinates and weight coefficient for Gauss-Legendre quadrature, as listed in A.5 and $\boldsymbol{\xi}(r_p, s_p)$ is calculated using equation (2.140).

2.3 LSM representation of complex material interface geometries

2.3.1 Level Set Method

The Level Set Method (LSM) is a convenient approach to track complex 2D and 3D geometries, which may be stationary and evolving. Originally developed by Osher and Sethian (1988), it was first used in conjunction with XFEM in Stolarska et al. (2001) for modeling cracks and Sukumar et al. (2001) for modeling inclusions or voids. In LSM, a curve in 2D or surface in 3D is implicitly represented by its zero level set (contour), instead of employing an explicit parametric description. The level set function is defined as the signed distance $\phi(\boldsymbol{x})$ from a point $\boldsymbol{x}$ to the curve or surface, as illustrated in figure 2.13. If the geometry of an inclusion is represented with LSM, the usual convention for the sign is: negative inside the inclusion and positive outside. The level set function is evaluated and stored at the nodes $\boldsymbol{x}_k$ of a mesh. Then, it is interpolated for any other point, using the same polynomial shape functions FEM uses:

$$\phi\left(\boldsymbol{x}\left(\boldsymbol{\xi}\right)\right) = \sum_{k=1}^{n_{nodes}} N_k\left(\boldsymbol{\xi}\right) \phi_k \tag{2.147}$$

where $\phi_k = \phi(\boldsymbol{x}_k)$. LSM complements XFEM nicely, since it uses the finite element mesh to represent the discontinuities and perform any geometric operations needed by XFEM. For example, $\phi(\boldsymbol{x})$ is needed to determine on which material phase a point lies and for the calculation of the enrichment functions. Furthermore, LSM can be used to identify the

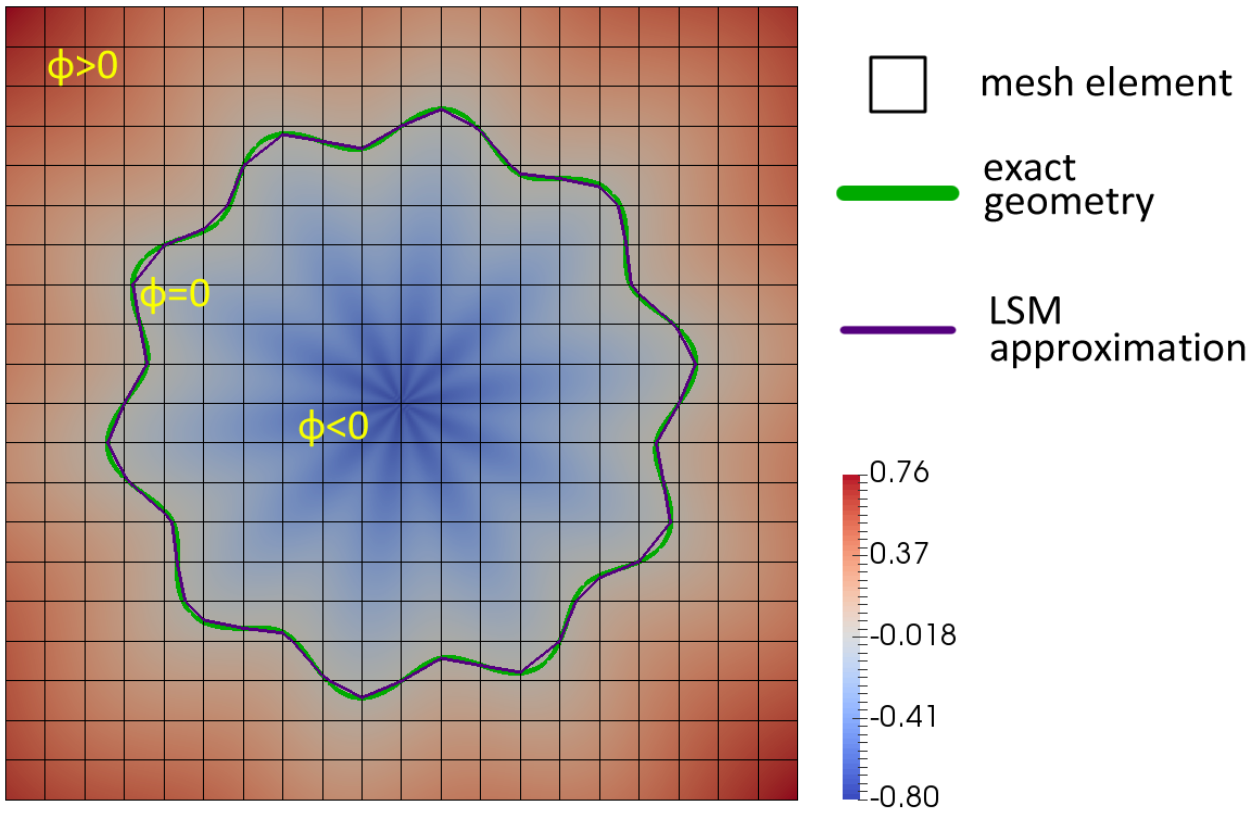


Figure 2.13: Signed distance function evaluated at nodes and interpolated inside elements¿

intersections of the material interfaces with the finite elements, which are needed for the numerical integration presented in section 2.2.7. However, the accuracy of LSM depends on the size of the elements. In order to represent geometries with high curvature or sharp turns, a very fine mesh is required, as shown in figure 2.14. Although the computational cost of LSM is very low, the same mesh is used by XFEM, where refining the mesh rapidly increases the time and memory requirements. The accuracy of XFEM does increase by refining the mesh, but usually convergence occurs at significantly lower mesh densities than the ones needed by LSM.

In this dissertation a *double-mesh LSM* approach has been developed, which uses a coarse mesh for efficient XFEM analysis and a fine mesh for accurate LSM representations. This section will elaborate the coupling of these two meshes and the geometric operations needed by the XFEM formulation presented in this chapter. The resulting method offers the advantage of independently adjusting the densities of the two meshes: the XFEM mesh must be fine enough to achieve the desired accuracy of the analysis, but coarse enough to not redundantly increase the computational effort, while the LSM mesh needs to be significantly refined, in order to capture complex geometries of inclusions, and match the XFEM mesh.

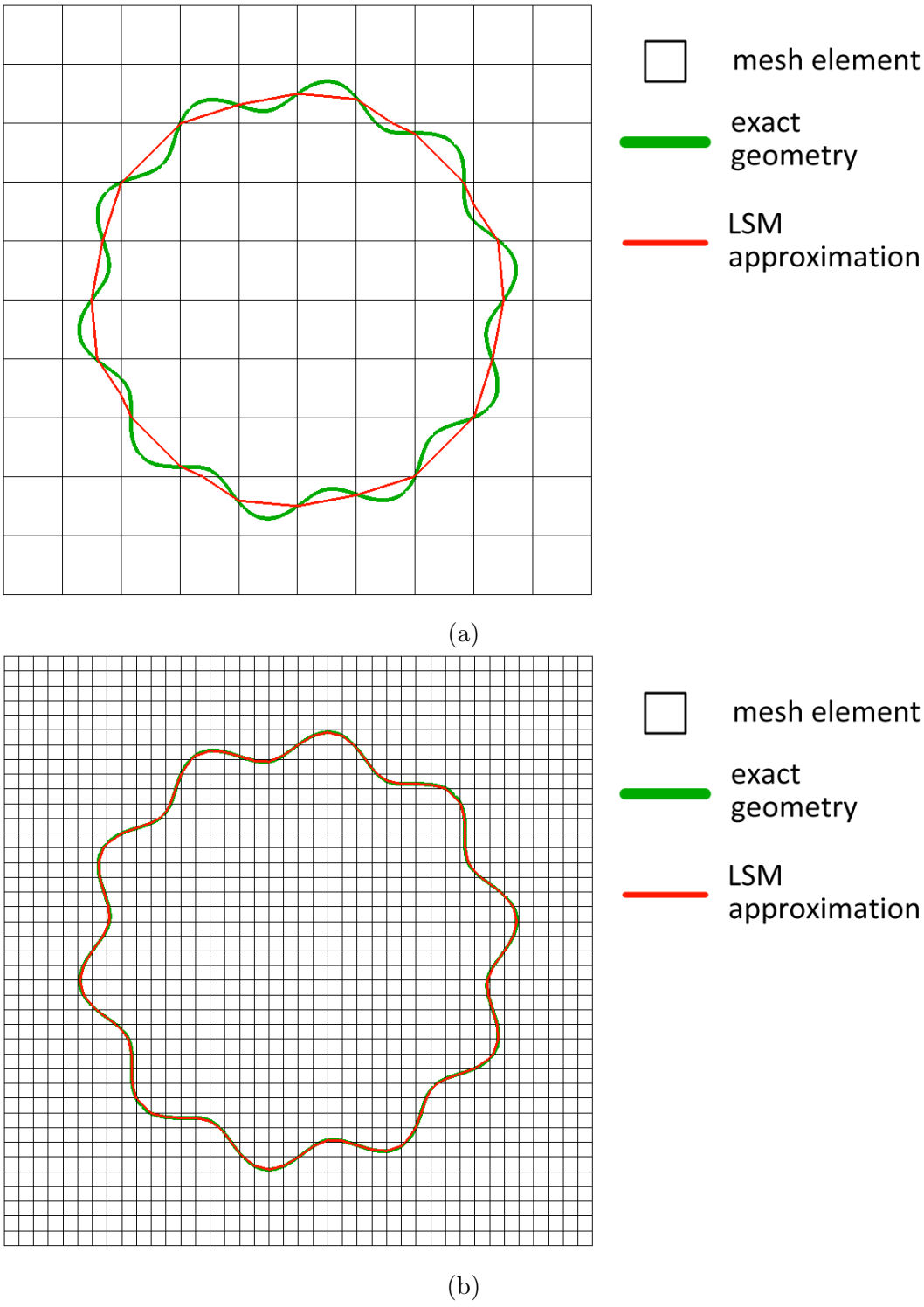


Figure 2.14: Variable accuracy of LSM representation. a) Coarse mesh. b) Fine mesh

2.3.2 The double-mesh LSM approach

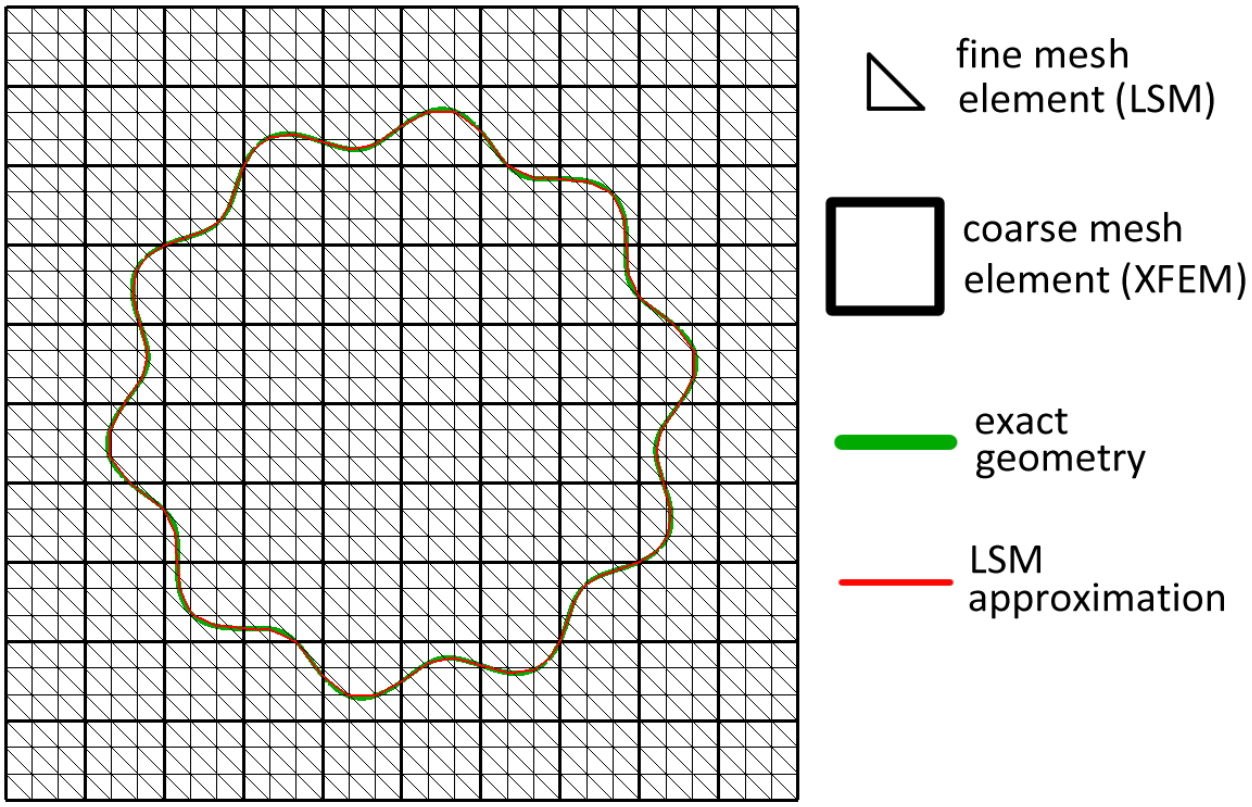


Figure 2.15: Example of coarse XFEM and fine LSM mesh.

Figure 2.15 illustrates the coarse mesh used for XFEM and the fine mesh used for LSM. It can be observed that all nodes of the coarse mesh coincide with some nodes of the fine mesh and that each coarse-mesh element contains a number of fine-mesh elements. The coarse mesh consists of 4-noded quadrilateral elements in 2D or 8-noded hexahedral elements in 3D. Triangular elements in 2D and tetrahedral elements in 3D are selected for the fine mesh, since they lead to simpler and more robust geometric operations, especially when intersections are considered (see section 2.3.3). Furthermore, triangles and tetrahedra produce a more accurate representation of the original curve or surface, since the level set approximation is linear inside each element. For example, 2 line segments inside 2 triangles can approximate a curve better than 1 line segment inside 1 quadrilateral.

Each fine-mesh triangle and tetrahedron has its own coordinate system $\boldsymbol{r}$, while the natural coordinates of the coarse-mesh element are $\boldsymbol{\xi}$, as shown in figure 2.16. In order to map between the two coordinate systems, a third auxiliary system $\widetilde{\boldsymbol{\xi}}$ is defined, along with its corresponding auxiliary elements. These auxiliary elements are generated by dividing a coarse-mesh quadrilateral into $(m_i \times m_j)$ 4-noded quadrilaterals, in 2D problems. In 3D

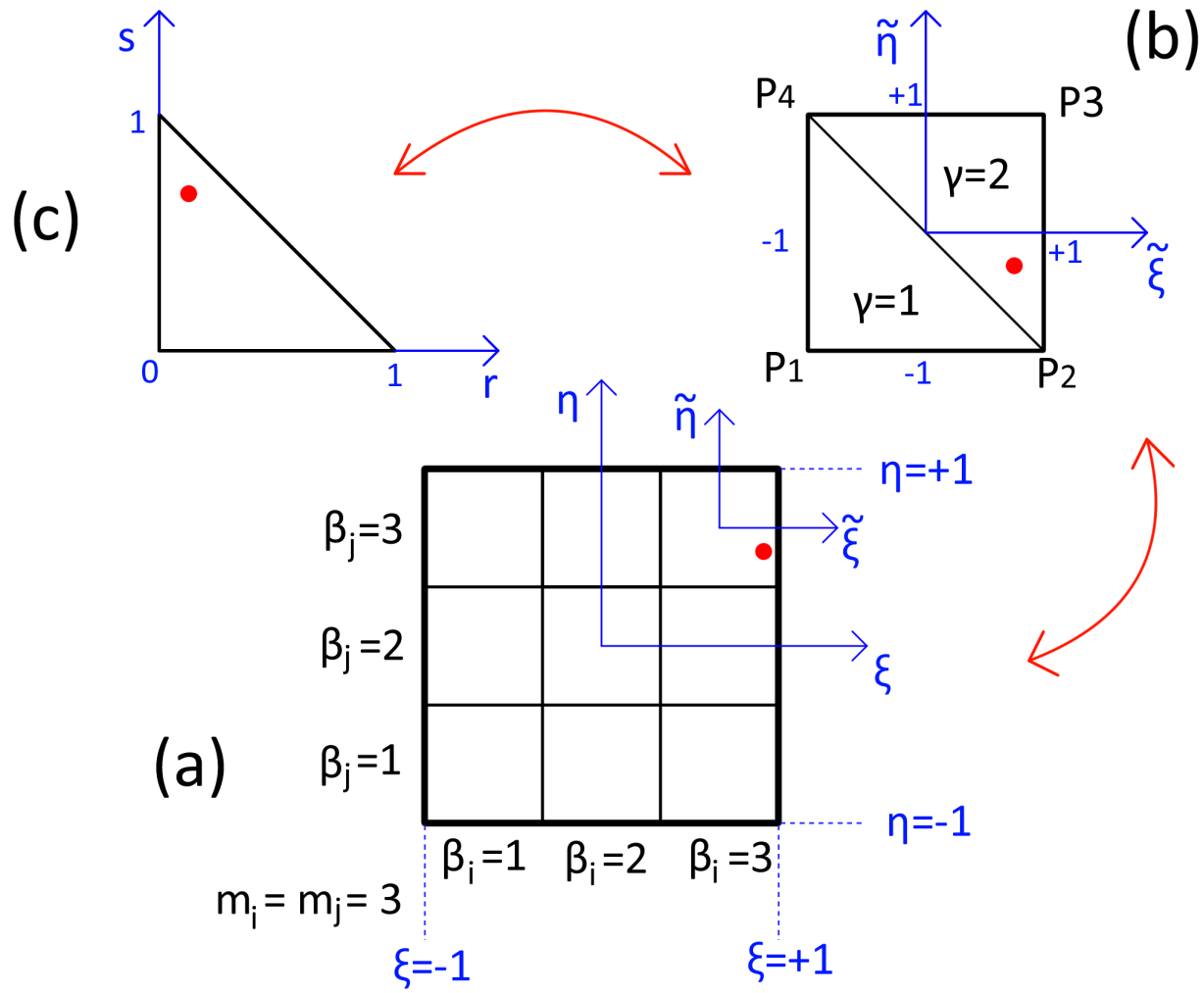


Figure 2.16: Natural coordinate systems of the elements. a) Coarse-mesh element system. b) Auxiliary element system. c) Fine-mesh element system.

problems, a coarse-mesh hexahedron is divided into $(m_i \times m_j \times m_k)$ 8-noded hexahedra. Therefore, the multiplicities m_i, m_j and m_k are integers that express the relative mesh density of the coarse and fine mesh, along the axes x, y and z respectively. Each of these auxiliary elements is assigned an index $\boldsymbol{\beta} = (\beta_i, \beta_j, \beta_k)$, where

$$\begin{aligned} \beta_i &= 1, \cdots m_i \\ \beta_j &= 1, \cdots m_j \\ \beta_k &= 1, \cdots m_k \end{aligned}$$

Essentially, mapping between the coarse and auxiliary element systems involves mapping

from $\boldsymbol{\xi} \longrightarrow (\widetilde{\boldsymbol{\xi}}, \boldsymbol{\beta})$:

$$\begin{aligned} \beta_i &= FLOOR\left(m_i \frac{1+\xi}{2}\right) \qquad \widetilde{\xi} = m_i \xi - 2\beta_i + m_i - 1 \\ \beta_j &= FLOOR\left(m_j \frac{1+\eta}{2}\right) \qquad \widetilde{\eta} = m_j \eta - 2\beta_j + m_j - 1 \\ \beta_k &= FLOOR\left(m_k \frac{1+\zeta}{2}\right) \qquad \widetilde{\zeta} = m_k \zeta - 2\beta_k + m_k - 1 \end{aligned} \tag{2.148}$$

where $FLOOR(\cdot)$ is an operator that keeps the integer part of a real number. Similarly, the mapping from $(\widetilde{\boldsymbol{\xi}}, \boldsymbol{\beta}) \longrightarrow \boldsymbol{\xi}$ is

$$\begin{aligned} \xi &= \frac{\widetilde{\xi} + 2\beta_i + 1}{m_i} - 1 \\ \eta &= \frac{\widetilde{\eta} + 2\beta_i + 1}{m_j} - 1 \\ \zeta &= \frac{\widetilde{\zeta} + 2\beta_i + 1}{m_k} - 1 \end{aligned} \tag{2.149}$$

Moreover, each auxiliary quadrilateral is divided into 2 triangular fine-mesh elements, as shown in figure 2.16. In order to map a point $\widetilde{\boldsymbol{\xi}}$ from the auxiliary system to the natural system of the fine-mesh elements, first we must identify which triangle contains the point $\widetilde{\boldsymbol{\xi}}$. Therefore and integer index $\gamma = 1$ or 2 is assigned to each of the 2 triangles. Assuming that the nodes of the auxiliary quadrilateral element are given in the specific order described in table A.1, table 2.1 lists the nodes of each triangle, as well as the region (in the auxiliary space) occupied by it. Mapping between the auxiliary and fine-mesh coordinate system is described in table 2.2.

Triangle	Nodes of quad4 element	Region
$\gamma = 1$	P_1, P_2, P_4	$\widetilde{\eta} \leq -\widetilde{\xi}$
$\gamma = 2$	P_3, P_4, P_2	$\widetilde{\eta} > -\widetilde{\xi}$

Table 2.1: Fine mesh triangles.

Triangle	$\widetilde{\boldsymbol{\xi}} \longrightarrow \boldsymbol{r}$		$\boldsymbol{r} \longrightarrow \widetilde{\boldsymbol{\xi}}$	
$\gamma = 1$	$r = \dfrac{1+\widetilde{\xi}}{2}$	$s = \dfrac{1+\widetilde{\eta}}{2}$	$\widetilde{\xi} = 2r - 1$	$\widetilde{\eta} = 2s - 1$
$\gamma = 2$	$r = \dfrac{1-\widetilde{\xi}}{2}$	$s = \dfrac{1-\widetilde{\eta}}{2}$	$\widetilde{\xi} = 1 - 2r$	$\widetilde{\eta} = 1 - 2s$

Table 2.2: Conversion between fine-mesh element system and auxiliary element system (2D case).

The 3D case is similar. Each auxiliary hexahedron is divided into 6 tetrahedral fine-mesh elements, which are assigned integer indices $\gamma = 1, \cdots 6$. Assuming that the nodes of the auxiliary hexahedron element are given in the specific order described in table A.2, table 2.3 lists the nodes of each tetrahedron, as well as the region (in the auxiliary space) occupied by it. Mapping between the auxiliary and fine-mesh coordinate system is described in table 2.4.

Triangle	Nodes of hexa8 element	Region
$\gamma = 1$	P_1, P_2, P_3, P_5	$\widetilde{\eta} \leq \widetilde{\xi}, \quad \widetilde{\eta} \leq -\widetilde{\xi}, \quad \widetilde{\eta} \leq -\widetilde{\zeta}$
$\gamma = 2$	P_1, P_3, P_4, P_5	$\widetilde{\eta} > \widetilde{\xi}, \quad \widetilde{\eta} \leq -\widetilde{\xi}, \quad \widetilde{\eta} \leq -\widetilde{\zeta}$
$\gamma = 3$	P_2, P_3, P_5, P_6	$\widetilde{\eta} \leq \widetilde{\xi}, \quad \widetilde{\eta} \geq -\widetilde{\xi}, \quad \widetilde{\eta} \leq -\widetilde{\zeta}$
$\gamma = 4$	P_7, P_6, P_5, P_3	$\widetilde{\eta} \leq \widetilde{\xi}, \quad \widetilde{\eta} \geq -\widetilde{\xi}, \quad \widetilde{\eta} > -\widetilde{\zeta}$
$\gamma = 5$	P_7, P_5, P_8, P_3	$\widetilde{\eta} > \widetilde{\xi}, \quad \widetilde{\eta} > -\widetilde{\xi}, \quad \widetilde{\eta} \geq -\widetilde{\zeta}$
$\gamma = 6$	P_4, P_5, P_3, P_8	$\widetilde{\eta} > \widetilde{\xi}, \quad \widetilde{\eta} < -\widetilde{\xi}, \quad \widetilde{\eta} > -\widetilde{\zeta}$

Table 2.3: Fine mesh tetrahedra.

Tetra	$\widetilde{\boldsymbol{\xi}} \longrightarrow \boldsymbol{r}$			$\boldsymbol{r} \longrightarrow \widetilde{\boldsymbol{\xi}}$		
$\gamma = 1$	$r = \dfrac{\widetilde{\xi} - \widetilde{\eta}}{2}$	$s = \dfrac{\widetilde{\eta} + 1}{2}$	$t = \dfrac{\widetilde{\zeta} + 1}{2}$	$\widetilde{\xi} = 2r + 2s - 1$	$\widetilde{\eta} = 2s - 1$	$\widetilde{\zeta} = 2t - 1$
$\gamma = 2$	$r = \dfrac{\widetilde{\xi} + 1}{2}$	$s = \dfrac{\widetilde{\eta} - \widetilde{\xi}}{2}$	$t = \dfrac{\widetilde{\zeta} + 1}{2}$	$\widetilde{\xi} = 2r - 1$	$\widetilde{\eta} = 2r + 2s - 1$	$\widetilde{\zeta} = 2t - 1$
$\gamma = 3$	$r = \dfrac{\widetilde{\eta} + 1}{2}$	$s = \dfrac{1 - \widetilde{\xi}}{2}$	$t = \dfrac{\widetilde{\xi} + \widetilde{\zeta}}{2}$	$\widetilde{\xi} = 1 - 2s$	$\widetilde{\eta} = 2r - 1$	$\widetilde{\zeta} = 2s + 2t - 1$
$\gamma = 4$	$r = \dfrac{\widetilde{\xi} - \widetilde{\eta}}{2}$	$s = \dfrac{1 - \widetilde{\xi}}{2}$	$t = \dfrac{1 - \widetilde{\zeta}}{2}$	$\widetilde{\xi} = 1 - 2s$	$\widetilde{\eta} = 1 - 2r - 2s$	$\widetilde{\zeta} = 1 - 2t$
$\gamma = 5$	$r = \dfrac{1 - \widetilde{\eta}}{2}$	$s = \dfrac{\widetilde{\eta} - \widetilde{\xi}}{2}$	$t = \dfrac{1 - \widetilde{\zeta}}{2}$	$\widetilde{\xi} = 1 - 2r - 2s$	$\widetilde{\eta} = 1 - 2r$	$\widetilde{\zeta} = 1 - 2t$
$\gamma = 6$	$r = \dfrac{1 - \widetilde{\eta}}{2}$	$s = \dfrac{\widetilde{\xi} + 1}{2}$	$t = \dfrac{\widetilde{\eta} + \widetilde{\zeta}}{2}$	$\widetilde{\xi} = 2s - 1$	$\widetilde{\eta} = 1 - 2r$	$\widetilde{\zeta} = 2r + 2t - 1$

Table 2.4: Conversion between fine-mesh element system and auxiliary element system (3D case).

Note that this approach is only viable for structured coarse meshes, consisting of quadrilateral (2D) or hexadedral (3D) elements. These meshes are perfect for modeling materials in Reference Volume Elements with XFEM. In the general case of an unstructured coarse mesh with triangular (2D) or tetrahedral (3D) elements, a similar approach can be adopted. Each triangle would be recursively divided into 3 subtriangles, using its nodes and centroid. Similarly, each tetrahedron would be recursively divided into 4 subtetrahedra, using its nodes and centroid.

2.3.3 Intersecting the finite elements

The LSM representation of the geometry of a material interface (curve in 2D problems or surface in 3D problems), can be used to find the intersection of the elements with that geometry. Let $\boldsymbol{r}_{P1}$ and $\boldsymbol{r}_{P2}$ be the nodal coordinates of an edge of a fine-mesh element, namely a triangle in 2D problems or tetrahedron in 3D problems, in the coordinate system of that element. Also let $\phi_{P1} = \phi(\boldsymbol{r}_{P1})$ and $\phi_{P2} = \phi(\boldsymbol{r}_{P2})$ be the level sets of the fine-mesh nodes P_1 and P_2 respectively. Then, the edge (P_1P_2) is intersected by the level representation of the curve or surface if

$$\begin{gathered} \phi_{P1} \cdot \phi_{P2} < 0 \\ \text{or} \\ \phi_{P1} = 0 \text{ and } \phi_{P2} \neq 0 \\ \text{or} \\ \phi_{P2} = 0 \text{ and } \phi_{P1} \neq 0 \end{gathered} \tag{2.150}$$

Assuming that the level set functions are linear inside the element, which holds for 3-noded triangles and 4-noded tetrahedra, the coordinates of the intersection point $\boldsymbol{r}_O$ with $\phi_O = 0$ are

$$\boldsymbol{r}_O = \boldsymbol{r}_{P1} + \frac{0 - \phi_{P1}}{\phi_{P2} - \phi_{P1}} (\boldsymbol{r}_{P2} - \boldsymbol{r}_{P1}) \tag{2.151}$$

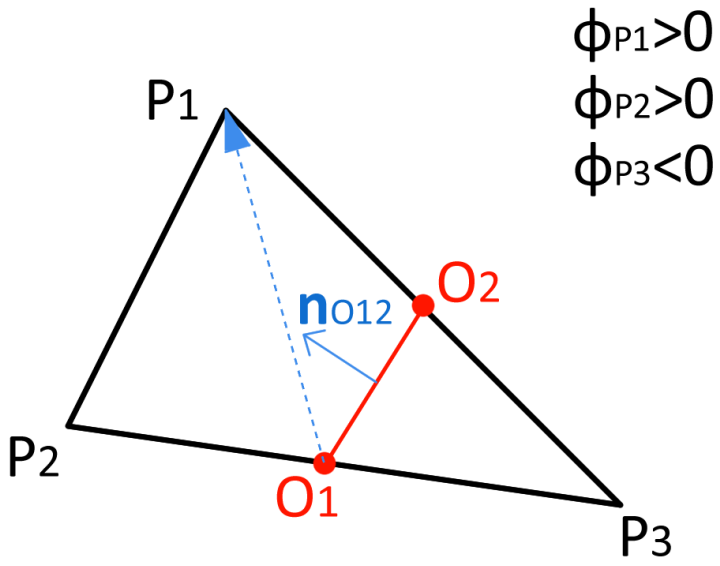


Figure 2.17: Intersection of LSM curve with a triangular element of the fine mesh.

On the other hand, the edge lies on the level set geometry, which means that fine-mesh element conforms to the level set geometry, if

$$\phi_{P1} = \phi_{P2} = 0 \tag{2.152}$$

2.3.3.1 2D problems

The intersection of a curve described by the proposed double-LSM approach and a coarse-mesh (XFEM) element is an 1D mesh consisting of line segments (actually just a series of

them) in 2D space, as depicted in figure 2.11. In order to find this *intersection mesh*, each coarse-mesh element (quadrilateral) is divided into the corresponding fine-mesh elements (triangles). Each triangle may by intersected by the level set representation of the curve along a line segment or not all. Iterating over the edges of the triangle, if an edge is intersected according to equation (2.150), the coordinates of the intersection point are calculated using equation (2.151). The triangle is intersected, if it has 2 unique intersection points $\boldsymbol{r}_{O1}$, $\boldsymbol{r}_{O2}$, as illustrated in figure 2.17. The segment (O_1O_2) must be oriented so that its normal vector

$$\boldsymbol{n}_{O12} = \begin{bmatrix} -(s_{O2} - s_{O1}) \\ r_{O2} - r_{O1} \end{bmatrix} \tag{2.153}$$

points towards the region where level set values are positive. Let one of the nodes with positive level sets be called P_1 ($\phi_{P1} > 0$). There must be at least one node with positive level set (and one with negative), if the triangle is intersected. The *intersection segment* (Q_1Q_2) is

$$(Q_1Q_2) = \begin{cases} (O_1O_2) & \text{if } \boldsymbol{n}_{O12} \cdot (\boldsymbol{r}_{P1} - \boldsymbol{r}_{O1}) > 0 \\ (O_2O_1) & \text{else} \end{cases} \tag{2.154}$$

Let $\mathcal{M}_I$ be the 1D intersection mesh, which consists of vertices and line segments. After identifying an intersection segment (Q_1Q_2) for each triangle (if the triangle is intersected), the points Q_1, Q_2 are sought in $\mathcal{M}_I$ and added to it, if they do not already exist. Then the segment (Q_1Q_2) is added to $\mathcal{M}_I$. After processing all fine-mesh elements (triangles) of a coarse-mesh element (quadrilateral), the coordinates of the vertices of the intersection mesh $\mathcal{M}_I$ are calculated in the natural system of that coarse-mesh element, using the mappings described in table 2.2 and equation (2.149).

Finally, an area mesh is generated for the integration operations of XFEM, as depicted in figure 2.9b. This mesh a) is defined in the natural coordinate system of the coarse mesh element, b) covers the element's area and c) conforms to the intersection mesh. Generating this conforming mesh can be done using the Constrained Delauny Triangulation (Chew, 1989), which takes as input the nodes of the coarse-mesh element and the vertices and segments of the intersection mesh.

2.3.3.2 3D problems

Similarly to the 2D case, the intersection of a surface described by the double-mesh LSM with a coarse-mesh element is a 2D mesh of triangles in 3D space, as shown in figure 2.12. Each coarse-mesh element (hexahedron) is divided into the corresponding fine-mesh elements (tetrahedra). A tetrahedron may be intersected by the level set surface along 1-2 triangles or not all. Iterating over the edges of the tetrahedron, if an edge is intersected according to equation (2.150), the coordinates of the intersection point are calculated using equation (2.151). There are two intersection cases, as can be observed in figure 2.18:

- 3 intersection points: This happens when there are 3 fine-mesh nodes with positive level sets and 1 with negative or 3 with negative and 1 with positive, as depicted in figure 2.18a. The intersection points O_1, O_2, O_3, which can be selected in any order, form a single triangle, which should be oriented, so that its normal vector $\boldsymbol{n}_{O123}$ points towards the region where level set values are positive:

$$\boldsymbol{n}_{O123} = (\boldsymbol{r}_{O2} - \boldsymbol{r}_{O1}) \times (\boldsymbol{r}_{O3} - \boldsymbol{r}_{O1}) \tag{2.155}$$

  where $\times$ is the cross product operator. Let one of the nodes with positive level sets be called P_1 ($\phi_{P1} > 0$). The intersection triangle $(Q_1Q_2Q_3)$ is

$$(Q_1Q_2Q_3) = \begin{cases} (O_1O_2O_3) & \text{if } \boldsymbol{n}_{O123} \cdot (\boldsymbol{r}_{P1} - \boldsymbol{r}_{O1}) > 0 \\ (O_1O_3O_2) & \text{else} \end{cases} \tag{2.156}$$

- 4 intersection points: This happens when there are 2 fine-mesh nodes with positive level sets and 2 with negative, as illustrated in figure 2.18b. Let the nodes with positive level sets be called P_1, P_2 and the nodes with negative level sets P_3, P_4. The order in which these nodes are selected is unimportant. Then the intersection points located on the edges between these nodes are

$$\begin{aligned} O_1 &\in (P_1P_3) \\ O_2 &\in (P_1P_4) \\ O_3 &\in (P_2P_3) \\ O_4 &\in (P_2P_4) \end{aligned} \tag{2.157}$$

  Then, two non-overlapping triangles $(O_1O_2O_3)$ and $(O_4O_3O_2)$ can be selected, although there are other valid choices. The normal vectors of these triangles are

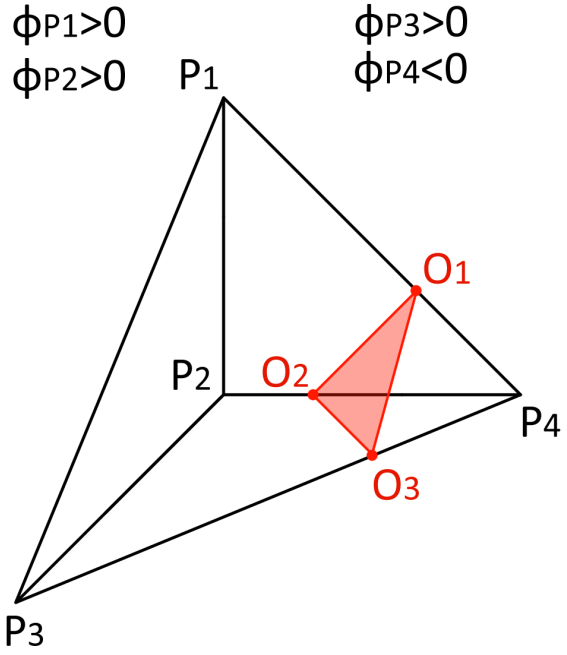
$$\begin{aligned} \boldsymbol{n}_{O123} &= (\boldsymbol{r}_{O2} - \boldsymbol{r}_{O1}) \times (\boldsymbol{r}_{O3} - \boldsymbol{r}_{O1}) \\ \boldsymbol{n}_{O432} &= (\boldsymbol{r}_{O3} - \boldsymbol{r}_{O4}) \times (\boldsymbol{r}_{O2} - \boldsymbol{r}_{O4}) \end{aligned} \tag{2.158}$$

  Taking into account that node P_1 lies on the positive side of the LSM surface ($\phi_{P1} > 0$), the two intersection triangles $(Q_1Q_2Q_3)$ and $(Q_4Q_5Q_6)$ are

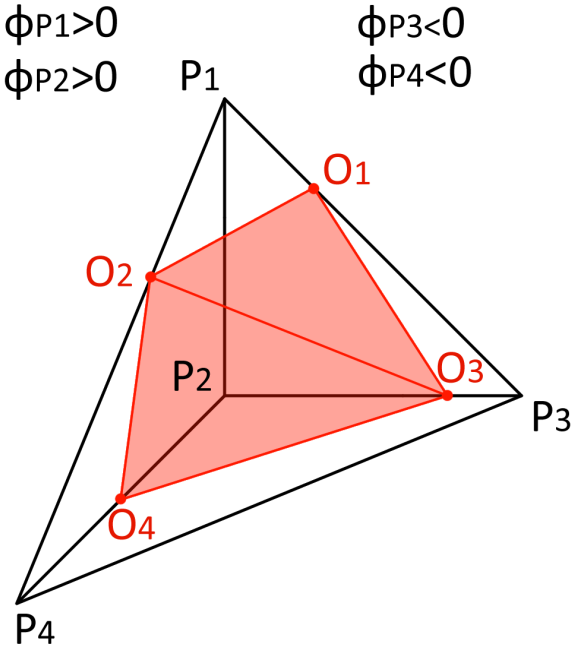
$$\begin{aligned} (Q_1Q_2Q_3) &= \begin{cases} (O_1O_2O_3) & \text{if } \boldsymbol{n}_{O123} \cdot (\boldsymbol{r}_{P1} - \boldsymbol{r}_{O1}) > 0 \\ (O_1O_3O_2) & \text{else} \end{cases} \\ (Q_4Q_5Q_6) &= \begin{cases} (O_4O_3O_2) & \text{if } \boldsymbol{n}_{O432} \cdot (\boldsymbol{r}_{P1} - \boldsymbol{r}_{O1}) > 0 \\ (O_4O_2O_3) & \text{else} \end{cases} \end{aligned} \tag{2.159}$$

Let $\mathcal{M}_I$ be the 2D intersection mesh, which consists of vertices and triangles. After identifying the intersection segments $(Q_1Q_2Q_3)$, $(Q_4Q_5Q_6)$ for each tetrahedron (if these intersections exist), the points Q_1-Q_6 are sought in $\mathcal{M}_I$ and added to it, if they do not already exist. Then the segments $(Q_1Q_2Q_3)$, $(Q_4Q_5Q_6)$ are added to $\mathcal{M}_I$. After processing all fine-mesh elements (tetrahedra) of a coarse-mesh element (hexahedron), the coordinates of the vertices of the intersection mesh $\mathcal{M}_I$ are calculated in the natural system of that coarse-mesh element, using the mappings described in table 2.4 and equation (2.149).

Finally, a volumetric mesh is generated for the integration operations of XFEM, as depicted in figure 2.10b. This mesh a) is defined in the natural coordinate system of the coarse mesh element, b) covers the element's area and c) conforms to the intersection mesh. Generating this conforming mesh can be done using the 3D version of the Constrained Delauny Triangulation (Shewchuk, 2008), which takes as input the nodes of the coarse-mesh element and the vertices and segments of the intersection mesh.



(a) 3 intersection points



(b) 4 intersection points

Figure 2.18: Intersection of LSM surface with a tetrahedral element of the fine mesh.

Chapter 3

Heat transfer analysis applications

In this chapter, the XFEM methodology proposed in chapter 2 is used in a series of numerical applications. To begin with, the computational homogenization method is presented, in order to obtain the macroscopic conductivity of composite materials. Subsequently, the proposed XFEM formulation is validated in a numerical benchmark, as well as in the analysis of multi-grain materials. Finally, the numerical model is used to simulate conductive heat transfer in polymer-CNT composites, with random complex 2D and 3D microstructure, and then calibrated using experimental macroscopic measurements.

3.1 Computational homogenization

In solid mechanics, homogenization is used to evaluate the parameters of the effective behavior of the macroscopic composite material by adopting a microscopic Representative Elementary Volume (RVE), on which predefined boundary conditions are applied. In this dissertation, linear boundary conditions are considered for the RVE, as described in Miehe and Koch (2002). Consider a square RVE denoted by Ω and its external boundary $\partial\Omega$. Ω is discretized by nodes $\boldsymbol{x}$ that can be partitioned into internal nodes $\boldsymbol{x}_i \in \Omega$ and boundary nodes $\boldsymbol{x}_b \in \partial\Omega$, as illustrated in figure 3.1. By applying the same internal-boundary partitioning to the DOFs of these nodes, the conductivity matrix can be written as

$$\boldsymbol{K} = \begin{bmatrix} \boldsymbol{K}_{ii} & \boldsymbol{K}_{ib} \\ \boldsymbol{K}_{bi} & \boldsymbol{K}_{bb} \end{bmatrix} \tag{3.1}$$

In computational homogenization, the internal DOFs are condensed by taking the Schur complement of $\boldsymbol{K}_{ii}$:

$$\widetilde{\boldsymbol{K}}_{bb} = \boldsymbol{K}_{bb} - \boldsymbol{K}_{bi}\boldsymbol{K}_{ii}^{-1}\boldsymbol{K}_{ib} \tag{3.2}$$

Due to the assumption of linear boundary conditions:

$$T_b = \nabla T \cdot \boldsymbol{x}_b = \boldsymbol{D}^T \cdot \nabla T \tag{3.3}$$

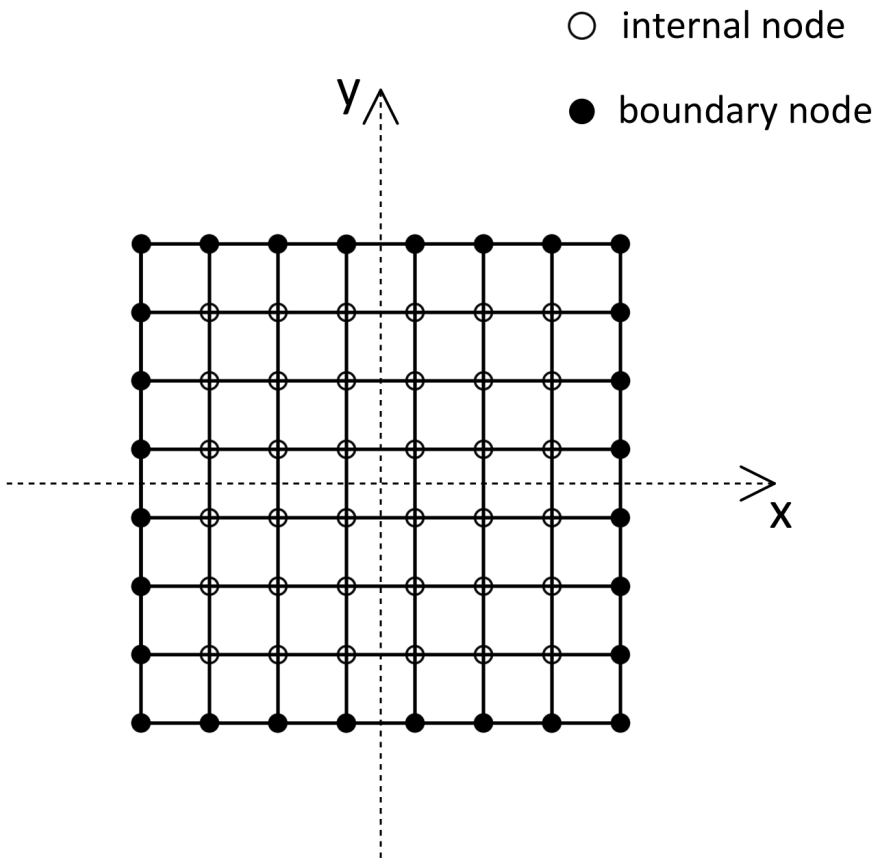


Figure 3.1: A 2D RVE with internal and boundary nodes.

with ∇T denoting any macroscopic temperature gradient vector applied on the RVE boundary and T_b the temperature of boundary node $\boldsymbol{x}_b$. The kinematic relationship matrix $\boldsymbol{D}$ contains the coordinates of each of the n_b boundary nodes in a coordinate system defined for the RVE, which is also depicted in figure 3.1. In case of 2D problems, it can be calculated as

$$\boldsymbol{D} = \begin{bmatrix} x_1 & x_2 & \cdots & x_{nb} \\ y_1 & y_2 & \cdots & y_{nb} \end{bmatrix} \tag{3.4}$$

and for 3D problems

$$\boldsymbol{D} = \begin{bmatrix} x_1 & x_2 & \cdots & x_{n_b} \\ y_1 & y_2 & \cdots & y_{n_b} \\ z_1 & z_2 & \cdots & z_{n_b} \end{bmatrix} \tag{3.5}$$

The effective macroscopic conductivity tensor can be now computed as

$$\widetilde{\boldsymbol{C}} = \frac{1}{\|\Omega\|} \boldsymbol{D} \widetilde{\boldsymbol{K}}_{bb} \boldsymbol{D}^T \tag{3.6}$$

where $\|\Omega\|$ is the volume of Ω. This computational homogenization method is combined with the XFEM procedure presented in chapter 2. For a target composite material, an RVE of the composite material is considered in the micro-scale. This RVE consists of multiple material phases, which are separated by interfaces that exhibit thermal resistance and can be dispersed randomly or according to case specific rules. The elaborated XFEM procedure is used to obtain the conductivity matrix $\boldsymbol{K}$, according to equation (2.113). Then equation (3.6) is used to approximate the macroscopic conductivity tensor of the composite material.

3.2 Application 1: Three-phase benchmark

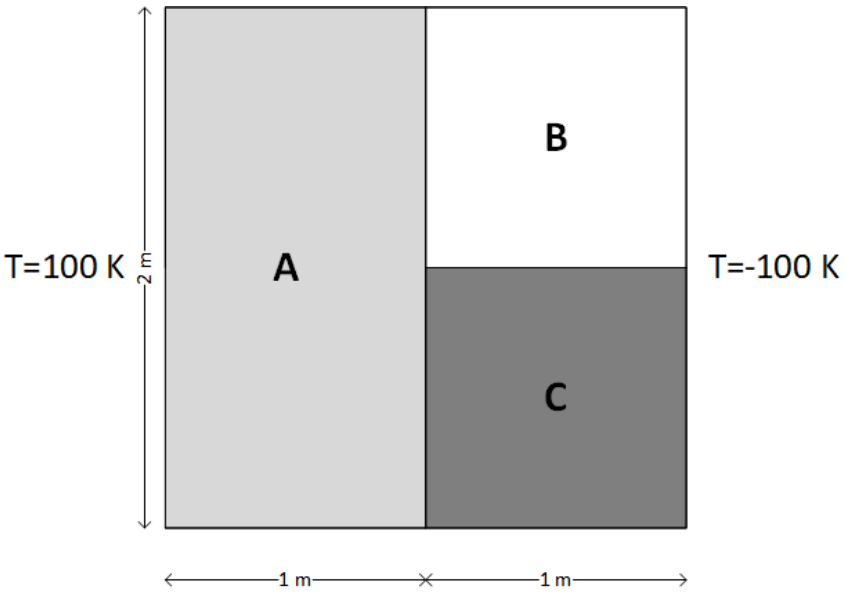


Figure 3.2: Material configuration

First of all, a simple a benchmark problem is investigated, in order to validate the correctness of the XFEM formulation proposed in chapter 2. In this example, the domain consists of a composite material with three phases, A, B and C, having different interface conductance along their boundaries. The configuration of the composite material is given in figure 3.2 along with the boundary conditions applied on the left and on the right side. The thickness of the material is equal to 1, while the conductivities of materials A, B and C are chosen $k_A = 100\frac{W}{mK}$, $k_B = 1000\frac{W}{mK}$ and $k_C = 1\frac{W}{mK}$, respectively. The conductance at

the interfaces A-B, A-C and B-C is $k_{AB} = 100\frac{W}{m^2K}$, $k_{AC} = 10\frac{W}{m^2K}$ and $k_{BC} = 1000\frac{W}{m^2K}$, respectively.

This example is first solved using the standard FEM software Abaqus (Dassault Systemes, 2020), where a 40×40 mesh of quadrilateral elements was deemed adequate for an accurate representation of the temperature field. In addition, appropriate cohesive interface elements were used in the FEM model so as to capture the temperature discontinuities at the interfaces between the different material phases. For XFEM, a nonconforming 39×39 mesh was used. The results of the analysis obtained from standard FEM and the proposed XFEM formulation are given in figures 3.3 and 3.4, respectively.

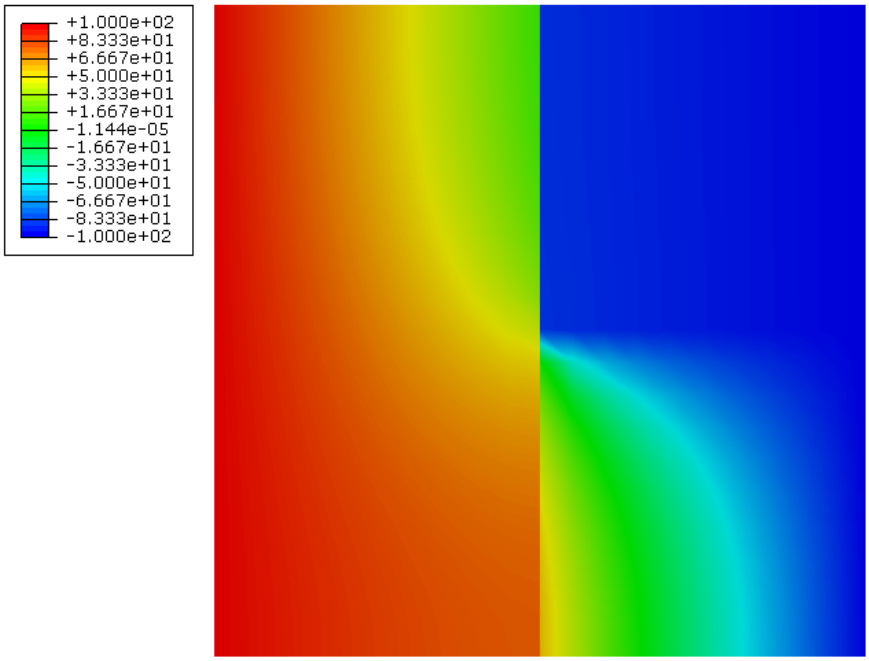


Figure 3.3: Temperature field from standard FEM analysis

Additionally, figures 3.5b, 3.5c and 3.5d depict a comparison of the temperatures along the different intersection lines shown in figure 3.5a. This investigation indicates the results obtained from the proposed approach are in almost perfect agreement with the FEM model. Lastly, figure 3.6 displays the distribution of the heat flux field at the interior of the material, where it becomes evident that heat 'chooses' to travel through the paths that have the highest conductivity, a result that matches our physical intuition.

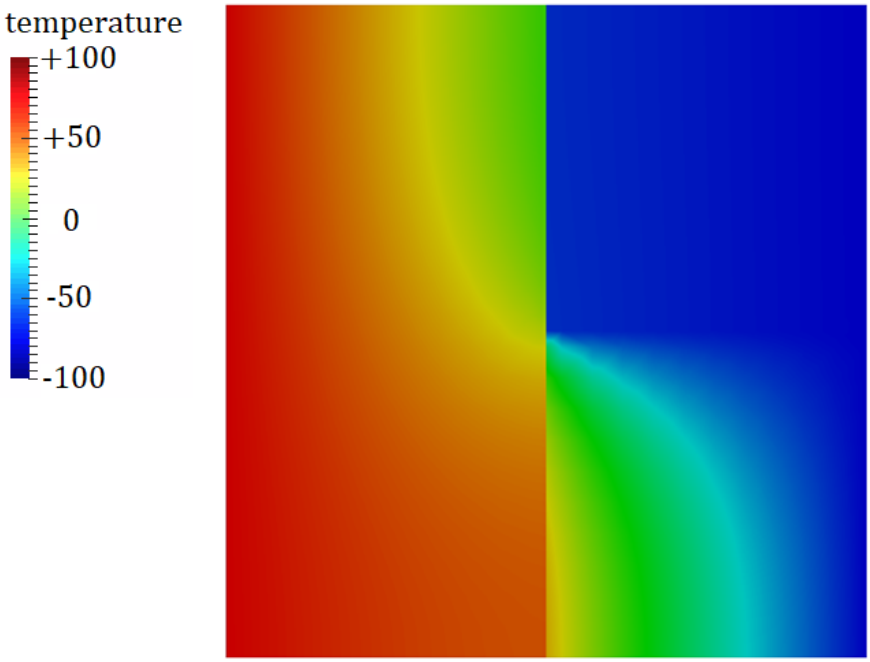
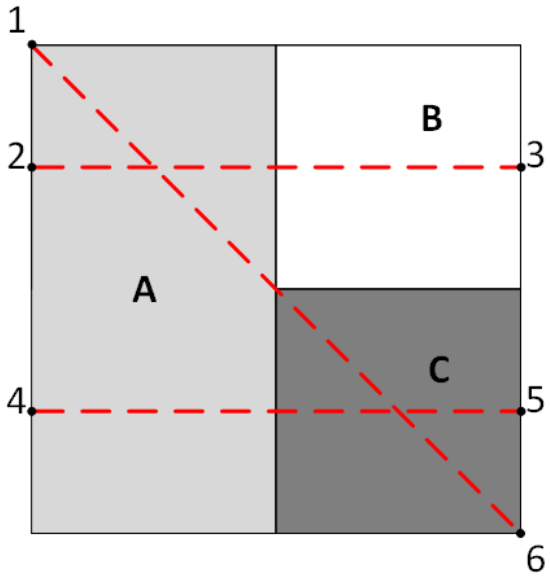
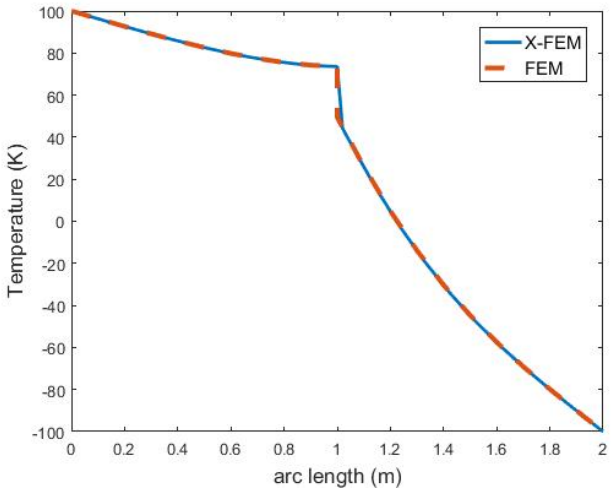


Figure 3.4: Temperature field from XFEM

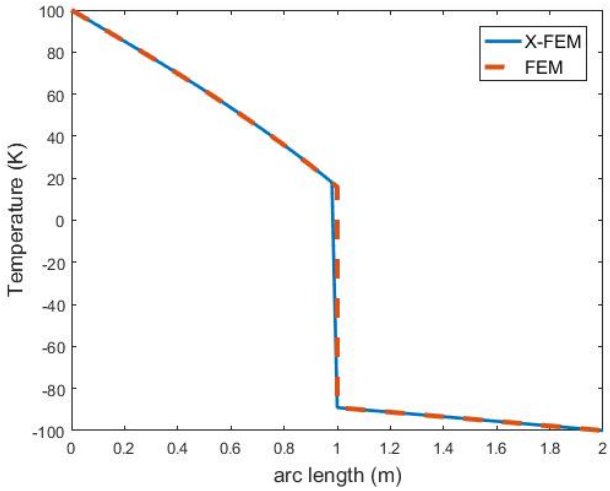


(a) Intersection lines paths

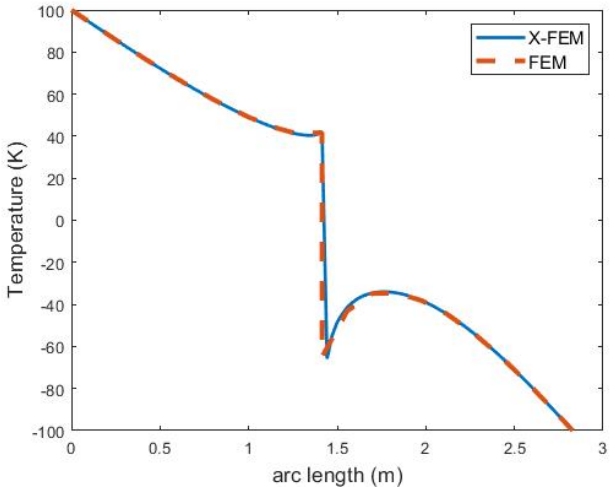


(b) Temperature along line 2-3

Figure 3.5: Comparison of temperature between FEM and proposed XFEM formulation.



(c) Temperature along line 4-5



(d) Temperature along line 1-6

Figure 3.5: Comparison of temperature between FEM and proposed XFEM formulation.

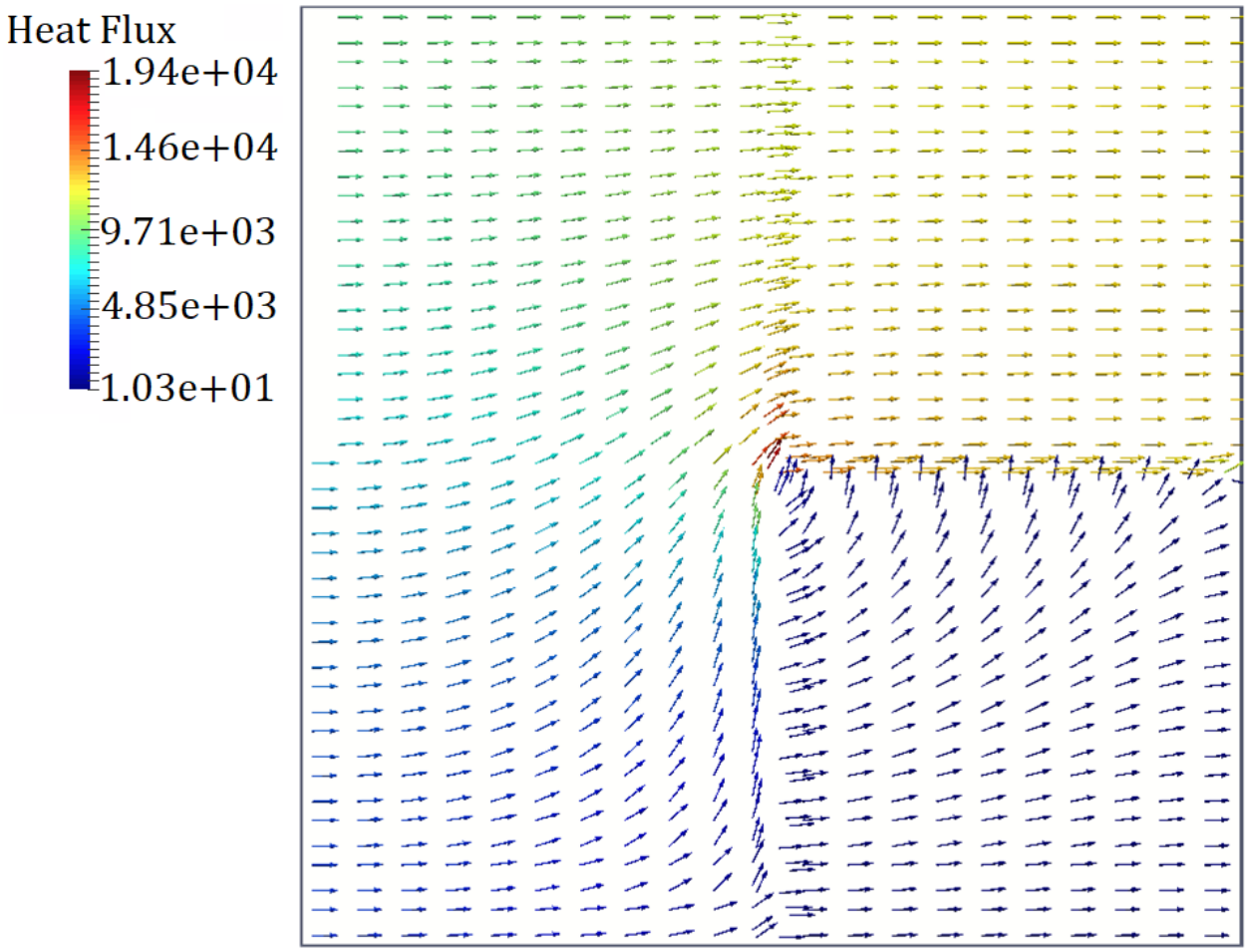


Figure 3.6: Heat fluxes

3.3 Application 2: Polycrystalline silicene

This application investigates thermal transport across grain boundaries in polycrystalline silicene. Silicene is a monolayer material, consisting of a honeycomb structure of silicon elements. It is commonly fabricated through chemical vapor deposition (CVD). However, this process leads to the formation of polycrystalline structures or, in other words, the formation of grains, with each grain having different crystalline orientation. As a consequence, temperature jumps appear across the grain boundaries due to phonon scattering. The thermal properties of this particular material have been extensively studied in Khalkhali et al. (2019), where the interfacial thermal conductance of grain boundaries was estimated at $k_{SB} = 2.46 \times 10^9 \frac{W}{m^2 K}$ using non-equilibrium molecular dynamics simulations. The conductivity of silicene is $k_{SI} = 41 \frac{W}{mK}$. Then, the effective conductivity, k_{eff}, as a function of the grain size GS can be estimated using the following formula (Mortazavi et al., 2014; Mortazavi et al., 2017)

$$k_{eff} = \frac{k_{SI} \times k_{SB} \times GS}{k_{SI} + k_{SB} \times GS} \tag{3.7}$$

which was also verified in Khalkhali et al. (2019).

The aim of this example is to apply the proposed XFEM methodology for the conductive heat transfer analysis of polycrystalline silicene, where the geometry of the interfaces is quite complex, and validate the XFEM results with those predicted by equation (3.7). In figure 3.7 the material configuration is depicted, where red lines depict the grain boundaries and black lines the structured finite element mesh used for the purposes of XFEM. The characteristic grain size is defined as $GS = \sqrt{(A/N)}$ with A being the total area of the silicene sheet and N the number of grains. The geometry of the grains was generated via Voronoi tessellation, while their characteristic size was considered to be a varying parameter. Without loss of generality, the same geometry was used for all RVEs, scaled accordingly to the GS size. As boundary conditions of the RVE, a temperature of $315K$ was applied to its leftmost edge and $285K$ to its rightmost. The RVE size was taken equal to $(20 \cdot GS) \times (20 \cdot GS)$ and various RVEs were analyzed with GS ranging from $2nm$ to $1000nm$.

To validate the proposed XFEM formulation, the effective thermal conductivity of polycrystalline silicene is evaluated from the XFEM model for various grain sizes. The results are then compared to those obtained from equation (3.7). As evidenced by figure 3.8, the two models are in very good agreement. From this figure, it can be surmised that the effective thermal conductivity of polycrystalline silicene increases monotonically with the size of grains, before reaching a plateau at the conductivity value of pure silicene. To estimate the sensitivity of the methodology with respect to the mesh size, figures 3.9a and 3.9b present two convergence diagrams, corresponding to the cases of $GS = 2nm$ and $GS = 1000nm$. It

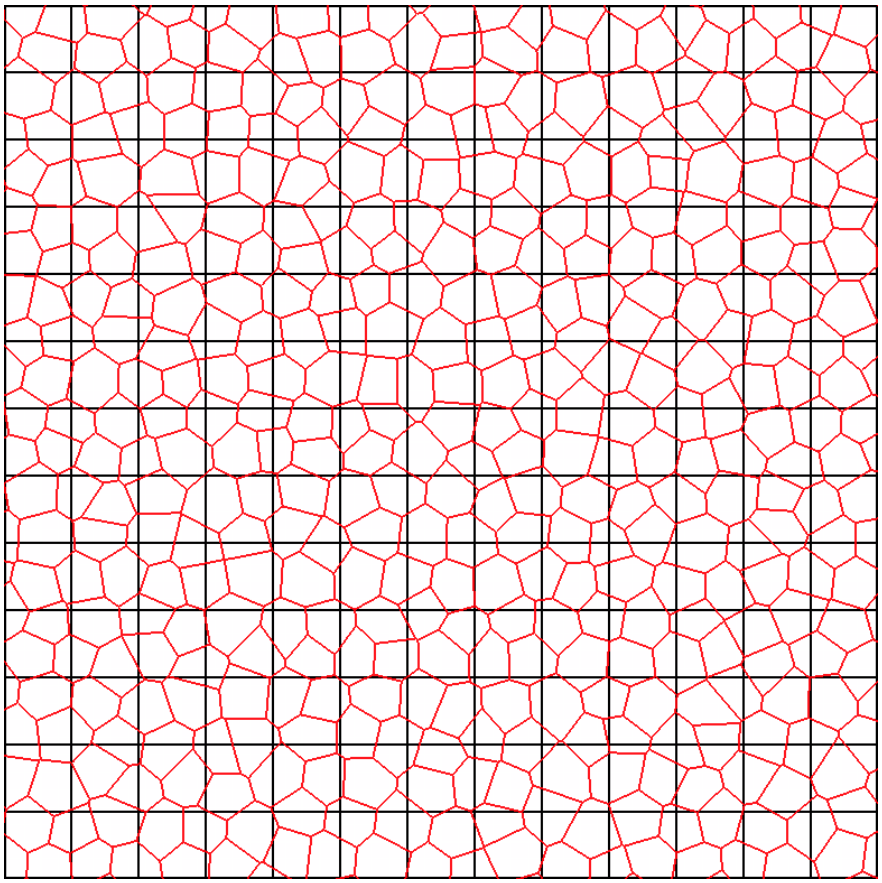


Figure 3.7: Material configuration and mesh (red lines: grain boundaries, black lines: XFEM mesh)

is obvious that in both these cases an adequate level of accuracy can be obtained even for a small number of elements.

Furthermore, figure 3.10 illustrates the temperature fields at steady-state for two RVEs with $GS = 2nm$ and $GS = 1000nm$. It becomes apparent that for a sample with small grain size (figure 3.10a), the temperature distribution is almost constant inside each grain. On the other hand, for samples with large grain sizes (figure 3.10b), a slight temperature gradient can be detected inside each grain. These observations suggest that for small grain sizes, the interface thermal resistance plays a dominant role in the temperature distribution but as the grain size becomes larger, its effect is significantly diminished. To further elucidate this, in figures 3.11a and 3.11b the temperature profiles along a section cut in the middle of the corresponding RVEs are presented. Specifically, in figure 3.11a it is clear that the temperature is nearly constant inside each grain and temperature jumps manifest at the boundaries of the grains. In contrast, in figure 3.11b the temperature profile is approximately linear inside the RVE and the temperature jumps are indiscernible.

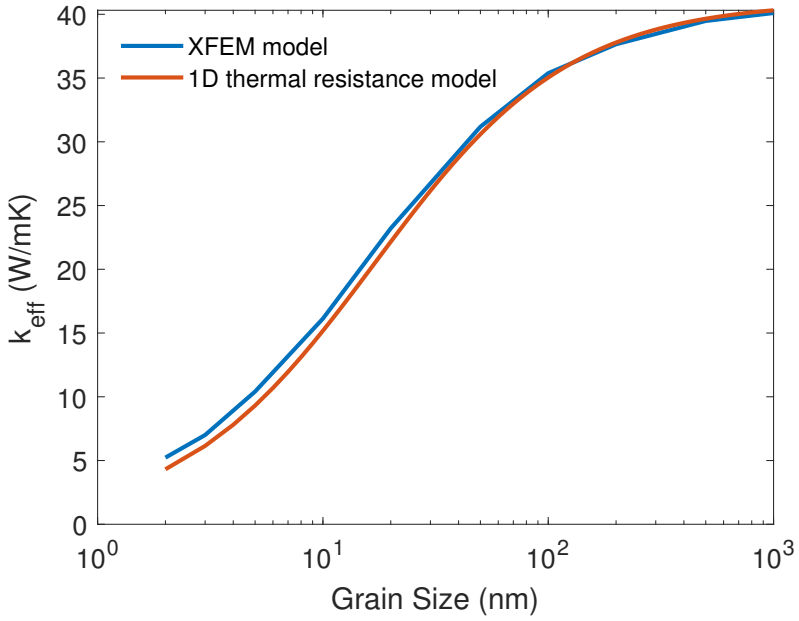
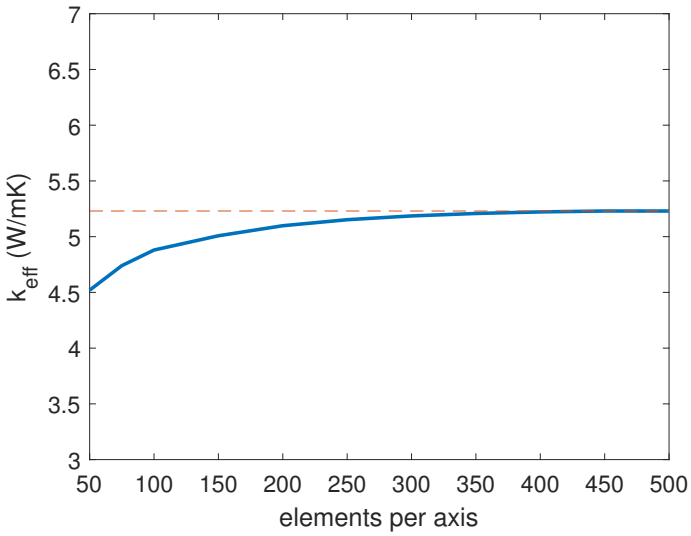
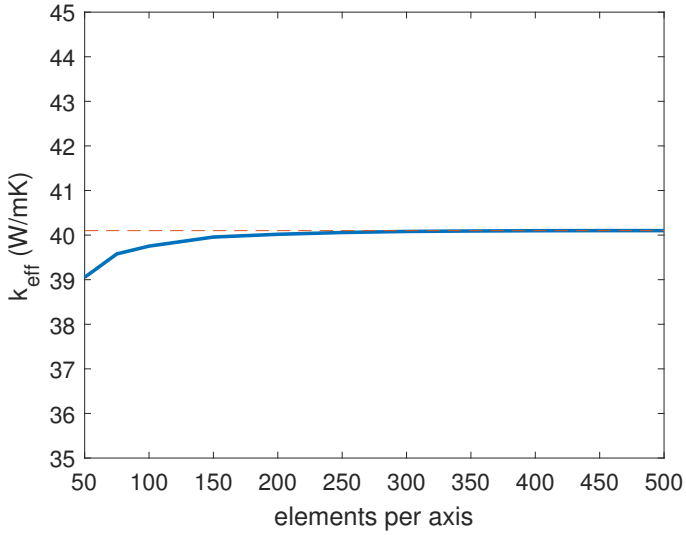


Figure 3.8: Effective thermal conductivity of polycrystalline silicene with various grain sizes based on the analytical thermal resistance model from Mortazavi et al. (2014) and Mortazavi et al. (2017) and the proposed XFEM formulation. Material properties: $k_{SI} = 41\frac{W}{mK}$, $k_{SB} = 2.46 \times 10^9 \frac{W}{m^2 K}$



(a) Mesh convergence for grain size $2nm$



(b) Mesh convergence for grain size $1000nm$

Figure 3.9: Mesh convergence plots for grain size $2nm$ and $1000nm$

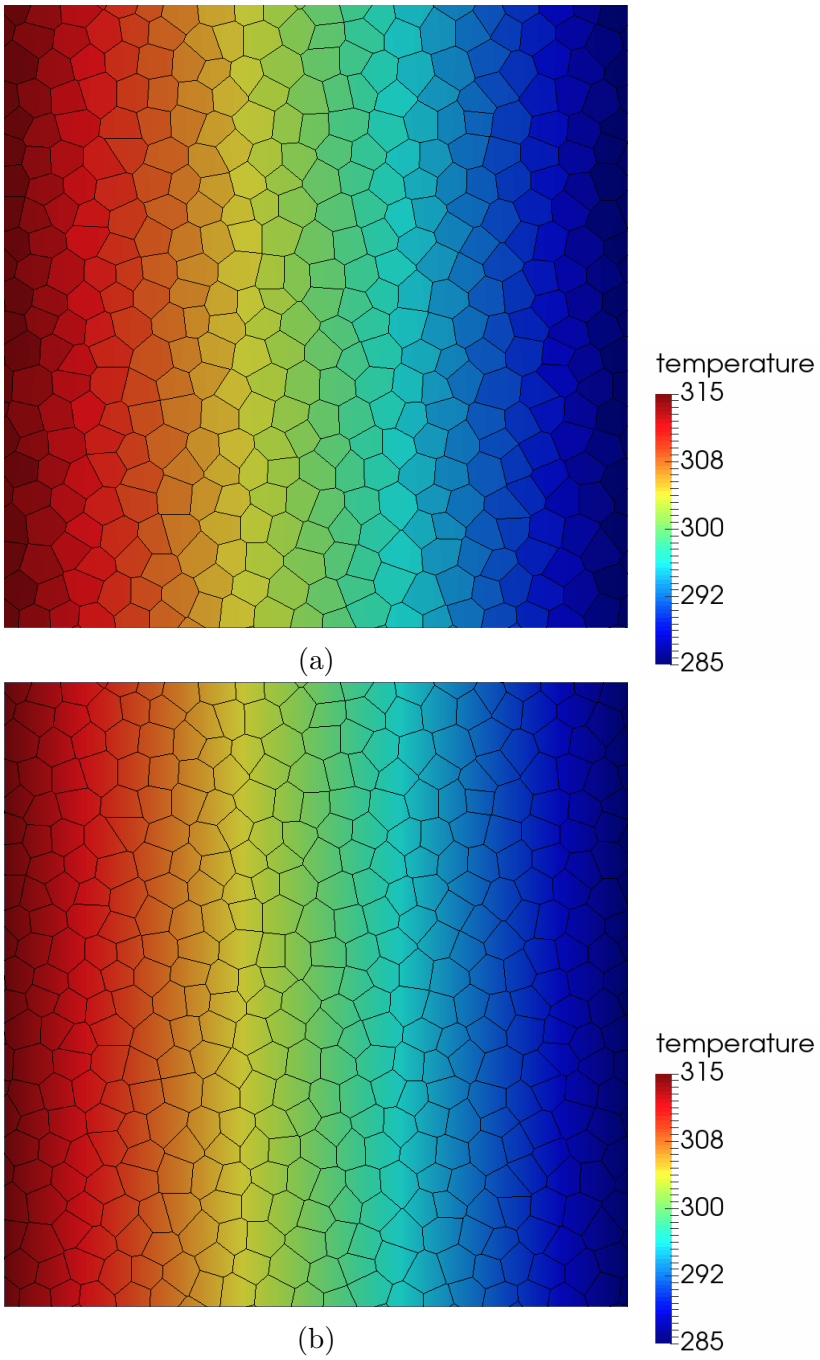
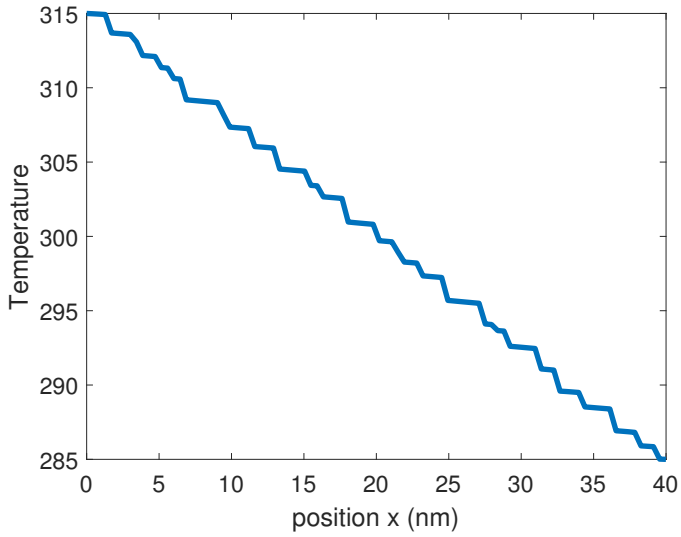
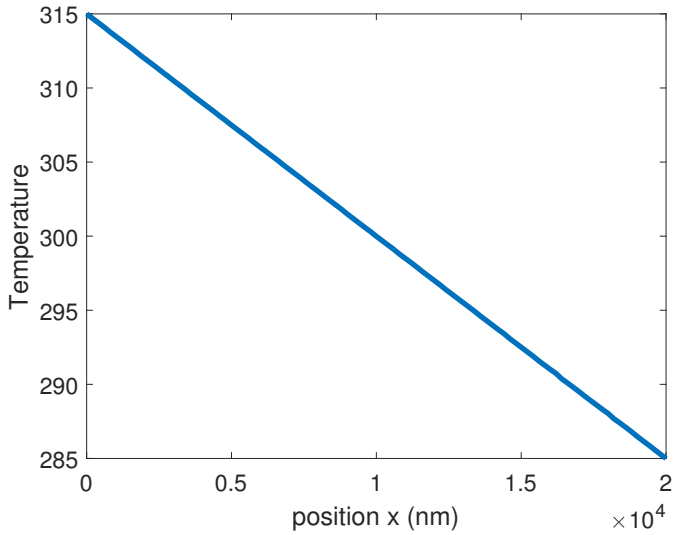


Figure 3.10: Temperature fields inside the polycrystalline material. Material properties: $k_{SI} = 41\frac{W}{mK}$, $k_{SB} = 2.46 \times 10^9 \frac{W}{m^2 K}$. (a) Grain size $2nm$, RVE size 40 $nm \times 40$ nm. (b) Grain size $1000nm$, RVE size 20000 $nm \times 20000$ nm



(a) temperature profile for grain size 2 nm



(b) temperature profile for grain size 1000 nm

Figure 3.11: Temperature profiles along a section cut in the middle of the RVEs for grain sizes $2nm$ and $1000nm$

3.4 Application 3: 2D polymer - carbon nanotube composite

In this example, a 2D RVE of a polypropylane (PP) polymeric matrix reinforced with carbon nanotubes is studied. Pure CNTs are excellent heat conductors with their conductivity estimated to be in the range from $3500\frac{W}{mK}$ (Pop et al., 2006) to $6600\frac{W}{mK}$ (Berber et al., 2000). Therefore, it has been theorized that the use of CNTs as inclusions in polymers will significantly enhance their conductivity. However, the results from several experimental works (Gojny et al., 2006; Moisala et al., 2006; Yunsheng et al., 2006) indicate only a marginal improvement in the polymer's conductivity, which is attributed to defects and impurities in the CNT lattice (Che et al., 2000), or, more critically, to the thermal resistance during heat transmission from one medium to the other (Marconnet et al., 2013; Singh et al., 2007). On the bright side, by increasing the volume fraction of CNTs in the polymer, a percolation network is created (Kumar et al., 2007), through which heat is being transferred unhindered, which results in increased material conductivity. Moreover, another property of CNTs, often neglected when used as inclusions in polymers, is that they act as heterogeneous nucleating agents for polymer crystallizing along the interface. This induces the formation of a transcrystalline layer (TL) that surrounds the CNT in a process known as CNT-induced polymer crystallization (S. Zhang et al., 2008). This layer has improved thermal properties compared to the amorphous polymer, which affects the overall thermal conductivity of the RVE.

In this regard, the proposed XFEM formulation is employed for the study of 2D RVEs of this three-phase material. CNTs are assumed to be randomly dispersed in the interior of the PP and each phase (PP, CNT, TL) has its own conductivity and interfacial conductance. More specifically, the PP conductivity was considered $k_{PP} = 0.20\frac{W}{mK}$ (Maier & Calafut, 1998), the CNT conductivity $k_{CNT} = 2000\frac{W}{mK}$ (Hussain et al., 2017) and in lack of any experimental knowledge over the thermal properties of the transcrystalline layer, it was taken equal to $k_{TL} = 0.30\frac{W}{mK}$, which is approximately the conductivity of the isotactic polypropylene (Laschet et al., 2017) (degree of crystallinity 30-60 %). Due to the inherent uncertainties concerning the interfacial conductances, they were treated as parameters in this model and a sensitivity analysis was performed to assess their influence on the effective conductivity of the composite material. Their initial values were $k_{PP/CNT} = 0.25\frac{W}{m^2K}$ and $k_{CNT/TL} = 1000\frac{W}{m^2K}$. Moreover, a justified assumption has been made that the TLs around the CNTs have different crystallinities and an interface is created at the edges where TLs of adjacent CNTs merge. The crystalline phases around each CNT exhibit different preferred orientations that can not be determined a priori, causing interfacial resistance. An analogous phenomenon was the formation of the grain boundaries in the polycrystalline material of

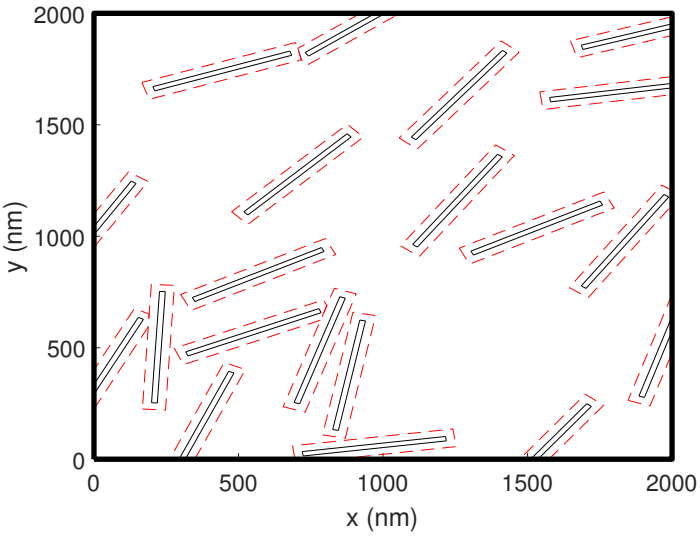
section 3.3. Therefore, the interfacial resistance between two TLs was also considered as a parameter under investigation.

The study of this material was performed on RVEs of size $2000nm \times 2000nm$, with varying volume fractions. Figure 3.12a depicts the RVE corresponding to a CNT volume fraction of 4.27%, while the RVEs of figures 3.12b, 3.12c and 3.12d correspond to volume fractions of 6.54%, 10.31% and 12.80%, respectively. In all cases, the CNTs were randomly scattered in the parent material and their orientations were also random, following a uniform distribution between 0 and 2π. The length of the CNTs was taken equal to $500nm$, their diameter $20nm$ and the width of the TL was $60nm$. Figure 3.13 displays in more detail the different interfaces formed in the interior of the composite material.

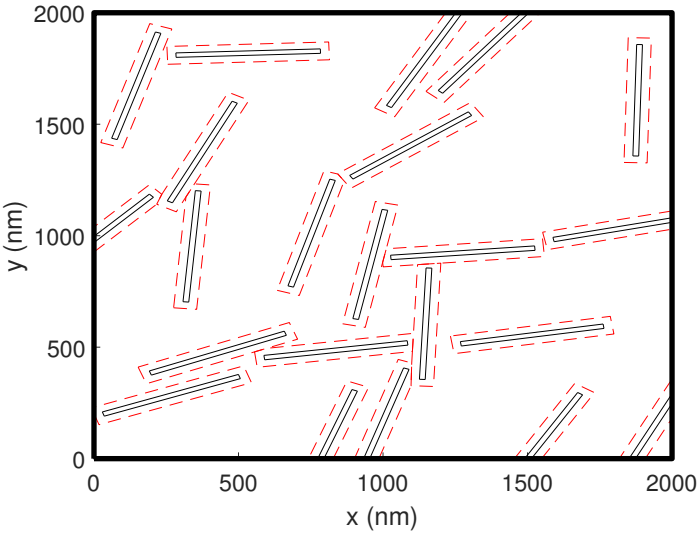
Next, a temperature gradient is applied at the x-direction of the RVEs by setting the temperature at the leftmost nodes equal to $100K$ and at the rightmost nodes equal to $-100K$. Figure 3.14 depicts the temperature field and the heat fluxes inside the corresponding RVEs at thermal equilibrium. This figure highlights the fact that heat fluxes inside CNTs are significantly larger in magnitude than those in the polymer matrix. Evidently, heat 'prefers' to travel mostly through CNTs, due to their excellent conductivity. CNTs oriented towards the y-axis exhibit smaller heat fluxes, as the temperature gradient for this example is applied only in the x-direction. However, when CNT clusters are formed, which is the case for higher volume fractions, these vertical CNTs act as bridges between horizontal heat paths. This is particularly noticeable in figure 3.14d, where several percolation networks can be detected.

Subsequently, the impact of the interfacial conductances $k_{TL/TL}$, $k_{PP/TL}$ and $k_{CNT/TL}$ on the overall material conductivity was assessed. To this end, a parametric investigation was performed, where the effective conductivity of each RVE was obtained via computational homogenization. More specifically, figure 3.15 illustrates the effect of the conductance $k_{TL/TL}$ of the interface between two TLs for various volume fractions, when considering it to be perfect insulator, perfect conductor or anything in-between. As shown from this figure, the effect $k_{TL/TL}$ has on the material's conductivity becomes more pronounced as the volume fraction increases. This is attributed to the fact that for higher volume fractions, more local networks between adjacent CNTs are being formed (see figure 3.12) and higher values of $k_{TL/TL}$ significantly facilitate conductive heat transfer through them. In contrast, for low volume fractions the absence of percolation networks renders the effect of $k_{TL/TL}$ insignificant.

Moreover, figure 3.16 depicts the surface plots of k_{eff} as a function of the interface conductances $k_{CNT/TL}$ and $k_{PP/TL}$ for all volume fractions. Upon inspection of these figures, it becomes apparent that the influence of $k_{PP/TL}$ and $k_{CNT/TL}$ on the effective conductivity, increases for higher volume fractions. This stems from the fact that by increasing the content of CNTs, more boundary interfaces are generated. Hence, higher values of interface conductance facilitate heat entering the highly conductive CNTs and flowing through them. For instance, for volume fraction 12.80%, the maximum attainable improvement in k_{eff} between the case of perfect insulation $\left(k_{CNT/TL} = k_{PP/TL} = 10^{-5}\dfrac{W}{m^2K}\right)$ and perfect conductance $\left(k_{CNT/TL} = k_{PP/TL} = 10^{3}\dfrac{W}{m^2K}\right)$ is 61.20%. These results indicate that



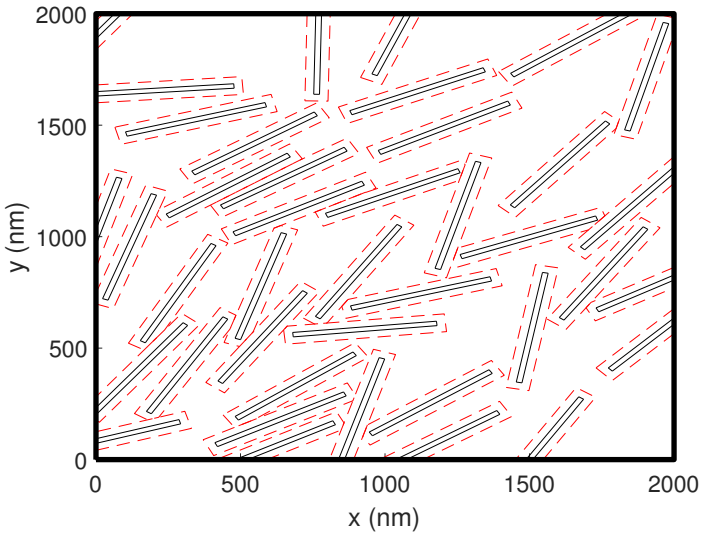
(a) RVE with vf 4.27%



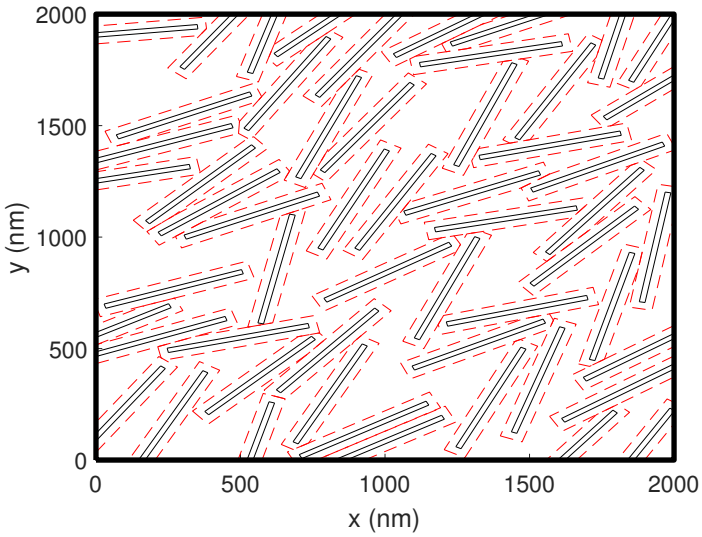
(b) RVE with vf 6.54%

Figure 3.12: RVEs for various volume fractions

$k_{CNT/TL}$ and $k_{PP/TL}$ play an important role in the overall material conductivity and should not be neglected in the analysis of nano-composites. Finally, as illustrated in these figures, the volume fraction of CNTs in the parent material is the most significant factor that enhances the composite's effective conductivity compared to the initial polymer's conductivity, $k_{PP} = 0.20\frac{W}{mK}$.



(c) RVE with vf 10.31%



(d) RVE with vf 12.80%

Figure 3.12: RVEs for various volume fractions

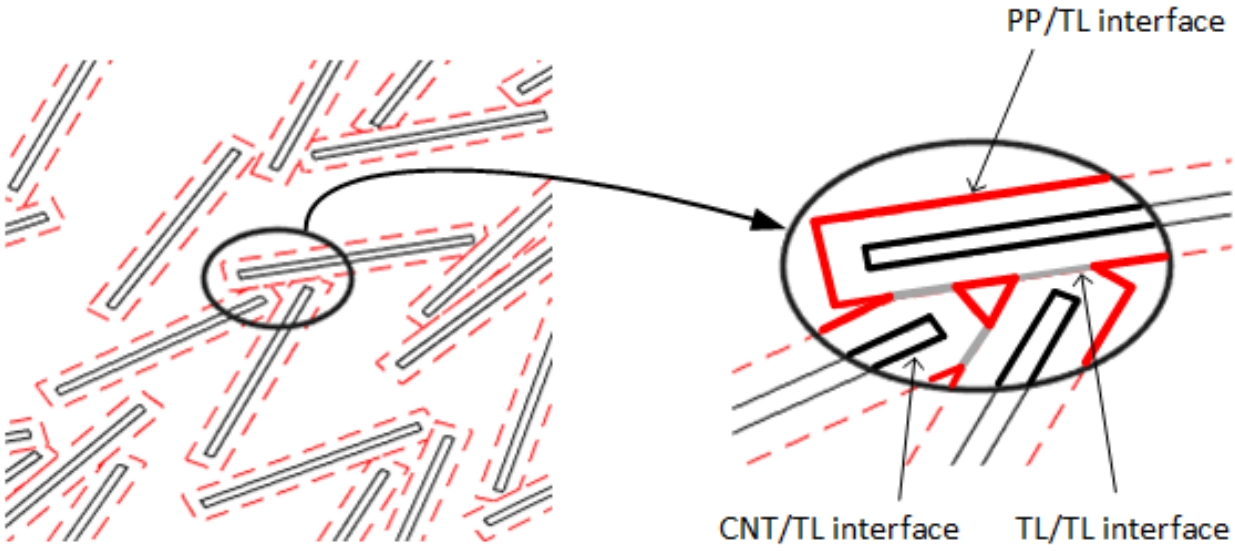
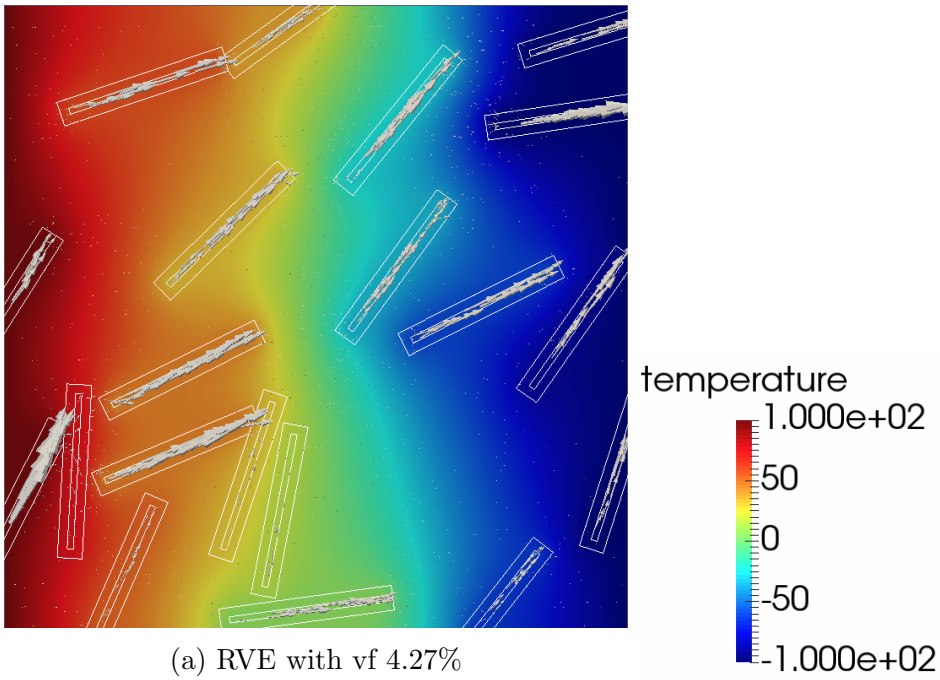
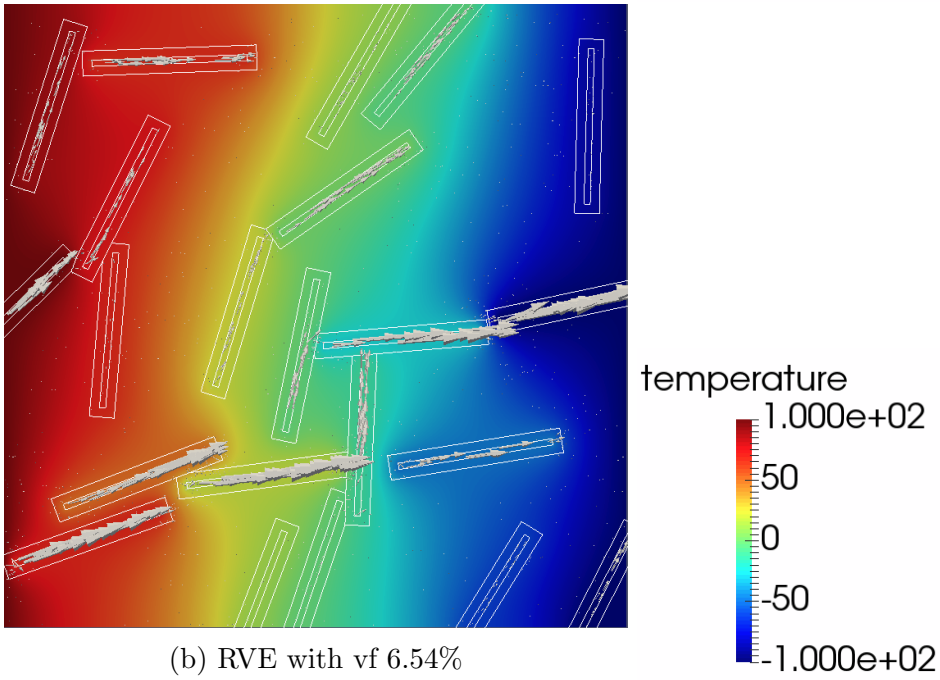


Figure 3.13: Detailed view of the different interfaces in the interior of the composite material

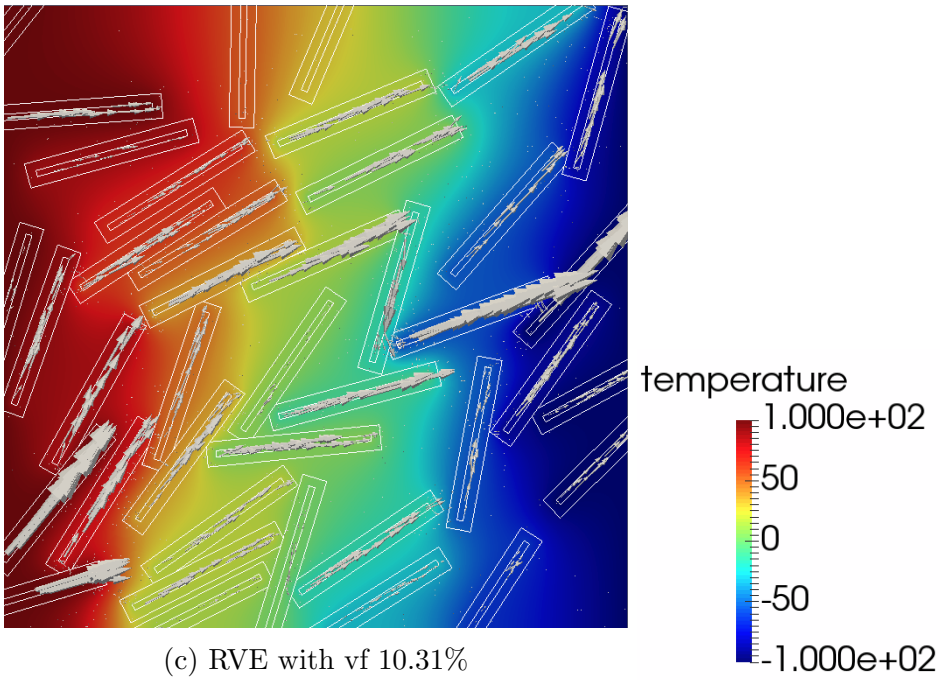


(a) RVE with vf 4.27%

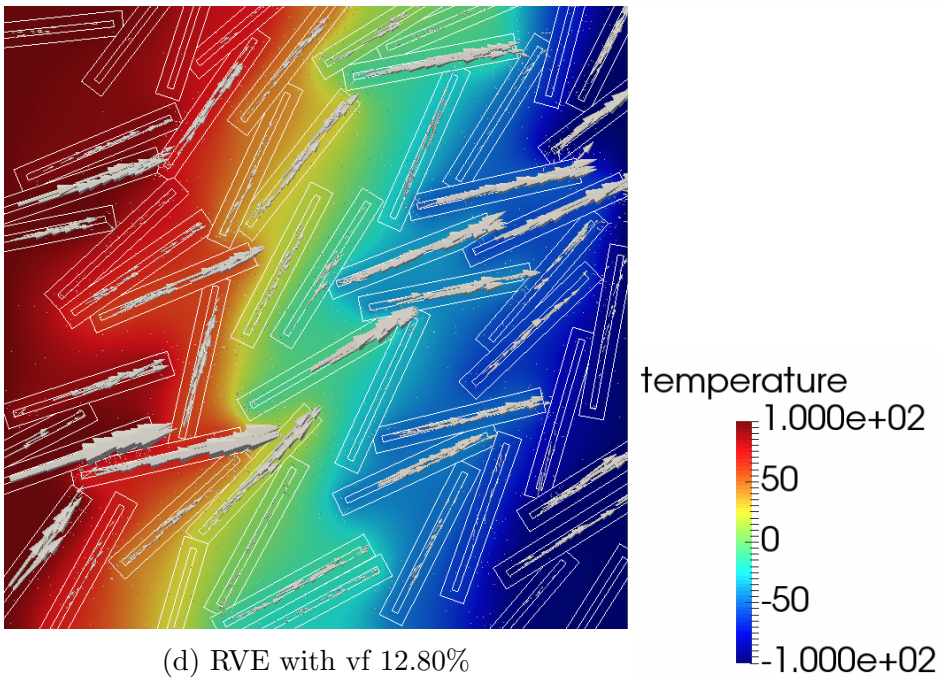


(b) RVE with vf 6.54%

Figure 3.14: Temperature field and heat fluxes inside the RVEs at thermal equilibrium. Material properties: $k_{CNT} = 2000\frac{W}{mK}$, $k_{PP} = 0.20\frac{W}{mK}$, $k_{TL} = 0.30\frac{W}{mK}$, $k_{PP/TL} = 0.25\frac{W}{m^2K}$, $k_{CNT/TL} = 1000\frac{W}{m^2K}$, $k_{TL/TL} = 1000\frac{W}{m^2K}$.



(c) RVE with vf 10.31%



(d) RVE with vf 12.80%

Figure 3.14: Temperature field and heat fluxes inside the RVEs at thermal equilibrium. Material properties: $k_{CNT} = 2000\frac{W}{mK}$, $k_{PP} = 0.20\frac{W}{mK}$, $k_{TL} = 0.30\frac{W}{mK}$, $k_{PP/TL} = 0.25\frac{W}{m^2K}$, $k_{CNT/TL} = 1000\frac{W}{m^2K}$, $k_{TL/TL} = 1000\frac{W}{m^2K}$.

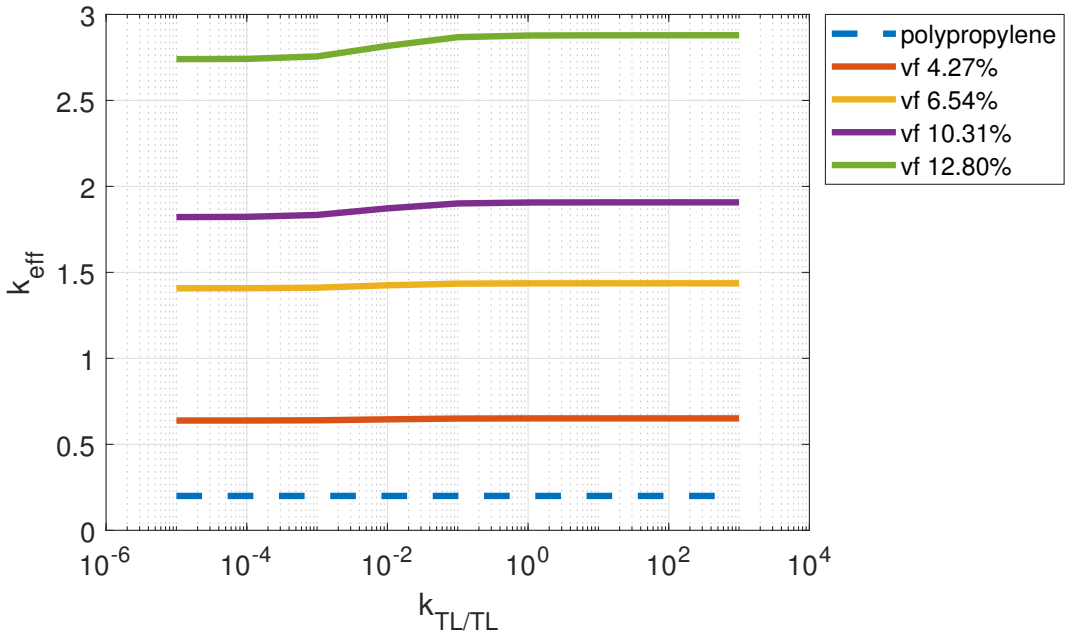
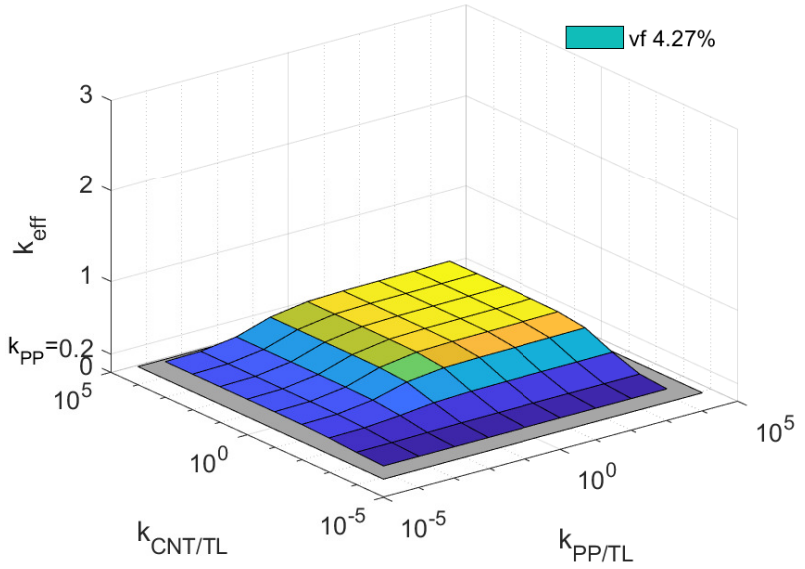
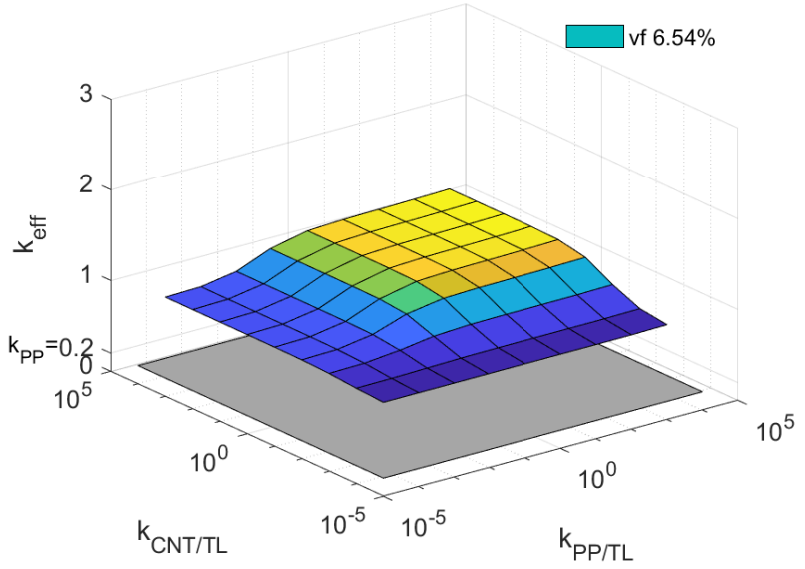


Figure 3.15: Parametric investigation on the effect of TL-TL interfacial conductance on effective conductivity of the composite. Material properties: $k_{CNT} = 2000\frac{W}{mK}$, $k_{PP} = 0.20\frac{W}{mK}$, $k_{TL} = 0.30\frac{W}{mK}$, $k_{PP/TL} = 0.25\frac{W}{m^2K}$, $k_{CNT/TL} = 1000\frac{W}{m^2K}$.

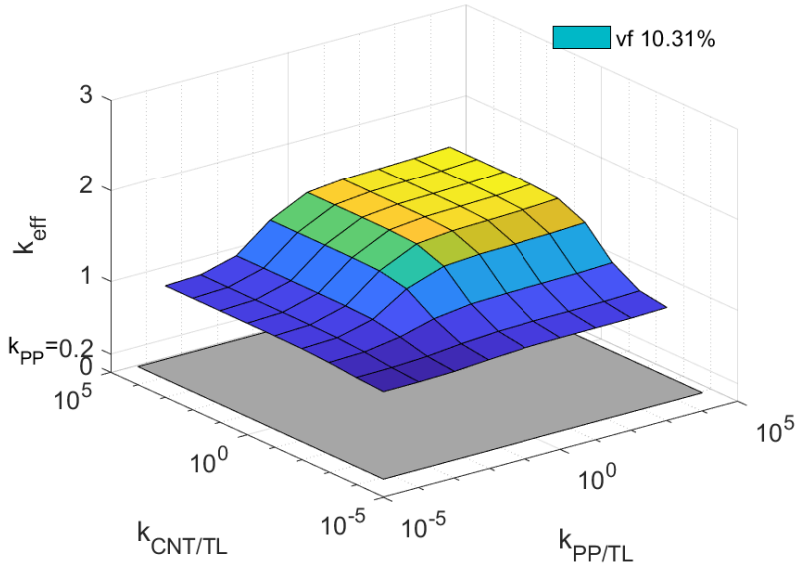


(a) Effective conductivity of RVE with volume fraction 4.27%

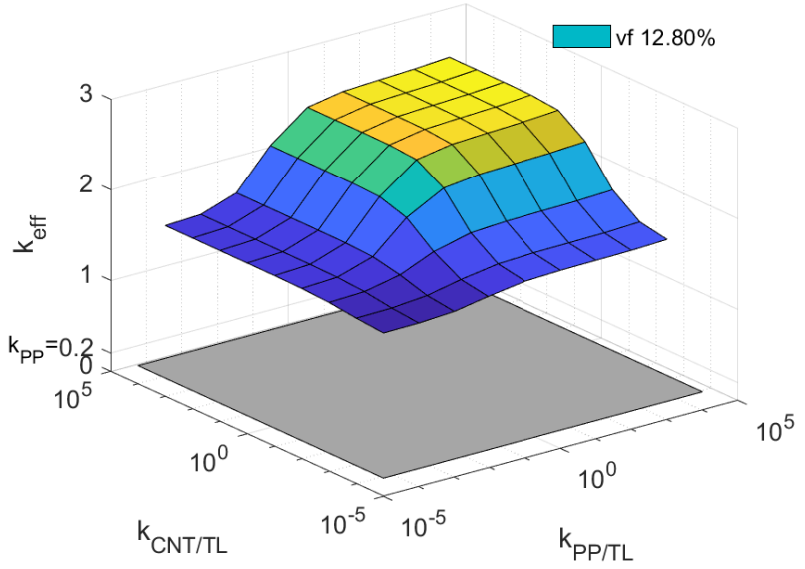


(b) Effective conductivity of RVE with volume fraction 6.54%

Figure 3.16: Parametric investigation on the effect of PP-TL and CNT-TL interfacial conductivities on the effective conductivity of the composite. Material properties: $k_{CNT} = 2000\frac{W}{mK}$, $k_{PP} = 0.20\frac{W}{mK}$, $k_{TL} = 0.30\frac{W}{mK}$, $k_{TL/TL} = 1000\frac{W}{m^2K}$.



(c) Effective conductivity of RVE with volume fraction 10.31%



(d) Effective conductivity of RVE with volume fraction 12.80%

Figure 3.16: Parametric investigation on the effect of PP-TL and CNT-TL interfacial conductivities on the effective conductivity of the composite. Material properties: $k_{CNT} = 2000\frac{W}{mK}$, $k_{PP} = 0.20\frac{W}{mK}$, $k_{TL} = 0.30\frac{W}{mK}$, $k_{TL/TL} = 1000\frac{W}{m^2K}$.

3.5 Application 4: 3D polymer - carbon nanotube composite

In this application, the XFEM model of the microstructure, which was elaborated in chapter 2, will be employed to investigate the thermal conductivity of CNT reinforced polyethylene (PE). In contrast to the previous example, in this case, a 3D XFEM model is used and the interface conductance is calibrated using experimental data.

3.5.1 Model calibration - Inference of the conductance between CNTs and polymers

In particular, the effective thermal conductivity of the composite will be calculated from an analysis of multiple SVEs of the micro structure, for different CNT concentrations and configurations. For an accurate estimation of the material's effective conductivity, it is of paramount importance to know the interface conductance, denoted as $\tilde{k}$, between the PE matrix and the CNTs. To infer $\tilde{k}$, a set of experimental measurements of the effective conductivity of CNT-reinforced PE provided by Konstantopoulos et al. (2021), will be utilized. In this work, specimens of the composite were studied, for the cases of 1%, 5% and 15% wt of CNTs in the parent material. The CNTs were synthesized by catalytic Chemical Vapor Deposition on a vertical setup, having an average diameter of $168 \pm 56\ nm$, while their length exceeded 10 μm, classifying them as "long". Masterbatches of CNTs were produced with polythelene glycol (PEG) in a $1:1$ weight ratio between CNTs and PEG, in order to ensure better dispersion of CNTs into PE. PEG however makes interfaces complex as a third phase in the composite. The experimental results regarding the composite's effective conductivity are summarized in table 3.1.

Reference samples	Effective conductivity $\left(\frac{W}{mK}\right)$			
	without CNTs	1 wt% CNTs	5 wt% CNTs	15 wt% CNTs
PE	0.250			
PE + 1 wt% PEG	0.229	0.416		
PE + 5 wt% PEG	0.219		0.403	
PE + 15 wt% PEG	0.210			0.422

Table 3.1: Effective thermal conductivity from experimental measurements in Konstantopoulos et al. (2021).

By denoting $\hat{k}_{eff}$ the composite's effective conductivity obtained by applying the homogenization scheme to the XFEM model, then this quantity is parametrized by the weight ratio $wt\%$ of the CNTs in the parent material, the interface conductance $\tilde{k}$, and a vector

of random parameters $\boldsymbol{\theta}$ which affect the architecture of CNTs. In mathematical notation, $\hat{k}_{eff} := \hat{k}_{eff}(wt\%,\ \tilde{k},\ \boldsymbol{\theta})$. To account for the randomness $\boldsymbol{\theta}$ in the material configuration, this work opts to analyze a number N of SVEs of the microstructure for different geometry realizations $\boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_N$ and extract average values of the effective conductivity, k_{eff} given by

$$k_{eff}(wt\%, \tilde{k}) = \frac{\sum_{i=1}^{N} \hat{k}_{eff}(wt\%,\ \tilde{k},\ \boldsymbol{\theta}_i)}{N} \tag{3.8}$$

which is considered to be representative for the composite. For the generation of the SVEs, a geometry generator was employed, where a $10nm \times 10nm \times 10nm$ polymer matrix was created, in which CNTs were added so as to attain a prescribed weight fraction. The CNTs were modeled as cylinders, randomly positioned and oriented inside the PE matrix, having length of 10 μm and diameter of 168 nm. It was also assumed that each CNT's center of mass follows a uniform distribution $\mathcal{U}([0, 10]^3)$ (nm) and periodic boundary conditions were applied. More specifically, the implementation aspects of the 3D SVE generator are the following:

1. The coordinates (x_i^c, y_i^c, z_i^c) of the center of mass of the i-th CNT cylinder are considered to be independent random variables, each following a uniform distribution between 0 and the length of the SVE. That is, $x_i^c, y_i^c, z_i^c \sim \mathcal{U}([0, 10])$ (nm). This is chosen because we do not have any information indicating the existence of more probable CNT locations or a correlation structure for the CNT dispersion in the host material.

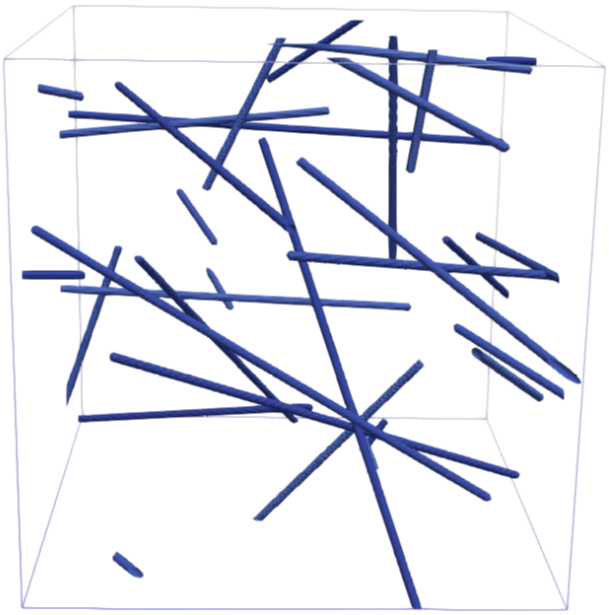
2. The orientation of the i-th CNT is characterized (in spherical coordinates) by the azimuthal angle ω_i^α and the polar angle ω_i^p , which are also considered to be random variables following uniform distributions, i.e. $\omega_i^\alpha \sim \mathcal{U}([-\pi, \pi])$ and $\omega_i^p \sim \mathcal{U}([0, \pi])$. Again, this is chosen since we do not have any evidence indicating towards a preferred CNT orientation during the manufacturing process.

3. A prescribed number of CNTs is added to the cubic volume element so as to achieve the target weight fraction. The positioning of the CNTs is performed in a serial manner, where for each new CNT, a random realization of $[x^c, y^c, z^c, \omega^\alpha, \omega^p]$ is drawn from the corresponding probability distributions. Two cases are examined here:

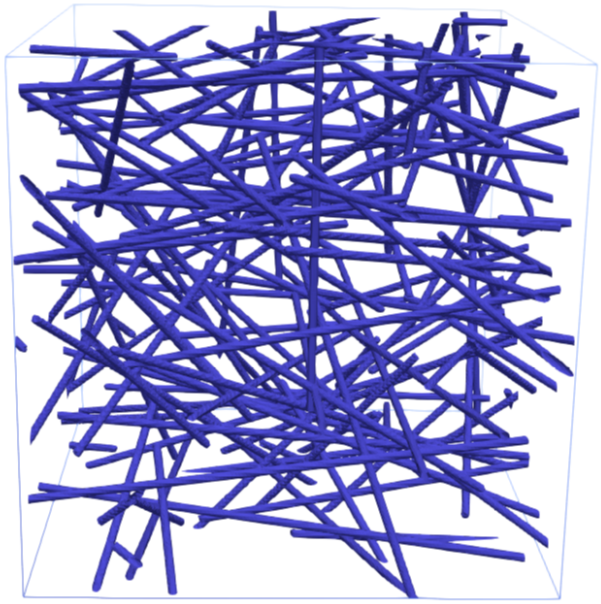
   a) If the i-th CNT intersects with a face of the cube, then the remaining part re-enters through the opposite face (periodic boundary positions). This allows us to insert the exact number of CNTs that will produce the desired weight fraction in the host material.

   b) If the i-th CNT penetrates a pre-existing one then its positioning process is repeated (Song et al., 2016).

In figure 3.17, three realizations of SVEs are illustrated for the cases of 1%wt of CNTs (figure 3.17a), 5%wt of CNTs (figure 3.17b) and 15%wt of CNTs (figure 3.17c).



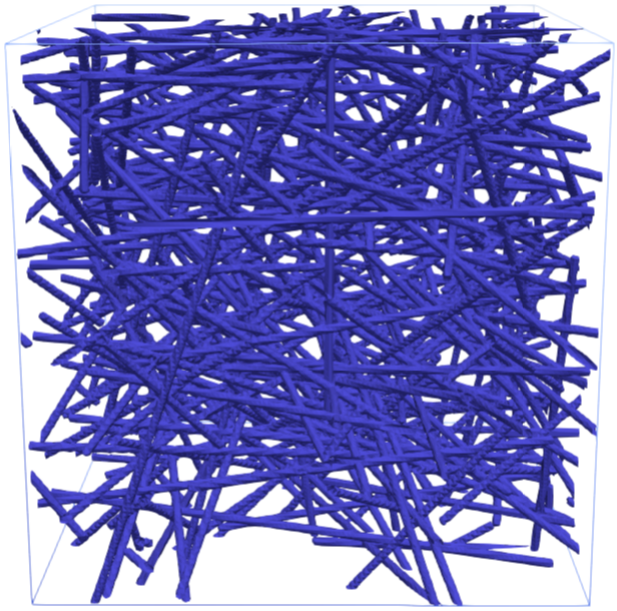
(a)



(b)

Figure 3.17: Statistical volume elements for different wt% of CNTs: (a) 1 $wt\%$, (b) 5 $wt\%$, (c) 15 $wt\%$

Based on the measurements of Table 3.1, the numerical model's agreement with the experimental measurements, is expressed through the error norm:



(c)

Figure 3.17: Statistical volume elements for different wt% of CNTs: (a) 1 $wt\%$, (b) 5 $wt\%$, (c) 15 $wt\%$

$$\epsilon(\tilde{k}) = \frac{\sqrt{\Big(k_{eff}(1wt\%, \tilde{k}) - 0.416\Big)^2 + \Big(k_{eff}(5wt\%, \tilde{k}) - 0.403\Big)^2 + \Big(k_{eff}(15wt\%, \tilde{k}) - 0.422\Big)^2}}{\sqrt{0.416^2 + 0.403^2 + 0.422^2}} \tag{3.9}$$

To obtain accurate statistical estimates of k_{eff}, for each $wt\%$ and each value of conductance $\tilde{k}$ in the parametric investigation, a number of $N = 100$ SVEs were analyzed. The results of this investigation are shown in figure 3.18, which plots the effective conductivity of the composite for varying values of $\tilde{k}$. Upon inspection of this figure, it quickly becomes evident that $\tilde{k}$ plays a major role in the composite's effective conductivity, since for values greater than 1 $\frac{W}{m^2K}$ a rapid increase in k_{eff} can be reported. This result is more prominent as the $wt\%$ of the CNTs increases, as expected.

Subsequently, the error ϵ given by equation (3.9) is plotted as a function of the conductance $\tilde{k}$ in figure 3.19. This figure indicates that the best agreement between the experimental measurements and the numerical predictions can be attained for $\tilde{k} = 0.73\ \frac{W}{m^2K}$, which results in an error $\epsilon = 7.41\%$.

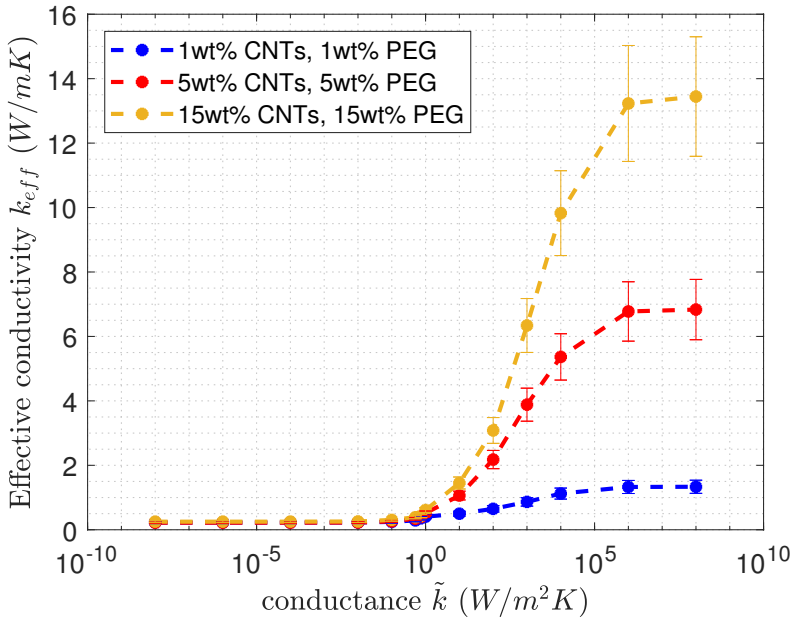


Figure 3.18: Parametric investigation for different $wt\%$ of CNTs and PEG. The markers indicate the mean value of k_{eff} and the length of the error bars is one standard deviation.

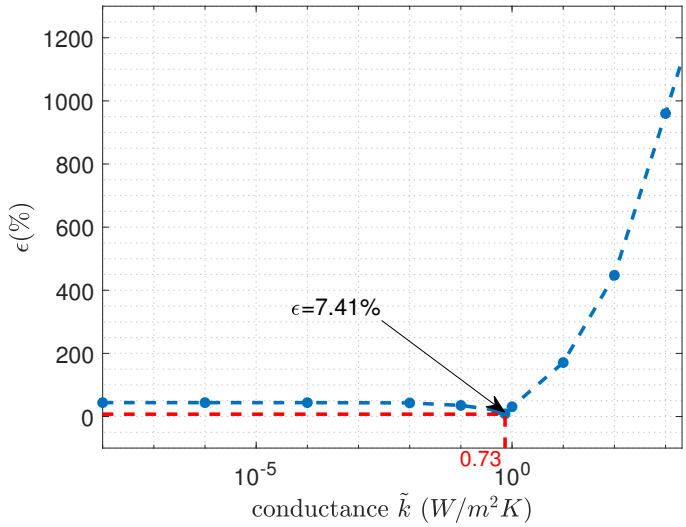
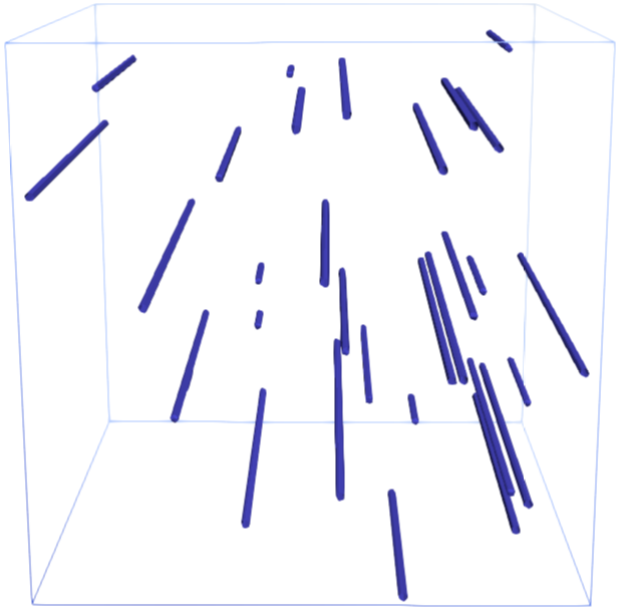


Figure 3.19: Error of effective conductivity k_{eff} as a function of interface conductance $\tilde{k}$

3.5.2 Theoretical investigation on the effective conductivity for optimal microstructural morphologies

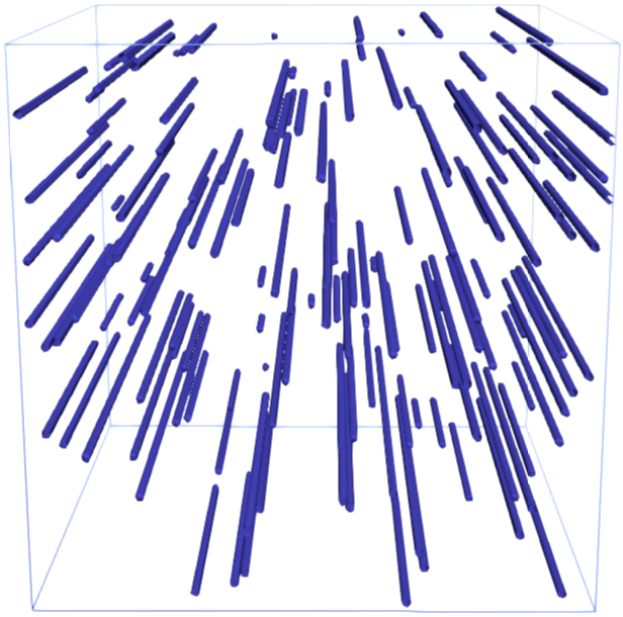
In this section, a numerical investigation will be conducted to assess the potential CNT-reinforced PE has as a conductive material, based on idealized microstructural morphologies, even if these are beyond current manufacturing capabilities. The focus in the section will be placed on: (i) the CNT weight fraction, (ii) the CNT orientation, (iii) the CNT aspect ratio and (iv) the interface conductance.

Initially, the effective conductivity of CNT reinforced PE is studied for varying weight ratios $wt\%$, assigning to $\tilde{k}$ the value 0.73 $\frac{W}{m^2K}$, found in the previous section. The investigation is performed for arbitrarily inserted CNTs in the polymer, as well as perfectly aligned CNTs. This way, upper estimates of k_{eff} can be obtained. In figure 3.20, the SVEs made up of horizontally aligned CNTs are depicted, as opposed to figure 3.17 which plotted SVEs with randomly oriented CNTs.

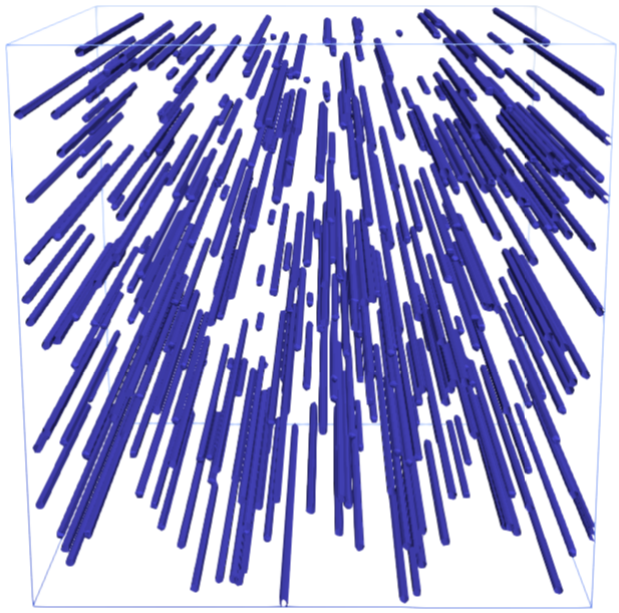


(a)

Figure 3.20: Statistical volume elements for different wt% of of horizontally aligned CNTs: (a) 1 $wt\%$, (b) 5 $wt\%$, (c) 15 $wt\%$



(b)



(c)

Figure 3.20: Statistical volume elements for different wt% of of horizontally aligned CNTs: (a) 1 $wt\%$, (b) 5 $wt\%$, (c) 15 $wt\%$

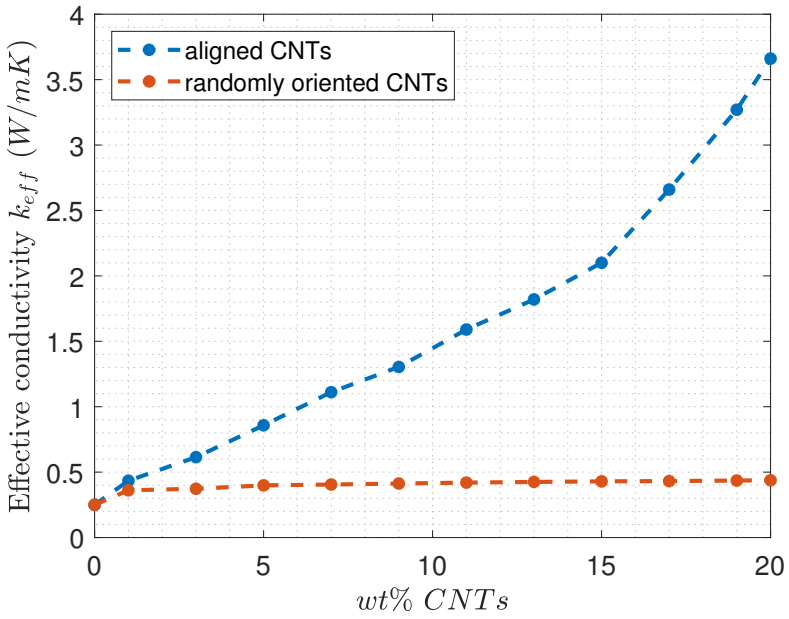
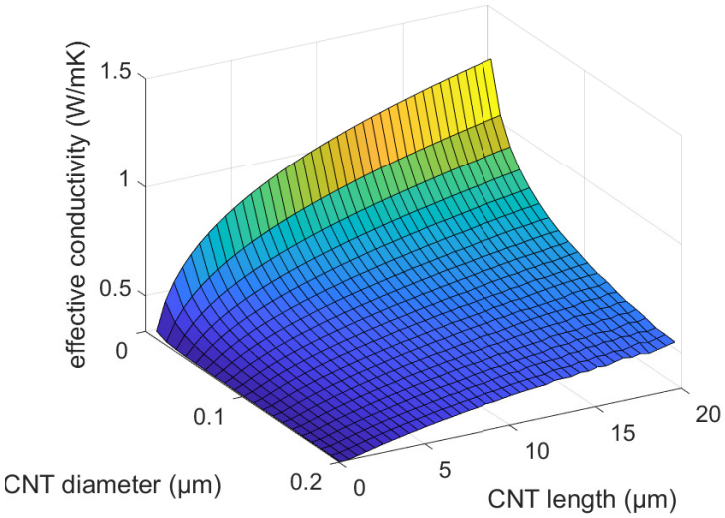


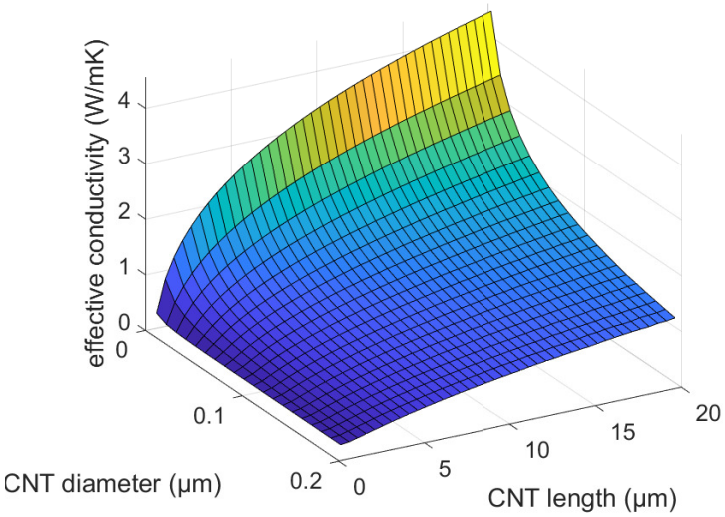
Figure 3.21: Comparison in the effective conductivity between randomly oriented CNTs and perfectly aligned

Figure 3.21 provides a comparison in the effective conductivity of the composite between randomly oriented CNTs and aligned CNTs, where a noticeable increase in conductivity can be reported for the latter configuration. Next, the role of the aspect ratio of the CNTs in the effective conductivity is examined. Figure 3.22 displays the effective conductivity of the composite for perfectly aligned CNTs, as a function of the CNT length and diameter. The conclusion is drawn that by increasing the aspect ratio of the CNTs, a significantly more conductive material can be produced.

Lastly, an idealized material is considered with perfectly aligned CNTs and an aspect ratio of 2000 (diameter 10 nm and length 20 μm), which is studied for varying interface conductance $\tilde{k}$ values. The reason for this investigation stems from the fact that the conductance between CNTs and PE could potentially be increased by coating the CNTs with other conductive materials (Wang et al., 2020), even though this area is not well researched yet. Figure 3.23 depicts the results of this analysis for $\tilde{k} = 0.73, 10, 100, 1000$, where a drastic improvement in k_{eff} can be reported as $\tilde{k}$ increases. Overall, the conclusions drawn from these analyses suggest that CNT-reinforced PE is not expected to have high conductivity values when using conventional manufacturing techniques. Nevertheless, by optimizing the CNT orientation and aspect ratio, as well as by increasing the conductance at the interface, a highly conductive material can be generated with great potential for industrial applications.

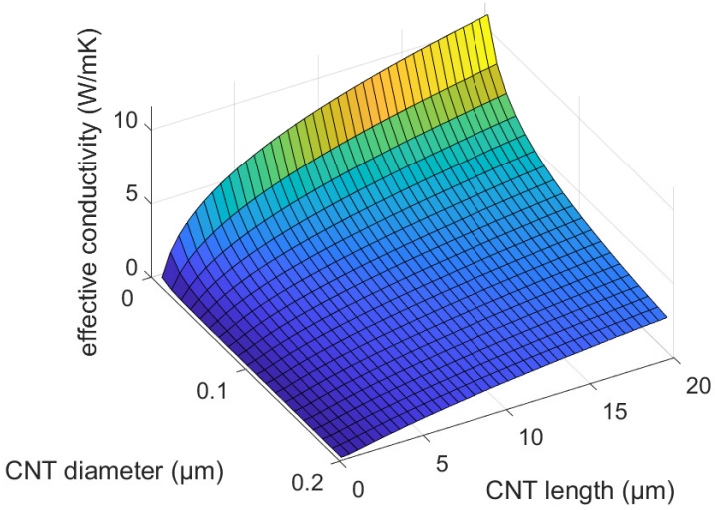


(a) 1%wt CNTs



(b) 5%wt CNTs

Figure 3.22: Effective conductivity k_{eff} as a function of the CNT length and diameter for various wt%, for the case of perfectly aligned CNTs



(c) 15%wt CNTs

Figure 3.22: Effective conductivity k_{eff} as a function of the CNT length and diameter for various wt%, for the case of perfectly aligned CNTs

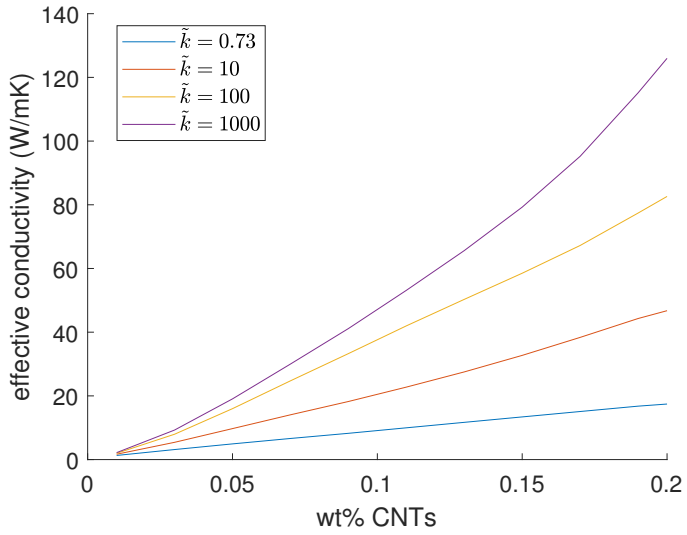


Figure 3.23: Effective conductivity k_{eff} for perfectly aligned CNTs with an aspect ratio of 2000, for different values of interface conductance $\tilde{k}$

3.6 Conclusions

This chapter investigated the numerical model based on XFEM and LSM, which was presented in chapter 2, in a series of examples involving heat transfer in composite materials with complex geometries. After validating the method's accuracy with data from the literature, it was employed for the study of polymer - carbon nanotube (CNT) composites. The thermal resistance between polyethylene (PE) and CNTs was estimated, by performing parameter inference using experimental measurements of the effective thermal conductivity of the composite. From the investigation of the role of CNT orientation, aspect ratio and interfacial resistance on the effective conductivity, the following conclusions can be drawn:

1. The resistance at the interface between PE and CNTs is quite high and as a result the effective conductivity that the composite can reach is far below theoretical expectations.

2. The theoretical investigations demonstrate that the most critical parameter for the production of a highly conductive composite is the resistance at the interface between CNTs and PE. Reduced values of thermal resistance, possibly achieved by coating CNTs with other conductive materials, can indeed lead to a composite with the desired properties for thermal applications.

3. In addition, microstructural morphologies consisting of perfectly aligned CNTs with high aspect ratio can lead to a noticeable improvement in the effective conductivity.

Chapter 4

3D crack propagation analysis

The aim of this chapter is to review crack propagation analysis using the Extended Finite Element Method (XFEM). The Boundary Value Problem (BVP) for the case of Linear Elastic Fracture Mechanics (LEFM) is stated and then solved in its weak form using XFEM. Furthermore, the geometric representation of 3D cracks, as well as their interaction with XFEM are presented. Finally, the propagation direction and increment of an existing crack are predicted.

4.1 Modeling crack propagation with XFEM

4.1.1 Strong form

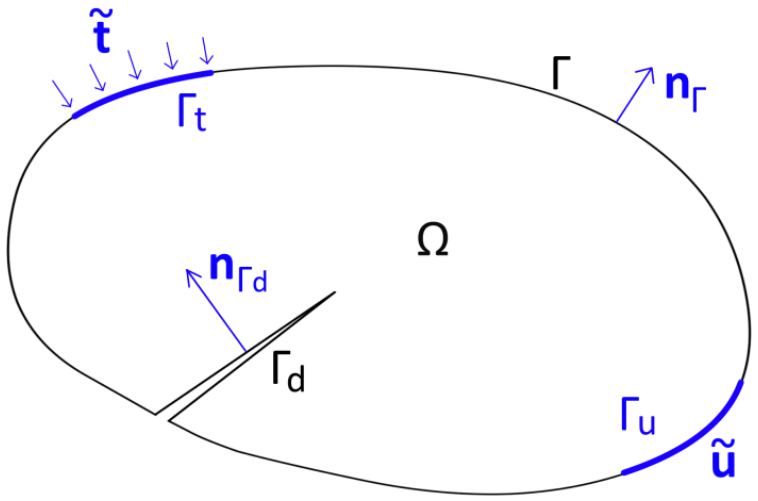


Figure 4.1: Domain with an edge crack

Let Ω be a domain containing a crack Γ_d, as illustrated in figure 4.1. Its external boundary of $\partial\Omega$ has an outward normal vector $\boldsymbol{n}_\Gamma$ and is divided into complementary parts $\partial\Omega_u$ and $\partial\Omega_t$, such that $\partial\Omega = \partial\Omega_u \cup \partial\Omega_t$. Dirichlet and Neumann boundary conditions are imposed on the boundaries Γ_u and Γ_t, respectively

$$\begin{aligned} \boldsymbol{u} &= \widetilde{\boldsymbol{u}} \text{ on } \Gamma_u \\ \boldsymbol{\sigma} \cdot \boldsymbol{n}_\Gamma &= \widetilde{\boldsymbol{t}} \text{ on } \Gamma_t \end{aligned} \tag{4.1}$$

where

- $\boldsymbol{u} = \boldsymbol{u}(\boldsymbol{x})$ is the displacement field

$$\boldsymbol{u} = \begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{4.2}$$

- $\boldsymbol{\sigma} = \boldsymbol{\sigma}(\boldsymbol{u})$ is the Cauchy stress tensor

$$\boldsymbol{\sigma} = \begin{bmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{zz} \\ \tau_{xy} \\ \tau_{yz} \\ \tau_{zx} \end{bmatrix} \tag{4.3}$$

- $\widetilde{\boldsymbol{u}}$ are prescribed displacements on Γ_u
- $\widetilde{\boldsymbol{t}}$ are prescribed tractions on Γ_t

The surface of the crack is assumed to be traction-free:

$$\boldsymbol{\sigma} \cdot \boldsymbol{n}_{\Gamma_d} = \boldsymbol{\sigma}^+ \cdot \boldsymbol{n}_{\Gamma_d} = \boldsymbol{\sigma}^- \cdot \boldsymbol{n}_{\Gamma_d} = \boldsymbol{0} \text{ on } \Gamma_d \tag{4.4}$$

where $\boldsymbol{n}_{\Gamma_d}$ is the normal vector of the crack Γ_d and $\boldsymbol{\sigma}^+$, $\boldsymbol{\sigma}^-$ is the stress tensor evaluated on each side of Γ_d. Assuming that the displacements remain small, the kinematic equations consist of the strain-displacement relation

$$\boldsymbol{\epsilon} = \boldsymbol{\epsilon}(\boldsymbol{u}) = \nabla_{sym}\boldsymbol{u} = \frac{\nabla\boldsymbol{u} + (\nabla\boldsymbol{u})^T}{2} \tag{4.5}$$

where $\nabla_{sym}\boldsymbol{u}$ denotes the symmetric part of the $\nabla\boldsymbol{u}$ tensor and $\boldsymbol{\epsilon}$ is the strain tensor

$$\boldsymbol{\epsilon} = \begin{bmatrix} \dfrac{\partial u}{\partial x} \\ \dfrac{\partial v}{\partial y} \\ \dfrac{\partial w}{\partial z} \\ \dfrac{\partial u}{\partial y} + \dfrac{\partial v}{\partial x} \\ \dfrac{\partial v}{\partial z} + \dfrac{\partial w}{\partial y} \\ \dfrac{\partial u}{\partial z} + \dfrac{\partial w}{\partial x} \end{bmatrix} \tag{4.6}$$

For a linear elastic material, the constitutive law is

$$\boldsymbol{\sigma} = \boldsymbol{\sigma}(\boldsymbol{u}) = \boldsymbol{C} : \boldsymbol{\epsilon}(\boldsymbol{u}) \tag{4.7}$$

where $\boldsymbol{C}$ is the constitutive tensor. For a 3D elastic isotropic material, this 4th-order tensor is constant and equal to

$$\boldsymbol{C} = \frac{E}{(1+v)(1-2v)} \begin{bmatrix} 1-v & v & v & 0 & 0 & 0 \\ v & 1-v & v & 0 & 0 & 0 \\ v & v & 1-v & 0 & 0 & 0 \\ 0 & 0 & 0 & \dfrac{1-2v}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & \dfrac{1-2v}{2} & 0 \\ 0 & 0 & 0 & 0 & 0 & \dfrac{1-2v}{2} \end{bmatrix} \tag{4.8}$$

where E is Young's modulus and v is Poisson's ratio. If $\boldsymbol{b} = \boldsymbol{b}(\boldsymbol{x})$ is the body force per unit volume applied to the entire Ω, then the equilibrium equation is

$$\nabla \cdot \boldsymbol{\sigma} + \boldsymbol{b} = \boldsymbol{0} \text{ in } \Omega \tag{4.9}$$

where the divergence of the stress tensor $\nabla \cdot \boldsymbol{\sigma}$ is a vector defined as

$$\nabla \cdot \boldsymbol{\sigma} = \frac{\partial \sigma_{ij}}{\partial x_j} \hat{\boldsymbol{e}}_i = \begin{bmatrix} \dfrac{\partial \sigma_{xx}}{\partial x} + \dfrac{\partial \sigma_{xy}}{\partial y} + \dfrac{\partial \sigma_{xz}}{\partial z} \\ \dfrac{\partial \sigma_{yx}}{\partial x} + \dfrac{\partial \sigma_{yy}}{\partial y} + \dfrac{\partial \sigma_{yz}}{\partial z} \\ \dfrac{\partial \sigma_{zx}}{\partial x} + \dfrac{\partial \sigma_{zy}}{\partial y} + \dfrac{\partial \sigma_{zz}}{\partial z} \end{bmatrix} \tag{4.10}$$

Given the equilibrium equation, kinematic relations, constitutive law and boundary conditions on the external boundary and the crack, the strong form of the BVP can be posed as: " Find a vector function $\boldsymbol{u}(\boldsymbol{x})$ for the displacement field, so that the following equations are satisfied:"

$$\begin{aligned} \nabla \cdot \boldsymbol{\sigma} + \boldsymbol{b} &= \mathbf{0} \text{ in } \Omega \\ \boldsymbol{\sigma} &= \boldsymbol{C} : \boldsymbol{\epsilon}(\boldsymbol{u}) \\ \boldsymbol{\epsilon}(\boldsymbol{u}) &= \frac{\nabla \boldsymbol{u} + (\nabla \boldsymbol{u})^T}{2} \\ \boldsymbol{u} &= \widetilde{\boldsymbol{u}} \text{ on } \Gamma_u \\ \boldsymbol{\sigma} \cdot \boldsymbol{n} &= \widetilde{\boldsymbol{t}} \text{ on } \Gamma_t \\ \boldsymbol{\sigma} \cdot \boldsymbol{n}_{\Gamma_d} &= \mathbf{0} \text{ on } \Gamma_d \end{aligned} \tag{4.11}$$

4.1.2 Divergence theorem in cracked domain

The divergence theorem is necessary to derive the weak form that is the basis of any finite element formulation. Given a continuous domain Ω, with boundary Γ and a continuous vector field $\boldsymbol{F}$, the integration of its divergence over the domain is equivalent to the integration of the field itself over the boundary:

$$\int_{\Omega} \nabla \cdot \boldsymbol{F} d\Omega = \int_{\Gamma} \boldsymbol{F} \cdot \boldsymbol{n}_{\Gamma} d\Gamma \tag{4.12}$$

where the divergence operator is used:

$$\nabla \cdot \boldsymbol{F} = \text{div}\boldsymbol{F} = \begin{bmatrix} \frac{\partial}{\partial x} & \frac{\partial}{\partial y} & \frac{\partial}{\partial z} \end{bmatrix} \cdot \begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} = \frac{\partial F_x}{\partial x} + \frac{\partial F_y}{\partial y} + \frac{\partial F_z}{\partial z} \tag{4.13}$$

In XFEM the domain Ω is discontinuous, but the divergence theorem can be applied by splitting Ω into continuous subdomains. Figure 4.2 depicts a domain Ω that is divided into two subdomains Ω^+ and Ω^- by a discontinuity Γ_d. The external boundaries of domains Ω^+ and Ω^- are denoted as Γ^+ and Γ^-, with their outward unit normal vectors being $\boldsymbol{n}_{\Gamma^+}$ and $\boldsymbol{n}_{\Gamma^-}$ respectively. The internal boundary Γ_d, with the unit normal vector $\boldsymbol{n}_{\Gamma_d}$ oriented towardsΩ^+, consists of the actual discontinuity Γ_{d1} and its extension Γ_{d2} with the unit normal vector $\boldsymbol{n}_{\Gamma_{d2}}$, both oriented towards$\Omega^+$. The divergence theorem can now be applied to Ω^+ and Ω^-, since $\boldsymbol{F}$ is continuous inside them:

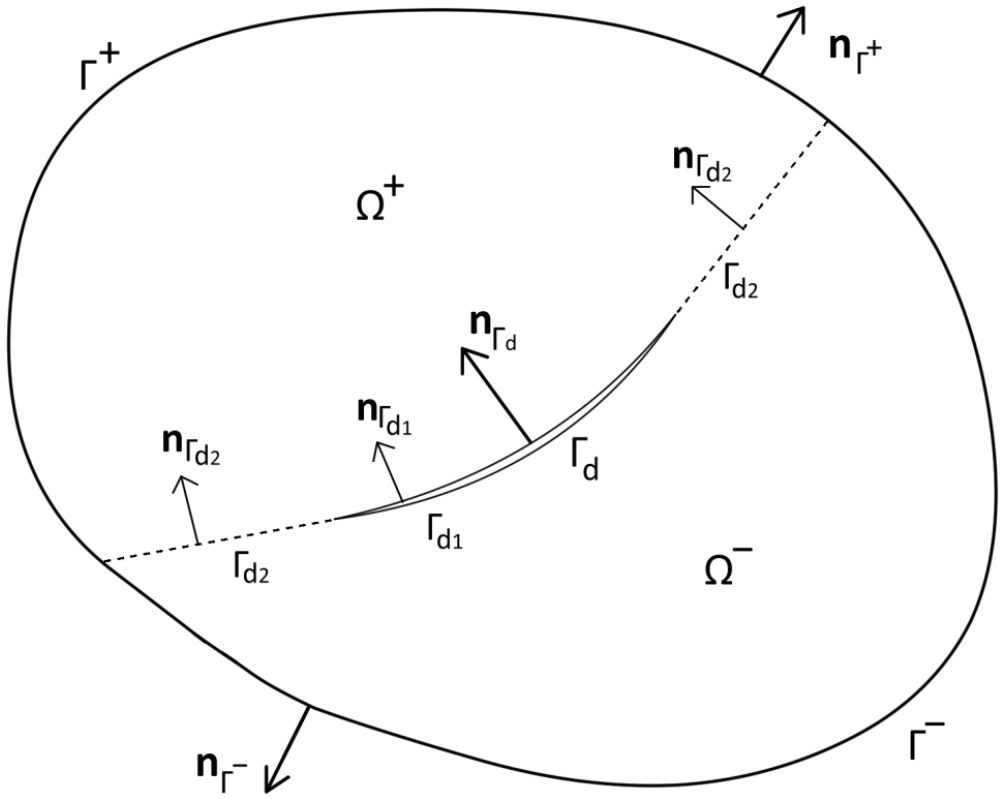


Figure 4.2: A domain Ω with an interior discontinuity Γ_d

$$\int_{\Omega^+} \nabla \cdot \boldsymbol{F} d\Omega = \int_{\Gamma^+} \boldsymbol{F} \cdot \boldsymbol{n}_{\Gamma^+} d\Gamma + \int_{\Gamma^+_{d1}} \boldsymbol{F}^+ \cdot (-\boldsymbol{n}_{\Gamma_{d1}})\, d\Gamma + \int_{\Gamma^+_{d2}} \boldsymbol{F} \cdot (-\boldsymbol{n}_{\Gamma_{d2}})\, d\Gamma \tag{4.14a}$$

$$\int_{\Omega^-} \nabla \cdot \boldsymbol{F} d\Omega = \int_{\Gamma^-} \boldsymbol{F} \cdot \boldsymbol{n}_{\Gamma^-} d\Gamma + \int_{\Gamma^-_{d1}} \boldsymbol{F}^- \cdot \boldsymbol{n}_{\Gamma_{d1}} d\Gamma + \int_{\Gamma^-_{d2}} \boldsymbol{F} \cdot \boldsymbol{n}_{\Gamma_{d2}} d\Gamma \tag{4.14b}$$

where the values of $\boldsymbol{F}$ along the two sides of the discontinuity Γ^+_{d1} and Γ^-_{d1} are different and denoted as $\boldsymbol{F}^+$ and $\boldsymbol{F}^-$ respectively. Since the two subdomains span the whole domain $\Omega = \Omega^+ \cup \Omega^-$ and $\Gamma = \Gamma^+ \cup \Gamma^-$ and by noticing that the contour integrals along the extension of the discontinuity Γ_{d2} in equation (4.14) cancel out:

$$\begin{aligned}\int_{\Omega} \nabla \cdot \boldsymbol{F} d\Omega &= \int_{\Omega^+} \nabla \cdot \boldsymbol{F} d\Omega + \int_{\Omega^-} \nabla \cdot \boldsymbol{F} d\Omega \\ &= \int_{\Gamma^+} \boldsymbol{F} \cdot \boldsymbol{n}_{\Gamma^+} d\Gamma + \int_{\Gamma^-} \boldsymbol{F} \cdot \boldsymbol{n}_{\Gamma^-} d\Gamma - \int_{\Gamma^+_{d1}} \boldsymbol{F}^+ \cdot \boldsymbol{n}_{\Gamma_{d1}} d\Gamma \\ &\qquad + \int_{\Gamma^-_{d1}} \boldsymbol{F}^- \cdot \boldsymbol{n}_{\Gamma_{d1}} d\Gamma - \int_{\Gamma^+_{d2}} \boldsymbol{F} \cdot \boldsymbol{n}_{\Gamma_{d2}} d\Gamma + \int_{\Gamma^-_{d2}} \boldsymbol{F} \cdot \boldsymbol{n}_{\Gamma_{d2}} d\Gamma \\ &= \int_{\Gamma} \boldsymbol{F} \cdot \boldsymbol{n}_{\Gamma} d\Gamma - \int_{\Gamma_d} \left(\boldsymbol{F}^+ - \boldsymbol{F}^- \right) \cdot \boldsymbol{n}_{\Gamma_d} d\Gamma \end{aligned} \tag{4.15}$$

By defining the jump of the vector field across Γ_d as $[\![\boldsymbol{F}]\!] = \boldsymbol{F}^+ - \boldsymbol{F}^-$ the previous equation becomes

$$\int_{\Omega} \nabla \cdot \boldsymbol{F} d\Omega = \int_{\Gamma} \boldsymbol{F} \cdot \boldsymbol{n}_{\Gamma} d\Gamma - \int_{\Gamma_d} [\![\boldsymbol{F}]\!] \cdot \boldsymbol{n}_{\Gamma_d} d\Gamma \tag{4.16}$$

For problems where the domain contains N_d discontinuities, the following should be used instead

$$\int_{\Omega} \nabla \cdot \boldsymbol{F} d\Omega = \int_{\Gamma} \boldsymbol{F} \cdot \boldsymbol{n}_{\Gamma} d\Gamma - \sum_{i=1}^{N_d} \int_{\Gamma_{d_i}} [\![\boldsymbol{F}_i]\!] \cdot \boldsymbol{n}_{\Gamma_{d_i}} d\Gamma \tag{4.17}$$

4.1.3 Weak form

The weak form of the BVP is posed as: Find a trial function $\boldsymbol{u}$ that belongs to the function space

$$\mathbb{U} = \{\boldsymbol{v} \in \mathbb{H} : \boldsymbol{v} = \widetilde{\boldsymbol{u}} \text{ on } \Gamma_u \ , \ \boldsymbol{v} \text{ discontinuous on } \Gamma_d\} \tag{4.18}$$

such that

$$\int_{\Omega} \boldsymbol{\epsilon}(\boldsymbol{w}) : \boldsymbol{C} : \boldsymbol{\epsilon}(\boldsymbol{u}) d\Omega = \int_{\Omega} \boldsymbol{w} \cdot \boldsymbol{b} \ d\Omega + \int_{\Gamma_t} \boldsymbol{w} \cdot \widetilde{\boldsymbol{t}} \ d\Gamma \tag{4.19}$$

for all test functions $\boldsymbol{w}$ belonging to the space

$$\mathbb{U}_0 = \{\boldsymbol{v} \in \mathbb{H} : \boldsymbol{v} = \boldsymbol{0} \text{ on } \Gamma_u \ , \ \boldsymbol{v} \text{ discontinuous on } \Gamma_d\} \tag{4.20}$$

where $\mathbb{H}$ is an H^1 Hilbert space of functions that are smooth in Ω, but discontinuous across Γ_d. To derive the weak form from the strong form, equation (4.9), is multiplied with an arbitrary test function $\boldsymbol{w}$, integrated and then the product rule of differentiation is applied:

$$\begin{aligned} \nabla \cdot \boldsymbol{\sigma}(\boldsymbol{u}) + \boldsymbol{b} &= \mathbf{0} \\ \iff \int_{\Omega} \boldsymbol{w} \cdot (\nabla \cdot \boldsymbol{\sigma}(\boldsymbol{u}) + \boldsymbol{b})\, d\Omega &= 0 \\ \iff \int_{\Omega} \boldsymbol{w} \cdot (\nabla \cdot \boldsymbol{\sigma}(\boldsymbol{u}))\, d\Omega + \int_{\Omega} \boldsymbol{w} \cdot \boldsymbol{b}\, d\Omega &= 0 \\ \iff \int_{\Omega} \nabla \cdot (\boldsymbol{w} \cdot \boldsymbol{\sigma}(\boldsymbol{u}))\, d\Omega - \int_{\Omega} \nabla \boldsymbol{w} : \boldsymbol{\sigma}(\boldsymbol{u})\, d\Omega + \int_{\Omega} \boldsymbol{w} \cdot \boldsymbol{b}\, d\Omega &= 0 \end{aligned} \tag{4.21}$$

The discontinuous Divergence theorem of equation (4.16) can now be applied on the first integral of equation (4.21)

$$\begin{aligned} \int_{\Omega} \nabla \cdot (\boldsymbol{w} \cdot \boldsymbol{\sigma}(\boldsymbol{u}))\, d\Omega &= \int_{\Gamma} \boldsymbol{w} \cdot \boldsymbol{\sigma}(\boldsymbol{u}) \cdot \boldsymbol{n}_{\Gamma}\, d\Gamma - \int_{\Gamma_d} [\![\boldsymbol{w} \cdot \boldsymbol{\sigma}(\boldsymbol{u})]\!] \cdot \boldsymbol{n}_{\Gamma_d}\, d\Gamma \\ &= \int_{\Gamma_t} \boldsymbol{w} \cdot \boldsymbol{\sigma}(\boldsymbol{u}) \cdot \boldsymbol{n}_{\Gamma}\, d\Gamma + \int_{\Gamma_u} \boldsymbol{w} \cdot \boldsymbol{\sigma}(\boldsymbol{u}) \cdot \boldsymbol{n}_{\Gamma}\, d\Gamma \\ &\quad - \int_{\Gamma_d} \left(\boldsymbol{w}^{+} \cdot \boldsymbol{\sigma}^{+}(\boldsymbol{u}) \cdot \boldsymbol{n}_{\Gamma_d} - \boldsymbol{w}^{-} \cdot \boldsymbol{\sigma}^{-}(\boldsymbol{u}) \cdot \boldsymbol{n}_{\Gamma_d}\right) d\Gamma \end{aligned} \tag{4.22}$$

By imposing the boundary conditions of equations (4.1, 4.4), the last two integrals are eliminated from the previous equation

$$\int_{\Omega} \nabla \cdot (\boldsymbol{w} \cdot \boldsymbol{\sigma}(\boldsymbol{u}))d\Omega = \int_{\Gamma_t} \boldsymbol{w} \cdot \tilde{\boldsymbol{t}} d\Gamma \tag{4.23}$$

The gradient of the test vector field can be written as

$$\nabla \boldsymbol{w} = \frac{\nabla \boldsymbol{w} + (\nabla \boldsymbol{w})^T}{2} + \frac{\nabla \boldsymbol{w} - (\nabla \boldsymbol{w})^T}{2} = \nabla_{sym} \boldsymbol{w} + \nabla_{ant} \boldsymbol{w} \tag{4.24}$$

where $\nabla_{sym}\boldsymbol{w}$ and $\nabla_{ant}\boldsymbol{w}$ are the symmetric and anti-symmetric parts of the $\nabla \boldsymbol{w}$ tensor. Since $\nabla_{ant}\boldsymbol{w}$ is anti-symmetric and $\boldsymbol{\sigma}(\boldsymbol{u})$ is symmetric, their product is

$$\nabla_{ant} \boldsymbol{w} : \boldsymbol{\sigma}(\boldsymbol{u}) = 0 \tag{4.25}$$

Therefore the second integral of equation (4.21) becomes

$$\begin{aligned}\int_{\Omega} \nabla \boldsymbol{w} : \boldsymbol{\sigma}\ d\Omega &= \int_{\Omega} \nabla_{sym}\boldsymbol{w} : \boldsymbol{\sigma}\left(\boldsymbol{u}\right)\ d\Omega + \int_{\Omega} \nabla_{ant}\boldsymbol{w} : \boldsymbol{\sigma}\left(\boldsymbol{u}\right)\ d\Omega \\ &= \int_{\Omega} \boldsymbol{\epsilon}(\boldsymbol{w}) : \boldsymbol{\sigma}(\boldsymbol{u})\ d\Omega\end{aligned} \tag{4.26}$$

Substituting equations (4.23) and (4.26) into equation (4.21) results in the weak form

$$\begin{aligned}&\int_{\Gamma_t} \boldsymbol{w} \cdot \tilde{\boldsymbol{t}}\ d\Gamma - \int_{\Omega} \boldsymbol{\epsilon}(\boldsymbol{w}) : \boldsymbol{\sigma}(\boldsymbol{u})\ d\Omega + \int_{\Omega} \boldsymbol{w} \cdot \boldsymbol{b}\ d\Omega = 0 \\ &\Leftrightarrow \int_{\Omega} \boldsymbol{\epsilon}(\boldsymbol{w}) : \boldsymbol{C} : \boldsymbol{u}\ d\Omega = \int_{\Omega} \boldsymbol{w} \cdot \boldsymbol{b}\ d\Omega + \int_{\Gamma_t} \boldsymbol{w} \cdot \tilde{\boldsymbol{t}}\ d\Gamma\end{aligned} \tag{4.27}$$

4.1.4 XFEM enrichment

In order to model the discontinuous displacement field $\boldsymbol{u}$, XFEM enriches the polynomial approximation space of standard FEM with non-smooth enrichment functions. The finite element mesh is independent from the crack geometry and does not need to conform to it. Instead, some elements are intersected by the crack surface or the crack front, as illustrated in figure 4.3b. The nodes of elements intersected by the crack front are enriched with 4 asymptotic crack-tip enrichment functions $F_m(\boldsymbol{x})$, derived from LEFM

$$\begin{aligned}\{F_m(\boldsymbol{x})\}_{m=1}^{4} &= \{F_m(r,\theta)\}_{m=1}^{4} \\ &= \left\{\sqrt{r}\sin(\frac{\theta}{2}); \sqrt{r}\cos(\frac{\theta}{2}); \sqrt{r}\sin(\frac{\theta}{2})\sin(\theta); \sqrt{r}\cos(\frac{\theta}{2})\sin(\theta)\right\}\end{aligned} \tag{4.28}$$

where $(r,\theta) = (r(\boldsymbol{x}), \theta(\boldsymbol{x}))$ are the coordinates of a point defined in a polar system at the crack front, as shown in figure 4.3a. Furthermore, the nodes belonging to elements that are intersected by the rest of the crack surface, are enriched with the Heaviside function $H(\boldsymbol{x})$

$$H(\boldsymbol{x}) = H(\phi(\boldsymbol{x})) = \begin{cases} +1, & \phi(\boldsymbol{x}) \geq 0 \\ -1, & \phi(\boldsymbol{x}) < 0 \end{cases} \tag{4.29}$$

where $\phi(\boldsymbol{x})$ is the signed distance of a point $\boldsymbol{x}$ to the crack surface, as shown in Fig.4.3a. This enrichment strategy of XFEM is localized around the crack, since the rest of the elements and nodes in the mesh do not interact with the crack. Let M, M_H and M_T be the sets of all nodes that are not enriched, enriched with the Heaviside function and enriched with the crack-tip functions, respectively. Then the enriched approximation space used in XFEM is

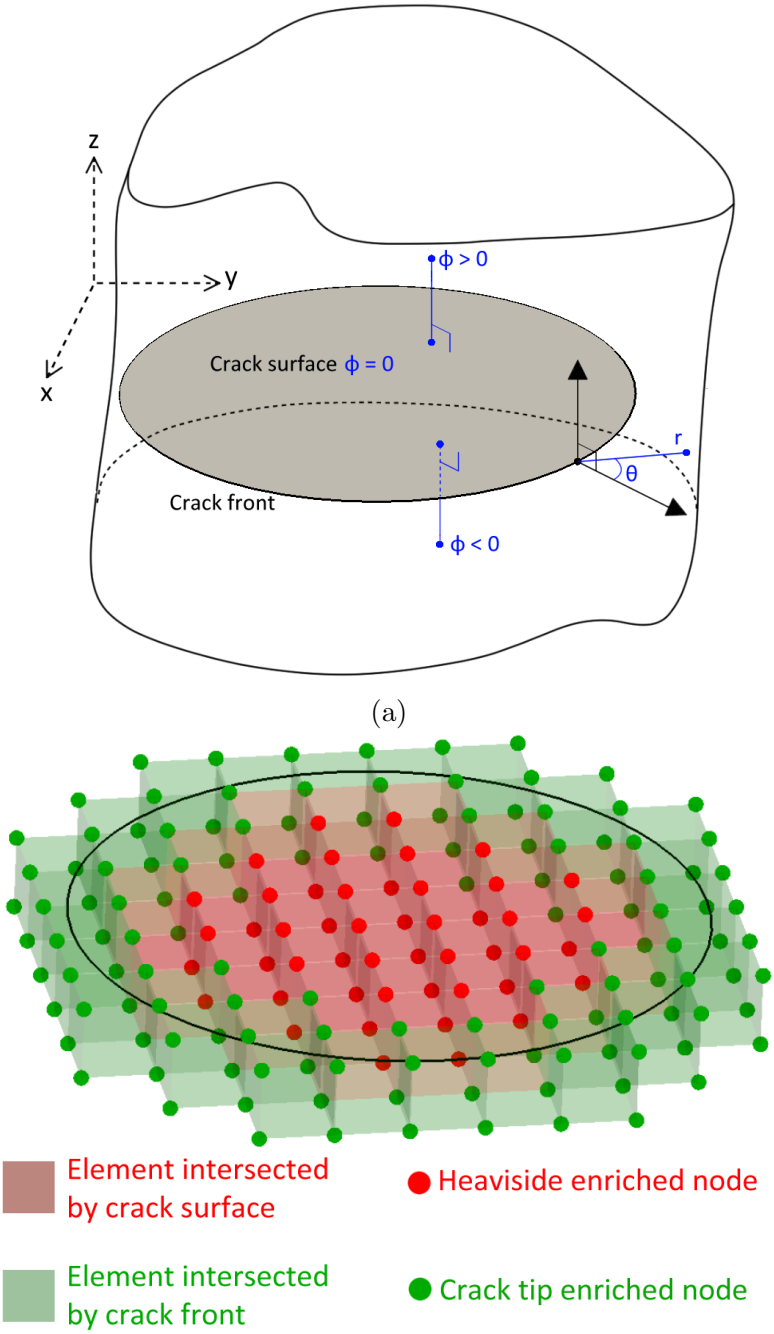


Figure 4.3: A crack surface inside a 3D body. (a) Signed distances ϕ from the crack surface and polar coordinates (r, θ) around the crack front. (b) Enriched nodes and elements intersected by the crack.

$$\begin{aligned}\boldsymbol{u}^h(\boldsymbol{x}) = &\sum_{k\in M} N_k(\boldsymbol{x})\boldsymbol{u}_k \\ &+ \sum_{k\in M_H} N_k(\boldsymbol{x})\left(H(\boldsymbol{x}) - H(\boldsymbol{x}_k)\right)\boldsymbol{u}_k^H \\ &+ \sum_{k\in M_T} N_k(\boldsymbol{x})\Big(\sum_{m=1}^{4}\left(F_m(\boldsymbol{x}) - F_m(\boldsymbol{x}_k)\right)\boldsymbol{u}_k^{Tm}\Big)\end{aligned} \tag{4.30}$$

where $\boldsymbol{u}_i$ are the standard DOFs expressing nodal displacements, while $\boldsymbol{u}_k^H$ and $\boldsymbol{u}_k^{Tm}$ are enriched DOFs introduced by XFEM at the nodes that are enriched with Heaviside and crack-tip functions, respectively. All $N_k(\boldsymbol{x})$ are polynomial shape functions, identical to the ones used in standard FEM. The above equation can be written more concisely by grouping all Heaviside and crack-tip enrichment functions and representing them collectively as $G^a(\boldsymbol{x})$

$$\boldsymbol{u}^h(\boldsymbol{x}) = \underbrace{\sum_{k\in M} N_k(\boldsymbol{x})\boldsymbol{u}_k}_{\boldsymbol{u}^{\text{std}}} + \underbrace{\sum_{k\in M_a} N_k(\boldsymbol{x})\left(G^a(\boldsymbol{x}) - G^a(\boldsymbol{x}_k)\right)\boldsymbol{u}_k^a}_{\boldsymbol{u}^{\text{enr}}} \tag{4.31}$$

where M_a are the nodes enriched with enrichment function G^a and $\boldsymbol{u}_k^a$ are the corresponding enriched DOFs. The first term ($\boldsymbol{u}^{\text{std}}$) on the right-hand side of equation (4.31) corresponds to the standard FEM approximation of the displacement field. The second term ($\boldsymbol{u}^{\text{enr}}$) contains enriched basis functions that allow the approximation space to model (i) displacement jumps across the crack surface (Heaviside enrichment) and (ii) stress/strain fields that are singular at the crack front (crack-tip enrichments). The derivatives of the crack-tip functions with respect to the polar coordinates are

$$\begin{bmatrix}\dfrac{\partial F_1}{\partial r} \\ \dfrac{\partial F_1}{\partial \theta}\end{bmatrix} = \begin{bmatrix}\dfrac{1}{2\sqrt{r}}\cdot\sin\dfrac{\theta}{2} \\ \dfrac{1}{2}\cdot\sqrt{r}\cdot\cos\dfrac{\theta}{2}\end{bmatrix} \tag{4.32}$$

$$\begin{bmatrix}\dfrac{\partial F_2}{\partial r} \\ \dfrac{\partial F_2}{\partial \theta}\end{bmatrix} = \begin{bmatrix}\dfrac{1}{2\sqrt{r}}\cdot\cos\dfrac{\theta}{2} \\ -\dfrac{1}{2}\cdot\sqrt{r}\cdot\sin\dfrac{\theta}{2}\end{bmatrix} \tag{4.33}$$

$$\begin{bmatrix}\dfrac{\partial F_3}{\partial r} \\ \dfrac{\partial F_3}{\partial \theta}\end{bmatrix} = \begin{bmatrix}\dfrac{1}{2\sqrt{r}}\cdot\sin\dfrac{\theta}{2}\cdot\sin\theta \\ \sqrt{r}\left(\dfrac{1}{2}\cdot\cos\dfrac{\theta}{2}\cdot\sin\theta + \sin\dfrac{\theta}{2}\cdot\cos\theta\right)\end{bmatrix} \tag{4.34}$$

$$\begin{bmatrix} \dfrac{\partial F_4}{\partial r} \\ \dfrac{\partial F_4}{\partial \theta} \end{bmatrix} = \begin{bmatrix} \dfrac{1}{2\sqrt{r}} \cdot \cos\dfrac{\theta}{2} \cdot \sin\theta \\ \sqrt{r}\left(-\dfrac{1}{2} \cdot \sin\dfrac{\theta}{2} \cdot \sin\theta + \cos\dfrac{\theta}{2} \cdot \cos\theta \right) \end{bmatrix} \tag{4.35}$$

4.1.5 Algebraic equations

Using the displacement field approximation of equation (4.31), the weak form of equation (4.19) results in a linear system with the nodal displacements $\boldsymbol{u}$ as unknowns

$$\begin{aligned} \boldsymbol{K}\ \boldsymbol{u} &= \boldsymbol{f} \\ \boldsymbol{K} &= \sum_{e=1}^{n_e} \boldsymbol{P}_e^T \cdot \boldsymbol{K}_e \cdot \boldsymbol{P}_e \\ \boldsymbol{f} &= \sum_{e=1}^{n_e} \boldsymbol{P}_e^T \cdot \boldsymbol{f}_e \end{aligned} \tag{4.36}$$

where n_e is the number of finite elements, $\boldsymbol{P}_e$ is a boolean matrix, namely it contains only 0 or 1 entries, that correlates each element DOF (row of $\boldsymbol{P}_e$) to one exactly global DOF (column of $\boldsymbol{P}_e$), $\boldsymbol{K}_e$ is the element stiffness matrix and $\boldsymbol{f}_e$ is the element force vector:

$$\boldsymbol{f} = \int_{\Omega_e} \boldsymbol{N}^T\left(\boldsymbol{x}\left(\boldsymbol{\xi}\right)\right) \boldsymbol{b} d\Omega + \int_{\Gamma_{te}} \boldsymbol{N}^T\left(\boldsymbol{x}\left(\boldsymbol{\xi}\right)\right) \widetilde{t} d\Gamma \tag{4.37}$$

where $\boldsymbol{\xi}$ are the coordinates of a point in the natural system of the element and $\boldsymbol{N}\left(\boldsymbol{x}\left(\boldsymbol{\xi}\right)\right)$ is a matrix containing the standard and enriched basis functions of the element:

$$\begin{aligned} \boldsymbol{N}\left(\boldsymbol{x}\left(\boldsymbol{\xi}\right)\right) &= \begin{bmatrix} \boldsymbol{N}^{std}\left(\boldsymbol{x}\left(\boldsymbol{\xi}\right)\right) & \boldsymbol{N}^{enr}\left(\boldsymbol{x}\left(\boldsymbol{\xi}\right)\right) \end{bmatrix} \\ \boldsymbol{N}^{std}\left(\boldsymbol{x}\left(\boldsymbol{\xi}\right)\right) &= \begin{bmatrix} N_k\left(\boldsymbol{x}\left(\boldsymbol{\xi}\right)\right) & 0 & 0 & \\ \cdots\ 0 & N_k\left(\boldsymbol{x}\left(\boldsymbol{\xi}\right)\right) & 0 & \cdots \\ 0 & 0 & N_k\left(\boldsymbol{x}\left(\boldsymbol{\xi}\right)\right) & \end{bmatrix} \\ \boldsymbol{N}^{enr}\left(\boldsymbol{x}\left(\boldsymbol{\xi}\right)\right) &= \begin{bmatrix} N_k^{enr}\left(\boldsymbol{x}\left(\boldsymbol{\xi}\right)\right) & 0 & 0 & \\ \cdots\ 0 & N_k^{enr}\left(\boldsymbol{x}\left(\boldsymbol{\xi}\right)\right) & 0 & \cdots \\ 0 & 0 & N_k^{enr}\left(\boldsymbol{x}\left(\boldsymbol{\xi}\right)\right) & \end{bmatrix} \end{aligned} \tag{4.38}$$

where N_k^{enr} are the enriched basis functions

$$N_k^{enr}\left(\boldsymbol{x}\left(\boldsymbol{\xi}\right)\right) = N_k\left(\boldsymbol{x}\left(\boldsymbol{\xi}\right)\right)\left(G^a\left(\boldsymbol{x}\left(\boldsymbol{\xi}\right)\right) - G^a\left(\boldsymbol{x}_k\right)\right) \tag{4.39}$$

where $G^a\left(\boldsymbol{x}\left(\boldsymbol{\xi}\right)\right)$ is any Heaviside or crack-tip enrichment, evaluated with respect to the natural coordinates $\boldsymbol{xi}$. Moreover, the element stiffness matrix can be calculaed as

$$\boldsymbol{K}_e = \int_\Omega \boldsymbol{B}^T(\boldsymbol{x}(\boldsymbol{\xi})) \cdot \boldsymbol{C}(\boldsymbol{x}(\boldsymbol{\xi})) \cdot \boldsymbol{B}(\boldsymbol{x}(\boldsymbol{\xi})) \det(\boldsymbol{J}_{NG}(\boldsymbol{\xi}))\, d\xi d\eta d\zeta \tag{4.40}$$

where $\det(\boldsymbol{J}_{NG}(\boldsymbol{\xi}))$ is the determinant of the Jacobian matrix $\boldsymbol{J}_{NG}(\boldsymbol{\xi})$ of the isoparametric mapping, which is defined in equation (A.3). The deformation matrix $\boldsymbol{B}(\boldsymbol{x}(\boldsymbol{\xi}))$ is

$$\boldsymbol{B}(\boldsymbol{x}(\boldsymbol{\xi})) = \begin{bmatrix} \boldsymbol{B}^{std}(\boldsymbol{x}(\boldsymbol{\xi})) & \boldsymbol{B}^{enr}(\boldsymbol{x}(\boldsymbol{\xi})) \end{bmatrix}$$

$$\boldsymbol{B}^{enr}(\boldsymbol{x}(\boldsymbol{\xi})) = \begin{bmatrix} & \dfrac{\partial N_k^{enr}(\boldsymbol{x}(\boldsymbol{\xi}))}{\partial x} & 0 & 0 & \\ & 0 & \dfrac{\partial N_k^{enr}(\boldsymbol{x}(\boldsymbol{\xi}))}{\partial y} & 0 & \\ & 0 & 0 & \dfrac{\partial N_k^{enr}(\boldsymbol{x}(\boldsymbol{\xi}))}{\partial z} & \\ \cdots & \dfrac{\partial N_k^{enr}(\boldsymbol{x}(\boldsymbol{\xi}))}{\partial y} & \dfrac{\partial N_k^{enr}(\boldsymbol{x}(\boldsymbol{\xi}))}{\partial x} & 0 & \cdots \\ & 0 & \dfrac{\partial N_k^{enr}(\boldsymbol{x}(\boldsymbol{\xi}))}{\partial z} & \dfrac{\partial N_k^{enr}(\boldsymbol{x}(\boldsymbol{\xi}))}{\partial y} & \\ & \dfrac{\partial N_k^{enr}(\boldsymbol{x}(\boldsymbol{\xi}))}{\partial z} & 0 & \dfrac{\partial N_k^{enr}(\boldsymbol{x}(\boldsymbol{\xi}))}{\partial x} & \end{bmatrix} \tag{4.41}$$

$\boldsymbol{B}^{std}(\boldsymbol{x}(\boldsymbol{\xi})) =$ similarly

The derivatives with respect to global cartesian coordinates $\boldsymbol{x}$ are calculated using the derivatives with respect to natural coordinates $\boldsymbol{\xi}$:

$$\begin{bmatrix} \dfrac{\partial N_k^{enr}(\boldsymbol{x}(\boldsymbol{\xi}))}{\partial x} \\ \dfrac{\partial N_k^{enr}(\boldsymbol{x}(\boldsymbol{\xi}))}{\partial y} \\ \dfrac{\partial N_k^{enr}(\boldsymbol{x}(\boldsymbol{\xi}))}{\partial z} \end{bmatrix} = (\boldsymbol{J}_{NG}(\boldsymbol{\xi}))^{-1} \cdot \begin{bmatrix} \dfrac{\partial N_k^{enr}(\boldsymbol{\xi})}{\partial \xi} \\ \dfrac{\partial N_k^{enr}(\boldsymbol{\xi})}{\partial \eta} \\ \dfrac{\partial N_k^{enr}(\boldsymbol{\xi})}{\partial \zeta} \end{bmatrix} \tag{4.42}$$

and

$$\begin{aligned} \frac{\partial N_k^{enr}(\boldsymbol{\xi})}{\partial \xi} &= \frac{\partial N_k(\boldsymbol{\xi})}{\partial \xi}(G^a(\boldsymbol{\xi}) - G^a(\boldsymbol{\xi}_k)) + N_k(\boldsymbol{\xi}) \frac{\partial G^a(\boldsymbol{\xi})}{\partial \xi} \\ \frac{\partial N_k^{enr}(\boldsymbol{\xi})}{\partial \xi} &= \frac{\partial N_k(\boldsymbol{\xi})}{\partial \eta}(G^a(\boldsymbol{\xi}) - G^a(\boldsymbol{\xi}_k)) + N_k(\boldsymbol{\xi}) \frac{\partial G^a(\boldsymbol{\xi})}{\partial \eta} \\ \frac{\partial N_k^{enr}(\boldsymbol{\xi})}{\partial \xi} &= \frac{\partial N_k(\boldsymbol{\xi})}{\partial \zeta}(G^a(\boldsymbol{\xi}) - G^a(\boldsymbol{\xi}_k)) + N_k(\boldsymbol{\xi}) \frac{\partial G^a(\boldsymbol{\xi})}{\partial \zeta} \end{aligned} \tag{4.43}$$

For the Heaviside enrichment function $H(\boldsymbol{x}(\boldsymbol{\xi}))$ of equation (4.29), the derivatives are

$$\frac{\partial H(\boldsymbol{\xi})}{\partial \xi} = \frac{\partial H(\boldsymbol{\xi})}{\partial \eta} = \frac{\partial H(\boldsymbol{\xi})}{\partial \zeta} = 0 \tag{4.44}$$

The derivatives of the crack-tip enrichment functions $F_m(\boldsymbol{x}(\boldsymbol{\xi}))$ of equation (4.28) with respect to the global coordinates $\boldsymbol{x}$ can be calculated as

$$\begin{bmatrix} \frac{\partial F_m}{\partial x} \\ \frac{\partial F_m}{\partial y} \\ \frac{\partial F_m}{\partial z} \end{bmatrix} = \underbrace{\begin{bmatrix} \frac{\partial \xi}{\partial x} & \frac{\partial \eta}{\partial x} & \frac{\partial \zeta}{\partial x} \\ \frac{\partial \xi}{\partial y} & \frac{\partial \eta}{\partial y} & \frac{\partial \zeta}{\partial y} \\ \frac{\partial \xi}{\partial z} & \frac{\partial \eta}{\partial z} & \frac{\partial \zeta}{\partial z} \end{bmatrix}}_{(\boldsymbol{J}_{NG})^{-1}} \cdot \begin{bmatrix} \frac{\partial F_m}{\partial \xi} \\ \frac{\partial F_m}{\partial \eta} \\ \frac{\partial F_m}{\partial \zeta} \end{bmatrix} \tag{4.45}$$

where $\boldsymbol{J}_{NG}$ is the Jacobian matrix of the isoparametric mapping: natural $\longrightarrow$ global coordinate system (NG), which is defined in equation (A.3). Then, the derivatives with respect to the natural coordinates $\boldsymbol{\xi}$ are

$$\begin{bmatrix} \frac{\partial F_m}{\partial \xi} \\ \frac{\partial F_m}{\partial \eta} \\ \frac{\partial F_m}{\partial \zeta} \end{bmatrix} = \underbrace{\begin{bmatrix} \frac{\partial r}{\partial \xi} & \frac{\partial \theta}{\partial \xi} \\ \frac{\partial r}{\partial \eta} & \frac{\partial \theta}{\partial \eta} \\ \frac{\partial r}{\partial \zeta} & \frac{\partial \theta}{\partial \zeta} \end{bmatrix}}_{\boldsymbol{J}_{NP}} \cdot \begin{bmatrix} \frac{\partial F_m}{\partial r} \\ \frac{\partial F_m}{\partial \theta} \end{bmatrix} \tag{4.46}$$

where $\boldsymbol{J}_{NP}$ is the Jacobian matrix of the mapping: natural $\longrightarrow$ polar coordinate system (NP). The derivatives $\frac{\partial F_m}{\partial r}$, $\frac{\partial F_m}{\partial \theta}$ of the crack-tip enrichment functions with respect to the polar coordinates (r, θ) are evaluated using equations (4.32–4.35). In this work, the Jacobian matrix $\boldsymbol{J}_{NP}$ is calculated using an auxiliary coordinate system (ϕ, ψ) defined by the representation of the crack's geometry, which will be elaborated in section 4.2.

4.2 Crack geometry representation

In combination with XFEM, implicit representations of crack geometries are usually employed, based on the Level Set Method (LSM). LSM was originally proposed in Osher and Sethian (1988) for tracking complicated moving interfaces, by computing their motion on a fixed finite element mesh. This implicit representation complements XFEM very well, since

it uses the same fixed mesh to efficiently calculate the signed distance of arbitrary points to the crack surface, as well as their polar coordinates (see equation (4.28)).

In this work, a hybrid explicit-implicit representation of cracks is used, instead of the purely implicit LSM developed by Stolarska and Chopp (2003) for simple 2D cracks. This hybrid approach was introduced in Fries and Baydoun (2012) and takes advantage of the synergy between implicit crack representations and XFEM, as well as the ease of updating explicit crack geometries in both 2D and 3D. A brief overview of the method will be given here, while interested readers should refer to Fries and Baydoun (2012) for more details. Note this approach is just one alternative for representing 3D cracks. The solution methods developed in chapter 5 will perform just as well with other crack geometry description methods.

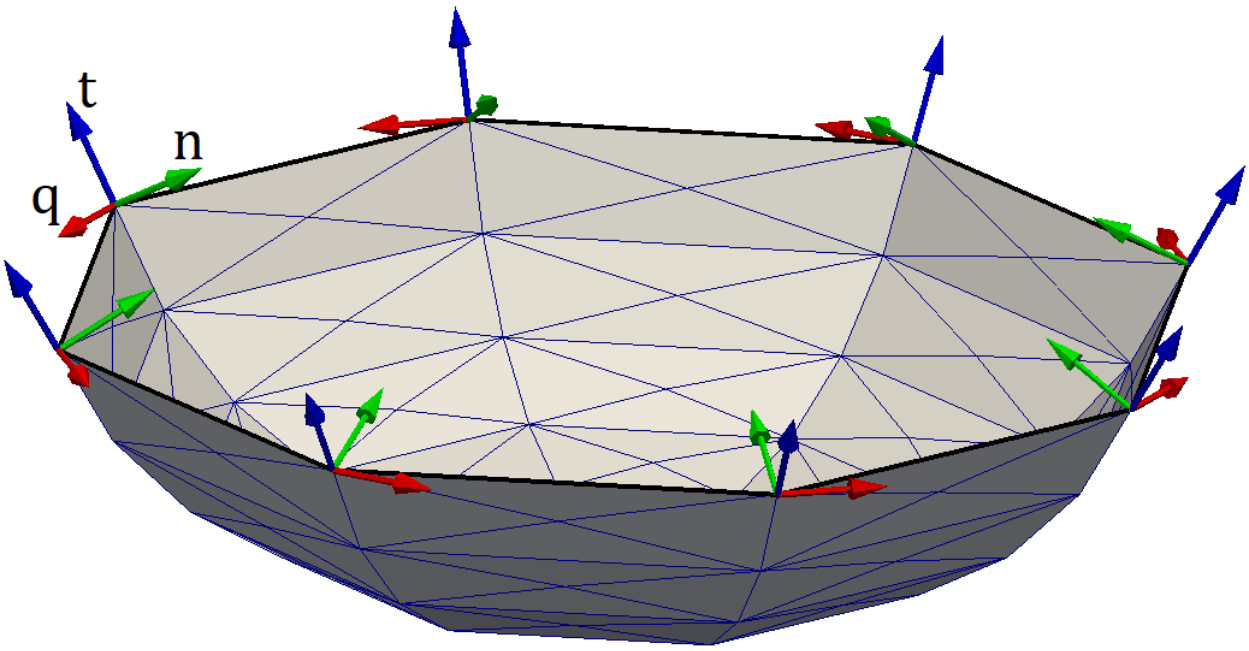


Figure 4.4: Explicit representation of a 3D crack as a mesh with triangular elements.

A crack surface is represented explicitly as a mesh of triangular cells in 3D space and the crack front consists of the surrounding vertices of this mesh, namely the crack tips. An example involving a planar crack is shown figure 4.4, along with local coordinate systems defined at each crack tip. This explicit representation is very convenient to model crack growth, since only new triangles need to be added along the crack front, as the crack propagates. Specifically, each crack tip $\boldsymbol{x}_i^{(t)}$ at step t will propagate towards $\boldsymbol{x}_i^{(t+1)}$. Update the crack mesh of step t consists simply of adding to it the triangles between crack tips $\boldsymbol{x}_i^{(t)}$ and $\boldsymbol{x}_i^{(t+1)}$.

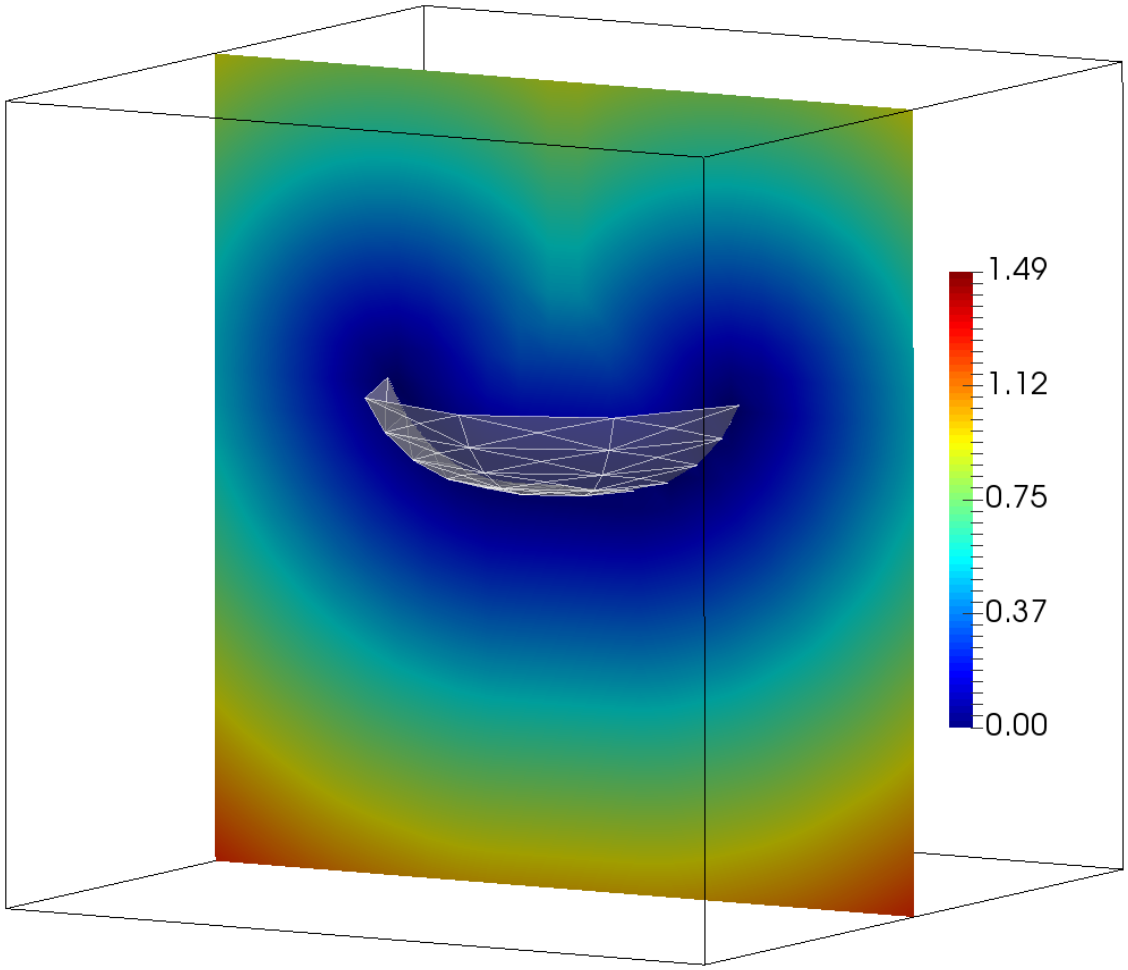


Figure 4.5: Implicit representation of a 3D crack: $\phi_1(\boldsymbol{x})$

Furthermore using this triangular crack mesh, 3 level set functions are evaluated at the finite element nodes:

- $\phi_1(\boldsymbol{x})$ is the unsigned distance of a point to the crack mesh.
- $\phi_2(\boldsymbol{x})$ is the unsigned distance of a point to the crack front.
- $\phi_3(\boldsymbol{x})$ is the signed distance of a point to the crack mesh, which defines a positive and a negative half-space depending on the normal vectors to the triangles.

Figures 4.5-4.7 illustrate these level set functions plotted over a plane that intersects the crack. In Fries and Baydoun (2012), these 3 level sets are used to evaluate the enrichment functions, identify elements intersected by the crack, etc. In this dissertation, a modified approach will be used, which avoids singular derivatives when $\phi_2(\boldsymbol{x}) = \phi_3(\boldsymbol{x})$. Two more level sets $\phi(\boldsymbol{x})$, $\psi(\boldsymbol{x})$ are defined here:

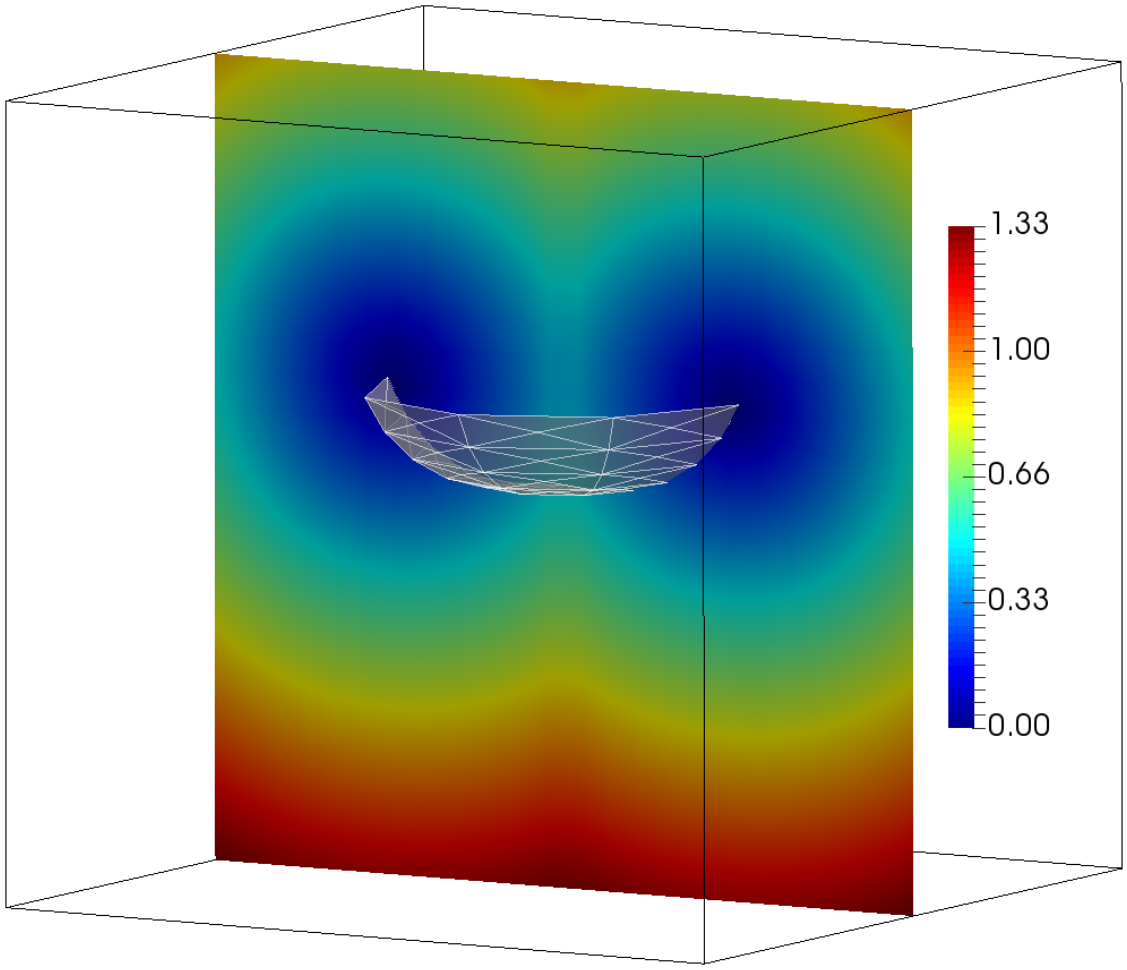


Figure 4.6: Implicit representation of a 3D crack: $\phi_2(\boldsymbol{x})$

$$\phi(\boldsymbol{x}) = \phi_3(\boldsymbol{x})$$

$$\psi(\boldsymbol{x}) = \begin{cases} \sqrt{(\phi_2(\boldsymbol{x}))^2 - (\phi_3(\boldsymbol{x}))^2} & \text{if } \phi_1(\boldsymbol{x}) \neq |\phi_3(\boldsymbol{x})| \\ -\sqrt{(\phi_2(\boldsymbol{x}))^2 - (\phi_3(\boldsymbol{x}))^2} & \text{if } \phi_1(\boldsymbol{x}) = |\phi_3(\boldsymbol{x})|, \phi_2(\boldsymbol{x}) \neq |\phi_3(\boldsymbol{x})|, \phi_3(\boldsymbol{x}) > 0 \\ -\sqrt{(\phi_2(\boldsymbol{x}))^2 - (\phi_3(\boldsymbol{x}))^2} & \text{if } \phi_1(\boldsymbol{x}) = |\phi_3(\boldsymbol{x})|, \phi_2(\boldsymbol{x}) \neq |\phi_3(\boldsymbol{x})|, \phi_3(\boldsymbol{x}) \leq 0 \\ 0 & \text{if } \phi_1(\boldsymbol{x}) = |\phi_3(\boldsymbol{x})|, \phi_2(\boldsymbol{x}) = |\phi_3(\boldsymbol{x})| \end{cases} \tag{4.47}$$

where it should be noted that $\phi_2(\boldsymbol{x}) > \phi_3(\boldsymbol{x})$ by definition. Figure 4.8 depicts the ψ level set, while ϕ can be seen in 4.7. Both these level sets functions are evaluated and stored at

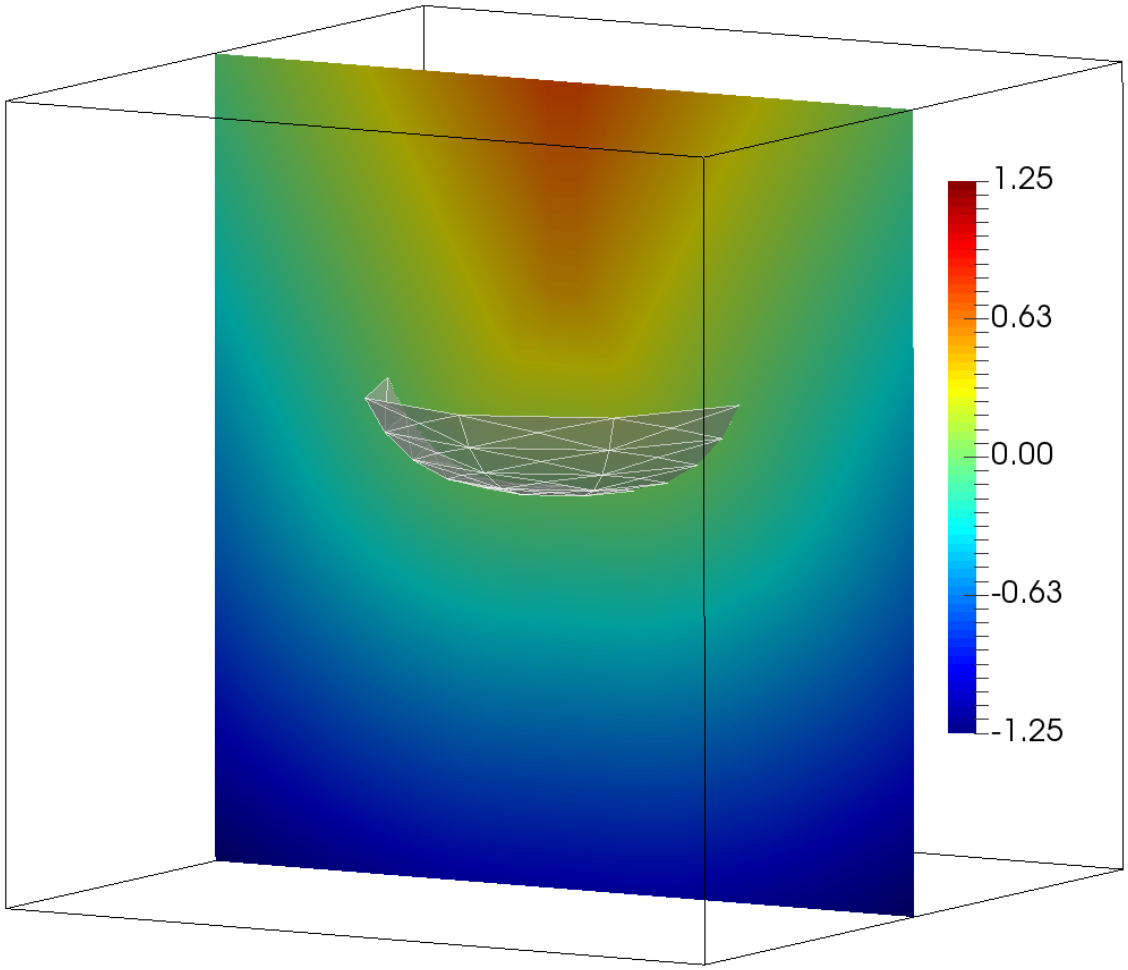


Figure 4.7: Implicit representation of a 3D crack: $\phi_3(\boldsymbol{x}) = \phi(\boldsymbol{x})$

mesh nodes $\boldsymbol{x}_k$. For any point inside the finite elements, the standard shape functions can be used to interpolate the stored nodal values ϕ_k, ψ_k:

$$\begin{aligned} \phi(\boldsymbol{x}(\boldsymbol{\xi})) &= \sum_{k=1}^{n_{nodes}} N_k(\boldsymbol{\xi})\,\phi_k \\ \psi(\boldsymbol{x}(\boldsymbol{\xi})) &= \sum_{k=1}^{n_{nodes}} N_k(\boldsymbol{\xi})\,\psi_k \end{aligned} \tag{4.48}$$

If ϕ^{min}, ϕ^{max}, ψ^{min}, ψ^{max} are the minimum and maximum values of the level set functions evaluated at the nodes of a finite element, then the element is intersected by the crack surface if

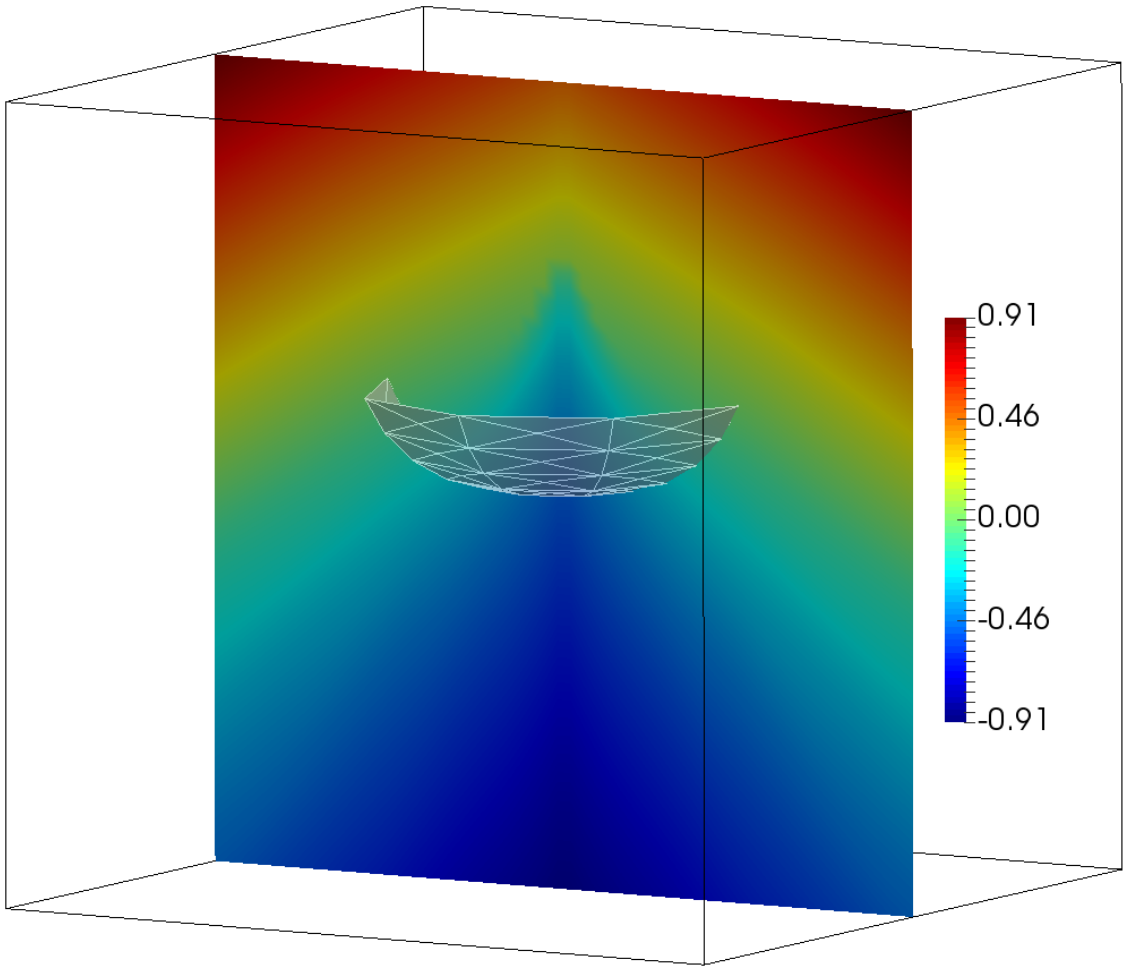


Figure 4.8: Implicit representation of a 3D crack: $\psi(\boldsymbol{x})$

$$\phi^{min} \cdot \phi^{max} \leq 0 \text{ and } \psi^{max} < 0 \tag{4.49}$$

Similarly, the element is intersected by the crack front if

$$\phi^{min} \cdot \phi^{max} \leq 0 \text{ and } \psi^{min} \cdot \psi^{max} \leq 0 \tag{4.50}$$

In addition, the (ϕ, ψ) level sets can be used to calculate the polar coordinates needed for the crack-tip enrichment functions in equation (4.28):

$$\begin{aligned} r &= \sqrt{\phi^2 + \psi^2} \\ \theta &= \arctan \frac{\phi}{\psi} = ATAN2(\phi, \psi) \end{aligned} \tag{4.51}$$

as well as evaluate the Heaviside enrichment function in equation (4.29):

$$H(\boldsymbol{x}) = H(\phi(\boldsymbol{x})) \tag{4.52}$$

Furthermore, (ϕ, ψ) define a "level-set" coordinate system that can be used to calculate the Jacobian matrix $\boldsymbol{J}_{NP}$, which is necessary for the evaluation of the derivatives of crack-tip enrichment functions, as described in equations (4.45, 4.46):

$$\begin{aligned} & \boldsymbol{J}_{NP} = \boldsymbol{J}_{N\Phi} \cdot \boldsymbol{J}_{\Phi P} \\ \Longleftrightarrow \quad & \begin{bmatrix} \dfrac{\partial r}{\partial \xi} & \dfrac{\partial \theta}{\partial \xi} \\ \dfrac{\partial r}{\partial \eta} & \dfrac{\partial \theta}{\partial \eta} \\ \dfrac{\partial r}{\partial \zeta} & \dfrac{\partial \theta}{\partial \zeta} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial \phi}{\partial \xi} & \dfrac{\partial \psi}{\partial \xi} \\ \dfrac{\partial \phi}{\partial \eta} & \dfrac{\partial \psi}{\partial \eta} \\ \dfrac{\partial \phi}{\partial \zeta} & \dfrac{\partial \psi}{\partial \zeta} \end{bmatrix} \cdot \begin{bmatrix} \dfrac{\partial r}{\partial \phi} & \dfrac{\partial \theta}{\partial \phi} \\ \dfrac{\partial r}{\partial \psi} & \dfrac{\partial \theta}{\partial \psi} \end{bmatrix} \end{aligned} \tag{4.53}$$

where $\boldsymbol{J}_{N\Phi}$ is the Jacobian matrix of the mapping: natural $\longrightarrow$ level-set coordinate system $(N\Phi)$ and can be calculated using equation (4.48):

$$\begin{bmatrix} \dfrac{\partial \phi}{\partial \xi} & \dfrac{\partial \psi}{\partial \xi} \\ \dfrac{\partial \phi}{\partial \eta} & \dfrac{\partial \psi}{\partial \eta} \\ \dfrac{\partial \phi}{\partial \zeta} & \dfrac{\partial \psi}{\partial \zeta} \end{bmatrix} = \begin{bmatrix} \displaystyle\sum_{k=1}^{n_{nodes}} \frac{\partial N_k(\boldsymbol{\xi})}{\partial \xi} \phi_k & \displaystyle\sum_{k=1}^{n_{nodes}} \frac{\partial N_k(\boldsymbol{\xi})}{\partial \xi} \psi_k \\ \displaystyle\sum_{k=1}^{n_{nodes}} \frac{\partial N_k(\boldsymbol{\xi})}{\partial \eta} \phi_k & \displaystyle\sum_{k=1}^{n_{nodes}} \frac{\partial N_k(\boldsymbol{\xi})}{\partial \eta} \psi_k \\ \displaystyle\sum_{k=1}^{n_{nodes}} \frac{\partial N_k(\boldsymbol{\xi})}{\partial \zeta} \phi_k & \displaystyle\sum_{k=1}^{n_{nodes}} \frac{\partial N_k(\boldsymbol{\xi})}{\partial \zeta} \psi_k \end{bmatrix} \tag{4.54}$$

Moreover, $\boldsymbol{J}_{\Phi P}$ is the Jacobian matrix of the mapping: level-set $\longrightarrow$ polar coordinate system $(N\Phi)$. Starting from equation (4.51), the derivatives of the polar coordinates with respect to the level sets are

$$\begin{aligned} \frac{\partial r}{\partial \phi} &= \frac{\partial(\phi^2 + \psi^2)}{\partial \phi} \Big/ 2\sqrt{\phi^2 + \psi^2} = \frac{2\phi}{2r} = \frac{\phi}{r} \\ \frac{\partial r}{\partial \psi} &= \frac{\partial(\phi^2 + \psi^2)}{\partial \psi} \Big/ 2\sqrt{\phi^2 + \psi^2} = \frac{2\psi}{2r} = \frac{\psi}{r} \\ \frac{\partial \theta}{\partial \phi} &= \frac{\psi}{\phi^2 + \psi^2} = \frac{\psi}{r^2} \\ \frac{\partial \theta}{\partial \psi} &= -\frac{\phi}{\phi^2 + \psi^2} = -\frac{\phi}{r^2} \end{aligned} \tag{4.55}$$

and the Jacobian matrix $\boldsymbol{J}_{\Phi P}$ is

$$\boldsymbol{J}_{\Phi P} = \begin{bmatrix} \dfrac{\partial r}{\partial \phi} & \dfrac{\partial \theta}{\partial \phi} \\ \dfrac{\partial r}{\partial \psi} & \dfrac{\partial \theta}{\partial \psi} \end{bmatrix} = \begin{bmatrix} \dfrac{\phi}{r} & \dfrac{\psi}{r^2} \\ \dfrac{\psi}{r} & -\dfrac{\phi}{r^2} \end{bmatrix} \tag{4.56}$$

To sum up, the derivatives of the crack-tip enrichment functions with respect to the global coordinates can be converted to the derivatives with respect to the polar coordinates of equations (4.32–4.35) as

$$\begin{bmatrix} \dfrac{\partial F_m}{\partial x} \\ \dfrac{\partial F_m}{\partial y} \\ \dfrac{\partial F_m}{\partial z} \end{bmatrix} = \underbrace{(\boldsymbol{J}_{NG})^{-1} \cdot \boldsymbol{J}_{N\Phi} \cdot \boldsymbol{J}_{\Phi P}}_{\boldsymbol{J}_{GP}} \cdot \begin{bmatrix} \dfrac{\partial F_m}{\partial r} \\ \dfrac{\partial F_m}{\partial \theta} \end{bmatrix} \tag{4.57}$$

where $\boldsymbol{J}_{GP}$ can be seen as the Jacobian matrix of an (indirect) mapping: polar $\longrightarrow$ global coordinate system (PG).

4.3 Crack propagation

For a given crack configuration, XFEM analysis is applied to calculate the displacement, strain and stress fields. In order to model the propagation of the crack surface from these results, the simplified approach proposed in Fries and Baydoun (2012) will be used in this work. According to this "$\sigma_{r\theta} = 0$"-criterion, the circumferential ($\sigma_{\theta\theta}$) and shear ($\sigma_{r\theta}$) stresses are calculated at a set of trial points around one of the crack tips, namely the vertices of the crack front. Then the propagation angle θ_{cr} for that tip is selected as the one where $\sigma_{r\theta} = 0$. If there are multiple such angles, then the angle with maximum $\sigma_{\theta\theta}$ among them is chosen. To begin with, the stress tensor at a trial point needs to be transformed to the local cartesian coordinate system of each crack tip

$$\boldsymbol{\sigma_{tnq}} = \boldsymbol{T}_{gl} \boldsymbol{\sigma_{xyz}} \boldsymbol{T}_{gl}^T = \begin{bmatrix} \sigma_{tt} & \sigma_{tn} & \sigma_{tq} \\ \sigma_{nt} & \sigma_{nn} & \sigma_{nq} \\ \sigma_{qt} & \sigma_{qn} & \sigma_{qq} \end{bmatrix} \tag{4.58}$$

where $\boldsymbol{n}$ is a vector normal to the crack surface, $\boldsymbol{q}$ is oriented along the crack front and $\boldsymbol{t}$ is oriented towards the tangential extension of the crack and orthogonal to the previous $\boldsymbol{n}$, $\boldsymbol{q}$, as illustrated in figure 4.4. The global-local rotation $\boldsymbol{T}_{gl}$ is defined as

$$\boldsymbol{T}_{gl} = \begin{bmatrix} \dfrac{\partial \boldsymbol{t}}{\partial x} & \dfrac{\partial \boldsymbol{t}}{\partial y} & \dfrac{\partial \boldsymbol{t}}{\partial z} \\ \dfrac{\partial \boldsymbol{n}}{\partial x} & \dfrac{\partial \boldsymbol{n}}{\partial y} & \dfrac{\partial \boldsymbol{n}}{\partial z} \\ \dfrac{\partial \boldsymbol{q}}{\partial x} & \dfrac{\partial \boldsymbol{q}}{\partial y} & \dfrac{\partial \boldsymbol{q}}{\partial z} \end{bmatrix} \tag{4.59}$$

Subsequently the polar stresses $\sigma_{\theta\theta}$ and $(\sigma_{r\theta})$ are calculated

$$\begin{aligned} \sigma_{\theta\theta} &= \sigma_{nn}\sin^2(\theta) + \sigma_{tt}\cos^2(\theta) - \sigma_{nt}\sin(2\theta) \\ \sigma_{r\theta} &= \sin(\theta)\cos(\theta)\left(\sigma_{nn} - \sigma_{tt}\right) + \sigma_{nt}\cos(2\theta) \end{aligned} \tag{4.60}$$

After estimating the direction of propagation θ_{cr} for a crack tip, the increment must be specified too. At each propagation step of the analysis, a maximum crack increment da is prescribed. This length is assigned to the crack tip with the maximum circumferential $\max(\sigma_{\theta\theta})$ stress, while the length of the crack increment at other tips will be scaled proportionally to their $\sigma_{\theta\theta}$. The trial points are placed on a $[-75°, +75°]$ arc in front of each crack tip at a distance r_c such that $0.1 \cdot da \leq r_c \leq da$, where da is the crack increment.

This simplified method for modelling crack propagation has been shown to produce reasonably accurate crack paths. However, other approaches, such as the J-integral method (Rice, 1968) and configurational forces (Gurtin, 1995) may result in higher accuracy, although they pose difficulties as well, especially in 3D problems. In any case, any method to estimate the crack propagation can be used without affecting the solvers that will be presented in chapter 5. The resulting procedure for crack propagation is therefore:

Algorithm 4.1 Quasi-static crack propagation analysis

```
Initialize crack geometry and level sets.
repeat
    Perform XFEM analysis to obtain linear system, as described in section 4.1.
    Solve linear system with one of the solvers, presented in chapter 5.
    for each crack tip do
        Identify trial points and calculate stress tensors.
        Estimate propagation direction and increment.
    end for
    Update the crack mesh and level sets, as described in section 4.2.
until the crack fully intersects the domain
```

This quasi-static analysis is suitable for simulating brittle crack propagation based on LEFM theory and has been used since the early XFEM works, such as Belytschko and Black (1999).

Chapter 5

Linear system solvers

Both FEM and XFEM convert the partial differentions of the boundary value problem into a system of algebraic equations

$$\boldsymbol{K}\boldsymbol{u} = \boldsymbol{f} \tag{5.1}$$

where the system matrix $\boldsymbol{K}$ is the stiffness or conductivity of the discretized domain, the right-hand-side (RHS) vector $\boldsymbol{f}$ expresses forces or thermal loads applied to the domain and the solution vector $\boldsymbol{u}$ contains the nodal displacements or temperatures. The system matrix $\boldsymbol{K}$ is symmetric positive definite. The solution of this linear system is by far the most computationally intensive part of FEM and XFEM, especially in large-scale and 3D problems. Inverting the system matrix is almost always prohibitive in terms of computing time and memory requirements. Instead, various solution algorithms have been developed and will be investigated in this chapter. Additionally, two novel solvers and their implementation in high performance computing environments will be proposed for the solution of linear systems resulting from crack propagation analysis in the framework of XFEM.

5.1 Direct methods

5.1.1 Cholesky solver

Direct methods solve the linear system by factorizing the system matrix and then applying back and substitution. For a positive definite system matrix, the most efficient direct methods are Cholesky solvers, which are based on the *Cholesky factorization*

$$\boldsymbol{K} = \boldsymbol{L}\boldsymbol{L}^T \tag{5.2}$$

where $\boldsymbol{L}$ is a lower triangular matrix. Most often, the *LDL factorization* variant is used:

$$\boldsymbol{K} = \boldsymbol{L}\boldsymbol{D}\boldsymbol{L}^T \tag{5.3}$$

where $\boldsymbol{L}$ is a lower triangular matrix with diagonal entries being equal to 1 and $\boldsymbol{D}$ is a diagonal matrix. Once the matrix is factorized, the $\boldsymbol{L}, \boldsymbol{D}$ factors can be used to solve one or more linear systems with different RHS vectors, according to the procedure described in algorithm 5.1:

Algorithm 5.1 Cholesky solver for linear systems $\boldsymbol{K}\boldsymbol{u}_t = \boldsymbol{f}_t$

1: Factorize: $\boldsymbol{K} = \boldsymbol{L}\boldsymbol{D}\boldsymbol{L}^T$
2: **for** $t = 1 \cdots n_{sys}$ **do**
3: Forward substitution: $\boldsymbol{L}\boldsymbol{x}_t = \boldsymbol{f}_t$
4: Diagonal solve: $\boldsymbol{D}\boldsymbol{y}_t = \boldsymbol{x}_t$
5: Back substitution: $\boldsymbol{L}^T\boldsymbol{u}_t = \boldsymbol{y}_t$
6: **end for**

where the subscript t denotes each of the n_{sys} RHS vectors. The LDL factorization is described in algorithm 5.2:

Algorithm 5.2 LDL factorization $\boldsymbol{K} = \boldsymbol{L}\boldsymbol{D}\boldsymbol{L}^T$

1: **for** $i = 1 \cdots n$ **do**
2: **for** $j = 1 \cdots i - 1$ **do**
3: $$L_{ij} = \frac{1}{D_{jj}}\Big(K_{ij} - \sum_{k=1}^{j-1} L_{ik}L_{jk}D_{kk}\Big)$$
4: **end for**
5: $$D_{ii} = K_{ii} - \sum_{k=1}^{i-1} L_{ik}L_{ik}D_{kk}$$
6: **end for**

where n is the number of rows (equal to the number of colums) and the subscripts ij denote the entry of the corresponding matrix at row i and column j. The forward substitution using the lower triangular matrix $\boldsymbol{L}$ is listed in algorithm 5.3:

Algorithm 5.3 Forward substitution $\boldsymbol{L}\boldsymbol{x} = \boldsymbol{f}$

1: $x_1 = f_1$
2: **for** $i = 2 \cdots n$ **do**
3: $$x_i = f_i - \sum_{j=1}^{i-1} L_{ij}x_j$$
4: **end for**

The solution of the system with the diagonal matrix $\boldsymbol{D}$ is listed in algorithm 5.4:

Algorithm 5.4 Diagonal system solution $\boldsymbol{Dy} = \boldsymbol{x}$

1: **for** $i = 1 \cdots n$ **do**
2: $\quad y_i = \dfrac{x_i}{D_{ii}}$
3: **end for**

The back substitution using the upper triangular matrix $\boldsymbol{L}^T$ is listed in algorithm 5.5:

Algorithm 5.5 Back substitution $\boldsymbol{L}^T\boldsymbol{u} = \boldsymbol{y}$

1: $u_n = y_n$
2: **for** $i = n - 1 \cdots 1$ **do**
3: $\quad u_i = y_i - \sum_{j=i+1}^{n} L_{ji} u_j$
4: **end for**

5.1.2 Sparse direct solvers

For matrices resulting from FEM and XFEM, most entries are equal to 0, while the *non-zero entries* are gathered around the diagonal. In this case, only the non-zero entries are explicitly stored, in order to greatly reduce the memory requirements of the solver and be able to solve much larger problems. The positions of the non-zero entries of a matrix is called its *sparsity pattern*. There are various *sparse matrix formats* used in practice, such as the skyline format (SKY). An example of a symmetric matrix stored in SKY format is:

$$\boldsymbol{K}^{SKY} = \begin{bmatrix} K_{11} & K_{12} & & K_{14} & & & & \\ & K_{22} & K_{23} & 0 & & & & \\ & & K_{33} & K_{34} & & K_{35} & & \\ & & & K_{44} & K_{45} & K_{46} & & \\ & & & & K_{55} & K_{56} & & K_{58} \\ & & & & & K_{66} & K_{67} & 0 \\ & & & & & & K_{77} & K_{78} \\ & & & & & & & K_{88} \end{bmatrix} \tag{5.4}$$

where only the non-zero entries of the upper triangle are stored, as well as any zero entries between the top non-zero entry to the diagonal entry of each column. Therefore the sparsity pattern of the matrix includes some zero entries. The active height of each column is defined as the distance between the top and diagonal non-zero entries and the *bandwidth* of the matrix as the maximum active height of all columns. In practice, the non-zero entries of matrices resulting from FEM and XFEM are very few (less than 20%), while the zero entries inside the active columns can reach a considerable number, albeit much less than the ignored zero entries outside the active columns. During factorization, any zero entries outside the sparsity pattern will remain as 0 and can thus be ignored in algorithms 5.2 - 5.5, which greatly

reduces the computational time. However, the zero entries inside the sparsity pattern will be changed to non-zero values, a process that is called *fill-in* and is the main obstacle of direct methods when solving large-scale problems.

A non-diagonal entry is non-zero when it corresponds to two DOFs that interact with each other, namely they belong to the same node or different nodes of the same element. The bandwidth of the matrix is the maximum distance in the ordering of these interacting DOFs, namely which unique number identifier is assigned to each DOF. Two nodes belong to the same element, when they are neighbors in the finite element mesh, but they can be ordered consecutively only along one axis, not all three axes x, y, z. As the mesh is refined, the difference in ordering of neighboring nodes increases, as does the ordering difference of their DOFs and thus the bandwidth of the matrix. Therefore, for large-scale problems, the bandwidth of the matrix can become so high that the fill-in causes the direct solver to become very inefficient, since it spends too much time operating on zero entries within the active heights of the columns, or even impossible due to the memory requirements of storing these zero entries. This problem does not affect 1D problems, where the DOF ordering is always consecutive. In 2D problems, the DOF ordering distance and thus bandwidth of the matrix are not very high, albeit higher than in 1D problem, therefore direct solvers are a very attractive choice. However, in 3D problems the increased ordering difference between neighboring nodes and their DOFs greatly increase the bandwidth and fill-in of the matrix, rendering direct solvers inefficient or unusable.

On the other hand, the bandwidth of the matrix can be reduced by applying a DOF reordering algorithm, such as the reverse Cuthill-McKee developed in Cuthill and McKee (1969), Approximate Minimum Degree (AMD) and its Column AMD (COLAMD) variant proposed in Amestoy et al. (2004) and Davis et al. (2004), respectively, as well as nested dissection by George (1973). These fill-reducing algorithms renumber the DOFs by using graph partitioning algorithms or heuristics, in order to minimize the bandwidth of the matrix. Their computational cost is much lower than factorizing the matrix, thus all direct solvers use a reordering algorithm prior to the factorization step in practice. In the numerical examples investigated in this dissertation, the direct method of choice is the supernodal sparse Cholesky solver (Y. Chen et al., 2008), which is implemented in the CHOLMOD package of the SuiteSparse library (Davis, 2022). This linear algebra library uses a sparse matrix format, which is similar to skyline and also provides AMD, COLAMD and nested dissection reordering algorithms, which are all applied and the best DOF ordering is kept.

5.1.3 Advantages and disadvantages

Advantages of direct solvers:

- The back/forward/diagonal substitutions require considerably less computational effort than the factorization step. Once the matrix is factorized, multiple linear systems with the same matrix, but different RHS vectors, can be solved very fast.
- In 1D problems, they are the most efficient solves available.

- In 2D problems, they are less efficient due to the increased bandwidth, but in many cases they are still the most attractive choice.
- They are reliable and not affected by ill-conditioning as much as iterative solvers.
- The number of operations can be predicted based on the system size and bandwidth.

Disadvantages of direct solvers:

- In 3D and large-scale problems they are very slow and can have prohibitive memory requirements, since the bandwidth of the matrix is very high, even after DOF reordering.
- The matrix must be explicitly stored in the memory space of a single computer. Using more computers to increase the available memory is pointless.
- The operations used in factorization and back/forward substitution are strongly coupled. Thus, they cannot be executed in parallel as efficiently as in iterative and domain decomposition solvers.

5.2 Iterative methods

Iterative solvers calculate an approximate solution of the linear system, by starting from an initial guess and then refining it over a number of iterations. The iterations end when the iterative method has converged, namely the error of the approximate solution is within a desired tolerance, or a maximum number of iterations has been reached. If $\boldsymbol{u}$ is the exact solution of the linear system and $\boldsymbol{u}_t$ is the approximate solution at iteration t, then the error is defined as

$$\boldsymbol{e}_t = \boldsymbol{u} - \boldsymbol{u}_t \tag{5.5}$$

Since the exact solution is not known, the residual vector

$$\boldsymbol{r}_t = \boldsymbol{f} - \boldsymbol{K}\boldsymbol{u}_t \tag{5.6}$$

is compared to the zero vector to estimate the accuracy of the solution vector $\boldsymbol{u}$.

5.2.1 The Conjugate Gradient method

The defacto iterative method for symmetric positive definite matrices is Conjugate Gradient. At each iteration of this method, the next approximation of the solution vector $\boldsymbol{u}_{t+1}$ is sought starting from the previous $\boldsymbol{u}_t$ and moving along a direction vector $\boldsymbol{d}$ for a step size a_t:

$$\boldsymbol{u}_{t+1} = \boldsymbol{u}_t + a_t \boldsymbol{d}_t \tag{5.7}$$

The step size, which is a scalar quantity, is calculated from line search as

$$a_t = \frac{\boldsymbol{r}_t^T \boldsymbol{r}_t}{\boldsymbol{d}_t^T \boldsymbol{K} \boldsymbol{d}_t} \tag{5.8}$$

The direction vectors are chosen from a basis of the vector space that contains vectors that are conjugate (also called $\boldsymbol{K}$-orthogonal) with each other:

$$\boldsymbol{d}_i \boldsymbol{K} \boldsymbol{d}_j = 0, \quad \text{if } i \neq j \tag{5.9}$$

which can be achieved if

$$\begin{aligned} \boldsymbol{d}_{t+1} &= \boldsymbol{r}_{t+1} + \beta_{t+1} \boldsymbol{d}_t \\ \beta_{t+1} &= \frac{\boldsymbol{r}_{t+1}^T \boldsymbol{r}_{t+1}}{\boldsymbol{r}_t^T \boldsymbol{r}_t} \end{aligned} \tag{5.10}$$

All operations are between vectors, except for the matrix-vector multiplications $\boldsymbol{K}\boldsymbol{d}_t$, $\boldsymbol{K}\boldsymbol{u}_t$. The matrix-vector multiplications are optimized by using sparse matrix storage formats, that are designed specifically for efficient matrix-vector multiplication operations. Two popular storage formats are the Compressed Sparse Rows (CSR) format for CPU implementations, which only stores the non-zero entries, their column indices and a compressed representation of their row indices, and the ELLPACK format for GPU implementations. Even then, the vector operations have negligible cost compared to the matrix-vector multiplications. Consequently, it is desirable to limit the latter into only one matrix-vector multiplication $\boldsymbol{K}\boldsymbol{d}_t$ per iteration t by calculating the residual vector as

$$\boldsymbol{r}_{t+1} = \boldsymbol{r}_t - a_t \boldsymbol{K} \boldsymbol{d}_t \tag{5.11}$$

where $\boldsymbol{K}\boldsymbol{d}_t$ has already been performed for the calculation of the step size a_t. In theory, conjugate gradient will find the exact solution after n iterations, where n is the number of rows/columns of the matrix. However, when the operations are executed by a computer, limited precision and round-off errors will increase the required number of iterations. In any case, once the residual vector $\boldsymbol{r}$ is within some tolerance ϵ, which happens after much fewer than n iterations for most ϵ chosen in practice, conjugate gradient has converged to a reasonably accurate solution $\boldsymbol{x}$. The convergence rate of Conjugate Gradient, like all iterative methods, is very sensitive to the *condition number* of the matrix

$$c(\boldsymbol{K}) = \frac{\lambda_{max}(\boldsymbol{K})}{\lambda_{min}(\boldsymbol{K})} \tag{5.12}$$

where $\lambda_{min}(\boldsymbol{K})$, $\lambda_{max}(\boldsymbol{K})$ are the the minimum and maximum, respectively, absolute values of the eigenvalues of $\boldsymbol{K}$. For a positive definite matrix $\boldsymbol{K}$, all eigenvalues are positive numbers. Additionally, the identity matrix I has all eigenvalues equal to 1, which leads to the minimum possible condition number $c(\boldsymbol{I}) = 1$. The convergence rate of Conjugate

Gradient depends on this condition number. Let $\|\boldsymbol{e}_t\|_{\boldsymbol{K}}$ be the energy norm of the error vector

$$\|\boldsymbol{e}_t\|_{\boldsymbol{K}} = \sqrt{\boldsymbol{e}_t^T \boldsymbol{K} \boldsymbol{e}_t} \tag{5.13}$$

Then, the convergence rate of Conjugate Gradient is quantified by the ratio of the current error norm to the error norm of the initial iteration

$$\frac{\|\boldsymbol{e}_t\|_{\boldsymbol{K}}}{\|\boldsymbol{e}_0\|_{\boldsymbol{K}}} = \left(\frac{c(\boldsymbol{K}) - 1}{c(\boldsymbol{K}) + 1} \right)^t \tag{5.14}$$

Matrices with a large spread of their eigenvalues will have high condition numbers and the rate at which the error decreases will be lower, therefore Conjugate Gradient will require a high number of iterations to converge to the desired tolerance. The matrices, linear systems and generally the mechanics problems themselves are call *ill-conditioned* in this case. Unfortunately, problems with heterogeneous materials, high Poisson ratios or the crack-tip enrichment functions used in XFEM (see equation(4.28)) are strongly ill-conditioned, which causes Conjugate Gradient and any other iterative solver to be inefficient.

5.2.2 Preconditioned Conjugate Gradient

The convergence rate of Conjugate Gradient can be increased with a technique called *preconditioning*. Instead of the original linear system $\boldsymbol{K}\boldsymbol{u} = \boldsymbol{f}$, the equivalent system

$$\boldsymbol{P}^{-1}\boldsymbol{K}\boldsymbol{u} = \boldsymbol{P}^{-1}\boldsymbol{f} \tag{5.15}$$

is solved. The matrix $\boldsymbol{P}^{-1}$ (or sometimes $\boldsymbol{P}$ itself) is called the *preconditioner* matrix. The preconditioner $\boldsymbol{P}^{-1}$ is usually an approximation of the inverse $\boldsymbol{K}^{-1}$, so that the matrix $\boldsymbol{P}^{-1}\boldsymbol{K}$ of the new linear system has a condition number $c\left(\boldsymbol{P}^{-1}\boldsymbol{K}\right)$ close to 1 or at least lower than the original $c\left(\boldsymbol{K}\right)$. Therefore, solving this linear system usually requires less iterations and computing time. The original Conjugate Gradient is modified to solve the preconditioned linear system leading to the Preconditioned Conjugate Gradient (PCG) method, which will be used in the rest of this dissertation and is elaborated in algorithm 5.6.

Algorithm 5.6 PCG solution of $\boldsymbol{K}\boldsymbol{u} = \boldsymbol{f}$, with preconditioner $\boldsymbol{P}^{-1}$

ϵ = desired tolerance, t_{max} = maximum iterations
$t = 0$
$\boldsymbol{u}_0 = \boldsymbol{0}$ or another initial guess
$\boldsymbol{r}_0 = \boldsymbol{f} - \boldsymbol{K}\boldsymbol{u}_0$
$\boldsymbol{d}_0 = \boldsymbol{s}_0 = \boldsymbol{P}^{-1}\boldsymbol{r}_0$
$\rho_0 = \boldsymbol{r}_0^T \boldsymbol{s}_0$
while $t < t_{max}$ **do**
    $\boldsymbol{q}_t = \boldsymbol{K}\boldsymbol{d}_t$
    $a_t = \dfrac{\boldsymbol{r}_t^T \boldsymbol{s}_t}{\boldsymbol{d}_t^T \boldsymbol{q}_t}$
    $\boldsymbol{u}_{t+1} = \boldsymbol{u}_t + a_t \boldsymbol{d}_t$
    $\boldsymbol{r}_{t+1} = \boldsymbol{r}_t - a_t \boldsymbol{q}_t$
    $\boldsymbol{s}_{t+1} = \boldsymbol{P}^{-1}\boldsymbol{r}_{t+1}$
    $\rho_{t+1} = \boldsymbol{r}_{t+1}^T \boldsymbol{s}_{t+1}$
    if $\rho_{t+1} < \epsilon\ \rho_0$ **then**
        Stop and return $\boldsymbol{u}_{t+1}$ ▷ PCG has converged
    end if
    $\beta_{t+1} = \dfrac{\rho_{t+1}}{\rho_t}$
    $\boldsymbol{d}_{t+1} = \boldsymbol{s}_{t+1} + \beta_{t+1} \boldsymbol{d}_t$
    $t = t + 1$
end while

Nevertheless, choosing a preconditioner $\boldsymbol{P}^{-1}$ is not straightforward and problem-specific preconditioners need to be developed usually. A good preconditioner must satisfy the following criteria:

1. $\boldsymbol{P}$ must be symmetric positive definite.
2. The number of iterations should be reduced as much as possible, namely the condition number $c\left(\boldsymbol{P}^{-1}\boldsymbol{K}\right)$ must be as low as possible.
3. The preconditioner $\boldsymbol{P}^{-1}$ must be calculated efficiently, in terms of memory and time required, before PCG starts.
4. The preconditioner $\boldsymbol{P}^{-1}$ must be applied effectively during the PCG iterations of algorithm 5.6, which is expressed by the multiplication $\boldsymbol{P}^{-1}\boldsymbol{r}_t$.

Unfortunately, criteria 2, 3, 4 are usually contradictory and a reasonable trade-off is sought. For example, the diagonal preconditioner (also called Jacobi preconditioner) is the most basic one and very efficient to both calculate:

$$\boldsymbol{P}_D^{-1} = \begin{bmatrix} \frac{1}{K_{11}} & & \\ & \ddots & \\ & & \frac{1}{K_{nn}} \end{bmatrix} \tag{5.16}$$

and apply during PCG:

$$\boldsymbol{P}_D^{-1}\boldsymbol{r} = \begin{bmatrix} \frac{r_1}{K_{11}} \\ \vdots \\ \frac{r_n}{K_{nn}} \end{bmatrix} \tag{5.17}$$

but does not reduce the number of iterations as much as more sophisticated options. Diagonal preconditioning is very effective in reducing the condition number when the matrix is strongly diagonally dominant. However, it can be applied to all problems, in contrast with more complicated preconditioners that can become unstable, has negligible computing cost and can be parallelized extremely efficiently in both shared memory and distributed memory computing systems. Consequently, it is always a good choice to try, unless a better option is available.

On the opposite end, the Cholesky (and LDL) methods can also be used for preconditioning. The initial calculation involves the factorization of the system matrix, while the application step involves performing the back/forward substitutions. By doing so, the preconditioner $\boldsymbol{P}^{-1}$ coincides with the inverse matrix $\boldsymbol{K}^{-1}$ and PCG converges after only 1 iteration. Of course, this preconditioner is the most costly to calculate and store, negating all the advantages PCG may have over the corresponding direct solver. While, this preconditioner is not used in practice, approximate versions of it, such as incomplete Cholesky, may be employed. Finally, the domain decomposition solvers investigated in this dissertation will define their own preconditioners.

5.2.3 Advantages and disadvantages

Advantages of iterative solvers:

- The memory requirements are minimized, since only non-zero entries need to be explicitly stored, without any fill-in occurring.
- In 3D and large-scale problems they usually outperform direct solvers.
- If a good preconditioning strategy is known for the specific problem, then the iterations and computation time can be greatly reduced.

- The convergence tolerance can be adjusted to improve performance at the cost of lower accuracy, if the latter is not important.
- The matrix-vector multiplications, vector-vector operations and some preconditioners can be parallelized very efficiently in both shared memory and distributed memory systems.
- There is no need to explicitly store the matrix $\boldsymbol{K}$ or the preconditioner $\boldsymbol{P}^{-1}$. Instead, only a way to perform the matrix vector multiplications $\boldsymbol{K}\cdot\boldsymbol{d}$ and $\boldsymbol{P}^{-1}\cdot\boldsymbol{r}$ is needed. This is a huge benefit, because it allows the decomposition of $\boldsymbol{K}$ and $\boldsymbol{P}^{-1}$ into submatrices that exist on different computers, without actually forming $\boldsymbol{K}$ and $\boldsymbol{P}^{-1}$ on any coputer explicitly. Instead the matrix-vector multiplications are performed by processing each submatrix independently and possibly in parallel and then summing the intermediate results. For example, this is the strategy employed when using PCG in the domain decomposition solvers that will be presented in the next sections.

Disadvantages of iterative solvers:

- They are very sensitive to ill-conditioning, unlike direct solvers.
- There is no universally good preconditioner. Instead, problem-specific preconditioners must be developed and tried for each case. The complexity and computational cost of these preconditioners often exceeds that of the iterative method itself.
- They converge to an approximation of the solution, instead of the exact one.
- Predicting the number of iterations and thus the time required is generally impossible, since calculating the eigenvalues needed for the condition number is actually more difficult than solving the linear system.

5.3 Domain decomposition methods

In this section, a review of the FETI-DP and P-FETI-DP domain decomposition methods (DDM) will be presented. Both belong to the finite element tearing and interconnecting (FETI) family of algorithms, the high performance of which has been established in standard FEM (Fragakis & Papadrakakis, 2003), meshless methods (Metsis & Papadrakakis, 2012) and isogeometric analysis (Stavroulakis et al., 2012). Moreover, both solvers have shown to exhibit high numerical and parallel scalability properties, since their convergence rate increases as more subdomains are used (Farhat et al., 2000; Fragakis & Papadrakakis, 2003). In FETI-DP and P-FETI-DP, an iterative algorithm, such as the preconditioned conjugate gradient (PCG), is used to connect individual subdomains and calculate the forces required for the equilibrium (FETI-DP) or the displacements required for the compatibility (P-FETI-DP) at boundary nodes. Boundary nodes are those that belong to two or more subdomains, while internal nodes belong to only one subdomain, as shown in figure 5.1.

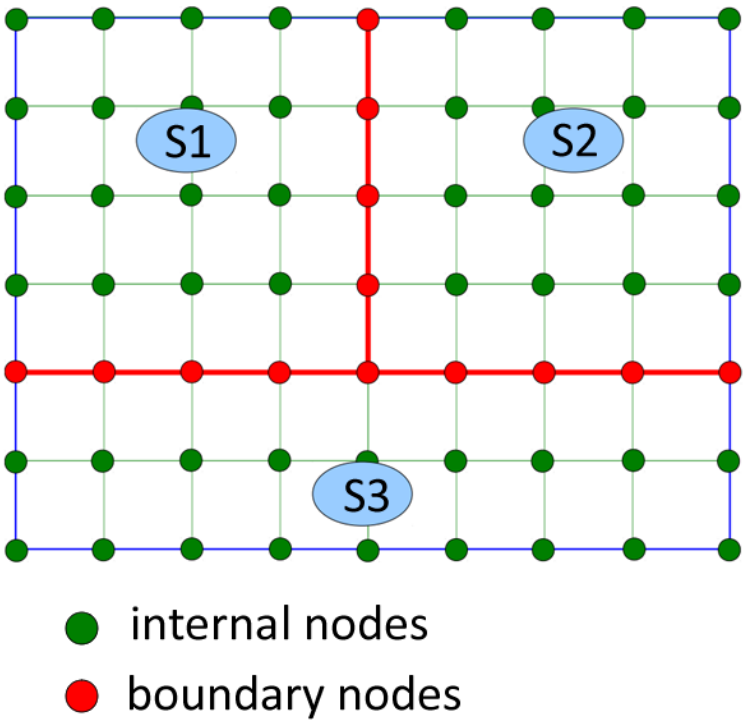


Figure 5.1: Boundary and internal nodes of subdomains.

5.3.1 FETI-DP

5.3.1.1 Primal and dual DOFs

The dual-primal FETI (FETI-DP) was introduced in Farhat et al. (2000) to improve the scalability of the original FETI method (Farhat & Roux, 1991). This is achieved by defining corner nodes, which are a subset of boundary nodes that lie on the beginning or end of each geometric edge of each subdomain, as illustrated in figure 5.2. Sometimes additional corner nodes can be used, to ensure that each subdomain has at least two (in 2D problems) or three (in 3D problems) non-colinear corner nodes. The corner DOFs associated with these corner nodes are indicated by the subscript c. The remainder DOFs of each subdomain are indicated by the subscript r. The stiffness matrix $\boldsymbol{K}^s$, displacement vector $\boldsymbol{u}^s$ and force vector $\boldsymbol{f}^s$ of subdomain s are therefore decomposed as follows:

$$\boldsymbol{K}^s = \begin{bmatrix} \boldsymbol{K}^s_{rr} & \boldsymbol{K}^s_{rc} \\ (\boldsymbol{K}^s_{rc})^T & \boldsymbol{K}^s_{cc} \end{bmatrix} \quad \boldsymbol{u}^s = \begin{bmatrix} \boldsymbol{u}^s_r \\ \boldsymbol{u}^s_c \end{bmatrix} \quad \boldsymbol{f}^s = \begin{bmatrix} \boldsymbol{f}^s_r \\ \boldsymbol{f}^s_c \end{bmatrix} \tag{5.18}$$

The remainder DOFs are further divided into internal DOFs, which are associated with nodes that belong to only one subdomain and are indicated by the subscript i, and boundary-remainder DOFs, which are indicated by the subscript b_r and are associated with boundary nodes that are not corners:

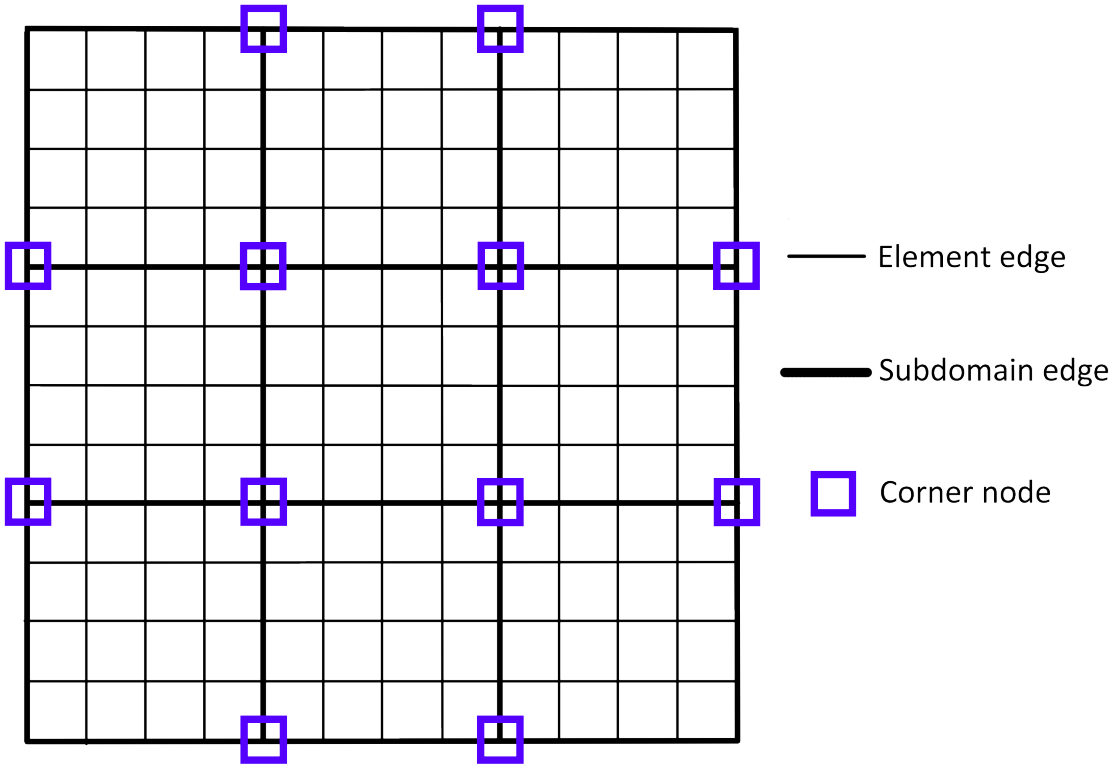


Figure 5.2: Definition of corner nodes.

$$\boldsymbol{K}_{rr}^{s} = \begin{bmatrix} \boldsymbol{K}_{ii}^{s} & \boldsymbol{K}_{ib_r}^{s} \\ (\boldsymbol{K}_{ib_r}^{s})^T & \boldsymbol{K}_{b_r b_r}^{s} \end{bmatrix} \quad \boldsymbol{u}_{r}^{s} = \begin{bmatrix} \boldsymbol{u}_{i}^{s} \\ \boldsymbol{u}_{b_r}^{s} \end{bmatrix} \quad \boldsymbol{f}_{r}^{s} = \begin{bmatrix} \boldsymbol{f}_{i}^{s} \\ \boldsymbol{f}_{b_r}^{s} \end{bmatrix} \tag{5.19}$$

The displacements corresponding to all corner DOFs of the global domain are gathered in the vector $\boldsymbol{u}_c$ of length n_c. The mapping between $\boldsymbol{u}_c$ and $\boldsymbol{u}_c^s$ is defined as boolean matrices $\boldsymbol{L}_c^s$ that have 0, 1 entries and dimensions $(n_c^s \times n_c)$, where n_c^s and n_c are the number of corner DOFs of subdomain s and the global domain, respectively. An entry i, j of $\boldsymbol{L}_c^s$ is equal to 1, if the corner DOF that corresponds to subdomain-level row i, is the same as the corner DOF that corresponds to global-level column j.

$$\boldsymbol{u}_c^s = \boldsymbol{L}_c^s \boldsymbol{u}_c \tag{5.20}$$

The continuity between the otherwise disconnected subdomains is retained by enforcing compatibility conditions for instances of the same boundary DOF in different subdomains:

$$\begin{bmatrix} 1 & -1 \end{bmatrix} \begin{bmatrix} u_k^{(s_i)} \\ u_k^{(s_j)} \end{bmatrix} = 0 \tag{5.21}$$

where the subscript k denotes any boundary-remainder DOF that belongs to subdomains s_i and s_j. These continuity equations are gathered into signed boolean matrices $\boldsymbol{B}_r^s$, which have 0, 1, -1 entries and dimensions $(n_\lambda \times n_r^s)$, where n_λ is the number of continuity equations of the global domain and n_r^s the number of remainder DOFs of subdomain s:

$$\sum_{s=1}^{n_s} \boldsymbol{B}_r^s \boldsymbol{u}_r^s = \boldsymbol{0} \tag{5.22}$$

To solve the global equilibrium equations $\boldsymbol{K}\boldsymbol{u} = \boldsymbol{f}$ in the presence of these constraints, the Lagrange multipliers $\boldsymbol{\lambda}$ are applied at boundary-remainder DOFs to enforce displacement compatibility, as illustrated in figure 5.3. It should be clarified that no Lagrange multipliers are applied at corner DOFs.

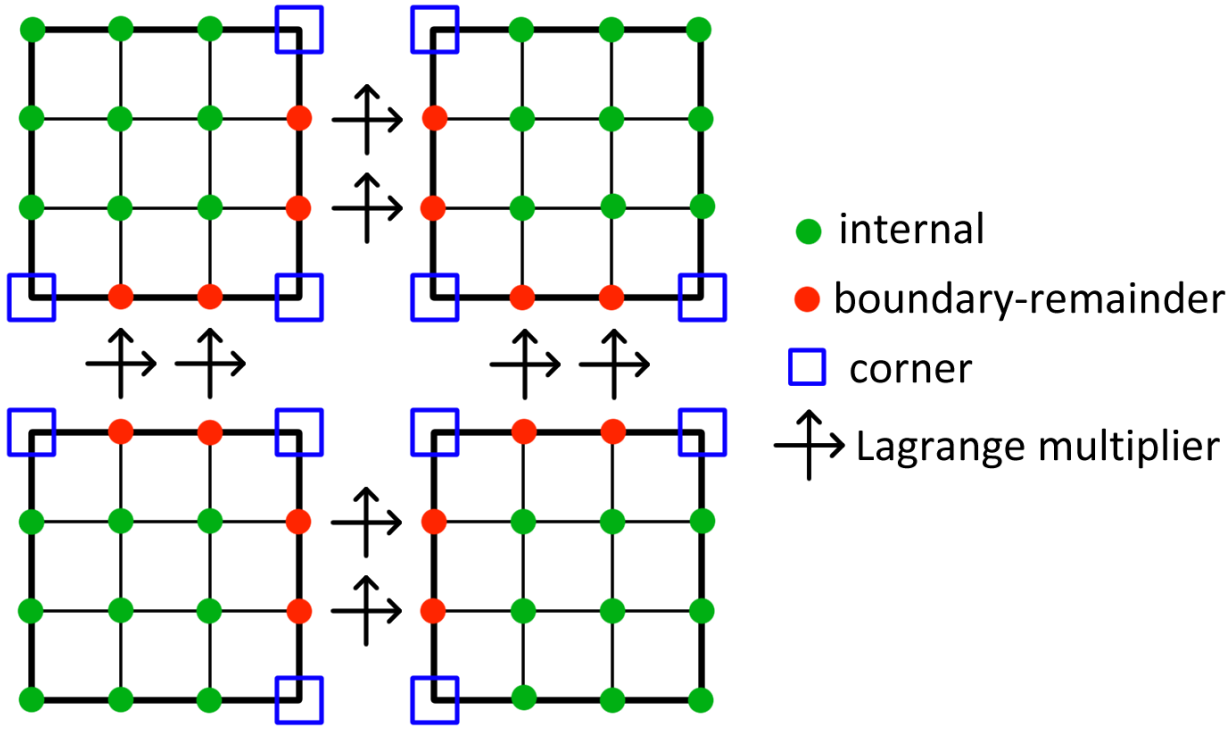


Figure 5.3: Langrange multipliers applied to boundary DOFs of subdomains. In the case of nodes belonging to 2 subdomains, only one Lagrange multiplier is needed per DOF.

In 3D problems, there are nodes that lie on the common edge between 3 or more subdomains, excluding its corners. For these cross-point nodes, multiple continuity equations and corresponding Lagrange multipliers per boundary DOF must be applied. This is achieved with the fully redundant constraint strategy (Farhat & Roux, 1994), which increases the

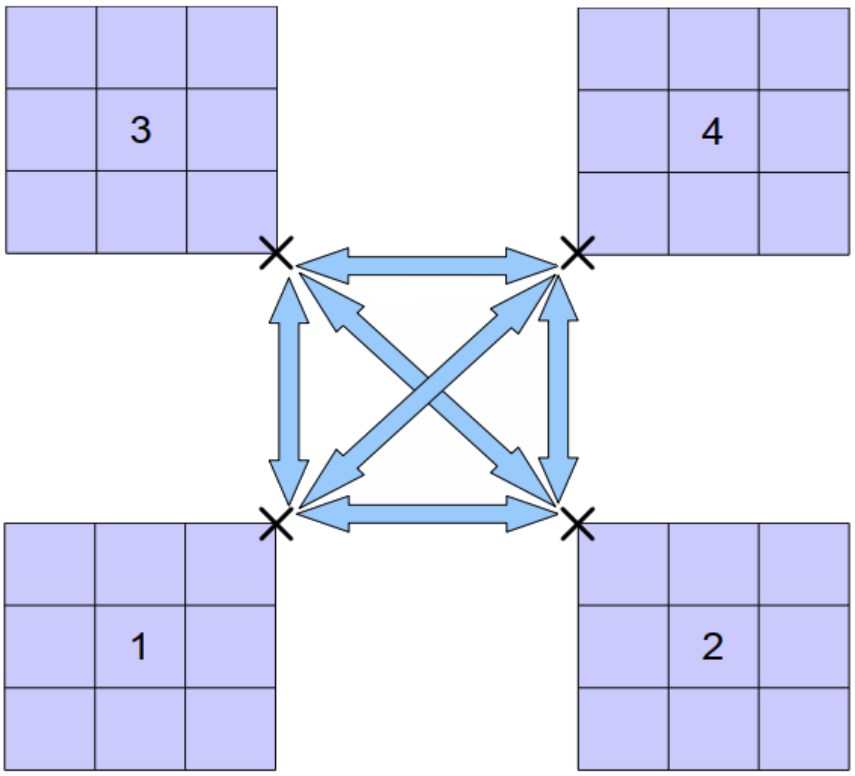


Figure 5.4: Langrange multipliers applied to boundary DOFs of subdomains. In the case of cross-points nodes belonging to 3 or more subdomains, multiple Lagrange multipliers are needed per DOF.

convergence rate of FETI-DP by applying a Lagrange multiplier between each pair of instances of the same DOF in different subdomains, as shown in figure 5.4. The equilibrium equations can then be written as

$$\boldsymbol{K}_{rr}^{s}\boldsymbol{u}_{r}^{s}+\boldsymbol{K}_{rc}^{s}\boldsymbol{L}_{c}^{s}\boldsymbol{u}_{c}+(\boldsymbol{B}_{r}^{s})^{T}\boldsymbol{\lambda}=\boldsymbol{f}_{r}^{s} \tag{5.23}$$

$$\sum_{s=1}^{n_s}(\boldsymbol{L}_{c}^{s})^{T}(\boldsymbol{K}_{rc}^{s})^{T}\boldsymbol{u}_{r}^{s}+\sum_{s=1}^{n_s}(\boldsymbol{L}_{c}^{s})^{T}(\boldsymbol{K}_{cc}^{s})^{T}\boldsymbol{L}_{c}^{s}\boldsymbol{u}_{c}^{s}=\sum_{s=1}^{n_s}(\boldsymbol{L}_{c}^{s})^{T}\boldsymbol{f}_{c}^{s} \tag{5.24}$$

5.3.1.2 Interface problem of FETI-DP

By performing static condensation of the remainder DOFs, the Schur complement $\boldsymbol{S}_{cc}^{s}$ of each subdomain's $\boldsymbol{K}_{rr}^{s}$ and the corresponding force vector $\boldsymbol{z}_{c}^{s}$ are calculated

$$\boldsymbol{S}_{cc}^{s}=\boldsymbol{K}_{cc}^{s}-(\boldsymbol{K}_{rc}^{s})^{T}(\boldsymbol{K}_{rr}^{s})^{-1}\boldsymbol{K}_{rc}^{s} \tag{5.25}$$

$$\hat{\boldsymbol{z}}_c^s = \boldsymbol{f}_c^s - (\boldsymbol{K}_{rc}^s)^T (\boldsymbol{K}_{rr}^s)^{-1} \boldsymbol{f}_r^s \tag{5.26}$$

Then these subdomain-level matrices and vectors are summed to obtain the global-level corner stiffness matrix $\boldsymbol{S}_{cc}$ and corresponding corner force vector $\boldsymbol{z}_c$

$$\boldsymbol{S}_{cc} = \sum_{s=1}^{n_s} (\boldsymbol{L}_c^s)^T \boldsymbol{S}_{cc}^s \boldsymbol{L}_c^s = \sum_{s=1}^{n_s} (\boldsymbol{L}_c^s)^T \left(\boldsymbol{K}_{cc}^s - (\boldsymbol{K}_{rc}^s)^T (\boldsymbol{K}_{rr}^s)^{-1} \boldsymbol{K}_{rc}^s \right) \boldsymbol{L}_c^s \tag{5.27}$$

$$\boldsymbol{z}_c = \sum_{s=1}^{n_s} (\boldsymbol{L}_c^s)^T \hat{\boldsymbol{z}}_c^s = \sum_{s=1}^{n_s} (\boldsymbol{L}_c^s)^T \left(\boldsymbol{f}_c^s - (\boldsymbol{K}_{rc}^s)^T (\boldsymbol{K}_{rr}^s)^{-1} \boldsymbol{f}_r^s \right) \tag{5.28}$$

With an appropriate selection of corner nodes, $\boldsymbol{K}_{rr}^s$ is a positive definite matrix, which means that its Schur complement $\boldsymbol{S}_{cc}^s$ and eventually $\boldsymbol{S}_{cc}$ are also positive definite matrices. Equation (5.23) can also be written as:

$$\boldsymbol{u}_r^s = (\boldsymbol{K}_{rr}^s)^{-1} \left(\boldsymbol{f}_r^s - (\boldsymbol{B}_r^s)^T \boldsymbol{\lambda} - \boldsymbol{K}_{rc}^s \boldsymbol{L}_c^s \boldsymbol{u}_c \right) \tag{5.29}$$

Substituting equation(5.29) into equations (5.22) and (5.24) and decoupling the unknowns u_c and λ, results in

$$\boldsymbol{u}_c = \boldsymbol{S}_{cc}^{-1} \left(\boldsymbol{z}_c + \boldsymbol{F}_{Icr} \boldsymbol{\lambda} \right) \tag{5.30}$$

$$\left(\boldsymbol{F}_{Irr} + \boldsymbol{F}_{Irc} \boldsymbol{S}_{cc}^{-1} \boldsymbol{F}_{Icr} \right) \boldsymbol{\lambda} = \boldsymbol{d}_r - \boldsymbol{F}_{Irc} \boldsymbol{S}_{cc}^{-1} \boldsymbol{z}_c \tag{5.31}$$

where

$$\boldsymbol{F}_{Irr} = \sum_{s=1}^{n_s} \boldsymbol{B}_r^s (\boldsymbol{K}_{rr}^s)^{-1} (\boldsymbol{B}_r^s)^T \tag{5.32}$$

$$\boldsymbol{F}_{Irc} = \sum_{s=1}^{n_s} \boldsymbol{B}_r^s (\boldsymbol{K}_{rr}^s)^{-1} \boldsymbol{K}_{rc}^s \boldsymbol{L}_c^s \tag{5.33}$$

$$\boldsymbol{F}_{Icr} = \boldsymbol{F}_{Irc}^T = \sum_{s=1}^{n_s} (\boldsymbol{L}_c^s)^T \boldsymbol{K}_{cr}^s (\boldsymbol{K}_{rr}^s)^{-1} (\boldsymbol{B}_r^s)^T \tag{5.34}$$

$$\boldsymbol{d}_r = \sum_{s=1}^{n_s} \boldsymbol{B}_r^s (\boldsymbol{K}_{rr}^s)^{-1} \boldsymbol{f}_r^s \tag{5.35}$$

Note that $\boldsymbol{F}_{Irr}$ is a flexibility matrix, while $\boldsymbol{d}_r$ expresses displacements. The linear system of equation (5.31) is the interface problem of FETI-DP. The matrix of equation (5.31) is positive definite and the solution for $\boldsymbol{\lambda}$ can be done using an iterative algorithm, such as the PCG method. During each matrix-vector multiplication of PCG, in the form of $\left(\boldsymbol{F}_{Irr} + \boldsymbol{F}_{Irc} \boldsymbol{S}_{cc}^{-1} \boldsymbol{F}_{Icr} \right) \cdot \boldsymbol{\lambda}$, the linear system

$$\boldsymbol{S}_{cc} \cdot \boldsymbol{x}_c = \boldsymbol{y}_c \tag{5.36}$$

needs to be solved, where $\boldsymbol{y}_c = \boldsymbol{F}_{Icr} \cdot \boldsymbol{\lambda}$. This linear system, defined as the coarse problem of FETI-DP, is a much smaller auxiliary problem that speeds up convergence by coupling the subdomain computations and globally propagating the error at each PCG iteration. After solving the interface problem and obtaining the Lagrange multipliers $\boldsymbol{\lambda}$, the displacements at corner and remainder DOFs can be calculated using equations (5.30) and (5.29). However, the displacements at different instances of the same boundary DOF will be slightly different in each subdomain. To obtain compatible displacements across all subdomains the following averaging operation needs to be done:

$$\check{\boldsymbol{u}}_b^s = \boldsymbol{L}_b^s (\boldsymbol{L}_b^s)^T \boldsymbol{W}_b^s \boldsymbol{u}_b^s \tag{5.37}$$

where $\boldsymbol{u}_b^s$ are the incompatible displacements at boundary DOFs of subdomain s, $\check{\boldsymbol{u}}_b^s$ are the corrected, compatible displacements at the same DOFs, $\boldsymbol{L}_b^s$ is a mapping matrix defined in section 5.3.2.1 and $\boldsymbol{W}_b^s$ is a scaling matrix defined in section 5.3.2.2.

5.3.1.3 Preconditioners of FETI-DP

Using a preconditioner is necessary to reduce the number of iterations of PCG. A preconditioner is an approximation to the flexibility-like matrix of the interface problem. Therefore, FETI-DP preconditioners incorporate the stiffness matrices of subdomains:

$$\widetilde{\boldsymbol{F}}_I^{-1} = \sum_{s=1}^{n_s} \boldsymbol{B}_{pb_r}^s \widetilde{\boldsymbol{S}}_{b_r b_r}^s (\boldsymbol{B}_{pb_r}^s)^T \tag{5.38}$$

where $\widetilde{\boldsymbol{S}}_{b_r b_r}^s$ is an approximation of the Schur complement of internal DOFs and $\boldsymbol{B}_{pb_r}^s$ is a scaling-mapping matrix. In elasticity (2nd order) problems with homogeneous stiffness distribution among subdomains, $\boldsymbol{B}_{pb_r}^s$ is given by

$$\boldsymbol{B}_{pb_r}^s = \boldsymbol{B}_{b_r}^s \boldsymbol{W}_{b_r}^s \tag{5.39}$$

where $\boldsymbol{B}_{b_r}^s$ are the columns of $\boldsymbol{B}_r^s$ that correspond to the boundary-remainder DOFs of subdomain s and $\boldsymbol{W}_{b_r}^s$ is the inverse of a diagonal matrix, whose entries are the multiplicities of these DOFs. The multiplicity of a DOF is defined as the number of subdomains that contain the node associated with that DOF. Depending on the definition of $\widetilde{\boldsymbol{S}}_{b_r b_r}^s$ the following preconditioners can be obtained:

$$\widetilde{\boldsymbol{S}}_{b_r b_r}^s = \begin{cases} \boldsymbol{K}_{b_r b_r}^s - (\boldsymbol{K}_{ib_r}^s)^T (\boldsymbol{K}_{ii}^s)^{-1} \boldsymbol{K}_{ib_r}^s & \text{Dirichlet preconditioner} \\ \boldsymbol{K}_{b_r b_r}^s & \text{lumped preconditioner} \end{cases} \tag{5.40}$$

Dirichlet preconditioner uses the full Schur complement of internal DOFs, therefore it is more costly to calculate and implement, but results in fewer PCG iterations, in comparison to the lumped preconditioner, which is less effective in improving the convergence of PCG.

5.3.2 P-FETI-DP

P-FETI-DP has been introduced in Fragakis and Papadrakakis (2003) as a hybrid approach that combines the high computational efficiency of FETI methods with the robustness of primal methods in ill-conditioned problems. Essentially, P-FETI-DP is equivalent to a primal substructuring method (PSM) that uses the first iteration of FETI-DP as its preconditioner. The resulting algorithm is a simpler, yet more efficient alternative to FETI-DP that avoids any need for Lagrange multipliers.

5.3.2.1 Interface problem of P-FETI-DP

In P-FETI-DP the stiffness matrix, displacement and force vectors are decomposed into parts corresponding to internal (subscript i) and boundary (subscript b) DOFs. In contrast to the boundary-remainder DOFs of FETI-DP, here boundary DOFs comprise all DOFs belonging to two or more subdomains, without considering corner nodes into account:

$$\boldsymbol{K}^s = \begin{bmatrix} \boldsymbol{K}^s_{ii} & \boldsymbol{K}^s_{ib} \\ (\boldsymbol{K}^s_{ib})^T & \boldsymbol{K}^s_{bb} \end{bmatrix} \quad \boldsymbol{u}^s = \begin{bmatrix} \boldsymbol{u}^s_i \\ \boldsymbol{u}^s_b \end{bmatrix} \quad \boldsymbol{f}^s = \begin{bmatrix} \boldsymbol{f}^s_i \\ \boldsymbol{f}^s_b \end{bmatrix} \tag{5.41}$$

The displacements corresponding to all boundary DOFs of the global domain are gathered in the vector $\boldsymbol{u}_b$ of length n_b. The mapping between $\boldsymbol{u}_b$ and $\boldsymbol{u}^s_b$ is defined as boolean matrices $\boldsymbol{L}^s_b$ that have 0, 1 entries and dimensions $(n^s_b \times n_b)$, where n^s_b and n_b are the number of boundary DOFs of subdomain s and the global domain, respectively. An entry i, j of $\boldsymbol{L}^s_b$ is equal to 1, if the boundary DOF that corresponds to subdomain-level row i, is the same as the boundary DOF that corresponds to global-level column j.

$$\boldsymbol{u}^s_b = \boldsymbol{L}^s_b \boldsymbol{u}_b \tag{5.42}$$

By performing static condensation of the internal DOFs, the Schur complement $\boldsymbol{S}^s_{bb}$ of each subdomain's $\boldsymbol{K}^s_{ii}$ and the corresponding force vector $\boldsymbol{z}^s_b$ are calculated

$$\boldsymbol{S}^s_{bb} = \boldsymbol{K}^s_{bb} - (\boldsymbol{K}^s_{ib})^T (\boldsymbol{K}^s_{ii})^{-1} \boldsymbol{K}^s_{ib} \tag{5.43}$$

$$\hat{\boldsymbol{z}}^s_b = \boldsymbol{f}^s_b - (\boldsymbol{K}^s_{ib})^T (\boldsymbol{K}^s_{ii})^{-1} \boldsymbol{f}^s_b \tag{5.44}$$

Then these subdomain-level matrices and vectors are summed to obtain the global-level boundary stiffness matrix $\boldsymbol{S}_{bb}$ and corresponding boundary force vector $\boldsymbol{z}_b$

$$\boldsymbol{S}_{bb} = \sum_{s=1}^{n_s} (\boldsymbol{L}^s_b)^T \boldsymbol{S}^s_{bb} \boldsymbol{L}^s_b = \sum_{s=1}^{n_s} (\boldsymbol{L}^s_b)^T \left(\boldsymbol{K}^s_{bb} - (\boldsymbol{K}^s_{ib})^T (\boldsymbol{K}^s_{ii})^{-1} \boldsymbol{K}^s_{ib} \right) \boldsymbol{L}^s_b \tag{5.45}$$

$$\boldsymbol{z}_b = \sum_{s=1}^{n_s} (\boldsymbol{L}^s_b)^T \hat{\boldsymbol{z}}^s_b = \sum_{s=1}^{n_s} (\boldsymbol{L}^s_b)^T \left(\boldsymbol{f}^s_b - (\boldsymbol{K}^s_{ib})^T (\boldsymbol{K}^s_{ii})^{-1} \boldsymbol{f}^s_i \right) \tag{5.46}$$

As in FETI-DP, $\boldsymbol{K}_{ii}^s$, $\boldsymbol{S}_{bb}^s$ and $\boldsymbol{S}_{bb}$ are positive definite matrices. The interface problem of P-FETI-DP is then solved iteratively by PCG:

$$\boldsymbol{S}_{bb} \cdot \boldsymbol{u}_b = \boldsymbol{z}_b \tag{5.47}$$

After obtaining the global boundary displacements $\boldsymbol{u}_b$, the boundary displacements of each subdomain s can be computed using equation(5.42) and the internal displacements $\boldsymbol{u}_i^s$ as

$$\boldsymbol{u}_i^s = \left(\boldsymbol{K}_{ii}^s\right)^{-1} \left(\boldsymbol{f}_i^s - \boldsymbol{K}_{ib}^s \boldsymbol{u}_b^s\right) \tag{5.48}$$

5.3.2.2 Preconditioner of P-FETI-DP

During the iterative solution of the interface problem, the first iteration of FETI-DP is used to provide the preconditioner for PCG. Fragakis and Papadrakakis (2003) proved that in this case, the Lagrange multipliers can be eliminated from the equations and the following matrix form of the preconditioner can be derived

$$\begin{aligned}
\widetilde{\boldsymbol{A}}_{bb}^{-1} &= \boldsymbol{A}_{bb}^r + \boldsymbol{A}_{bc}\boldsymbol{A}_{cc}\boldsymbol{A}_{cb} \\
\boldsymbol{A}_{bb}^r &= \sum_{s=1}^{n_s} \left(\boldsymbol{L}_{pr}^s\right)^T \left(\boldsymbol{K}_{rr}^s\right)^{-1} \boldsymbol{L}_{pr}^s \\
\boldsymbol{A}_{cb} &= -\sum_{s=1}^{n_s} \left(\boldsymbol{L}_c^s\right)^T \boldsymbol{K}_{cr}^s \left(\boldsymbol{K}_{rr}^s\right)^{-1} \boldsymbol{L}_{pr}^s + \boldsymbol{N}_{c,b} \\
\boldsymbol{A}_{bc} &= \boldsymbol{A}_{cb}^T \\
\boldsymbol{A}_{cc} &= \boldsymbol{S}_{cc}^{-1}
\end{aligned} \tag{5.49}$$

where the matrices $\boldsymbol{K}_{rr}^s$, $\boldsymbol{K}_{rc}^s$, $\boldsymbol{K}_{cr}^s$, $\boldsymbol{S}_{cc}$ and $\boldsymbol{L}_c^s$ were defined in section 5.3.1. $\boldsymbol{N}_{c,b}$ is a boolean matrix (0, 1 as entries) with dimensions $(n_c \times n_b)$ that maps the boundary DOFs of the global domain (columns) to the corner DOFs of the global domain (rows). In addition

$$\boldsymbol{L}_{pr}^s = \boldsymbol{N}_{r,b}^s \boldsymbol{L}_{pb}^s \tag{5.50}$$

where $\boldsymbol{N}_{r,b}^s$ is a boolean matrix (0, 1 as entries) with dimensions $(n_r^s \times n_b^s)$ that maps the boundary DOFs of a subdomain (columns) to its remainder DOFs (rows). $\boldsymbol{L}_{pb}^s$ is a scaling-mapping matrix and in elasticity (2nd order) problems with homogeneous stiffness distribution among subdomains, it can be calculated as

$$\boldsymbol{L}_{pb}^s = \boldsymbol{L}_b^s \boldsymbol{W}_b^s \tag{5.51}$$

where $\boldsymbol{W}_b^s$ is the inverse of a diagonal matrix whose entries are the multiplicities of the boundary DOFs of subdomain s. Since there are no Lagrange multipliers involved, the matrix form of the preconditioner in equation (5.49) is a lot simpler to implement and the

size of the interface problem of P-FETI-DP in 3D problems is not increased, unlike FETI-DP where multiple Lagrange multipliers are applied at cross-point nodes. It can be observed that the operations performed by P-FETI-DP and Dirichlet-preconditioned FETI-DP are identical, with the exception of some multiplications with boolean matrices. Since these are extremely sparse matrices, the computing cost of these multiplications is negligible, therefore the amount of work done per iteration of the PCG algorithm is the same for both methods. However, P-FETI-DP tends to converge faster, particularly for ill-conditioned problems, making it superior in terms of computational time.

5.4 Domain decomposition methods for XFEM

In this dissertation two domain decomposition methods, namely the FETI-DP and P-FETI-DP solvers described in section 5.3, are proposed for the solution of the linear systems resulting from XFEM at each step of the crack propagation analysis. Both solvers are particular effective for this type of problems for a number of reasons that will be elaborated in this section. A main advantage over direct solvers developed especially for XFEM crack propagation, such as the incremental Cholesky algorithm of Pais et al. (2012), is that the proposed solvers involve matrices which can be factorized significantly more efficiently, especially in large scale 3D problems. In particular, the matrices $\boldsymbol{K}_{rr}^{s}$, $\boldsymbol{K}_{ii}^{s}$ and $\boldsymbol{S}_{cc}$ of equations (5.18, 5.19, 5.41, 5.27) require factorization, but these are substantially smaller than the global stiffness matrix of a direct solver. Moreover, their bandwidth is reduced, since DOFs of nearby elements, which interact and contribute non-zero entries to stiffness matrices, follow a local DOF ordering at each subdomain with much fewer total DOFs than the global domain. As a result, FETI-DP and P-FETI-DP can achieve significant computation gains in terms of time and memory requirements, especially in 3D problems.

Additionally, both FETI-DP and P-FETI-DP can benefit from their high parallelizable features, since all subdomain-level operations are independent from each other and can be executed concurrently. This inherent parallelism allows the full utilization of multi-core and distributed memory computing systems to further reduce the computation cost. Other domain decomposition solvers for XFEM crack propagation have been developed in X. Chen and Cai (2022), Menk and Bordas (2011), and Waisman and Berger-Vergiat (2013) and can also be executed in parallel, however without being tested in 3D computationally intensive problems. In contrast, this work is focused on producing high performance solvers for large-scale 3D problems, as reflected by the numerical examples investigated in chapter 6.

5.4.1 Resolving crack-specific singularities

FETI-DP and P-FETI-DP solvers, as described in sections 5.3.1 and 5.3.2, were developed for continuum mechanics problems. Thus, it is assumed that the subdomains are sufficiently supported and their $\boldsymbol{K}_{rr}^{s}$ matrices are not singular, as long as corner nodes are selected according to the rules listed in section 5.3.1.1. However, this assumption is not valid in

fracture mechanics, where one or more cracks propagate through a discontinuous domain and some subdomains may be entirely intersected by them. An example of this is shown in figure 5.5, where a single crack propagates throughout a 2D domain and interacts with three subdomains, two of which are completely bisected by the crack. A 2D example is used, since it is can be illustrated more clearly. However, the proposed method can also be applied to 3D problems. In the same figure, the boundary and corner nodes defined in the original FETI-DP and P-FETI-DP algorithms are also depicted.

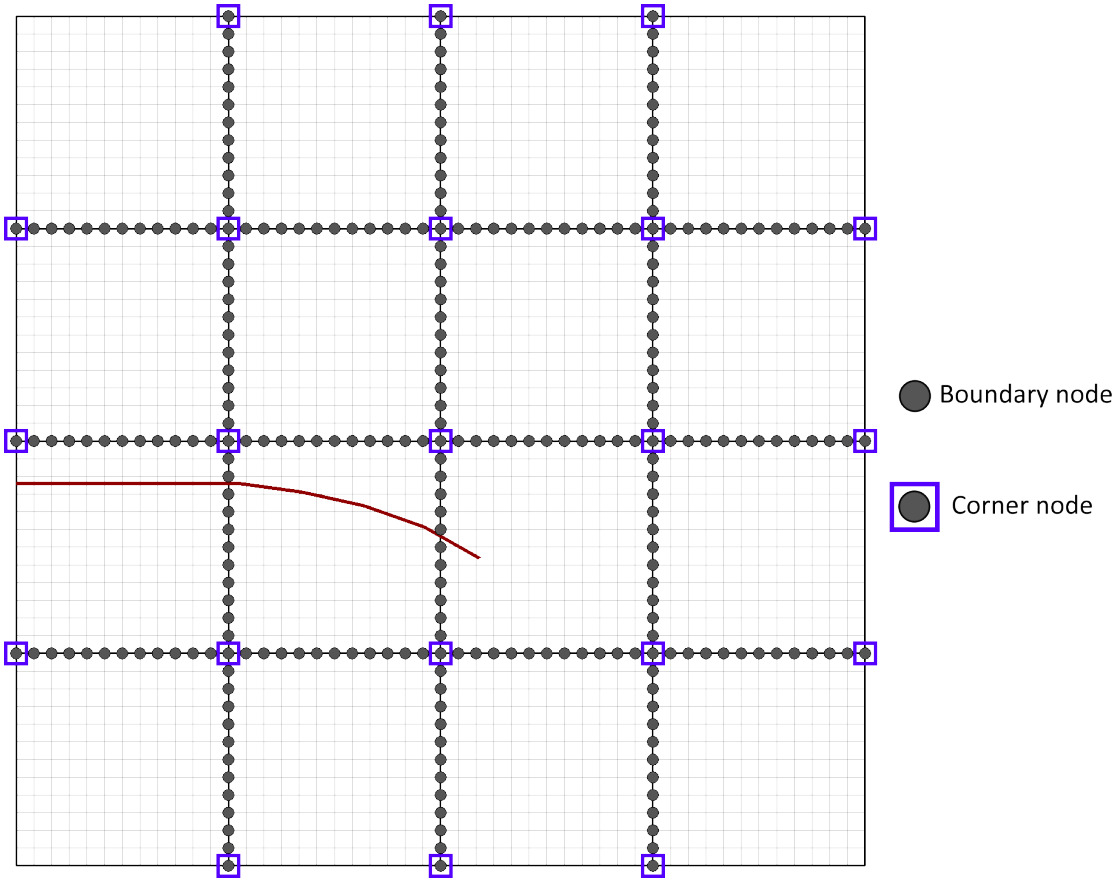


Figure 5.5: Boundary and corner nodes of the original FETI-DP and P-FETI-DP algorithms.

The XFEM enrichment described in section 4.1.4 models the jump in the displacement field around a crack, by introducing discontinuous basis functions and corresponding enriched DOFs. When the crack completely intersects a subdomain s, the rows and columns of K^s_{rr} corresponding to these enriched DOFs become linearly dependent. In this case, the subdomain is essentially divided into two floating rigid parts that can move independently from one another, as illustrated in figure 5.6, where two subdomains are bisected by the crack, and the resulting independent rigid parts are denoted as shaded regions. There is

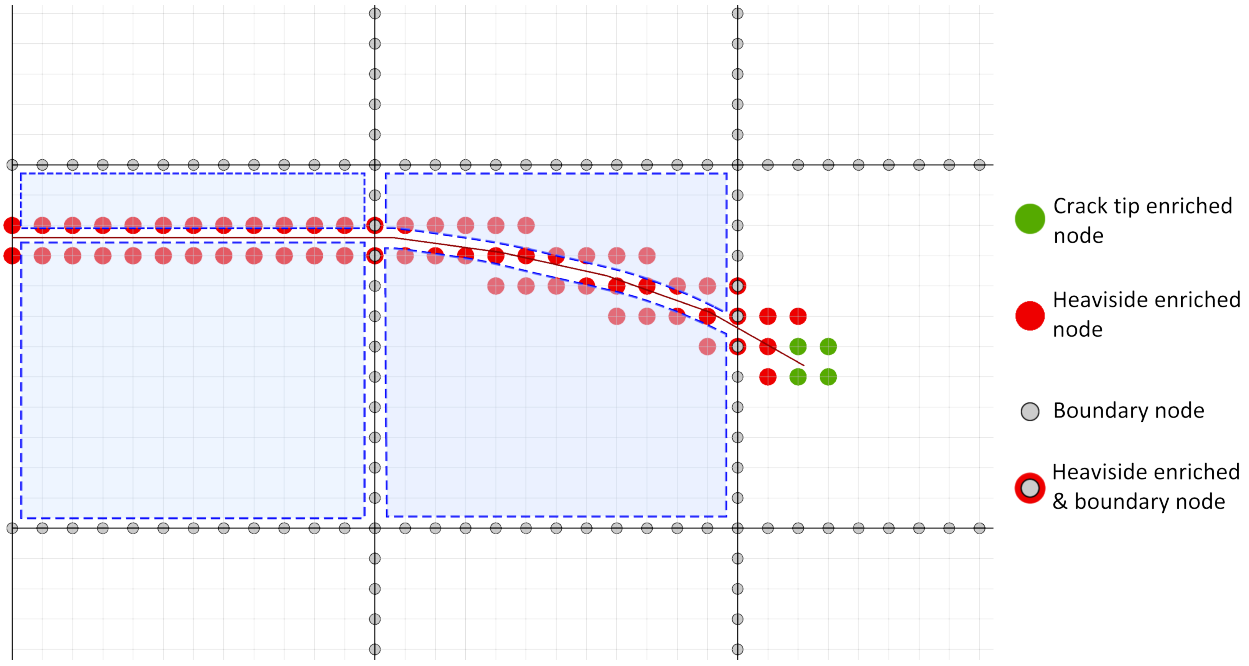


Figure 5.6: Floating rigid parts of subdomains that are completely intersected by a crack.

also another subdomain that interacts with the crack and contains the crack tip, but no mechanism is developed there, since the crack does not run through the entire subdomain.

In order to overcome this singularity of $\boldsymbol{K}_{rr}^{s}$, the following procedure is proposed. First, the linearly dependent rows and columns of $\boldsymbol{K}_{rr}^{s}$ are identified by locating their corresponding enriched DOFs. Let $\mathbb{M}_b$ be the set of boundary DOFs, namely DOFs belonging to 2 or more subdomains. The Heaviside enriched DOFs $\boldsymbol{a}_j$ of equation(4.31) belong to the set $\mathbb{M}_H$, while the enriched DOFs $\boldsymbol{b}_k^1$ applied for the first crack tip function F_1 of equation(4.28) belong to the set $\mathbb{M}_{T^1}$. If a subdomain is fully intersected by a crack, then the DOFs that belong to the set $\mathbb{M}_b \cap (\mathbb{M}_H \cup \mathbb{M}_{T^1})$ are responsible for introducing the jump in the displacement field and for developing internal mechanisms for that subdomain. Subsequently, these DOFs are promoted to corner DOFs, thus removing the corresponding linearly dependent rows/columns from $\boldsymbol{K}_{rr}^{s}$ and restoring its invertibility. If $\mathbb{M}_{c,std}$ is the set of corner DOFs defined in section 5.3.1.1, then the proposed modification consists of identifying the set

$$\mathbb{M}_c = \mathbb{M}_{c,std} \cup (\mathbb{M}_b \cap \mathbb{M}_H) \cup (\mathbb{M}_b \cap \mathbb{M}_{T^1}) \tag{5.52}$$

and using it to define the corner DOFs of FETI-DP and P-FETI-DP, instead of $\mathbb{M}_{c,std}$, as illustrated in figure 5.7. These boundary enriched DOFs are then included in the subdomain's stiffness submatrix $\boldsymbol{K}_{cc}^{s}$ and ultimately in the global coarse problem matrix $\boldsymbol{S}_{cc}$ of equation(5.36). However, contrary to subdomain-level matrices, $\boldsymbol{S}_{cc}$ corresponds to the global domain, albeit in terms of its corner DOFs only. While the crack propagates through-

out the domain, some subdomains will be fully intersected, but this will not happen for the entire domain until the last propagation step of the analysis, when collapse occurs. Therefore, the proposed approach avoids the singularity of $\boldsymbol{S}_{cc}$, since no internal mechanisms will be developed for the reduced order model corresponding to the coarse problem. As a result, the definition of subdomains does not depend on the location of cracks, as was the case in X. Chen and Cai (2022), Waisman and Berger-Vergiat (2013), and Wyart et al. (2008), and the entire domain can be arbitrarily partitioned, unlike Menk and Bordas (2011) where only the enriched DOFs were decomposed into subdomains. Furthermore, with the proposed approach cracks can intersect any subdomains, which can then be selected with the objective of minimizing memory requirements and computation time.

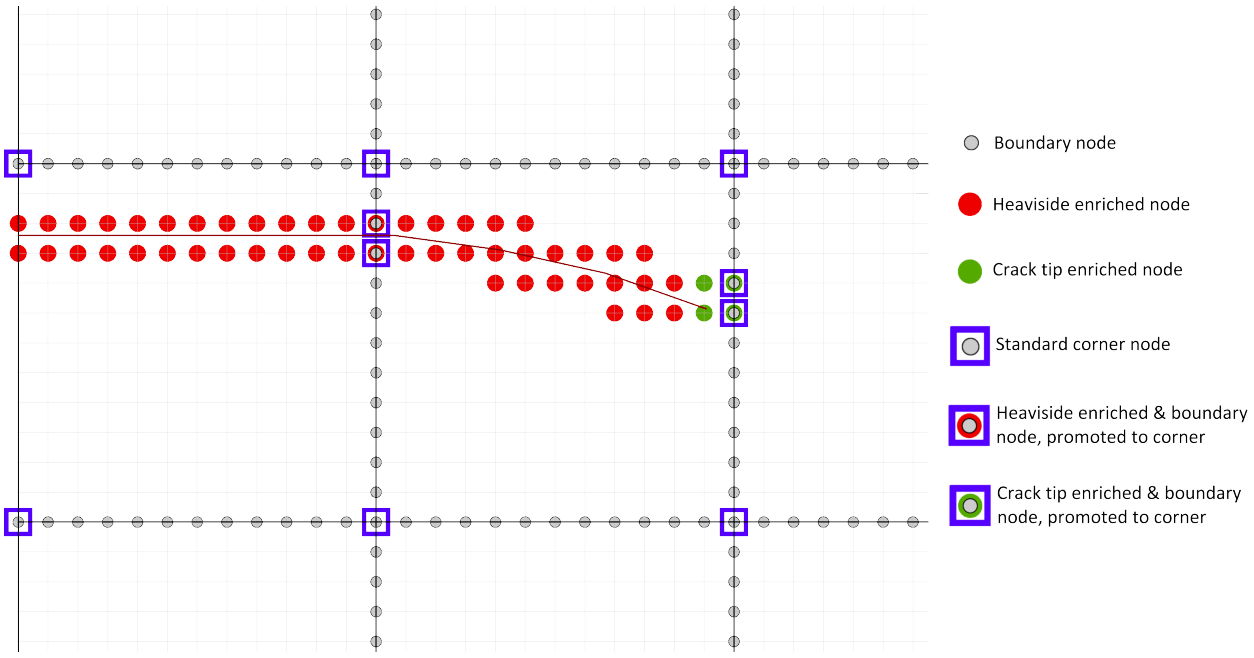


Figure 5.7: Boundary nodes enriched with Heaviside and crack tip enrichment functions are promoted to corner nodes, to avoid singular $\boldsymbol{K}_{rr}^{s}$ matrices in FETI-DP and P-FETI-DP.

5.4.2 Elimination of XFEM-related ill-conditioning

Another difficulty that arises in crack propagation simulations with XFEM is the ill-conditioning of the stiffness matrices, due to the significant difference of stiffness entries corresponding to the crack-tip enriched DOFs from entries corresponding to standard and Heaviside enriched DOFs. To speed up convergence, a preconditioner suitable for this particular source of ill-conditioning is essential. Unfortunately, general purpose preconditioners are not as effective in the case of XFEM. Problem specific preconditioners based on domain decomposition methods have been developed for this purpose in X. Chen and Cai (2022), Menk and

Bordas (2011), and Waisman and Berger-Vergiat (2013), where the domain is separated into well-conditioned subdomains (standard and possibly Heaviside DOFs) and ill-conditioned subdomains (crack-tip enriched DOFs and possibly Heaviside or even a few standard DOFs). Then direct solvers are used for the subproblems defined at the ill-conditioned subdomains, while the well-conditioned ones are treated with inexact methods, such as diagonal preconditioning (Menk & Bordas, 2011), algebraic multigrid (Waisman & Berger-Vergiat, 2013) and incomplete LU factorization (X. Chen & Cai, 2022).

In the present formulation, the terms causing ill-conditioning are restricted only to the interface problem of FETI-DP and P-FETI-DP and can be further eliminated by modifying the coarse problem appropriately. Let $\mathbb{M}_{T^m}$, $m = 1, \cdots 4$ be the set of $\boldsymbol{b}_k^m$ DOFs introduced for each crack-tip enrichment function of equation(4.28). The ill-conditioned terms of the interface problem correspond to DOFs belonging to $\mathbb{M}_b \cap \mathbb{M}_{T^m}$, $m = 1, \cdots 4$. The proposed technique treats these DOFs as corner DOFs, as depicted in figure 5.7, thus making the coarse problem of equation(5.36) more efficient in distributing the error between subdomains at each PCG iteration. Therefore, the preconditioner of P-FETI-DP can overcome ill-conditioning due to XFEM enrichments, since it includes this coarse problem, as described in equation(5.49). Although the preconditioner of FETI-DP does not depend on corner DOFs and is not affected by the modification, the condition of the interface problem matrix $\boldsymbol{F}_{Irr} + \boldsymbol{F}_{Irc}\boldsymbol{S}_{cc}^{-1}\boldsymbol{F}_{Icr}$ is improved, because i) the coarse problem is directly embedded into this matrix and ii) the problematic DOFs in $\mathbb{M}_b \cap \mathbb{M}_{T^m}$, $m = 1, \cdots 4$ are removed from it. Taking into account the modification of section 5.4.1 to avoid singular $\boldsymbol{K}_{rr}^s$ matrices, the proposed method uses the set

$$\mathbb{M}_c = \mathbb{M}_{c,std} \cup (\mathbb{M}_b \cap \mathbb{M}_H) \bigcup_{m=1}^{4} (\mathbb{M}_b \cap \mathbb{M}_{T^m}) \tag{5.53}$$

to define the corner DOFs of FETI-DP and P-FETI-DP. It should be pointed out, however, that even without this improvement, the ill-conditioning due to XFEM enrichments could be avoided during some crack propagation steps, if the crack front does not interact with nodes located on the boundary between subdomains, as illustrated in figure 5.6. However, this is unlikely in 3D problems, since the crack front is substantially more extensive and interacts with multiple elements, as will be shown in the numerical examples of chapter 6. In any case, when ill-conditioning does appear, FETI-DP and P-FETI-DP exhibit a substantial increase in the iterations required for convergence. In the test case of section 6.2, an increase of up to 245% is reported. By using equation(5.53) to define the corner DOFs, this XFEM related ill-conditioning is entirely eliminated and the proposed FETI-DP and P-FETI-DP become insensitive to the location and number of enriched nodes.

Moreover, by avoiding these iteration spikes, the scalability of FETI-DP and P-FETI-DP solvers is retained. As will be shown in the numerical examples of chapter 6, the iterations required for convergence decrease as more subdomains are used. All DMM solvers for XFEM crack propagation developed earlier (X. Chen & Cai, 2022; Menk & Bordas, 2011; Waisman & Berger-Vergiat, 2013) were not scalable and by increasing the number of subdomains,

an increase in the required iterations was observed. In contrast, the scalability of the proposed FETI-DP and P-FETI-DP solvers permits their implementation in cluster computing environments, where numerous networked multi-core CPUs and/or GPUs can be used to solve the resulting equations in parallel. In these high perfomance computing systems, the computational power and available memory can be arbitrarily increased, simply by adding more processors, each with its own memory. The ability to assign a lot of subdomains to all these processors, without increasing the required iterations for convergence is essential for the exploitation of cluster computing environments and makes the proposed FETI-DP and P-FETI-DP very attractive solvers for large scale problems.

5.4.3 Reusing data from previous steps

In brittle crack propagation with XFEM, only a few entries of the stiffness matrices change from one analysis step to the next. These entries correspond to localized DOFs near the crack front, specifically Heaviside and crack-tip enriched DOFs that are introduced in the current step, as well as crack-tip enriched DOFs introduced in previous steps and then removed. Yet, XFEM-oriented solvers based on domain decomposition methods, such as Menk and Bordas (2011), Waisman and Berger-Vergiat (2013), and Wyart et al. (2008) overlook this potential to reduce the computational cost of the solution phase by reusing data calculated during previous crack propagation steps. In Pais et al. (2012), a reanalysis solver was proposed, that takes advantage of this opportunity and partially reuses the matrix factorization of the previous step. Nevertheless, this approach has limited success in 3D problems or when the percentage of modified columns of the stiffness matrix is not extremely small.

In the present FETI-DP and P-FETI-DP formulations, the need to update only a part of the total stiffness is exploited for further computational gains in a very natural manner. Due to the domain partitioning, many subdomains do not interact with the crack front at each propagation step. In this case, all corresponding subdomain stiffness matrices, vectors and related data remain unaltered and can be reused from the last step when they were updated. Note that these reusable data include the Schur complements of the matrices $\boldsymbol{K}_{rr}^{s}$ and $\boldsymbol{K}_{ii}^{s}$, which require time consuming operations. Figures 5.8a and 5.8b represent two successive steps, as a crack propagates through multiple subdomains. The set of nodes with modified stiffness contains only those enriched with crack tip functions and the newly added Heaviside functions. Therefore, only the corresponding quantities of subdomains $S6$ and $S7$ need to be updated. In addition, increasing of the number of subdomains will enhance the effectiveness of this reanalysis feature, due to further localization of modified DOFs and the existence of more unmodified subdomains.

Furthermore, a re-initialization technique is proposed for the solution of the interface problem of FETI-DP and P-FETI-DP, in order to reduce the number of iterations. Specifically, when solving the interface problem of P-FETI-DP (see equation(5.47)), an initial guess $\widetilde{\boldsymbol{u}}_b$ is required for the first PCG iteration. In the first crack propagation step, the zero vector is assumed as an initial guess $\widetilde{\boldsymbol{u}}_b^{t=0} = \boldsymbol{0}$. Let $\boldsymbol{u}_b^t$ be the solution of the interface problem at propagation step t

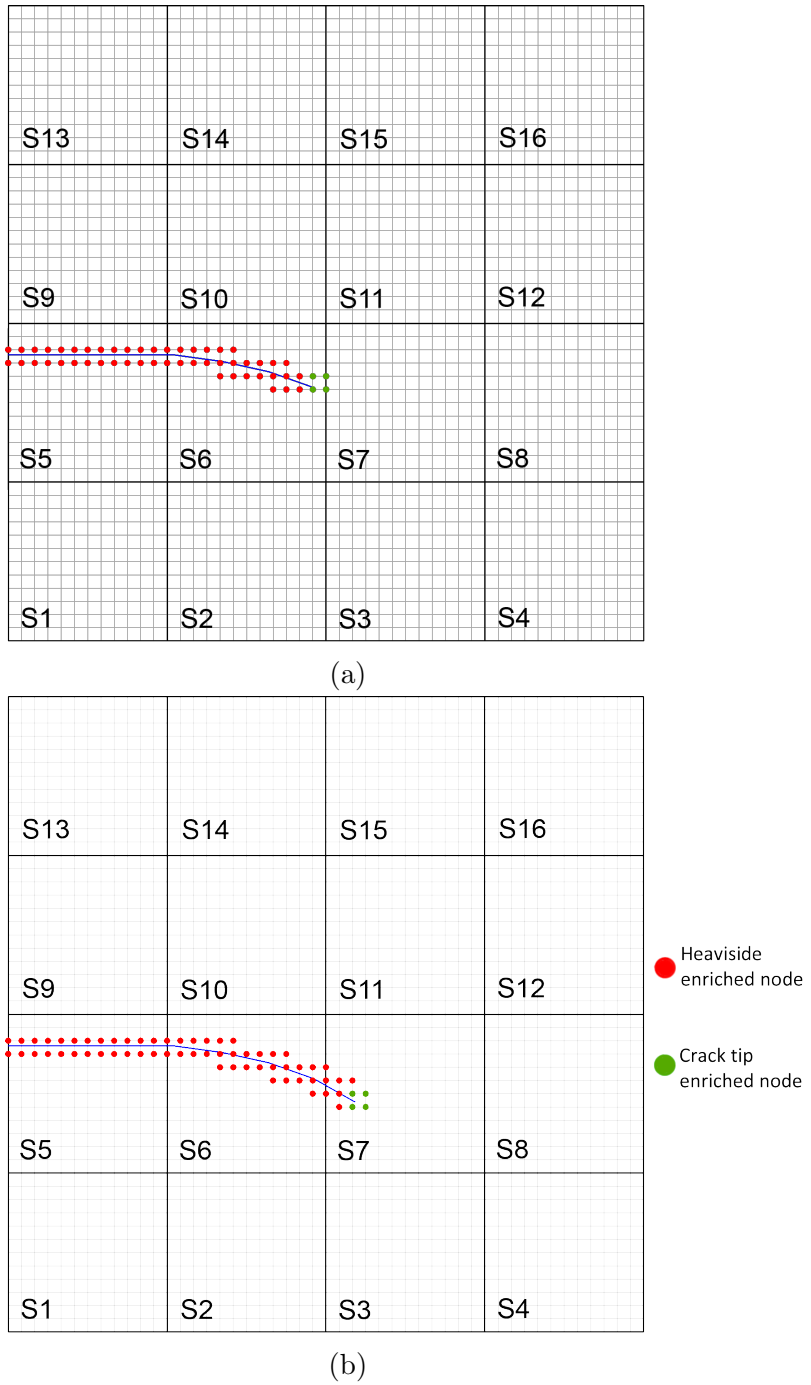


Figure 5.8: Two consecutive analysis steps as the crack propagates. Only subdomains S6 and S7 will have modified DOFs and stiffness between propagation steps (a) i and (b) i + 1.

$$\boldsymbol{u}_b^t = \left[\begin{array}{ccc} \boldsymbol{u}_{std}^t & \boldsymbol{u}_H^t & \boldsymbol{u}_T^t \end{array} \right] \tag{5.54}$$

where $\boldsymbol{u}_{std}^t$ are the displacements of standard boundary DOFs, $\boldsymbol{u}_H^t$ of Heaviside boundary DOFs and $\boldsymbol{u}_T^t$ of crack-tip boundary DOFs. Then the displacements of standard and Heaviside DOFs can be reused as an initial guess during the next propagation step

$$\widetilde{\boldsymbol{u}}_b^{t+1} = \left[\begin{array}{cccc} \widetilde{\boldsymbol{u}}_{std}^{t+1} & \widetilde{\boldsymbol{u}}_{H1}^{t+1} & \widetilde{\boldsymbol{u}}_{H2}^{t+1} & \widetilde{\boldsymbol{u}}_T^{t+1} \end{array} \right] = \left[\begin{array}{cccc} \boldsymbol{u}_{std}^t & \boldsymbol{u}_H^t & \boldsymbol{0} & \boldsymbol{0} \end{array} \right] \tag{5.55}$$

where $\widetilde{\boldsymbol{u}}_{H1}^{t+1}$ corresponds to Heaviside boundary DOFs that are present in both steps t, $t+1$, while $\widetilde{\boldsymbol{u}}_{H2}^{t+1}$ corresponds to Heaviside boundary DOFs that were newly introduced in step $t+1$. Conversely, $\widetilde{\boldsymbol{u}}_{H2}^{t+1}$ and $\widetilde{\boldsymbol{u}}_T^{t+1}$ correspond to DOFs that were not present during the propagation step t, therefore the initial guess for them is $\boldsymbol{0}$. Nevertheless, these make up a small fraction of the total boundary DOFs and do not markedly affect the improvement of the re-initialization technique.

A similar re-initialization technique is proposed for FETI-DP, where the interface problem is expressed in terms of Lagrange multipliers applied to boundary-remainder DOFs (see equation(5.31)), instead of all boundary DOFs in P-FETI-DP (see equation(5.47)). Using equation(5.53), all enriched boundary-remainder DOFs are promoted to corner DOFs, thus removing the corresponding Lagrange multipliers. This complements the re-initialization technique, since it ensures that the interface problem of FETI-DP involves only standard boundary-remainder DOFs, which are the same during all crack propagation steps. As a result, the entire solution vector of one propagation step $\boldsymbol{\lambda}^t$ can be reused as an initial guess for the next step

$$\widetilde{\boldsymbol{\lambda}}^{t+1} = \boldsymbol{\lambda}^t \tag{5.56}$$

Both the re-initialization of the solution vector guess and the reuse of previously computed subdomain matrices and factorizations, can be easily implemented to further increase the computational efficiency of FETI-DP and P-FETI-DP solvers. In the numerical examples of chapter 6, it is shown that the re-initialization technique can reduce the number of iterations across all propagation steps of the analysis by up to 40% for P-FETI-DP and 37% for FETI-DP and in combination with the reuse of subdomain data, the time required for the solution can be reduced by up to 50% for both methods.

5.5 HPC implementation

This section specifies the parallel implementation of the FETI-DP and P-FETI-DP solvers in high performance computing systems, specifically computer clusters. By modifying the original FETI-DP and P-FETI-DP equations, the programming endeavor can be greatly simplified.

5.5.1 Cluster computing

There is a large number of parallel computing environments and programming paradigms available nowadays. This dissertation focuses on cluster computing, which is popular for computational simulations. Computer clusters are computing environments consisting of multiple networked computers that collaborate to solve a single problem. These units are often called "computing nodes", but the term "computers" will be used here, to avoid confusion with FEM nodes. Generally, a cluster consists of readily available personal computers, complete with their own processors, memory and storage devices. These are connected via a Local Area Network (LAN), such as conventional Ethernet. Usually, the computers have the same or similar specifications, which helps in *load balancing*, namely evenly distributing the required work to all computers to increase performance. Supercomputers may also be computer clusters, in which case high speed buses are used instead of Ethernet, to improve the bandwidth and latency of data transfers.

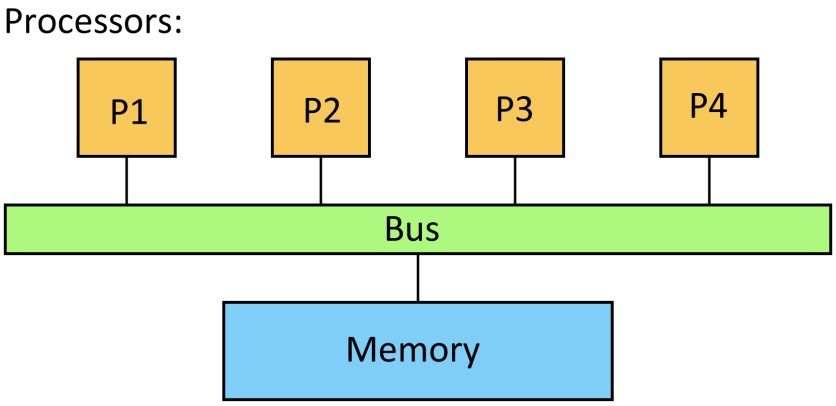


Figure 5.9: Shared memory system

Parallel computing environments may use shared memory, distributed memory or a combination of both. A *shared memory* system employs central (typically large) block of Random Access Memory (RAM), which can be accessed by multiple processing units in parallel, as illustrated in figure 5.9. These processing units may be separate processors or different threads executed on the same CPU. The main advantages of this approach is the very fast memory accesses. Additionally, it leads to simplified programming, albeit race-conditions need to be avoided, due to the unified memory address space that allows processors to communicate, simply by accessing the same data. However, shared memory systems are not scalable, since the number of processors and the amount of available RAM cannot be increased indefinitely. On the other hand, figure 5.10 depicts a *purely distributed memory* system, where each processor has its own private memory and communication between them happens by passing messages over a network. This approach complicates communication and requires a middleware (software) to handle the message passing between processors. Nevertheless, it allows an

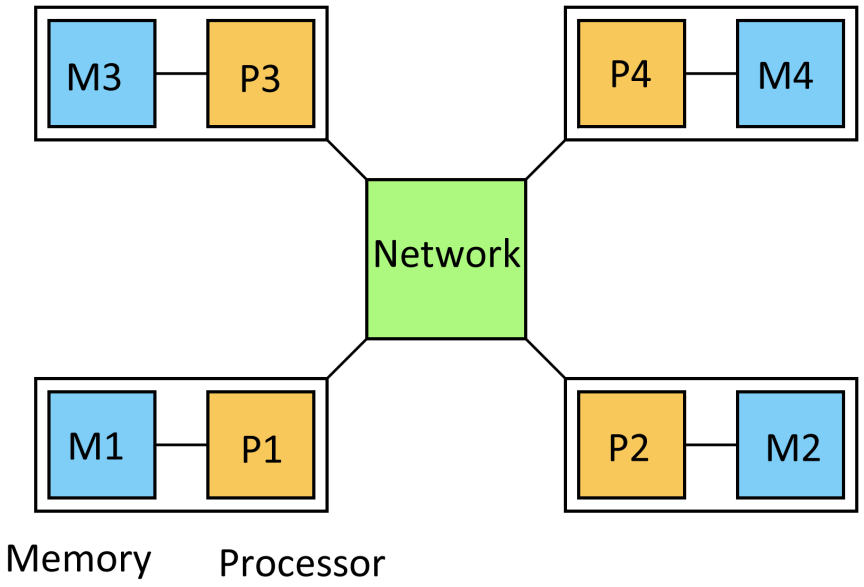


Figure 5.10: Distributed memory system

indefinite increase of processing power and available memory by adding an arbitrary number of processors and their memory.

Computer clusters typically employ a *hybrid distributed memory* model, where each computer has a multi-processor and its own private memory, but this memory is shared among the multiple processing units, as shown in figure 5.11. Therefore, each computer is a shared memory machine, but the system of all networked computers is distributed. In this hybrid system, communication between the processors of the same computer is much faster and should be prioritized over communication between processors belonging to remote computers. All in all, cluster computing environments offer the following advantages:

- Performance. Programs can be executed in parallel.
- Scalability. Processing power and available memory can be increased indefinitely by adding more computers, which have their own RAM and multicore CPUs.
- Cost effectiveness. Obtaining the desired level of performance with a cluster of several low-end computers is usually cheaper than using a single high-end computer.
- Reliability. Failure or maintenance of one or more computers does not incapacitate the whole system. Instead, its performance is lowered until fixing or replacing the affected computers.

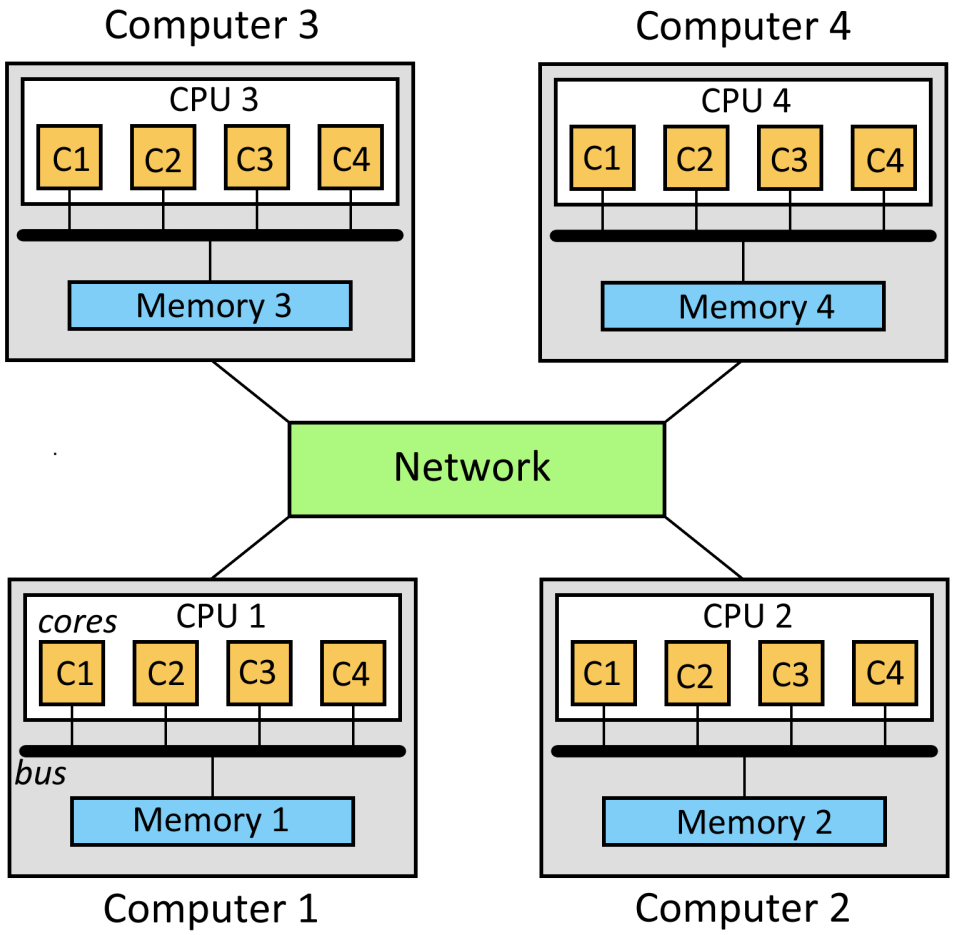


Figure 5.11: Computer cluster

5.5.1.1 Message Passing Interface

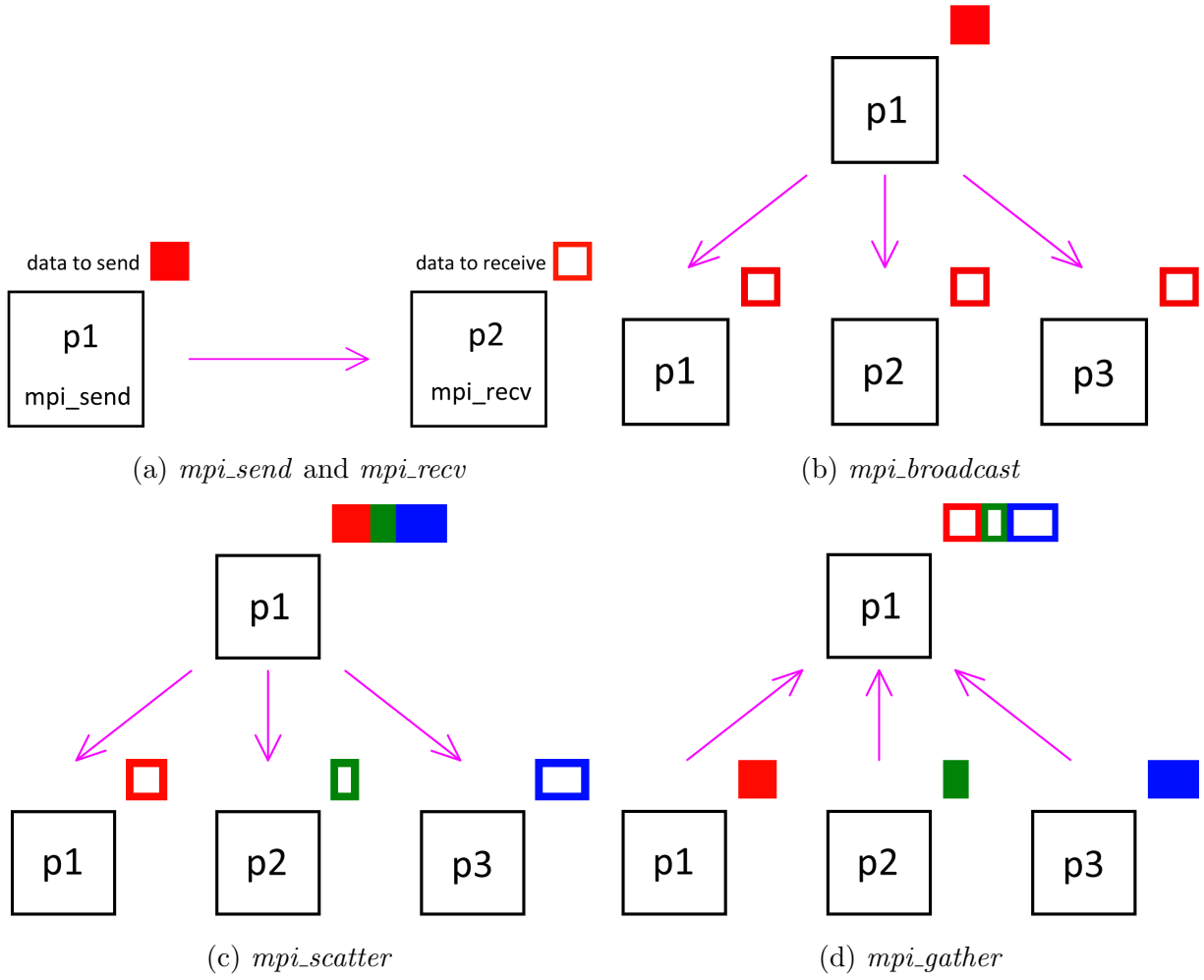
In this dissertation, communication between different computers of a cluster is performed using the Message Passing Interface (MPI). The MPI standard defines a set of commands for transferring data between physically distinct memory spaces and is implemented by different libraries for various operating systems. In MPI terminology, a "process" is an independent thread of execution with has its own private memory and can be mapped to a physical computer of the cluster. Using multiple MPI processes on the same computer is possible, but not used here. This section lists the most basic commands supported by all implementations:

- *mpi_send* / *mpi_recv*. Process i sends a chunk of data to process j. Process j receives the same chunk of data from process j. These point-to-point commands are the most basic ones and they must always appear in pairs; each *mpi_send* must have a corre-

sponding *mpi_recv*. They can also be executed asynchronously, which is exploited in this dissertation for further performance gains. See figure 5.12a.

- *mpi_broadcast*. One process, called the *root* process, sends a chunk of data to all other processes. This is collective command sends the same data to all processes. See figure 5.12b.
- *mpi_scatter*. Another collective command, where one root process sends data to all other processes, but each processes receives a different chunk of data. See figure 5.12c.
- *mpi_gather*. This collective command is the opposite of *mpi_scatter*: one root process i receives data from all other processes, but each processes sends a different chunk of data. See figure 5.12d.

There are numerous other commands, but they are not available in all MPI implementations. In any case, other commands can be viewed as optimized or shorthand versions of a series of the above basic ones.



(a) *mpi_send* and *mpi_recv* (b) *mpi_broadcast*

(c) *mpi_scatter* (d) *mpi_gather*

Figure 5.12: Basic point-to-point and collective MPI commands.

5.5.1.2 DDM solvers on clusters

When implementing the FETI-DP and PFETI-DP solvers, which were discussed in the previous sections, on a computer cluster, the total number of subdomains is divided into groups and each group is assigned to one computer. This is called static scheduling and happens once, before the analysis starts. Load balancing is achieved by forming equally sized subdomain groups, if the computers have the same specifications, or groups of size proportional to the processing power of each different computer. Within a computer, the number of subdomains is generally much higher than the number of processing units (usually CPU cores). In this case, a producer-consumer strategy is employed instead of static scheduling, since it is more flexible and can balance the computational load more effectively. Specifically, the subdomain-level tasks are put on a queue. When a CPU core has finished the rest of its

computations, it is assigned to a new subdomain, which is then removed from the queue.

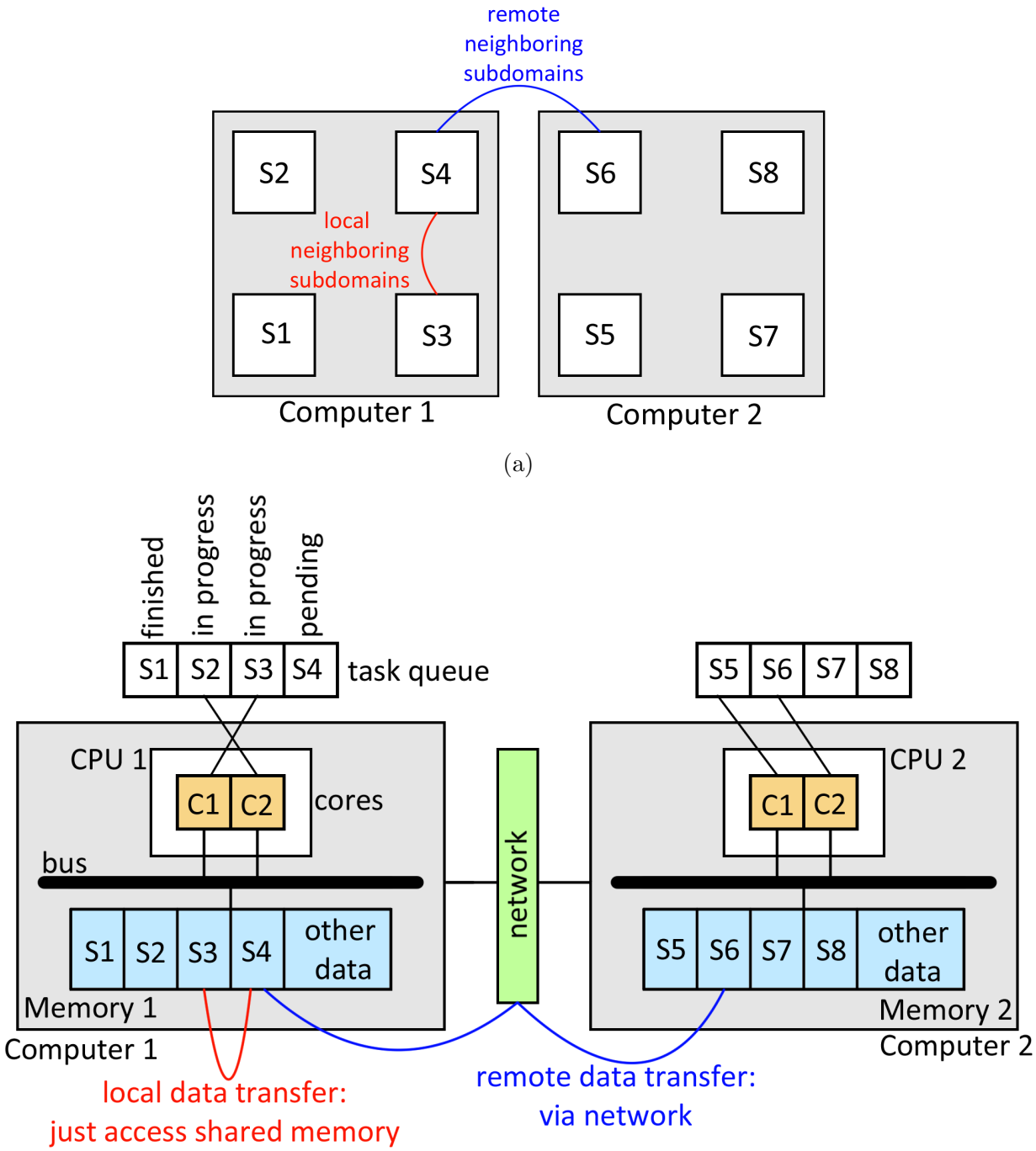


Figure 5.13: Allocation of subdomains to computers and communication between them.

Figure 5.13 illustrates this subdomains-to-computers allocation for a 2D example. In the same figure, the communication between neighboring subdomain can be seen, where two subdomains are considered as neighbors, if they have common nodes. Data transfers between two *local* neighboring subdomains, namely subdomains allocated to the same computer, is done very fast via accessing the shared memory of that computer. In contrast, data transfers between two *remote* neighboring subdomains, namely subdomains allocated to different computers, is performed over the network using MPI commands and is, thus, much slower. When allocating subdomains to computers, these number and frequency of these remote transfers should be minimized to increase the performance of the solvers.

5.5.2 P-FETI-DP

The HPC implementation of P-FETI-DP will be covered first, since it is simpler than FETI-DP. To begin with, the *expanded domain* is defined, which contains all DOFs, but each boundary DOF appears multiple times, once for each corresponding subdomain. In contrast, the *global domain* contains exactly one instance of all DOFs in the model. A 2D example of this distinction for a mesh with 4×2 elements is given in figure 5.14, where the DOFs along axes x,y are shown for the global and expanded domain. In the remainder of this section, superscript s will denote a subdomain-level quantity, superscript e will denote an expanded-domain-level quantity and no superscript will denote a global-level quantity.

5.5.2.1 Interface problem

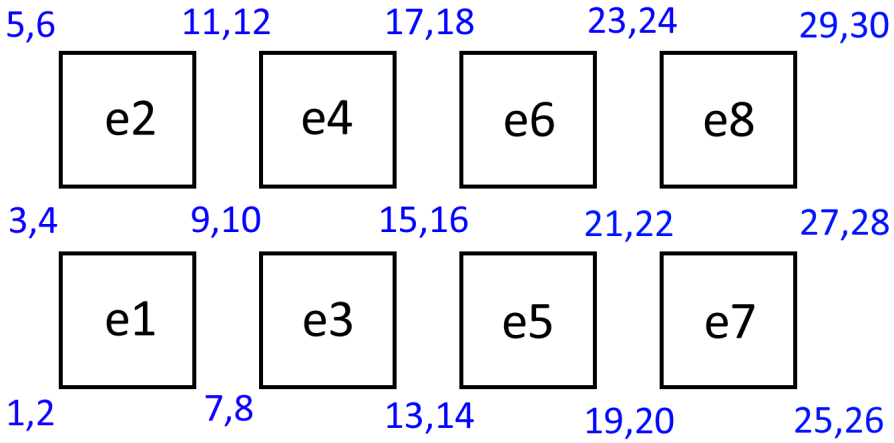
In section 5.3.2, $\boldsymbol{u}_b$ $(n_b \times 1)$ was defined as the global boundary displacements and $\boldsymbol{u}_b^s$ $(n_b^s \times 1)$ as subdomain boundary displacements. Here, the expanded boundary displacements $\boldsymbol{u}_b^e$ $(n_b^e \times 1)$ are also defined as

$$\boldsymbol{u}_b^e = \begin{bmatrix} \boldsymbol{u}_b^1 \\ \vdots \\ \boldsymbol{u}_b^{n_s} \end{bmatrix} \tag{5.57}$$

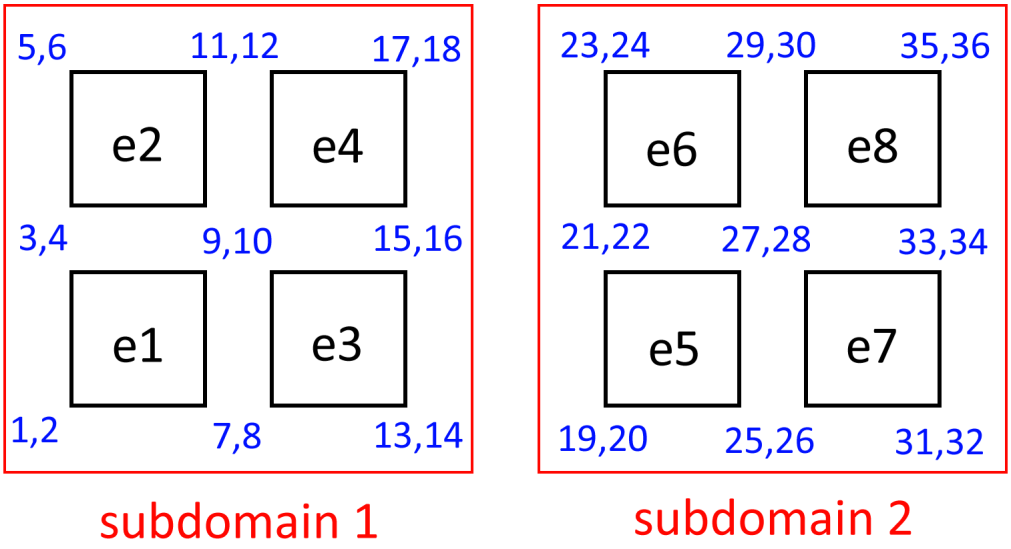
where n_s is the number of subdomains, n_b is the number of global boundary DOFs, n_b^s is the number of boundary DOFs of subdomain s and n_b^e is the number of boundary DOFs of the expanded domain. The boolean mapping matrices $\boldsymbol{L}_b^s$ defined in section 5.3.2 compose the expanded matrix $\boldsymbol{L}_b^e$ $(n_b^e \times n_b)$:

$$\boldsymbol{L}_b^e = \begin{bmatrix} \boldsymbol{L}_b^1 \\ \vdots \\ \boldsymbol{L}_b^{n_s} \end{bmatrix} \tag{5.58}$$

This mapping matrix can be used to extract subdomain-level displacements from global ones:



(a)



(b)

Figure 5.14: a) Global domain: each DOF appears once. b)Expanded domain: each boundary DOF appears once for each subdomain it belongs to. Here the following DOFs coincide: $13 \equiv 19$, $14 \equiv 20$, $15 \equiv 21$, $16 \equiv 22$, $17 \equiv 23$, $18 \equiv 24$

$$\begin{aligned} \boldsymbol{u}_b^s &= \boldsymbol{L}_b^s \boldsymbol{u}_b \\ \boldsymbol{u}_b^e &= \boldsymbol{L}_b^e \boldsymbol{u}_b \end{aligned} \tag{5.59}$$

Similarly, to calculate global right-hand-side (RHS) forces $\boldsymbol{z}_b$ from their subdomain contributions $\boldsymbol{z}_b^s$, which compose the expanded vector $\hat{\boldsymbol{z}}_b^e$, the following are used

$$\begin{aligned} \boldsymbol{z}_b &= \sum_{s=1}^{n_s} (\boldsymbol{L}_b^s)^T \hat{\boldsymbol{z}}_b^s = (\boldsymbol{L}_b^e)^T \hat{\boldsymbol{z}}_b^e \\ \hat{\boldsymbol{z}}_b^e &= \begin{bmatrix} \hat{\boldsymbol{z}}_b^1 \\ \vdots \\ \hat{\boldsymbol{z}}_b^{n_s} \end{bmatrix} \end{aligned} \tag{5.60}$$

The stiffness matrices can also be written in expanded domain form:

$$\boldsymbol{K}_{bb}^e = \begin{bmatrix} \boldsymbol{K}_{bb}^1 & & \\ & \ddots & \\ & & \boldsymbol{K}_{bb}^{n_s} \end{bmatrix} \quad \boldsymbol{K}_{bi}^e = \begin{bmatrix} \boldsymbol{K}_{bi}^1 & & \\ & \ddots & \\ & & \boldsymbol{K}_{bi}^{n_s} \end{bmatrix} \quad \boldsymbol{K}_{ii}^e = \begin{bmatrix} \boldsymbol{K}_{ii}^1 & & \\ & \ddots & \\ & & \boldsymbol{K}_{ii}^{n_s} \end{bmatrix} \tag{5.61}$$

and for the inverses and Schur complements:

$$\begin{aligned} (\boldsymbol{K}_{ii}^e)^{-1} &= \begin{bmatrix} (\boldsymbol{K}_{ii}^1)^{-1} & & \\ & \ddots & \\ & & (\boldsymbol{K}_{ii}^{n_s})^{-1} \end{bmatrix} \\ \boldsymbol{S}_{bb}^e &= \begin{bmatrix} \boldsymbol{S}_{bb}^1 & & \\ & \ddots & \\ & & \boldsymbol{S}_{bb}^{n_s} \end{bmatrix} = \boldsymbol{K}_{bb}^e - (\boldsymbol{K}_{ib}^e)^T (\boldsymbol{K}_{ii}^e)^{-1} \boldsymbol{K}_{ib}^e \end{aligned} \tag{5.62}$$

Then, the matrix of the interface problem of P-FETI-DP is

$$\boldsymbol{S}_{bb} = \sum_{s=1}^{n_s} (\boldsymbol{L}_b^s)^T \boldsymbol{S}_{bb}^s \boldsymbol{L}_b^s = (\boldsymbol{L}_b^e)^T \boldsymbol{S}_{bb}^e \boldsymbol{L}_b^e \tag{5.63}$$

and the interface problem can be rewritten in terms of the expanded domain matrices and vectors

$$\begin{aligned} \boldsymbol{S}_{bb} \cdot \boldsymbol{u}_b = \boldsymbol{z}_b &\iff (\boldsymbol{L}_b^e)^T \boldsymbol{S}_{bb}^e \boldsymbol{L}_b^e \boldsymbol{u}_b = \boldsymbol{z}_b \\ &\iff (\boldsymbol{L}_b^e)^T \boldsymbol{S}_{bb}^e \boldsymbol{u}_b^e = (\boldsymbol{L}_b^e)^T \hat{\boldsymbol{z}}_b^e \end{aligned} \tag{5.64}$$

This interface problem system is solved using the PCG method. At each PCG iteration, the following matrix-vector multiplication is performed:

$$\boldsymbol{y}_b = \sum_{s=1}^{n_s} (\boldsymbol{L}_b^s)^T \boldsymbol{S}_{bb}^s \boldsymbol{x}_b^s = (\boldsymbol{L}_b^e)^T \boldsymbol{S}_{bb}^e \boldsymbol{x}_b^e \tag{5.65}$$

where $\boldsymbol{x}_b^e$ ($n_b^e \times 1$) is the approximation of the solution $\boldsymbol{u}_b^e$ for this iteration and $\boldsymbol{y}_b$ ($n_b \times 1$) is a force vector. This operation is performed in two stages.

- The matrix-vector multiplication is executed at subdomain level (or equivalently expanded domain level)

$$\begin{aligned} \hat{\boldsymbol{y}}_b^s &= \boldsymbol{S}_{bb}^s \boldsymbol{x}_b^s \\ \hat{\boldsymbol{y}}_b^e &= \boldsymbol{S}_{bb}^e \boldsymbol{x}_b^e \end{aligned} \tag{5.66}$$

- A map-reduce operation is performed to obtain the global vector $\boldsymbol{y}_b$ from the subdomain contributions $\hat{\boldsymbol{y}}_b^s$

$$\boldsymbol{y}_b = \sum_{s=1}^{n_s} (\boldsymbol{L}_b^s)^T \hat{\boldsymbol{y}}_b^s = (\boldsymbol{L}_b^e)^T \hat{\boldsymbol{y}}_b^e \tag{5.67}$$

where $\hat{\boldsymbol{y}}_b^s$ $(n_b^s \times 1)$ and $\hat{\boldsymbol{y}}_b^e$ $(n_b^e \times 1)$ are intermediate force vectors, meaning that their entries at different instances of the same boundary DOF (one instance per subdomain) are different. In contrast, these different entries for the same boundary DOF are summed and appear only once in $\boldsymbol{y}_b$.

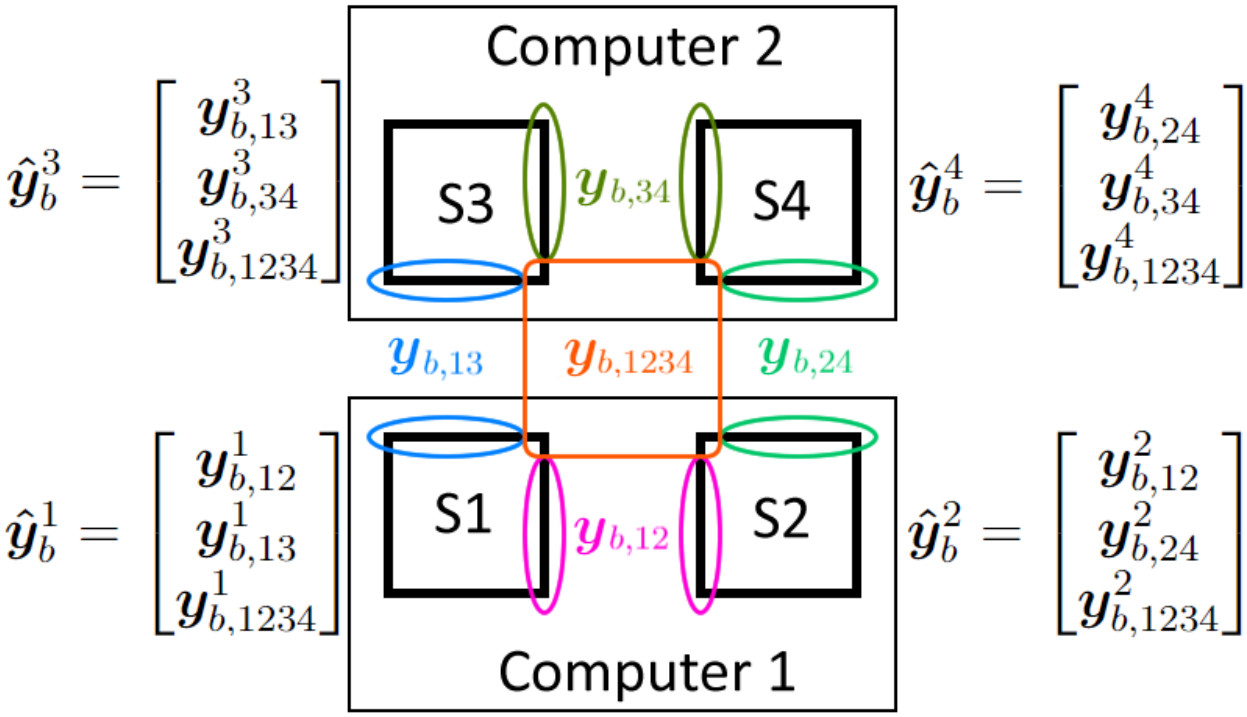


Figure 5.15: Example of neighboring subdomains and displacements along their common boundary DOFs.

It can be observed that $\boldsymbol{x}_b^e$, $\boldsymbol{z}_b^e$ and $\boldsymbol{S}_{bb}^e$ can be calculated and stored so that each subdomain owns only its corresponding matrices and vectors. When implementing P-FETI-DP in a distributed memory system, each computer will be assigned to a group of subdomains. Thus the matrices and vectors of each subdomain will only exist in the memory space of the computer assigned to the subdomain's group. In contrast, the global vectors $\boldsymbol{y}_b$ and $\boldsymbol{z}_b$ refer to global boundary DOFs, that do not correspond to any computer. Trying to implement the equations above, would require transfering the subdomain vectors to one computer, e.g. by using the MPI operation *mpi_gather*. The global vectors would be calculated on that computer and then sent to all others, e.g. with the MPI operation *mpi_broadcast*, to continue with the rest of the algorithm. An example is given in figure 5.15, where the vectors $\hat{\boldsymbol{y}}_b$ at boundary DOFs of 4 subdomains, which are allocated to 2 computers, are shown before the map-reduce operation. Figure 5.16 illustrates the data transfers from the memory spaces of the subdomains to the memory space where global operations will be performed, the global-level map-reduce operation $\boldsymbol{y}_b = \sum_{s=1}^{n_s} (\boldsymbol{L}_b^s)^T \hat{\boldsymbol{y}}_b^s$ and the data transfer of the global vector $\boldsymbol{y}_b$ back to the memory spaces of the subdomains.

While this centralized approach is definitely doable, it is not efficient in terms of communication, since it forces significant data transfers to a single computer. For large-scale problems with thousands of subdomains, this communication pattern will lead to congestion of the network at the computer that handles global vectors, and become a computational bottleneck. Additionally, forcing one computer to execute the summation of vectors from all subdomains, while the rest are idle, leads to imbalances in the load distribution. To overcome these problems, a distributed implementation is developed in this dissertation, which requires a modifyiing the original P-FETI-DP (and FETI-DP) equations.

For each ordered pair of subdomains (s, t), an unsigned boolean matrix, namely a matrix that has only 0, 1 as values, $\boldsymbol{M}_b^{st}$ $(n_b^s \times n_b^t)$ is defined, which maps the boundary DOFs of t into the boundary DOFs of s. Specifically, an entry i, j of $\boldsymbol{M}_b^{st}$ is 1, only if row i and column j correspond to the same boundary DOF in subdomains s and t, respectively. Let us define two subdomains as neighbors, if they have common boundary nodes, and thus DOFs. If two subdomains s, t are not neighbors, then they do not have common DOFs and the matrices are $\boldsymbol{M}_b^{st} = 0$, $\boldsymbol{M}_b^{ts} = 0$. On the other hand, for the same subdomain s: $\boldsymbol{M}_b^{ss} = I$. Map-reduce operations of subdomain vectors into a global one are performed in a distributed fashion using $\boldsymbol{M}_b^{st}$. For the vector $\hat{\boldsymbol{y}}_b^s$ of a given subdomain s, summing the corresponding entries of other sumbdomains can be performed as

$$\boldsymbol{y}_b^s = \sum_{\substack{t=1,\cdots n_s \\ t \neq s}} \boldsymbol{M}_b^{st} \hat{\boldsymbol{y}}_b^s \tag{5.68}$$

The corresponding expanded domain matrix $\boldsymbol{M}_b^e$ $(n_b^e \times n_b^e)$ is defined as

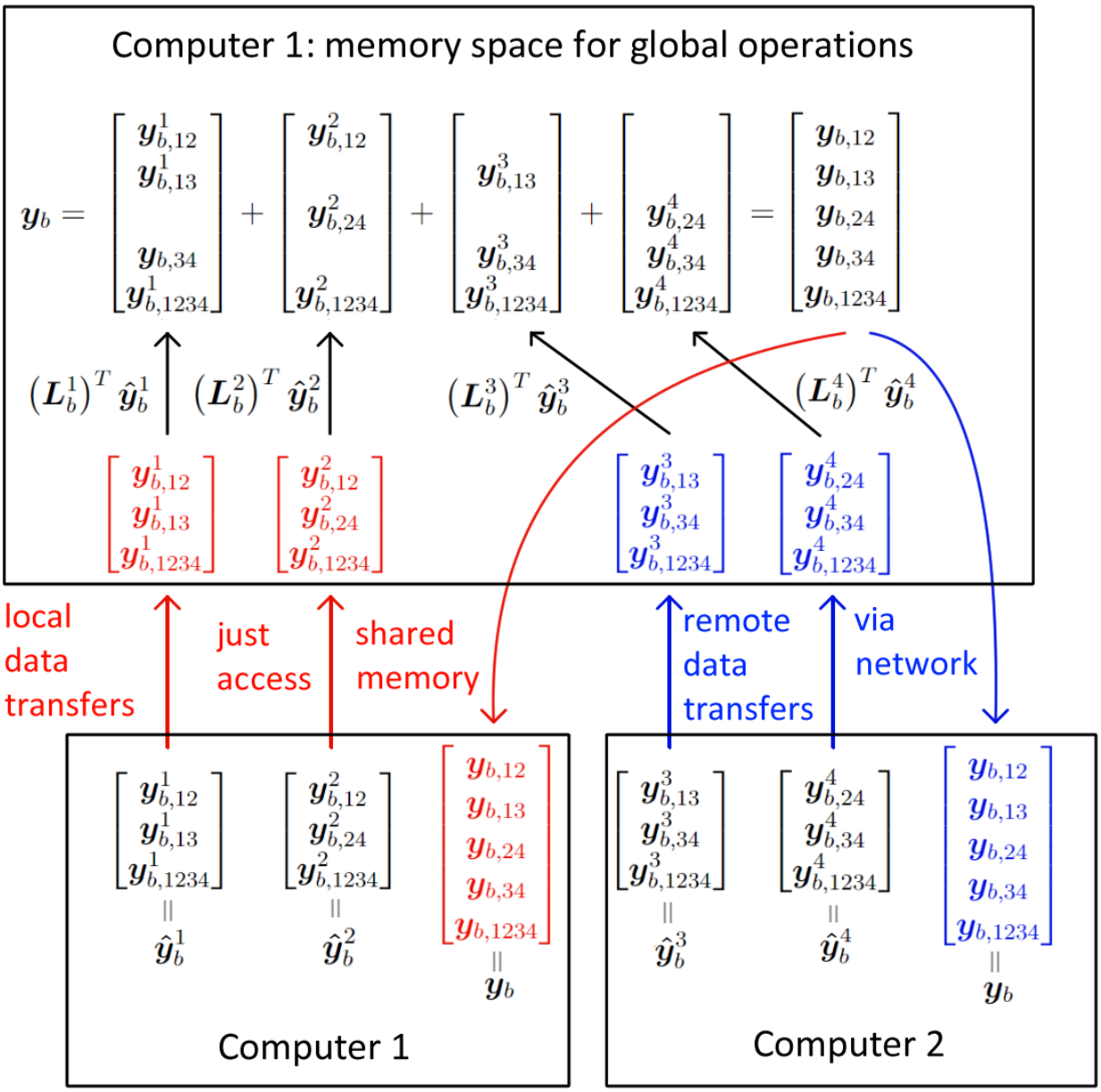


Figure 5.16: Map-reduce operation for global-level vectors.

$$\boldsymbol{M}_b^e = \begin{bmatrix} \boldsymbol{M}_b^{11} & \boldsymbol{M}_b^{12} & \cdots & \boldsymbol{M}_b^{1n_s} \\ \boldsymbol{M}_b^{21} & \boldsymbol{M}_b^{22} & \cdots & \boldsymbol{M}_b^{2n_s} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{M}_b^{n_s1} & \boldsymbol{M}_b^{n_s2} & \cdots & \boldsymbol{M}_b^{n_sn_s} \end{bmatrix} \tag{5.69}$$

and summing all vectors of the expanded domain can be done as

$$\boldsymbol{y}_b^e = \begin{bmatrix} \boldsymbol{y}_b^1 \\ \vdots \\ \boldsymbol{y}_b^{n_s} \end{bmatrix} = \boldsymbol{M}_b^e \hat{\boldsymbol{y}}_b^e \tag{5.70}$$

where $\boldsymbol{y}_b^e$ ($n_b^e \times 1$) is an expanded-domain force vector that contains the same entries as the global vector $\boldsymbol{y}_b$ ($n_b \times 1$). Similarly to the $\boldsymbol{u}_b^e$, $\boldsymbol{u}_b$ pair of vectors, entries corresponding to the same DOFs are listed only once in $\boldsymbol{y}_b$, while in $\boldsymbol{y}_b^e$ they are listed once per subdomain vector $\boldsymbol{y}_b^s$, provided that DOF belongs to subdomain s. Therefore, the following conclusion will be used to transform the original P-FETI-DP and FETI-DP equations:

> "The *centralized* map-reduce operation $\boldsymbol{y}_b = (\boldsymbol{L}_b^e)^T \hat{\boldsymbol{y}}_b^e$ can be replaced with the *distributed* equivalent $\boldsymbol{y}_b^e = \boldsymbol{M}_b^e \hat{\boldsymbol{y}}_b^e$."

This distributed alternative of map-reduce operations can be implemented much more efficiently in computer clusters. Each subdomain s only needs to exchange the common entries of its sumbdomain-level vector $\hat{\boldsymbol{y}}_b^s$ with its neighbors, while for the rest $\boldsymbol{M}_b^{st} = \boldsymbol{0}$ and $\boldsymbol{M}_b^{ts} = \boldsymbol{0}$. This means that:

- Neighboring subdomains that belong to the same group, therefore the same computer, exchange vector data with negligible cost, since these entries exist in the same memory space.
- Neighboring subdomains that belong to different groups, therefore different computers, exchange vector data over the network, using *mpi_send* and *mpi_receive* or equivalent MPI commands, if they are supported by the MPI library.
- To further minimize communication, only the common entries of $\hat{\boldsymbol{y}}_b^s$, $\hat{\boldsymbol{y}}_b^t$ are exchanged between two subdomains s, t, instead of the whole vectors.

With the above procedure, the communication between computers is minimized, as well as distributed evenly across the network, while there are no centralized data transfers to congest the network at any point, as is the case with the original map-reduce operation. Furthermore, instead of performing global-level map-reduce operations at a single computer, while the rest are idle, the proposed implementation distributes the computations evenly across all computers. An example is depicted in figure 5.17 for the case described in figure 5.15. Using the matrices $\boldsymbol{M}_b^{1s}$ isolates the entries at boundary DOFs that are common between subdomain 1 and each other subdomain s. Only these common entries are then transferred to the memory space where subdomain 1 exists and finally added to the vector of subdomain 1. While these steps are executed for subdomain 1, all other subdomains are also processed in parallel and with the same steps.

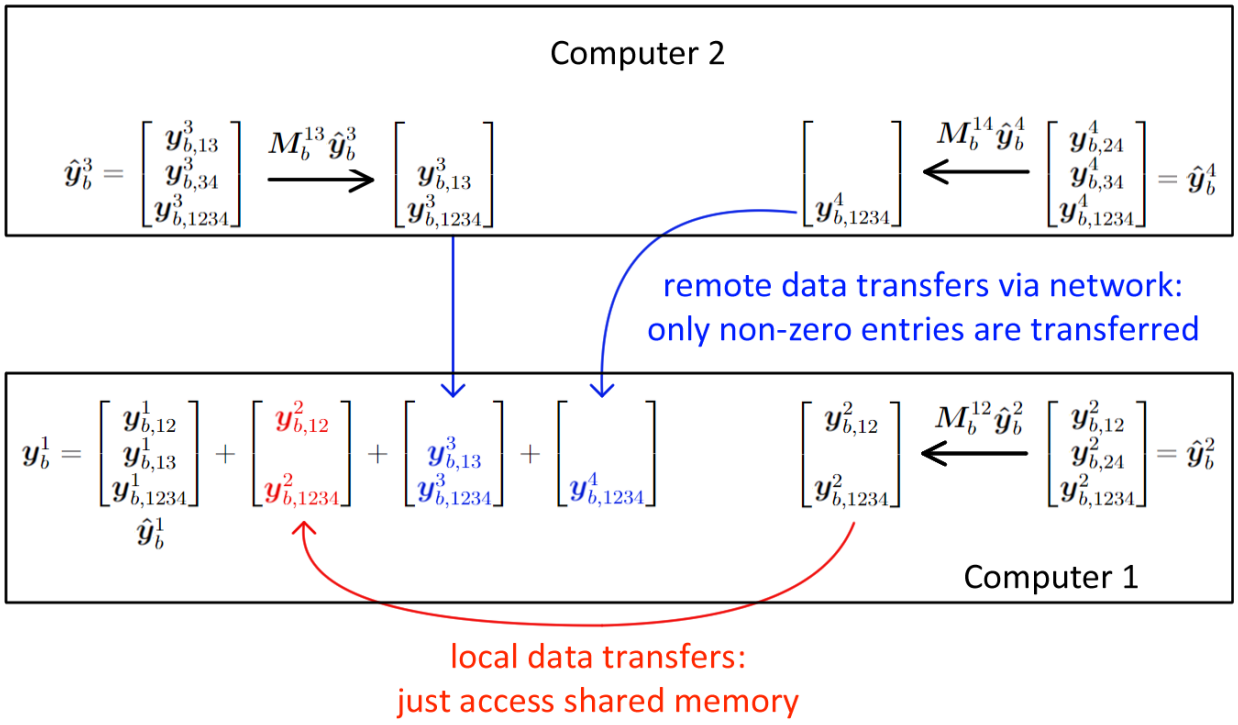


Figure 5.17: Distributed version of the map-reduce operation.

By using the above logic, the distributed alternative for the RHS vector $\boldsymbol{z}_b = (\boldsymbol{L}_b^e)^T \hat{\boldsymbol{z}}_b^e$ of P-FETI-DP's interface problem is

$$\boldsymbol{z}_b^e = \boldsymbol{M}_b^e \hat{\boldsymbol{z}}_b^e \tag{5.71}$$

The matrix-vector operations $(\boldsymbol{L}_b^e)^T \boldsymbol{S}_{bb}^e \boldsymbol{x}_b^e$ can be written as

$$\boldsymbol{y}_b^e = \boldsymbol{M}_b^e \boldsymbol{S}_{bb}^e \boldsymbol{x}_b^e \tag{5.72}$$

and finally the linear system that expresses the interface problem is equivalent to

$$\boldsymbol{M}_b^e \boldsymbol{S}_{bb}^e \boldsymbol{x}_b^e = \boldsymbol{M}_b^e \hat{\boldsymbol{z}}_b^e \tag{5.73}$$

5.5.2.2 Coarse problem

The distributed implementation of the coarse problem of P-FETI-DP, which is the same as FETI-DP, is similar to the interface problem's implementation. The original coarse problem from equation(5.36) is

$$\boldsymbol{S}_{cc} \cdot \boldsymbol{x}_c = \boldsymbol{y}_c$$

where $\boldsymbol{S}_{cc}$ $(n_c \times n_c)$, $\boldsymbol{x}_c$ $(n_c \times 1)$, $\boldsymbol{y}_c$ $(n_c \times 1)$ are the stiffness matrix, displacement vector and force vector, respectively, defined at global corner DOFs. The number of global corner DOFs is n_c, while the number of corner DOFs of subdomain s is n_c^s. Similarly, n_c^e is the number of corner DOFs of the expanded domain, where each corner DOF is included once per subdomain it belongs to. Mapping between global vectors $\boldsymbol{x}_c$, $\boldsymbol{y}_c$ and subdomain vectors $\boldsymbol{x}_c^s$ $(n_c^s \times 1)$, $\hat{\boldsymbol{y}}_c^s$ $(n_c^s \times 1)$ is done with the unsigned boolean mapping matrices $\boldsymbol{L}_c^s$ $(n_c^s \times n_c)$:

$$\begin{aligned} \boldsymbol{x}_c^s &= \boldsymbol{L}_c^s \boldsymbol{x}_c \\ \boldsymbol{y}_c &= \sum_{s=1}^{n_s} \left(\boldsymbol{L}_c^s\right)^T \hat{\boldsymbol{y}}_c^s \end{aligned} \tag{5.74}$$

and for the expanded domain

$$\begin{aligned} \boldsymbol{x}_c^e &= \boldsymbol{L}_c^e \boldsymbol{x}_c \\ \boldsymbol{y}_c &= \left(\boldsymbol{L}_c^e\right)^T \hat{\boldsymbol{y}}_c^e \end{aligned} \tag{5.75}$$

where $\boldsymbol{x}_c^e$ $(n_c^e \times 1)$, $\hat{\boldsymbol{y}}_c^e$ $(n_c^e \times 1)$ and $\boldsymbol{L}_c^e$ $(n_c^e \times n_c)$:

$$\boldsymbol{L}_c^e = \begin{bmatrix} \boldsymbol{L}_c^1 \\ \vdots \\ \boldsymbol{L}_c^{n_s} \end{bmatrix} \tag{5.76}$$

The coarse problem matrix can be written as

$$\boldsymbol{S}_{cc} = \sum_{s=1}^{n_s} (\boldsymbol{L}_c^s)^T \boldsymbol{S}_{cc}^s \boldsymbol{L}_c^s = (\boldsymbol{L}_c^e)^T \boldsymbol{S}_{cc}^e \boldsymbol{L}_c^e \tag{5.77}$$

where $\boldsymbol{S}_{cc}^s$ $(n_c^s \times n_c^s)$ is given by equation(5.25) and $\boldsymbol{S}_{cc}^e$ $(n_c^e \times n_c^e)$ is the corresponding expanded domain matrix. Therefore the coarse problem can be written as

$$\boldsymbol{S}_{cc} \cdot \boldsymbol{x}_c = \boldsymbol{y}_c \iff (\boldsymbol{L}_c^e)^T \boldsymbol{S}_{cc}^e \boldsymbol{L}_c^e \boldsymbol{x}_c = \boldsymbol{y}_c \tag{5.78}$$

or equivalently

$$(\boldsymbol{L}_c^e)^T \boldsymbol{S}_{cc}^e \boldsymbol{x}_c^e = \left(\boldsymbol{L}_c^e\right)^T \hat{\boldsymbol{y}}_c^e \tag{5.79}$$

Similarly to the boundary DOFs, for each ordered pair of subdomains (s, t), an unsigned boolean matrix, namely a matrix that has only 0, 1 as values, $\boldsymbol{M}_c^{st}$ $(n_c^s \times n_c^t)$ is defined, which maps the corner DOFs of t into the corner DOFs of s. Specifically, an entry i, j of $\boldsymbol{M}_c^{st}$ is 1, only if row i and column j correspond to the same corner DOF in subdomains s and t, respectively. If two subdomains s, t are not neighbors, then they do not have common DOFs and the matrices are $\boldsymbol{M}_c^{st} = \boldsymbol{0}$, $\boldsymbol{M}_c^{ts} = \boldsymbol{0}$. On the other hand, for the same subdomain s: $\boldsymbol{M}_c^{ss} = I$. Map-reduce operations of subdomain vectors into a global one are performed in

a distributed fashion using $\boldsymbol{M}_c^{st}$. For the vector $\hat{\boldsymbol{y}}_c^s$ of a given subdomain s, summing the corresponding entries of other sumbdomains can be performed as

$$\boldsymbol{y}_c^s = \sum_{\substack{t=1,\cdots n_s \\ t \neq s}} \boldsymbol{M}_c^{st} \hat{\boldsymbol{y}}_c^s \tag{5.80}$$

The corresponding expanded domain matrix $\boldsymbol{M}_c^e$ $(n_b^e \times n_b^e)$ is defined as

$$\boldsymbol{M}_c^e = \begin{bmatrix} \boldsymbol{M}_c^{11} & \boldsymbol{M}_c^{12} & \cdots & \boldsymbol{M}_c^{1n_s} \\ \boldsymbol{M}_c^{21} & \boldsymbol{M}_c^{22} & \cdots & \boldsymbol{M}_c^{2n_s} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{M}_c^{n_s 1} & \boldsymbol{M}_c^{n_s 2} & \cdots & \boldsymbol{M}_c^{n_s n_s} \end{bmatrix} \tag{5.81}$$

and summing all vectors of the expanded domain can be done as

$$\boldsymbol{y}_c^e = \begin{bmatrix} \boldsymbol{y}_c^1 \\ \vdots \\ \boldsymbol{y}_c^{n_s} \end{bmatrix} = \boldsymbol{M}_c^e \hat{\boldsymbol{y}}_c^e \tag{5.82}$$

where $\boldsymbol{y}_c^e$ $(n_c^e \times 1)$ is an expanded force vector that contains the same entries as the global vector $\boldsymbol{y}_c$ $(n_c \times 1)$. Entries corresponding to the same DOFs are listed only once in $\boldsymbol{y}_c$, while in $\boldsymbol{y}_c^e$ they are listed once per subdomain vector $\boldsymbol{y}_c^s$, provided that DOF belongs to subdomain s. Finally, the matrix-vector operations $(\boldsymbol{L}_c^e)^T \boldsymbol{S}_{cc}^e \boldsymbol{x}_c^e$ can be written as $\boldsymbol{M}_c^e \boldsymbol{S}_{cc}^e \boldsymbol{x}_c^e$ and the linear system that expresses the coarse problem in equation(5.79) is equivalent to

$$\boldsymbol{M}_c^e \boldsymbol{S}_{cc}^e \boldsymbol{x}_c^e = \boldsymbol{M}_c^e \hat{\boldsymbol{y}}_c^e \tag{5.83}$$

In this dissertation, the following approaches are investigated for solving the coarse problem of P-FETI-DP (and FETI-DP). Note that this solution has to performed once per iteration of the PCG method used to solve the interface problem of P-FETI-DP (and FETI-DP).

5.5.2.2.1 Distributed iterative strategy

The linear system of equation(5.83) is solved using a nested PCG method. In this case, the subdomain-level matrices $\boldsymbol{S}_{cc}^s$ and vectors $\boldsymbol{x}_c^s$, $\boldsymbol{y}_c^s$ exist only in the memory space of the corresponding computers, without the need to transfer them to a central computer, so that the global coarse problem matrix $\boldsymbol{S}_{cc}$ can be explicitly formed and factorized. Preconditioning of the nested PCG is parallely executed using a diagonal preconditioner, which can be done in a distributed fashion as

$$\begin{aligned} \boldsymbol{P}_D^{-1} &= (\boldsymbol{D}_{cc})^{-1} \\ \boldsymbol{D}_{cc} &= \boldsymbol{M}_c^e \hat{\boldsymbol{D}}_{cc}^e \end{aligned} \tag{5.84}$$

where $\boldsymbol{P}_D^{-1}$ $(n_c^e \times n_c^e)$ is the diagonal preconditioner and $\hat{\boldsymbol{D}}_{cc}^e$ $(n_c^e \times n_c^e)$ is the expanded domain matrix containing the diagonals $\hat{\boldsymbol{D}}_{cc}^s$ $(n_c^s \times n_c^s)$ of the subdomain-level matrices $\boldsymbol{S}_{cc}^s$. The calculation of $\boldsymbol{D}_{cc}$ and $\boldsymbol{P}_D^{-1}$ is performed before the iterations of the interface problem PCG and the nested coarse problem PCG start.

The *distributed iterative approach* minimizes the memory requirements of the coarse problem. Additionally, it ensures that the stored data and operations needed to solve the coarse problem are distributed evenly across all computers. Therefore, it can be employed for extremely large scale problems, where the number of subdomains is so high, that explicitly storing and factorizing the coarse problem in a single computer is not viable. On the other hand, repeatedly using the nested PCG to solve linear systems with the same matrix is not optimal in terms of computing time.

The iterations required for the coarse-problem PCG can be reduced by relaxing the corresponding convergence tolerance, which will result in less accurate coarse-problem solutions. As far as P-FETI-DP is concerned, the coarse problem is part of the preconditioner of equation(5.49). Therefore, less accurate coarse-problem solutions lower the accuracy of the preconditioner, which leads to more iterations of the interface-problem PCG. In many cases, the convergence decrease of the interface-problem PCG is not very severe and relaxing the coarse-problem PCG ends up reducing the overall computing time. On the other hand, this technique cannot be used in FETI-DP, because the coarse problem is included in the interface-problem matrix of equation(5.31), instead of the precondtioner. Consequently, reducing the accuracy of the coarse-problem solution will lead to an incorrect solution, instead of just decreasing the convergence rate.

5.5.2.2.2 Centralized direct strategy

The linear system $\boldsymbol{S}_{cc} \cdot \boldsymbol{x}_c = \boldsymbol{y}_c$ of equation(5.36) is solved using a direct solver, such as supernodal Cholesky. Before starting the solution of the coarse and interface problems, algorithm 5.7 is used to prepare the coarse problem matrix:

Algorithm 5.7 Preparation of the centralized direct coarse problem

1: Gather subdomain matrices $\boldsymbol{S}_{cc}^s$ from all computers to a central one.

2: Explicitly form the global matrix $\boldsymbol{S}_{cc} = \sum_{s=1}^{n_s} \left(\boldsymbol{L}_c^s\right)^T \boldsymbol{S}_{cc}^s \boldsymbol{L}_c^s$ in the central computer.

3: Factorize the global matrix using Cholesky factorization ($\boldsymbol{S}_{cc} = \boldsymbol{U}_{cc}^T \boldsymbol{U}_{cc}$) in the central computer

During each iteration of the interface problem PCG, the coarse problem needs to be solved with the same matrix but a different RHS vector $\boldsymbol{M}_c^e \hat{\boldsymbol{y}}_c^e$. Algorithm 5.8 is used, which has the same input and output, namely RHS vector $\hat{\boldsymbol{y}}_c^e$ and solution vector $\boldsymbol{x}_c^e$, as the *distributed iterative approach*.

Algorithm 5.8 Solution of the centralized direct coarse problem

1: Input: Subdomain vectors $\hat{\boldsymbol{y}}_c^s$
2: Gather subdomain vectors $\hat{\boldsymbol{y}}_c^s$ from all computers to the central one.
3: Map-reduce these subdomain vectors $\boldsymbol{y}_c = \sum_{s=1}^{n_s} (\boldsymbol{L}_c^s)^T \hat{\boldsymbol{y}}_c^s$ in the central computer.
4: Solve the linear system $\boldsymbol{S}_{cc}\boldsymbol{x}_c = \boldsymbol{y}_c$ by performing back & forward substitution with the Cholesky factors $\boldsymbol{U}_{cc}^T$, $\boldsymbol{U}_{cc}$ in the central computer.
5: Isolate the subdomain displacement vectors $\boldsymbol{x}_c^s = \boldsymbol{L}_c \boldsymbol{x}_c$ in the central computer.
6: Scatter the subdomain vectors $\boldsymbol{x}_c^s$ to their corresponding computers.

This approach is usually significantly faster than the *distributive iterative* solution of the coarse problem, since most of the work is done only once during the factorization of the coarse problem matrix. The back & forward substitutions, which are performed once per iteration of the interface problem PCG, are much faster in comparison. In contrast, the memory requirements are increased for the central computer, since the global matrix $\boldsymbol{S}_{cc}$ needs to be stored and factorized there. Nevertheless, this is usually a preferable trade-off, because the coarse problem is significantly smaller than the global and interface problems. Only in extremely large-scale problems with thousands of subdomains, are the memory requirements potentially too high for this centralized approach. Moreover, gathering/scattering subdomain-level matrices and vectors to/from a centralized computer may cause a congestion of the network at that computer, but, once more, that becomes a concern when the problem grows beyond a certain size. Finally, there are load imbalances hindering performance, since the factorization of $\boldsymbol{S}_{cc}$ and back/forward substitutions are all performed by a single computer, while the rest are idle.

5.5.2.2.3 Distributed direct strategy

This is the same as the *centralized direct approach*, but the coarse problem solution is performed on all computers. In this case, the subdomain matrices $\boldsymbol{S}_{cc}^s$ are gathered to all computers, where the formation and factorization of the global matrix $\boldsymbol{S}_{cc}$ are performed. Algorithms 5.9 and 5.10 describe the procedure.

Algorithm 5.9 Preparation of the distributed direct coarse problem

1: Gather subdomain matrices $\boldsymbol{S}_{cc}^s$ from all computers to all computers.
2: Explicitly form the global matrix $\boldsymbol{S}_{cc} = \sum_{s=1}^{n_s} (\boldsymbol{L}_c^s)^T \boldsymbol{S}_{cc}^s \boldsymbol{L}_c^s$ in all computers.
3: Factorize the global matrix using Cholesky factorization ($\boldsymbol{S}_{cc} = \boldsymbol{U}_{cc}^T \boldsymbol{U}_{cc}$) in all computers.

Algorithm 5.10 Solution of the distributed direct coarse problem

1: Input: Subdomain vectors $\hat{\boldsymbol{y}}_c^s$
2: Gather subdomain vectors $\hat{\boldsymbol{y}}_c^s$ from all computers to all computers.
3: Map-reduce these subdomain vectors $\boldsymbol{y}_c = \sum_{s=1}^{n_s} \left(\boldsymbol{L}_c^s\right)^T \hat{\boldsymbol{y}}_c^s$ in all computers.
4: Solve the linear system $\boldsymbol{S}_{cc}\boldsymbol{x}_c = \boldsymbol{y}_c$ by performing back & forward substitution with the Cholesky factors $\boldsymbol{U}_{cc}^T$, $\boldsymbol{U}_{cc}$ in all computers.
5: Isolate the displacement vector $\boldsymbol{x}_c^s = \boldsymbol{L}_c\boldsymbol{x}_c$ of each subdomain s, but only in the computer corresponding to subdomain s.

This redundancy increases the memory requirements of all computers, contrary to the *centralized direct approach*, where only one central computer requires more memory. Nevertheless, many distributed memory systems consist of computers that have the same available memory, in which case this redundancy is inconsequential, since the extra memory would be unoccupied either way. On the other hand, the *distributed direct approach* can be more efficient, due to better load balancing during the solution of the coarse problem. The factorized $\boldsymbol{S}_{cc}$ matrix is available on all computers, thus each one solves the coarse problem independently, instead of idly waiting the central computer to finish the solution and scatter the corresponding subdomain vectors.

5.5.2.3 Preconditioner

The expanded domain form equation(5.49), which defines the P-FETI-DP preconditioner, is

$$\begin{aligned}
\widetilde{\boldsymbol{A}}_{bb}^{-1} &= \boldsymbol{A}_{bb}^{r,e} + \boldsymbol{A}_{bc}^e \boldsymbol{A}_{cc} \boldsymbol{A}_{cb}^e \\
\boldsymbol{A}_{bb}^{r,e} &= \left(\boldsymbol{L}_{pr}^e\right)^T \left(\boldsymbol{K}_{rr}^e\right)^{-1} \boldsymbol{L}_{pr}^e \\
\boldsymbol{A}_{cb}^e &= -\left(\boldsymbol{L}_c^e\right)^T \boldsymbol{K}_{cr}^e \left(\boldsymbol{K}_{rr}^e\right)^{-1} \boldsymbol{L}_{pr}^e + \boldsymbol{N}_{c,b} \\
\boldsymbol{A}_{bc}^e &= \left(\boldsymbol{A}_{cr}^e\right)^T = \left(\boldsymbol{N}_{c,b}\right)^T - \left(\boldsymbol{L}_{pr}^e\right)^T \left(\boldsymbol{K}_{rr}^e\right)^{-1} \boldsymbol{K}_{rc}^e \boldsymbol{L}_c^e \\
\boldsymbol{A}_{cc} &= \boldsymbol{S}_{cc}^{-1}
\end{aligned} \tag{5.85}$$

where

$$\boldsymbol{L}_{pr}^e = \boldsymbol{N}_{r,b}^e \boldsymbol{L}_{pb}^e \tag{5.86}$$

and

$$\boldsymbol{L}_{pb}^e = \boldsymbol{W}_b^e \boldsymbol{L}_b^e \tag{5.87}$$

Therefore the preconditioner can be rewritten as

$$
\begin{aligned}
\widetilde{\boldsymbol{A}}_{bb}^{-1} &= \widetilde{\boldsymbol{A}}_{bb}^{r,e} + \widetilde{\boldsymbol{A}}_{bc}^{e} \boldsymbol{A}_{cc} \widetilde{\boldsymbol{A}}_{cb}^{e} \\
\widetilde{\boldsymbol{A}}_{bb}^{r,e} &= \left(\boldsymbol{L}_b^e\right)^T \boldsymbol{W}_b^e \left(\boldsymbol{N}_{r,b}^e\right)^T \left(\boldsymbol{K}_{rr}^e\right)^{-1} \boldsymbol{N}_{r,b}^e \boldsymbol{W}_b^e \boldsymbol{L}_b^e \\
\widetilde{\boldsymbol{A}}_{cb}^{e} &= -\left(\boldsymbol{L}_c^e\right)^T \boldsymbol{K}_{cr}^e \left(\boldsymbol{K}_{rr}^e\right)^{-1} \boldsymbol{N}_{r,b}^e \boldsymbol{W}_b^e \boldsymbol{L}_b^e + \boldsymbol{N}_{c,b} \\
\widetilde{\boldsymbol{A}}_{bc}^{e} &= \left(\widetilde{\boldsymbol{A}}_{cr}^{e}\right)^T = \left(\boldsymbol{N}_{c,b}\right)^T - \left(\boldsymbol{L}_b^e\right)^T \boldsymbol{W}_b^e \left(\boldsymbol{N}_{r,b}^e\right)^T \left(\boldsymbol{K}_{rr}^e\right)^{-1} \boldsymbol{K}_{rc}^e \boldsymbol{L}_c^e \\
\boldsymbol{A}_{cc} &= \boldsymbol{S}_{cc}^{-1}
\end{aligned}
\tag{5.88}
$$

where $\left(\boldsymbol{W}_b^e\right)^T = \boldsymbol{W}_b^e$ has been used, since $\boldsymbol{W}_b^e$ $(n_b^e \times n_b^e)$ is diagonal. Let $\boldsymbol{N}_{c,b}^s$ be a boolean matrix (0, 1 as entries) with dimensions $(n_c^s \times n_b^s)$ that maps the boundary DOFs of the subdomain s (columns) to the corner DOFs of subdomain s (rows). Then, its expanded domain form is $\boldsymbol{N}_{c,b}^e$ $(n_c^e \times n_b^e)$. For a force vector $\boldsymbol{y}_b$, the following multiplications are equivalent:

$$
\boldsymbol{N}_{c,b} \boldsymbol{y}_b = \left(\boldsymbol{L}_c^e\right)^T \boldsymbol{N}_{c,b}^e \boldsymbol{W}_b^e \boldsymbol{L}_b^e \boldsymbol{y}_b = \left(\boldsymbol{L}_c^e\right)^T \boldsymbol{N}_{c,b}^e \boldsymbol{W}_b^e \boldsymbol{y}_b^e \tag{5.89}
$$

where it should be noted that multiplying with $\left(\boldsymbol{L}_c^e\right)^T$ will sum the contributions from all subdomains for any DOF, thus they are scaled with the matrix $\boldsymbol{W}_b^e$ to replicate the action of $\left(\boldsymbol{L}_c^e\right)^T$. As a result, multiplying a force vector $\boldsymbol{y}_b$ with $\widetilde{\boldsymbol{A}}_{cb}^e$ is equivalent to

$$
\begin{aligned}
\widetilde{\boldsymbol{A}}_{cb}^{e} \boldsymbol{y}_b &= -\left(\boldsymbol{L}_c^e\right)^T \boldsymbol{K}_{cr}^e \left(\boldsymbol{K}_{rr}^e\right)^{-1} \boldsymbol{N}_{r,b}^e \boldsymbol{W}_b^e \boldsymbol{L}_b^e \boldsymbol{y}_b + \boldsymbol{N}_{c,b} \boldsymbol{y}_b \\
&= -\left(\boldsymbol{L}_c^e\right)^T \boldsymbol{K}_{cr}^e \left(\boldsymbol{K}_{rr}^e\right)^{-1} \boldsymbol{N}_{r,b}^e \boldsymbol{W}_b^e \boldsymbol{y}_b^e + \left(\boldsymbol{L}_c^e\right)^T \boldsymbol{N}_{c,b}^e \boldsymbol{W}_b^e \boldsymbol{y}_b^e \\
&= \left(\boldsymbol{L}_c^e\right)^T \Big(-\boldsymbol{K}_{cr}^e \left(\boldsymbol{K}_{rr}^e\right)^{-1} \boldsymbol{N}_{r,b}^e + \boldsymbol{N}_{c,b}^e \Big) \boldsymbol{W}_b^e \boldsymbol{y}_b^e
\end{aligned}
\tag{5.90}
$$

Furthermore, for a displacement vector $\boldsymbol{x}_c$, the multiplication

$$
\widetilde{\boldsymbol{A}}_{bc}^{e} \boldsymbol{x}_c = \left(\boldsymbol{N}_{c,b}\right)^T \boldsymbol{x}_c - \left(\boldsymbol{L}_b^e\right)^T \boldsymbol{W}_b^e \left(\boldsymbol{N}_{r,b}^e\right)^T \left(\boldsymbol{K}_{rr}^e\right)^{-1} \boldsymbol{K}_{rc}^e \boldsymbol{L}_c^e \boldsymbol{x}_c \tag{5.91}
$$

can be done in a distributed fashion as

$$
\begin{aligned}
&\left(\boldsymbol{N}_{c,b}^e\right)^T \boldsymbol{L}_c^e \boldsymbol{x}_c - \boldsymbol{M}_b^e \boldsymbol{W}_b^e \left(\boldsymbol{N}_{r,b}^e\right)^T \left(\boldsymbol{K}_{rr}^e\right)^{-1} \boldsymbol{K}_{rc}^e \boldsymbol{L}_c^e \boldsymbol{x}_c \\
&= \Big(\left(\boldsymbol{N}_{c,b}^e\right)^T - \boldsymbol{M}_b^e \boldsymbol{W}_b^e \left(\boldsymbol{N}_{r,b}^e\right)^T \left(\boldsymbol{K}_{rr}^e\right)^{-1} \boldsymbol{K}_{rc}^e \Big) \boldsymbol{L}_c^e \boldsymbol{x}_c
\end{aligned}
\tag{5.92}
$$

Additionally, for a force vector $\boldsymbol{y}_b$, the multiplication

$$
\widetilde{\boldsymbol{A}}_{bb}^{r,e} \boldsymbol{y}_b = \left(\boldsymbol{L}_b^e\right)^T \boldsymbol{W}_b^e \left(\boldsymbol{N}_{r,b}^e\right)^T \left(\boldsymbol{K}_{rr}^e\right)^{-1} \boldsymbol{N}_{r,b}^e \boldsymbol{W}_b^e \boldsymbol{L}_b^e \boldsymbol{y}_b \tag{5.93}
$$

can be done in a distributed fashion as

$$\boldsymbol{M}_b^e \boldsymbol{W}_b^e \left(\boldsymbol{N}_{r,b}^e\right)^T \left(\boldsymbol{K}_{rr}^e\right)^{-1} \boldsymbol{N}_{r,b}^e \boldsymbol{W}_b^e \boldsymbol{y}_b^e \tag{5.94}$$

Consequently, the preconditioner takes the following distributed form

$$\begin{aligned}
(\widetilde{\boldsymbol{A}}_{bb}^e)^{-1} &= \boldsymbol{A}_{bb}^{r,e} + \boldsymbol{A}_{bc}^e \boldsymbol{A}_{cc}^e \boldsymbol{A}_{cb}^e \\
\boldsymbol{A}_{bb}^{r,e} &= \boldsymbol{M}_b^e \boldsymbol{W}_b^e \left(\boldsymbol{N}_{r,b}^e\right)^T \left(\boldsymbol{K}_{rr}^e\right)^{-1} \boldsymbol{N}_{r,b}^e \boldsymbol{W}_b^e \\
\boldsymbol{A}_{cb}^e &= \Big(- \boldsymbol{K}_{cr}^e \left(\boldsymbol{K}_{rr}^e\right)^{-1} \boldsymbol{N}_{r,b}^e + \boldsymbol{N}_{c,b}^e \Big) \boldsymbol{W}_b^e \\
\boldsymbol{A}_{bc}^e &= \left(\boldsymbol{N}_{c,b}^e\right)^T - \boldsymbol{M}_b^e \boldsymbol{W}_b^e \left(\boldsymbol{N}_{r,b}^e\right)^T \left(\boldsymbol{K}_{rr}^e\right)^{-1} \boldsymbol{K}_{rc}^e \\
\boldsymbol{A}_{cc}^e &= \boldsymbol{L}_c^e \boldsymbol{S}_{cc}^{-1} \left(\boldsymbol{L}_c^e\right)^T
\end{aligned} \tag{5.95}$$

Note that during PCG, this preconditioner will be multiplied with force-like vectors (residual vectors to be exact) $\boldsymbol{y}_b^e$ and produce displacement-like vectors (preconditioned-residual vectors to be exact) $\boldsymbol{x}_b^e$. All these matrices and vectors refer to the expanded domain and can be stored and operated on in a distributive fashion. The only global operations left are in the solution of the coarse problem, which is performed by multiplying a force vector $\boldsymbol{y}_c$ with the matrix $\boldsymbol{A}_{cc}^e$:

$$\boldsymbol{x}_c = \boldsymbol{A}_{cc}^e \boldsymbol{y}_c = \boldsymbol{L}_c^e \boldsymbol{S}_{cc}^{-1} \left(\boldsymbol{L}_c^e\right)^T \boldsymbol{y}_c \tag{5.96}$$

As elaborated in section 5.5.2.2, the above can be done in two ways:

- Direct solution of the global system

$$\left(\boldsymbol{L}_c^e\right)^T \boldsymbol{S}_{cc} \boldsymbol{L}_c^e \boldsymbol{x}_c = \boldsymbol{y}_c = \left(\boldsymbol{L}_c^e\right)^T \hat{\boldsymbol{y}}_c^e$$

- Iterative solution of the equivalent distributed system

$$\boldsymbol{M}_c^e \boldsymbol{S}_{cc}^e \boldsymbol{x}_c^e = \boldsymbol{M}_c^e \hat{\boldsymbol{y}}_c^e$$

5.5.2.4 Implementation details

This section describes some details pertaining to the parallel implementation of P-FETI-DP, while also avoiding duplicate operations. Specifically, algorithm 5.11 describes the matrix-vector multiplications performed during the solution of the interface problem in equation (5.73), while algorithm 5.12 describes the application of equation (5.95) for the preconditioning step of PCG.

Algorithm 5.11 Matrix-vector multiplication for interface problem of P-FETI-DP.

Input: displacement vector $\boldsymbol{x}_b^e$. Output: force vector $\boldsymbol{y}_b^e$
for each subdomain s **do** ▷ In parallel
  $\boldsymbol{v}_1 = \boldsymbol{K}_{bb}^s \cdot \boldsymbol{x}_b^s$
  $\boldsymbol{v}_2 = \boldsymbol{K}_{ib}^s \cdot \boldsymbol{v}_1$
  $\boldsymbol{v}_3 = (\boldsymbol{K}_{ii}^s)^{-1} \cdot \boldsymbol{v}_2$ ▷ Back & forward substitutions with factors of $\boldsymbol{K}_{ii}^s$
  $\boldsymbol{v}_4 = \boldsymbol{K}_{bi}^s \cdot \boldsymbol{v}_3$
  $\hat{\boldsymbol{y}}_b^s = \boldsymbol{v}_1 + \boldsymbol{v}_3$
end for
$\boldsymbol{y}_b^e = \boldsymbol{M}_b^e \cdot \hat{\boldsymbol{y}}_b^e$ ▷ Communication between subdomains

Algorithm 5.12 Preconditioner of P-FETI-DP.

Input: force vector $\boldsymbol{y}_b^e$. Output: displacement vector $\boldsymbol{x}_b^e$
for each subdomain s **do** ▷ In parallel
  $\boldsymbol{v}_1 = \boldsymbol{W}_b^s \cdot \boldsymbol{y}_b^s$
  $\boldsymbol{v}_2 = \boldsymbol{N}_{cb}^s \cdot \boldsymbol{v}_1$
  $\boldsymbol{v}_3 = \boldsymbol{N}_{rb}^s \cdot \boldsymbol{v}_1$
  $\boldsymbol{v}_4 = (\boldsymbol{K}_{rr}^s)^{-1} \cdot \boldsymbol{v}_3$ ▷ Back & forward substitutions with factors of $\boldsymbol{K}_{rr}^s$
  $\boldsymbol{v}_5 = \boldsymbol{K}_{cr}^s \cdot \boldsymbol{v}_4$
  $\hat{\boldsymbol{y}}_c^s = \boldsymbol{v}_2 - \boldsymbol{v}_5$
end for
Solve coarse problem $\boldsymbol{x}_c^e = \boldsymbol{A}_{cc}^e \cdot \hat{\boldsymbol{y}}_c^e$ ▷ See section 5.5.2.2
for each subdomain s **do** ▷ In parallel
  $\boldsymbol{v}_6 = \boldsymbol{K}_{rc}^s \cdot \boldsymbol{x}_c^s$
  $\boldsymbol{v}_7 = (\boldsymbol{K}_{rr}^s)^{-1} \cdot \boldsymbol{v}_6$ ▷ Back & forward substitutions with factors of $\boldsymbol{K}_{rr}^s$
  $\boldsymbol{v}_8 = \boldsymbol{v}_4 - \boldsymbol{v}_7$
  $\boldsymbol{v}_9 = (\boldsymbol{N}_{rb}^s)^T \cdot \boldsymbol{v}_8$
  $\hat{\boldsymbol{v}}_b^s = \boldsymbol{W}_b^s \cdot \boldsymbol{v}_9$
end for
$\boldsymbol{v}_b^e = \boldsymbol{M}_b^e \cdot \hat{\boldsymbol{v}}_b^e$ ▷ Communication between subdomains
for each subdomain s **do** ▷ In parallel
  $\boldsymbol{v}_{10} = (\boldsymbol{N}_{cb}^s)^T \cdot \boldsymbol{x}_c^s$
  $\boldsymbol{x}_b^s = \boldsymbol{v}_b^s + \boldsymbol{v}_{10}$
end for

5.5.3 FETI-DP

In FETI-DP, the unknowns of the interface problem are Lagrange multipliers, namely inter-subdomain forces at boundary-remainder DOFs. Therefore, its left-hand-side (LHS) vectors are force-like quantities and its RHS vectors are displacement-like quantities.

5.5.3.1 Interface problem

A Lagrange multiplier is defined between the corresponding instances of the same boundary-remainder DOF of exactly two subdomains. Let n_λ, n_λ^s and $n_\lambda^e = \sum_{s=1}^{n_s} n_\lambda^s$ be the Lagrange multipliers of the global domain, subdomain s and expanded domain, respectively. Each Lagrange multiplier is counted only once in n_λ. In contrast, a Lagrange multiplier, which is applied to different instances of the same boundary-remainder DOF in two subdomains, is counted twice in in n_λ^e. Similarly, the vectors containing the Lagrange multipliers of the global domain, subdomain s and expanded domain are $\boldsymbol{\lambda}$ $(n_\lambda \times 1)$, $\boldsymbol{\lambda}^s$ $(n_\lambda^s \times 1)$ and $\boldsymbol{\lambda}^e$ $(n_\lambda^e \times 1)$.

In order to obtain a distributed form of the FETI-DP equations, the global Lagrange multipliers vector $\boldsymbol{\lambda}$ needs to be replaced by the expanded domain vector $\boldsymbol{\lambda}^e$, whereas the multiplications with signed boolean matrices $\boldsymbol{B}_r^s$ $(n_\lambda \times n_r^s)$ need to be be replaced by equivalent actions. In the implementation developed here, another signed boolean mapping matrix $\boldsymbol{C}_r^s$ $(n_\lambda^s \times n_r^s)$ is introduced. This $\boldsymbol{C}_r^s$ matrix maps the remainder DOFs of subdomain s (columns) to the Lagrange multipliers of subdomain s (rows), instead of mapping to the global Lagrange multipliers, as done by $\boldsymbol{B}_r^s$. The rules for ± 1 signs of $\boldsymbol{C}_r^s$ are identical with the rules for $\boldsymbol{B}_r^s$. The corresponding expanded domain matrix $\boldsymbol{C}_r^e$ $(n_\lambda^e \times n_r^e)$ is

$$\boldsymbol{C}_r^e = \begin{bmatrix} \boldsymbol{C}_r^1 & & \\ & \ddots & \\ & & \boldsymbol{C}_r^{n_s} \end{bmatrix} \tag{5.97}$$

Using the boolean matrices $\boldsymbol{C}_r^s$ and $\boldsymbol{C}_r^e$, multiplications of force-like vectors $(\boldsymbol{B}_r^s)^T \cdot \boldsymbol{\lambda}$ will be replaced by $(\boldsymbol{C}_r^s)^T \cdot \boldsymbol{\lambda}^s$ and $(\boldsymbol{C}_r^e)^T \cdot \boldsymbol{\lambda}^e$. Let $\boldsymbol{v}_r^s$ $(n_r^s \times 1)$ be a vector that contains displacement quantities along the n_r^s remainder DOFs of subdomain s and $\boldsymbol{v}_r^e$ $(n_r^e \times 1)$ the corresponding expanded domain vector. Multiplications with these displacement-like vectors $\boldsymbol{B}_r^s \cdot \boldsymbol{v}_r^s$ will be replaced by $\boldsymbol{C}_r^s \cdot \boldsymbol{v}_r^s$:

$$\hat{\boldsymbol{\delta}}^s = \boldsymbol{C}_r^s \cdot \boldsymbol{v}_r^s \tag{5.98}$$

where $\hat{\boldsymbol{\delta}}^s$ $(n_\lambda^s \times 1)$ is a vector containing displacement quantities of subdomain s along its Lagrange multipliers. For each ordered pair of subdomains (s, t), an unsigned boolean matrix, namely a matrix that has only 0, 1 as values, $\boldsymbol{M}_\lambda^{st}$ $(n_\lambda^s \times n_\lambda^t)$ is defined, which maps the Lagrange multipliers of t into the Lagrange multipliers of s. Specifically, an entry i, j of $\boldsymbol{M}_\lambda^{st}$ is 1, only if row i and column j correspond to instances of the same Lagrange multiplier in subdomains s and t, respectively. If two subdomains s, t are not neighbors, then they do not have common Lagrange multipliers and the matrices are $\boldsymbol{M}_\lambda^{st} = 0$, $\boldsymbol{M}_\lambda^{ts} = 0$. On the other hand, for the same subdomain s: $\boldsymbol{M}_b^{ss} = I$. Map-reduce operations of subdomain vectors into a global one are performed in a distributed fashion using the $\boldsymbol{M}_\lambda^{st}$ matrices. For the vector $\hat{\boldsymbol{\delta}}^s$ of a given subdomain s, summing the corresponding entries of other sumbdomains can be performed as

$$\boldsymbol{\delta}^s = \sum_{\substack{t=1,\cdots n_s \\ t \neq s}} \boldsymbol{M}_\lambda^{st} \hat{\boldsymbol{\delta}}^s \tag{5.99}$$

and summing all vectors of the expanded domain into $\boldsymbol{\delta}^e$ $(n_\lambda^e \times 1)$ can be done as

$$\boldsymbol{\delta}^e = \boldsymbol{M}_\lambda^e \hat{\boldsymbol{\delta}}^e \tag{5.100}$$

where the expanded domain matrix $\boldsymbol{M}_\lambda^e$ $(n_b^e \times n_b^e)$ is defined as

$$\boldsymbol{M}_\lambda^e = \begin{bmatrix} \boldsymbol{M}_\lambda^{11} & \boldsymbol{M}_\lambda^{12} & \cdots & \boldsymbol{M}_\lambda^{1n_s} \\ \boldsymbol{M}_\lambda^{21} & \boldsymbol{M}_\lambda^{22} & \cdots & \boldsymbol{M}_\lambda^{2n_s} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{M}_\lambda^{n_s 1} & \boldsymbol{M}_\lambda^{n_s 2} & \cdots & \boldsymbol{M}_\lambda^{n_s n_s} \end{bmatrix} \tag{5.101}$$

As a result, the map-reduce operation $\sum_{s=1}^{n_s} (\boldsymbol{B}_r^s \cdot \boldsymbol{v}_r^s)$ will be replaced by the distributed alternative $\boldsymbol{M}_\lambda^e \boldsymbol{C}_r^e \cdot \boldsymbol{v}_r^e$. Applying the aforementioned modifications, the distributed form of the interface problem of FETI-DP is

$$\left(\boldsymbol{F}_{Irr}^e + \boldsymbol{F}_{Irc}^e \boldsymbol{A}_{cc}^e \boldsymbol{F}_{Icr}^e\right) \boldsymbol{\lambda}^e = \boldsymbol{d}_r^e - \boldsymbol{F}_{Irc}^e \boldsymbol{A}_{cc}^e \hat{\boldsymbol{z}}_c^e \tag{5.102}$$

where

$$\begin{aligned} \boldsymbol{F}_{Irr}^e &= \boldsymbol{M}_\lambda^e \boldsymbol{C}_r^e (\boldsymbol{K}_{rr}^e)^{-1} (\boldsymbol{C}_r^e)^T \\ \boldsymbol{F}_{Irc}^e &= \boldsymbol{M}_\lambda^e \boldsymbol{C}_r^e (\boldsymbol{K}_{rr}^e)^{-1} \boldsymbol{K}_{rc}^e \\ \boldsymbol{F}_{Icr}^e &= \boldsymbol{K}_{cr}^e (\boldsymbol{K}_{rr}^e)^{-1} (\boldsymbol{C}_r^e)^T \\ \boldsymbol{A}_{cc}^e &= \boldsymbol{L}_c^e \boldsymbol{S}_{cc}^{-1} (\boldsymbol{L}_c^e)^T \\ \boldsymbol{d}_r^e &= \boldsymbol{M}_\lambda^e \boldsymbol{C}_r^e (\boldsymbol{K}_{rr}^e)^{-1} \boldsymbol{f}_r^e \end{aligned} \tag{5.103}$$

where $\boldsymbol{f}_r^e$ and $\hat{\boldsymbol{z}}_c^e$ are the expanded domain forms of the corresponding vectors defined in equations (5.18, 5.26). The linear system of equation(5.111) is solved in parallel using the PCG method. It can be observed that only the matrix $\boldsymbol{A}_{cc}^e$ involves global operations. Multiplying a vector with $\boldsymbol{A}_{cc}^e$ is equivalent to solving the coarse problem. In fact, $\boldsymbol{A}_{cc}^e$ in the interface problem of FETI-DP is identical with $\boldsymbol{A}_{cc}^e$ in the preconditioner of P-FETI-DP. Therefore, the observations relating to the parallel execution of this operation, given in section 5.5.2.3, apply here as well. After solving of the interface problem and obtaining the Lagrange multipliers $\boldsymbol{\lambda}^e$, the displacements at corner and remainder DOFs can be calculated as

$$\boldsymbol{u}_c^e = \boldsymbol{A}_{cc}^e \left(\hat{\boldsymbol{z}}_c^e + \boldsymbol{F}_{Icr}^e \boldsymbol{\lambda}^e\right) \tag{5.104}$$

$$\boldsymbol{u}_r^e = (\boldsymbol{K}_{rr}^e)^{-1} \left(\boldsymbol{f}_r^e - (\boldsymbol{C}_r^e)^T \boldsymbol{\lambda}^e - \boldsymbol{K}_{rc}^e \boldsymbol{u}_c^e\right) \tag{5.105}$$

However, the above equations will result in the displacements at different instances of the same boundary DOF being slightly different in each subdomain. To obtain compatible displacements across all subdomains, the following averaging operation needs to be done:

$$\check{\boldsymbol{u}}_b^e = \boldsymbol{M}_b^e \boldsymbol{W}_b^e \boldsymbol{u}_b^e \tag{5.106}$$

where $\boldsymbol{u}_b^e$ are the incompatible displacements at boundary DOFs of subdomain s and $\check{\boldsymbol{u}}_b^e$ are the corrected, compatible displacements at the same DOFs.

5.5.3.2 Preconditioners

In the original FETI-DP preconditioners, the signed boolean matrix $\boldsymbol{B}_{b_r}^s$ $(n_\lambda \times n_{b_r}^s)$ is used for mapping the $n_{b_r}^s$ boundary-remainder DOFs of subdomain s (columns) to the n_λ global Lagrange multipliers (rows). In the developed implementation, this will be replaced by $\boldsymbol{C}_{b_r}^s$ $(n_\lambda^s \times n_{b_r}^s)$, which is also signed boolean matrix (0,-1,+1 entries) and maps the boundary-remainder DOFs of subdomain s (columns) to the Lagrange multipliers of subdomain s (rows). The rules for ± 1 signs of $\boldsymbol{C}_{b_r}^s$ are identical with the rules for $\boldsymbol{B}_{b_r}^s$. The corresponding expanded domain matrix $\boldsymbol{C}_{b_r}^e$ $(n_\lambda^e \times n_{b_r}^e)$ of the mapping operation is

$$\boldsymbol{C}_{b_r}^e = \begin{bmatrix} \boldsymbol{C}_{b_r}^1 & & \\ & \ddots & \\ & & \boldsymbol{C}_{b_r}^{n_s} \end{bmatrix} \tag{5.107}$$

and the expanded domain matrix $\boldsymbol{W}_{b_r}^e$ $(n_{b_r}^e \times n_{b_r}^e)$ of the scaling operation is

$$\boldsymbol{W}_{b_r}^e = \begin{bmatrix} \boldsymbol{W}_{b_r}^1 & & \\ & \ddots & \\ & & \boldsymbol{W}_{b_r}^{n_s} \end{bmatrix} \tag{5.108}$$

where the subdomain-level scaling matrices $\boldsymbol{W}_{b_r}^s$ $(n_{b_r}^s \times n_{b_r}^s)$ were defined in section 5.3.1.3. The FETI-DP preconditioner has the following general form

$$(\widetilde{\boldsymbol{F}}_{Irr}^e)^{-1} = \boldsymbol{M}_\lambda^e \boldsymbol{C}_{b_r}^e \boldsymbol{W}_{b_r}^e \widetilde{\boldsymbol{S}}_{b_r b_r}^e \boldsymbol{W}_{b_r}^e (\boldsymbol{C}_{b_r}^e)^T \tag{5.109}$$

Depending on the definition of $\widetilde{\boldsymbol{S}}_{b_r b_r}^e$ the following preconditioners can be obtained:

$$\widetilde{\boldsymbol{S}}_{b_r b_r}^e = \begin{cases} \boldsymbol{K}_{b_r b_r}^e - (\boldsymbol{K}_{ib_r}^e)^T (\boldsymbol{K}_{ii}^e)^{-1} \boldsymbol{K}_{ib_r}^e & \text{Dirichlet preconditioner} \\ \boldsymbol{K}_{b_r b_r}^e & \text{lumped preconditioner} \end{cases} \tag{5.110}$$

The observations of section 5.3.1.3, concerning the performance properties of the Dirichlet and lumped preconditioner, apply here as well.

5.5.3.3 Implementation details

This section describes some details pertaining to the parallel implementation of FETI-DP, while also avoiding duplicate operations. Specifically, algorithm 5.13 describes the matrix-vector multiplications performed during the solution of the interface problem in equation (5.111), while algorithm 5.14 describes the application of the equation (5.109) (Dirichlet-preconditioner version) for the preconditioning step of PCG.

Algorithm 5.13 Matrix-vector multiplication for interface problem of FETI-DP.

Input: force vector $\boldsymbol{\lambda}^e$. Output: displacement vector $\boldsymbol{\delta}^e$
for each subdomain s **do** ▷ In parallel
  $\boldsymbol{v}_1 = (\boldsymbol{C}_r^s)^T \cdot \boldsymbol{\lambda}^s$
  $\boldsymbol{v}_2 = (\boldsymbol{K}_{rr}^s)^{-1} \cdot \boldsymbol{v}_1$ ▷ Back & forward substitutions with factors of $\boldsymbol{K}_{rr}^s$
  $\hat{\boldsymbol{y}}_c^s = \boldsymbol{K}_{cr}^s \cdot \boldsymbol{v}_2$
end for
Solve coarse problem $\boldsymbol{x}_c^e = \boldsymbol{A}_{cc}^e \cdot \hat{\boldsymbol{y}}_c^e$ ▷ See section 5.5.2.2
for each subdomain s **do** ▷ In parallel
  $\boldsymbol{v}_3 = \boldsymbol{K}_{rc}^s \cdot \boldsymbol{x}_c^s$
  $\boldsymbol{v}_4 = (\boldsymbol{K}_{rr}^s)^{-1} \cdot \boldsymbol{v}_3$ ▷ Back & forward substitutions with factors of $\boldsymbol{K}_{rr}^s$
  $\boldsymbol{v}_5 = \boldsymbol{v}_2 + \boldsymbol{v}_4$
  $\hat{\boldsymbol{\delta}}^s = \boldsymbol{C}_r^s \cdot \boldsymbol{v}_5$
end for
$\boldsymbol{\delta}^e = \boldsymbol{M}_\lambda^e \cdot \hat{\boldsymbol{\delta}}^e$ ▷ Communication between subdomains

Algorithm 5.14 Dirichlet preconditioner of FETI-DP.

Input: displacement vector $\boldsymbol{\delta}^e$. Output: force vector $\boldsymbol{\lambda}^e$
for each subdomain s **do** ▷ In parallel
  $\boldsymbol{v}_1 = \left(\boldsymbol{C}_{b_r}^s\right)^T \cdot \boldsymbol{\delta}^s$
  $\boldsymbol{v}_2 = \boldsymbol{W}_{b_r}^s \cdot \boldsymbol{v}_1$
  $\boldsymbol{v}_3 = \boldsymbol{K}_{b_r b_r}^s \cdot \boldsymbol{v}_2$
  $\boldsymbol{v}_4 = \boldsymbol{K}_{ib_r}^s \cdot \boldsymbol{v}_2$
  $\boldsymbol{v}_5 = (\boldsymbol{K}_{ii}^s)^{-1} \cdot \boldsymbol{v}_4$ ▷ Back & forward substitutions with factors of $\boldsymbol{K}_{ii}^s$
  $\boldsymbol{v}_6 = \boldsymbol{K}_{bi}^s \cdot \boldsymbol{v}_5$
  $\boldsymbol{v}_7 = \boldsymbol{v}_3 - \boldsymbol{v}_6$
  $\boldsymbol{v}_8 = \boldsymbol{W}_{b_r}^s \cdot \boldsymbol{v}_7$
  $\hat{\boldsymbol{\lambda}}^s = \boldsymbol{C}_{b_r}^s \cdot \boldsymbol{v}_8$
end for
$\boldsymbol{\lambda}^e = \boldsymbol{M}_\lambda^e \cdot \hat{\boldsymbol{\lambda}}^e$ ▷ Communication between subdomains

Finally, the RHS vector of the interface problem in equation (5.111) can be written as

$$\begin{aligned} \boldsymbol{e}_\lambda^e &= \boldsymbol{d}_r^e - \boldsymbol{F}_{Irc}^e \boldsymbol{A}_{cc}^e \hat{\boldsymbol{z}}_c^e \\ &= \boldsymbol{M}_\lambda^e \boldsymbol{C}_r^e \left(\boldsymbol{K}_{rr}^e\right)^{-1} \boldsymbol{f}_r^e - \boldsymbol{M}_\lambda^e \boldsymbol{C}_r^e \left(\boldsymbol{K}_{rr}^e\right)^{-1} \boldsymbol{K}_{rc}^e \boldsymbol{A}_{cc}^e \left(\boldsymbol{f}_c^e - \boldsymbol{K}_{cr}^e \left(\boldsymbol{K}_{rr}^e\right)^{-1} \boldsymbol{f}_r^e\right) \end{aligned} \tag{5.111}$$

and its implementation is described by algorithm 5.15

Algorithm 5.15 RHS of interface problem of FETI-DP.

for each subdomain s **do** ▷ In parallel
  $\boldsymbol{v}_1 = \left(\boldsymbol{K}_{rr}^s\right)^{-1} \cdot \boldsymbol{f}_r^s$ ▷ Back & forward substitutions with factors of $\boldsymbol{K}_{rr}^s$
  $\hat{\boldsymbol{y}}_c^s = \boldsymbol{f}_c^s - \boldsymbol{K}_{cr}^s \cdot \boldsymbol{v}_1$
end for
Solve coarse problem $\boldsymbol{x}_c^e = \boldsymbol{A}_{cc}^e \cdot \hat{\boldsymbol{y}}_c^e$ ▷ See section 5.5.2.2
for each subdomain s **do** ▷ In parallel
  $\boldsymbol{v}_2 = \boldsymbol{K}_{rc}^s \cdot \boldsymbol{x}_c^s$
  $\boldsymbol{v}_3 = \left(\boldsymbol{K}_{rr}^s\right)^{-1} \cdot \boldsymbol{v}_2$ ▷ Back & forward substitutions with factors of $\boldsymbol{K}_{rr}^s$
  $\boldsymbol{v}_4 = \boldsymbol{v}_1 - \boldsymbol{v}_3$
  $\hat{\boldsymbol{e}}_\lambda^s = \boldsymbol{C}_r^s \cdot \boldsymbol{v}_4$
end for
$\boldsymbol{e}_\lambda^e = \boldsymbol{M}_\lambda^e \cdot \hat{\boldsymbol{e}}_\lambda^e$ ▷ Communication between subdomains

Chapter 6

Crack propagation applications

6.1 Hardware and software setup

In this chapter the proposed FETI-DP and P-FETI-DP algorithms are used to solve two 3D problems involving brittle crack propagation. Their scalability is investigated and their performance in terms of computation time and iterations is compared with other solvers on a computer with the following specifications: Intel(R) Core(TM) i7-X980 CPU (3.33GHz, 6 cores/12 threads) and 24GB RAM. Moreover, the performance of the developed HPC implementations of FETI-DP and P-FETI-DP is investigated in sections 6.2.5, 6.2.6, 6.3.6 and 6.3.7. To this end, a computer cluster is used, which consists of 1-6 computers with the aforementioned specifications, linked via an Ethernet LAN.

The first solver used for comparison is a direct solver based on the supernodal sparse Cholesky factorization (Y. Chen et al., 2008) and the fill-reducing DOF ordering operations, available from the CHOLMOD package of SuiteSparse library (Davis, 2022). SuiteSparse provides high-performance implementations of direct sparse solvers, which are written in the C programming language and take full advantage of the multicore architecture of modern CPUs. Since the main portion of the XFEM code is written in C#, SuiteSparse is compiled and linked with as few calls as possible to reduce any overhead. This solver will be abbreviated as Direct-S in the remaining of this section.

Both FETI-DP and P-FETI-DP are implemented in C# as part of the MSolve (MGroup, 2022) open-source software for computational mechanics. In order to solve the linear systems at subdomain level (operations involving $(\boldsymbol{K}_{rr}^{s})^{-1}$ and $(\boldsymbol{K}_{ii}^{s})^{-1}$) and the coarse problem (operations involving $(\boldsymbol{S}_{cc})^{-1}$), SuiteSparse is used once more. Apart from the parallelism exploited by SuiteSparse for these operations, the subdomains are processed concurrently, taking advantege of the domain decomposition formulation. The DDM solvers tested are following: Dirichlet-preconditioned FETI-DP (abbreviated as FETI-DP-D), lumped-preconditioned FETI-DP (abbreviated as FETI-DP-L) and the P-FETI-DP. Each of these methods is also executed with the improvements discussed in section 5.4.3 and these versions of the solvers will be abbreviated as FETI-DP-D-I, FETI-DP-L-I and P-FETI-DP-I.

From equations (5.31) and (5.47), it can be observed that unknowns and residual vectors of the iterative PCG algorithm are defined in terms of Lagrange multipliers (FETI-DP) and boundary displacements (P-FETI-DP), respectively. In order to objectively compare the performance and convergence rate of these solvers, a unified residual based convergence criterion is used in the following numerical examples:

$$\frac{||\boldsymbol{K}\boldsymbol{u} - \boldsymbol{f}||}{||\boldsymbol{f}||} \leq \epsilon \tag{6.1}$$

where $\boldsymbol{K}$, $\boldsymbol{u}$ and $\boldsymbol{f}$ are the global stiffness matrix, displacements and external forces of the whole domain, and $\epsilon = 10^{-7}$. In practical applications, a more relaxed tolerance may be used, which will require less computational time for FETI-DP and P-FETI-DP. The objective criterion of equation (6.1) is expensive to apply, because at each PCG iteration the Lagrange multipliers (FETI-DP) or boundary displacements (P-FETI-DP) need to be converted to the global displacements $\boldsymbol{u}$ and the difference $\boldsymbol{K}\boldsymbol{u} - \boldsymbol{f}$ needs to be calculated. In practical applications, the default convergence criterion of PCG is used instead and the global displacements $\boldsymbol{u}$ are calculated once, after PCG has converged. Since equation (6.1) is applied only for the sake of objective comparison, the time required for it is not included in the comparisons presented.

Another solver used for comparison is the incremental Cholesky solver (abbreviated as Direct-I), an reanalysis solver proposed in Pais et al. (2012), specifically for brittle crack propagation problems, which achieved significant performance improvements over the Direct-S solver. This method implements a partial factorization connected to the modified DOFs at each propagation step. Row-add operations are used for the stiffness of new Heaviside and tip enriched DOFs of the current step, while row-delete operations for the tip enriched DOFs of the previous step. Before the analysis starts, the approximate minimum degree (AMD) algorithm (Amestoy et al., 2004) is used to obtain an effective DOF ordering for all standard and possibly enriched DOFs. The aforementioned row-add, row-delete and AMD operations are all implemented by the high performance library SuiteSparse.

Finally, a typical PCG solver with Jacobi preconditioning (abbreviated as PCG-D) is included in the comparison, where the stiffness matrix is stored in compressed sparse rows (CSR) format and Intel math kernel library (MKL) (Intel Corporation, 2022) is used for the execution of matrix-vector multiplications, vector-vector operations and the application of the preconditioner in each PCG iteration. Therefore, this is an very optimized parallel implementation that takes full advantage of the multicore computing system. Similarly to the domain decomposition solvers, the objective criterion of equation (6.1) with a tolerance of $\epsilon = 10^{-7}$ is employed for checking the convergence of this iterative solver and the additional cost of this criterion is subtracted, in order to compare its performance with the other solvers impartially.

6.2 Test case 1: Plate under impact loading

6.2.1 Problem description

The first test case is based on the Kalthoff experiment (Kalthoff & Winkler, 1987). As illustrated in figure 6.1, a steel plate with an initial notch (crack) is impacted by a cylindrical projectile. The projectile is modeled with prescribed displacements $u_0 = 50\ mm$ around the point of impact and an initial crack propagates from the cylindrical notch (Fries & Baydoun, 2012). For low projectile velocities, brittle crack propagation occurs, in a $60 \div 70°$ angle through the domain, indicating Mode II loading. The material properties are $E = 3 \cdot 10^7\ N/mm^2$ and $v = 0.3$. Similarly to Fries and Baydoun (2012) and other studies, the crack is allowed to propagate quasi-statically with a constant increment $da = 5\ mm$, until it reaches the domain boundary, which happens after 16 propagation steps. At each propagation step, XFEM analysis is performed using a uniform mesh of hexahedral elements with 8 nodes. The whole quasi-static crack propagation is repeated for various mesh densities and for each solver. The resulting crack path after 16 propagation steps and the nodes enriched with Heaviside and crack tip functions are depicted in figure 6.2. The number of DOFs varies per propagation step, since new enriched DOFs are introduced as the crack propagates. The initial and final number of DOFs for each mesh are listed in table 6.1.

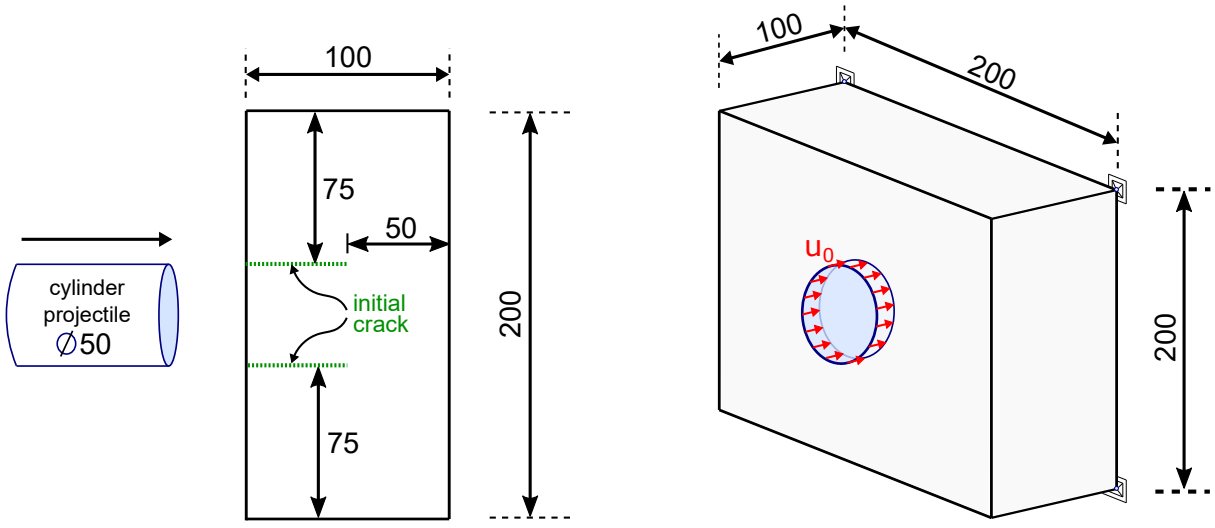
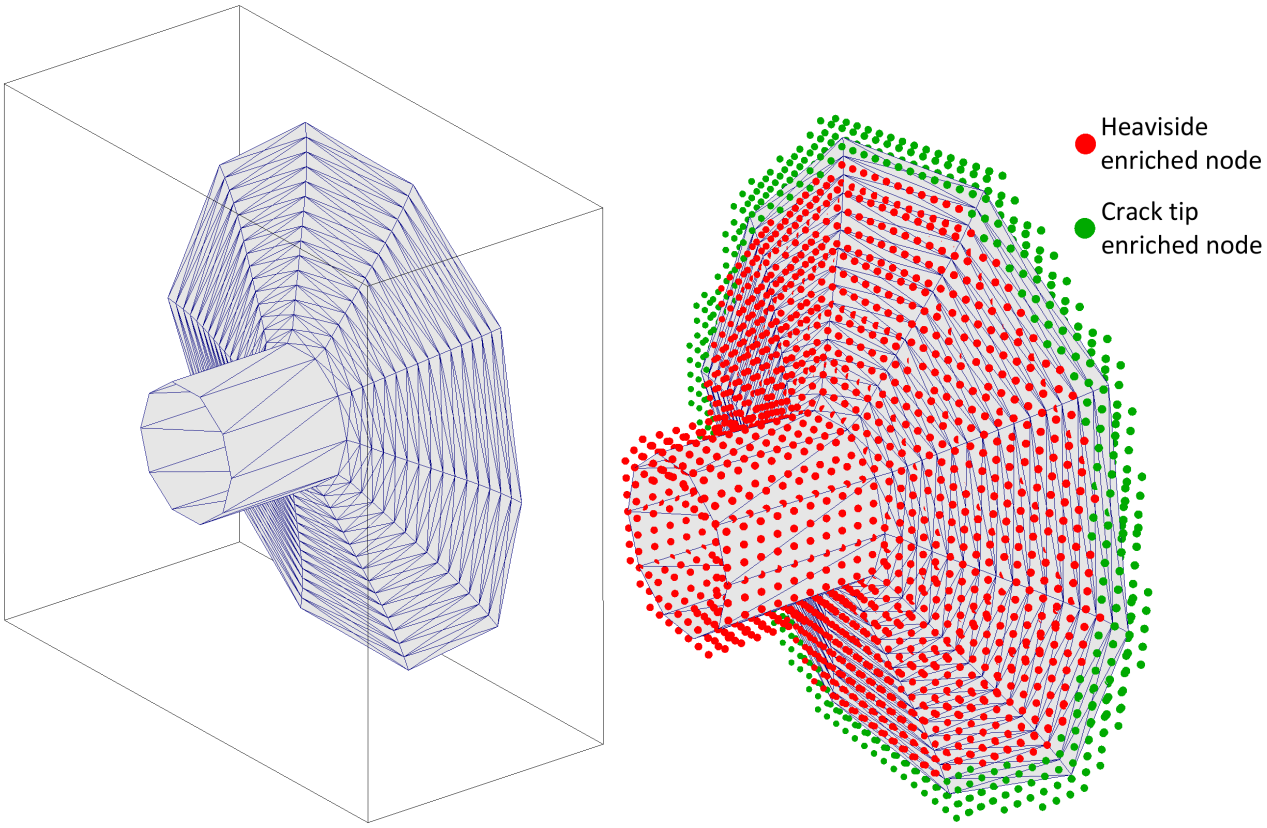


Figure 6.1: Test case 1. Description of the plate under impact example. All dimensions are in mm.



(a) (b)

Figure 6.2: Test case 1. (a) Crack path and (b) enriched nodes, at the last propagation step.

Mesh	DOFs at first step	DOFs at last step	Average
$10 \times 5 \times 10$	2,791	4,357	3,532
$20 \times 10 \times 20$	15,960	20,502	17,983
$30 \times 15 \times 30$	48,507	56,709	52,083
$40 \times 20 \times 40$	109,386	122,052	115,169
$50 \times 25 \times 50$	208,385	226,577	216,456
$60 \times 30 \times 60$	353,784	379,140	364,641
$70 \times 35 \times 70$	554,171	587,357	568,222
$80 \times 40 \times 80$	818,770	860,224	836,817
$90 \times 45 \times 90$	1,157,761	1,208,929	1,179,569

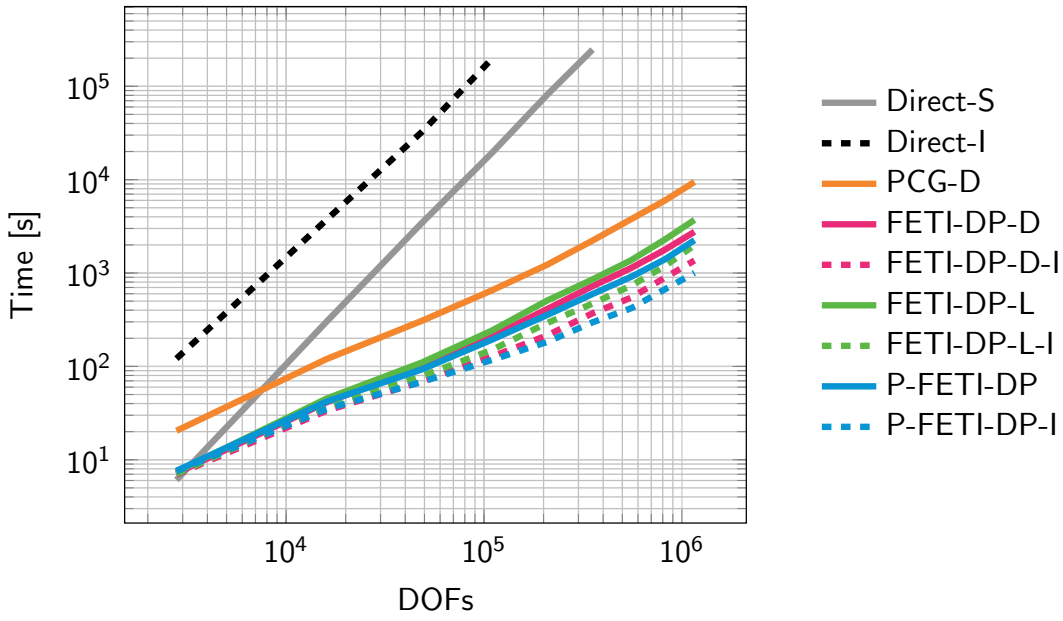
Table 6.1: Test case 1. Number of DOFs per mesh.

6.2.2 Comparison of various solvers

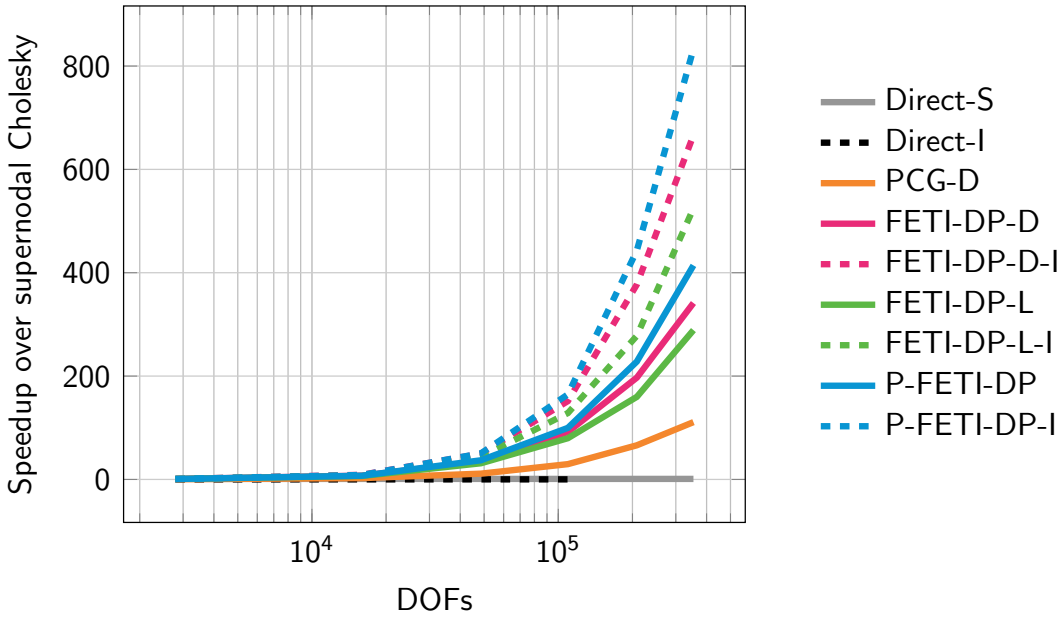
The performances of the supernodal Cholesky solver (Direct-S), the incremental Cholesky solver (Direct-I), the iterative solver PCG-D and the FETI-DP and P-FETI-DP solvers are compared. The total time required by each for the solution phases of the whole analysis can be observed in figure 6.3a and the speedup with respect to the default direct solver Direct-S in figure 6.3b: $speedup = t\,(\text{(Direct-S)}\ /\ t\,(\text{solver})$, where $t(\cdot)$ is the total time required by each solver for the full crack propagation analysis. In both figures, the x-axis corresponds to the initial number of DOFs for the meshes listed in table 6.1. Note that supernodal Cholesky can be used for meshes up to $3.5 \cdot 10^5$ DOFs only, while incremental Cholesky runs out of memory at approximately $1.1 \cdot 10^5$ DOFs.

This increased memory requirement of Direct-D is attributed to the increased bandwidth of the stiffness matrix compared to the matrix of Direct-S, although both solvers use the AMD reordering algorithm to reduce the bandwidth. In the case of Direct-I, many inactive enriched DOFs are included in the set of total DOFs and the effectiveness of AMD decreases, whereas in the Direct-S solver, the AMD algorithm can optimize the DOF ordering. Additionally, Direct-I uses extra temporary memory for implementing the row-add operations. In terms of computation time, incremental Cholesky is slower than the supernodal Cholesky solver in this test case, in contrast to 2D applications, where incremental Cholesky solver provides significant speedups (Pais et al., 2012). This is caused by the increase of crack tip enriched DOFs, since the crack front is more extensive and interacts with a larger part of the domain in 3D problems. Section 6.3 provides a detailed comparison of this case with a 3D application, where the reanalysis features of incremental Cholesky are effective and outperform supernodal Cholesky.

The PCG-D solver is up to 100 times faster than supernodal Cholesky. Although PCG-D requires a very large number of iterations to converge, the computational effort remains reasonably low, due to the simple operations performed by the very optimized linear algebra library (Intel MKL). Nevertheless, the improvement of the FETI-DP and P-FETI-DP solvers is superior, especially for larger problems, even though they are implemented by an unoptimized C# code. The maximum speedups that have been achieved due to the memory limitations of the direct solver are 340 (FETI-DP-D), 290 (FETI-DP-L), 410 (P-FETI-DP), 670 (FETI-DP-D-I), 530 (FETI-DP-L-I) and 830 (P-FETI-DP-I).



(a)



(b)

Figure 6.3: Test case 1. Performance comparison of the solvers for the plate under impact: (a) Computing time (in seconds). (b) Speedup of the solvers relative to supernodal Cholesky solver.

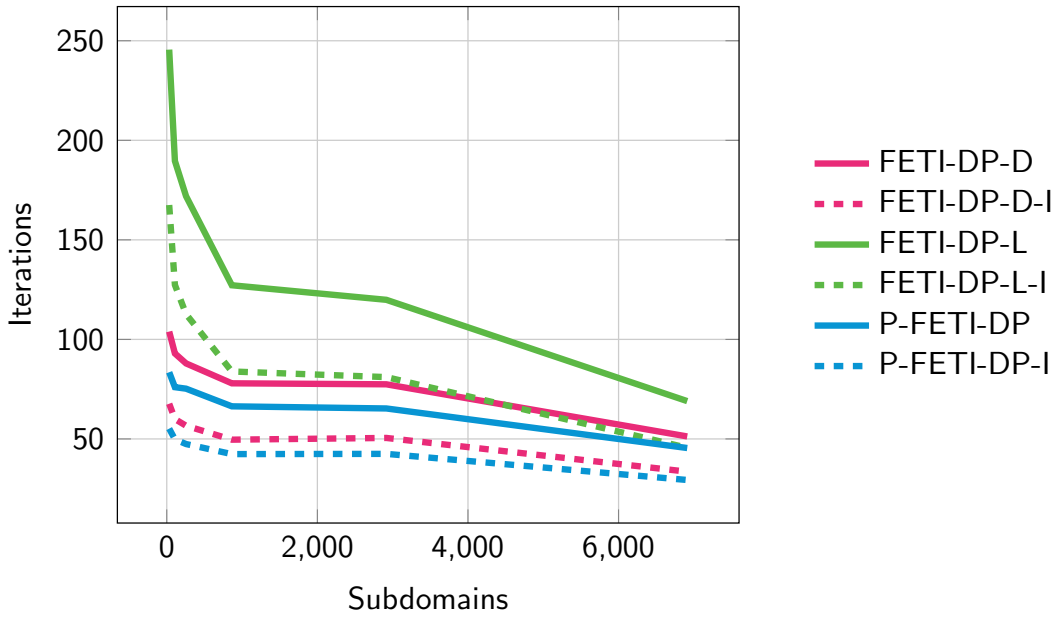
6.2.3 Numerical scalability investigation

The numerical scalability of FETI-DP and P-FETI-DP with respect to the number of subdomains (strong scalability), is depicted in figure 6.4. It can be seen that as the subdomains increase for a constant mesh size, the iterations of PCG required for the solution of the interface problem of FETI-DP and P-FETI-DP decrease, thus both methods scale well with respect to the number of subdomains. The constant mesh used is $(72 \times 36 \times 72)$, corresponding to $616,115$ global DOFs on average. For various partitions of the constant mesh, table 6.2 lists the number of subdomains, boundary DOFs, corner DOFs and Lagrange multipliers (relevant only for FETI-DP). Although the iterations always decrease as the number of subdomains increases, this is not the case for the solution time, where there is an optimum number of subdomains for each solver resulting in the best solution time. Adding more subdomains also increases the number of corner DOFs of FETI-DP and P-FETI-DP, which in turn increases the size and bandwidth of the coarse problem matrix $\boldsymbol{S}_{cc}$. Consequently, after that optimum is reached, the benefit of reduced iterations cannot counterbalance the increased computing cost of the factorization of $\boldsymbol{S}_{cc}$ and the corresponding back/forward substitutions.

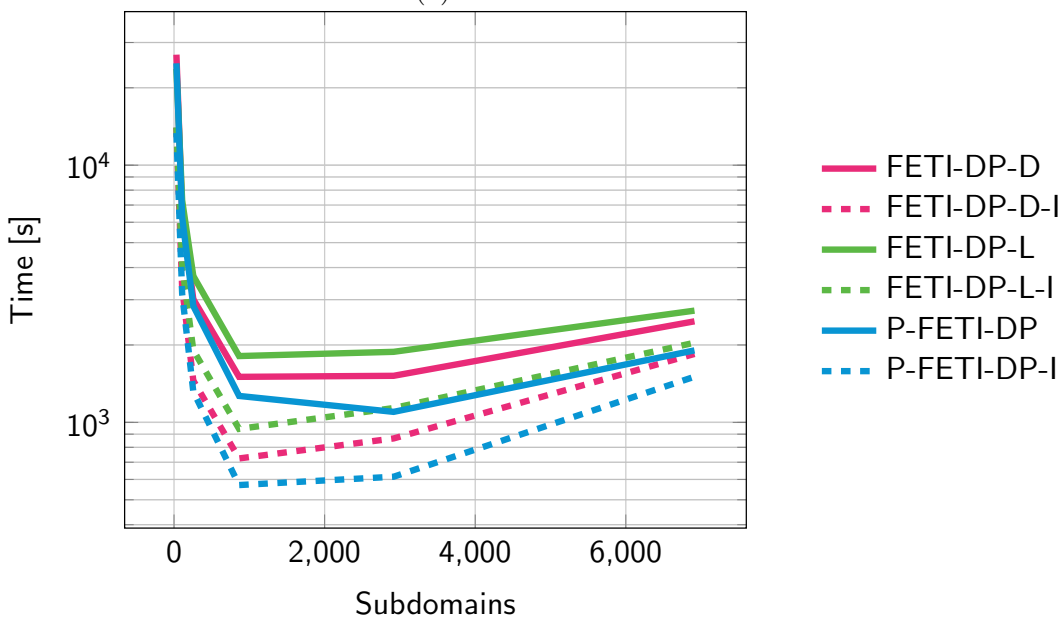
Subdomains	Number of subdomains	Boundary DOFs	Corner DOFs	Lagrange multipliers
$4 \times 2 \times 4$	32	65,735	3,563	72,892
$6 \times 3 \times 6$	108	111,156	5,607	137,734
$8 \times 4 \times 8$	256	154,970	8,780	210,040
$12 \times 6 \times 12$	864	232,434	13,190	372,784
$18 \times 9 \times 18$	2,916	332,774	24,443	645,906
$24 \times 12 \times 24$	6,912	414,101	41,303	927,808

Table 6.2: Test case 1 ($616,115$ global DOFs). Number of subdomains, boundary DOFs, corner DOFs and Lagrange multipliers.

Figure 6.5 illustrates the behaviour of FETI-DP and P-FETI-DP, when the ratio of subdomain to element size remaints constant $H/h = 5$ and the mesh size, along with the number of subdomains, increases. The x-axis contains the initial number of DOFs for each mesh, as listed in table 6.1, while the y-axis corresponds to the iterations required for PCG-D and for the interface problem of FETI-DP and P-FETI-DP. Although the mesh size increases, the number of iterations performed by FETI-DP and P-FETI-DP remains constant, contrary to the PCG-D solver that requires increasingly more iterations to converge. Specifically, when using the zero vector as an initial guess for the interface problems, the number of iterations reaches a plateau at 69 (FETI-DP-D), 117 (FETI-DP-L) and 59 (P-FETI-DP), while when using equations (5.56) and (5.55) the iterations are 44 (FETI-DP-D-I), 77 (FETI-DP-L-I) and 37 (P-FETI-DP-I). This result confirms the numerical scalability of FETI-DP and PFETI-DP with respect to the problem size (weak scalability), a property that is essential



(a)



(b)

Figure 6.4: Test case 1. Scalability analysis with respect to the number of subdomains. Constant mesh ($72 \times 36 \times 72$), variable number of subdomains. (a) Iterations required to solve the interface problem (b) Time (in seconds) required for the solution phase.

for solving arbitrarily large problems with constant iterations, by increasing the number of subdomains and the corresponding processors of a distributed memory system.

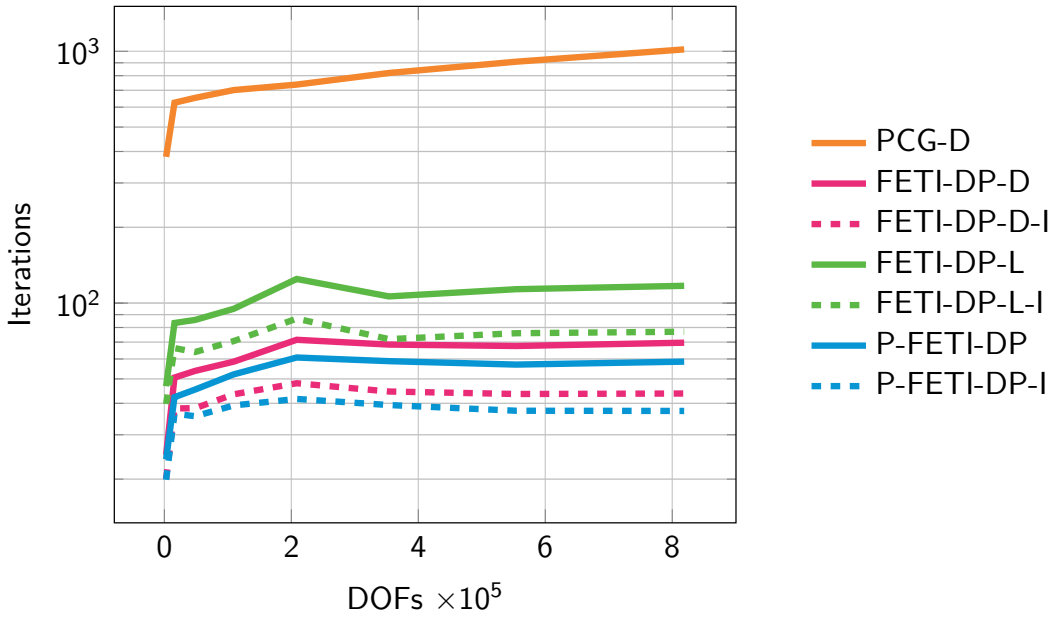


Figure 6.5: Test case 1. Scalability analysis with respect to the problem size: Variable mesh size, but constant ratio of subdomain to element size $H/h = 5$.

As can be seen in Figs 6.3-6.5, when using Eqs (5.56, 5.55) as initial guesses for the solution of the interface problem, both FETI-DP and P-FETI-DP exhibit substantial computational gains. Specifically, the number of iterations is reduced by up to 37% (FETI-DP-D), 36% (FETI-DP-L) and 40% (P-FETI-DP-I). By reusing subdomain data from previous propagation steps and the non-zero initial guesses for the interface problems, the total solution time is reduced by 49% (FETI-DP-D-I), 45% (FETI-DP-L-I) and 50% (P-FETI-DP-I).

6.2.4 Effectiveness of XFEM-specific modifications

Next, the effectiveness of the proposed solvers in eliminating the ill-conditioning caused by XFEM is investigated by comparing two corner DOF schemes. The first scheme requires the minimum modification to the original FETI-DP and P-FETI-DP for avoiding singular $\boldsymbol{K}_{rr}^{s}$ matrices, by using equation (5.52) to treat boundary DOFs that are enriched with Heaviside and only the first crack tip function as corner DOFs. The second scheme, which in fact is used in all previous investigations, uses equation (5.53) to treat boundary DOFs that are enriched with Heaviside and all four crack tip functions as corner DOFs. Figure 6.6 depicts the iterations required for solving the interface problems of FETI-DP and P-FETI-DP, when each of these schemes are used. Four sample meshes are shown and the x-axis

corresponds to crack propagation steps. The ratio of subdomain to element size is kept constant at $H/h = 5$ and the initial guesses for the interface problems are taken equal to $\mathbf{0}$. When using equation (5.52), sharp increases in the number of iterations are observed during some propagation steps. This XFEM-related ill-conditioning occurs only when boundary DOFs become enriched due to their proximity to the crack front, which happens at different propagation steps for each mesh. Nevertheless, when it does happen, an iteration increase of up to 245% can be observed. On the other hand, iteration spikes during those crack propagation steps are not observed when using equation (5.53), which ensures that XFEM ill-conditioning is completely eliminated with this version of FETI-DP and P-FETI-DP.

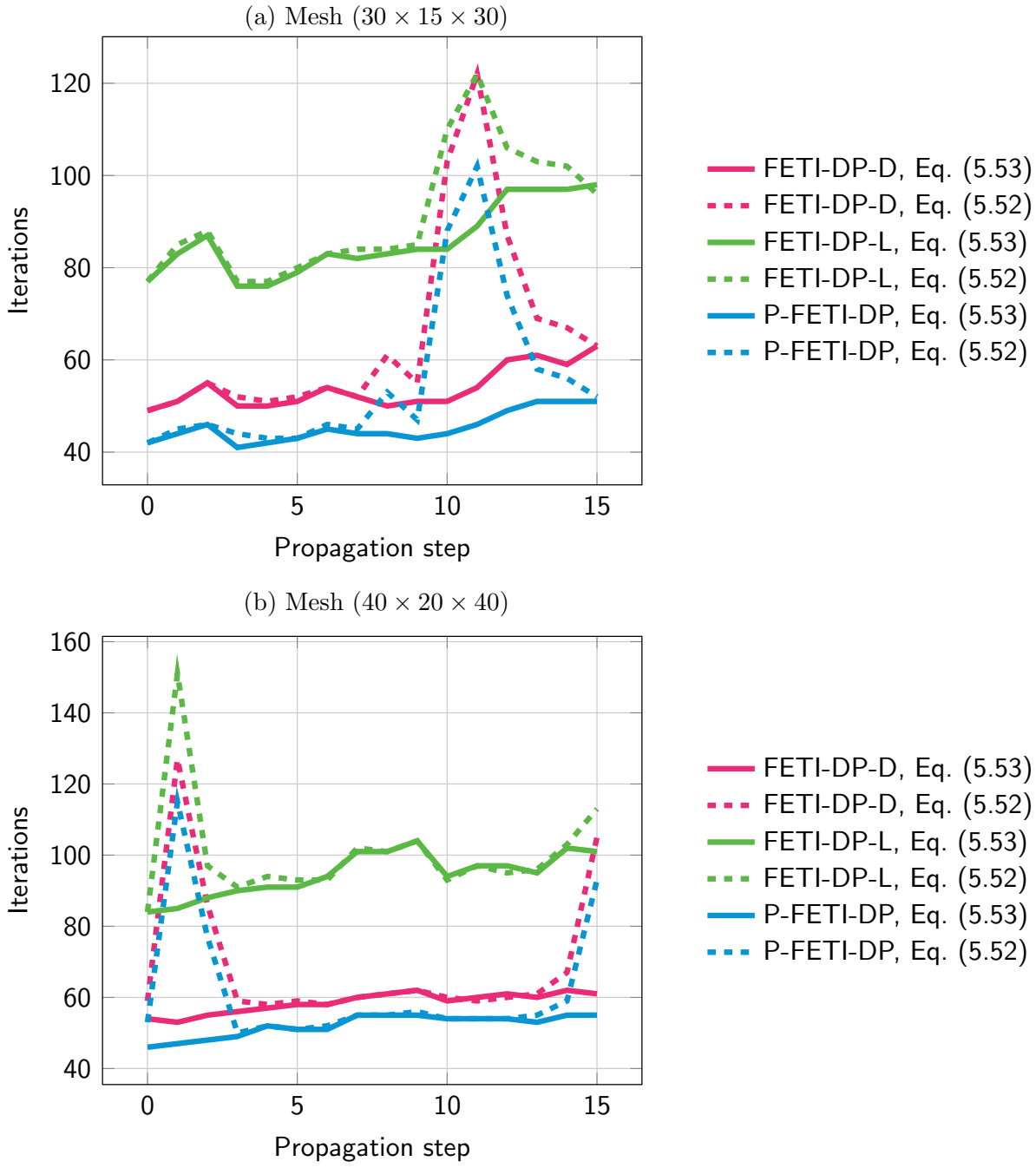


Figure 6.6: Test case 1. Iterations required for solving the interface problem at each propagation step for different mesh sizes, when using equations (5.52) or (5.53) to define the corner DOFs.

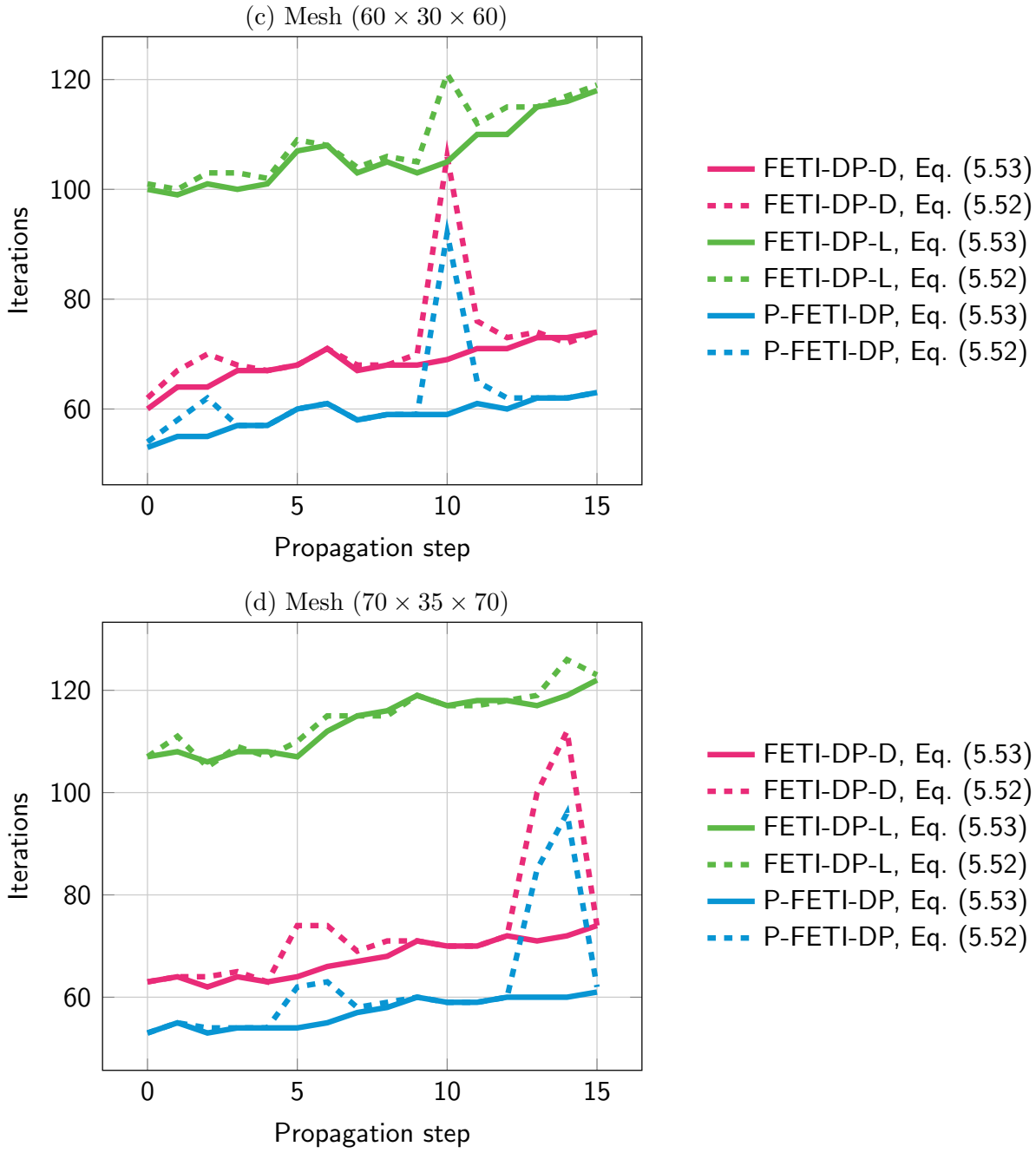


Figure 6.6: Test case 1. Iterations required for solving the interface problem at each propagation step for different mesh sizes, when using equations (5.52) or (5.53) to define the corner DOFs.

6.2.5 Parallel scalability investigation

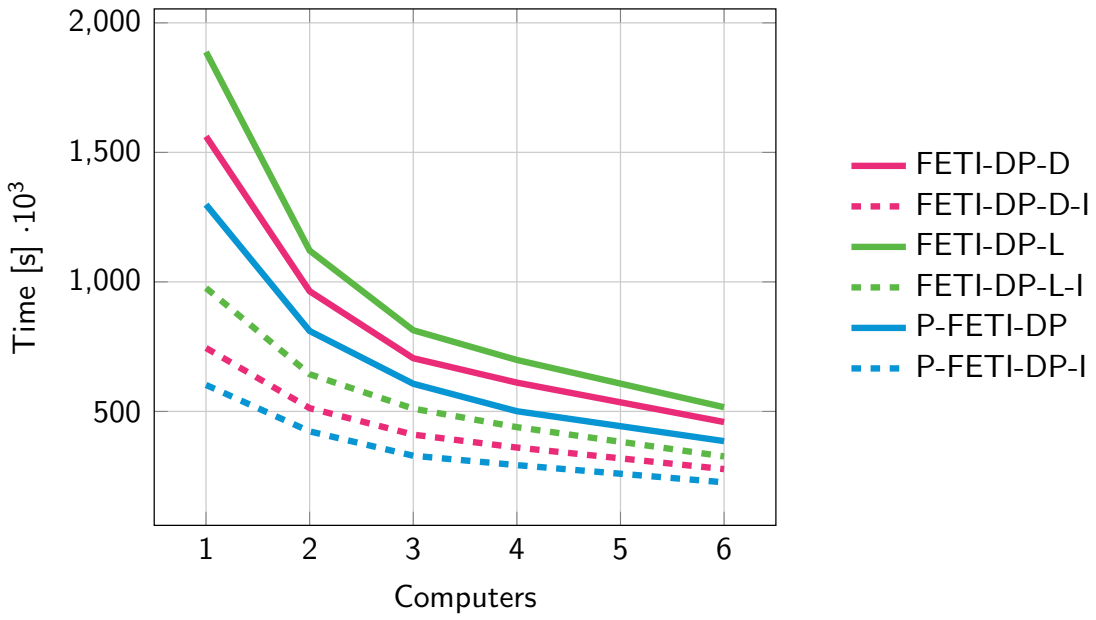
This section investigates the performance of the proposed DDM solvers when executed on the HPC system described in section 6.1. A mesh with $(72 \times 36 \times 72)$ elements is partitioned into $(12 \times 6 \times 12)$ subdomains (864 in total). The number of subdomain DOFs, global DOFs, boundary DOFs (size of P-FETI-DP's interface problem), corner DOFs (size of coarse problem) and Lagrange multipliers (size of FETI-DP's interface problem) is listed in table 6.3. Since new enriched DOFs are introduced at each crack propagation step, the minimum, maximum and average values are shown. This problem is then solved by each solver on a cluster computing environment, which consists of 1,2,4 or 6 computers, with the specifications given in section 6.1. The 864 subdomains are evenly divided among the available computers in each case, to ensure balanced loads. Furthermore, the communication needed between the computers is minimized, by minimizing the boundary DOFs that belong to subdomains allocated to different computers and maximizing the boundary DOFs that belong to subdomains allocated to the same machine. The solution of the coarse problem of FETI-DP and P-FETI-DP is performed according to section 5.5.2.2.3

	Min	**Max**	**Average**
Subdomain DOFs	1,026	1,773	1,074
Global DOFs	601,050	635,400	616,115
Boundary DOFs	226,690	238,816	232,434
Corner DOFs	7,446	19,572	13,190
Lagrange multipliers	372,784	372,784	372,784

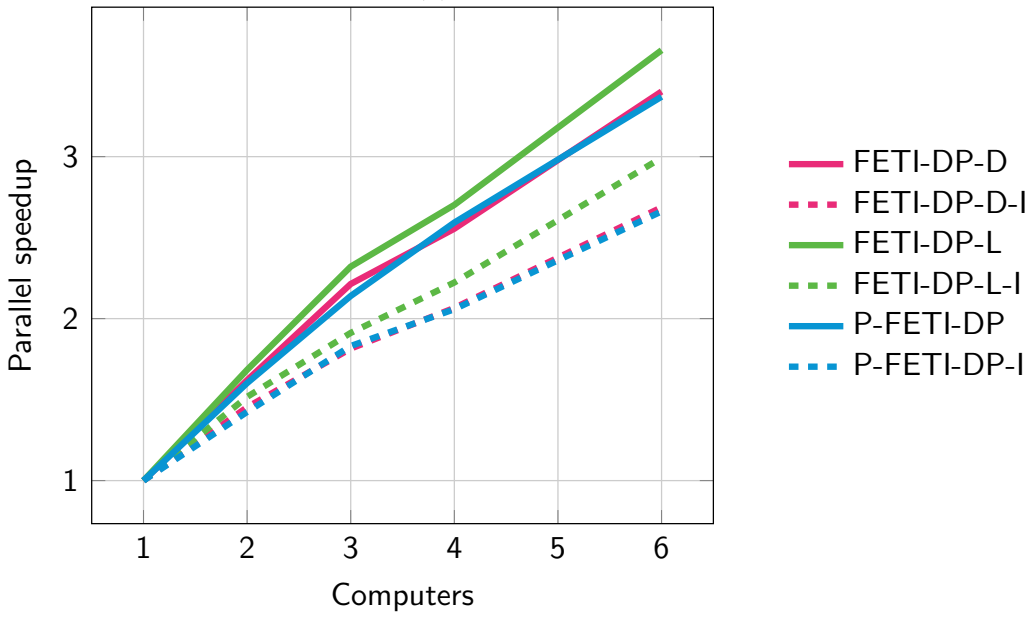
Table 6.3: Test case 1. Number of DOFs for the case of $(72 \times 36 \times 72)$ elements and $(12 \times 6 \times 12)$ subdomains.

Fig. 6.7a depicts the time required for the solution phase of the XFEM analysis by each of the solvers. As previously, P-FETI-DP is faster than FETI-DP-D and FETI-DP-L and the improved versions P-FETI-DP-I, FETI-DP-D-I, FETI-DP-L-I are even faster. However, as more resources (computers) are added, the relative performance differences between the 6 solvers become less pronounced. The parallel speedup, namely $speedup = \dfrac{t(1 \text{ computer})}{t(\text{n computers})}$ is shown in Fig. 6.7b. It can be observed that all solvers scale well, when more computers are added to the system.

The main obstacle in the efficient implementation of these DDM solvers for HPC systems is the amount of data that need to be transferred between different computers. As section 5.5 elaborates, this communication has been minimized in the proposed P-FETI-DP solver by replacing the map-reduce operations on vectors involving multiple subdomains $\boldsymbol{y}_b = \sum_{s=1}^{n_s} (\boldsymbol{L}_b^s)^T \hat{\boldsymbol{y}}_b^s$ with the more distributed version $\boldsymbol{y}_b^e = \boldsymbol{M}_b^e \hat{\boldsymbol{y}}_b^e$, as explained in equation (5.70).



(a)



(b)

Figure 6.7: Test case 1. Parallel scalability analysis. Constant elements ($72 \times 36 \times 72$) and subdomains ($12 \times 6 \times 12$), variable number of computers. (a) Time (in seconds) required for the solution phase (b) Parallel speedup.

Similarly, $\boldsymbol{\delta}^e = \boldsymbol{M}_\lambda^e \hat{\boldsymbol{\delta}}^e$ is used in the proposed FETI-DP solver, as explained in equation (5.100). These map reduce operations occur:

- Once per iteration of the PCG used to solve the interface-problem of P-FETI-DP, when the interface-problem matrix is multiplied with a vector corresponding to displacements at boundary DOFs. See equation (5.73).
- Once per iteration of the PCG used to solve the interface-problem of P-FETI-DP, when the preconditioner is multiplied with a residual vector corresponding to forces at boundary DOFs. See equation (5.95).
- Once per iteration of the PCG used to solve the interface-problem of FETI-DP, when the interface-problem matrix is multiplied with a vector corresponding to Lagrange multiplies. See equation (5.111).
- Once per iteration of the PCG used to solve the interface-problem of FETI-DP, when the preconditioner is multiplied with a residual vector corresponding to displacement quantities along the Lagrange multipliers. See equation (5.109).

In order to estimate the magnitude of these communications, Fig. 6.8 presents the data transferred in MB. Data transfers between subdomains that are assigned to the same computer are denoted as "local" and have negligible cost in the current implementation, since they are performed by accessing the shared memory of that computer. In contrast, "remote" data transfers, namely transfers between subdomains assigned to different computers, go through the network connecting these computers. Therefore, they are much slower and can easily become a computational bottleneck if their size, frequency or distribution in the network topology are not optimized. As illustrated in Fig. 6.8, the remote data transfers are significantly lower than the local ones in this application. Additionally, FETI-DP requires transferring less data than P-FETI-DP, which can be attributed to the removal of enriched DOFs from the interface-problem of FETI-DP (see section 5.4.2), but not from the interface problem of P-FETI-DP.

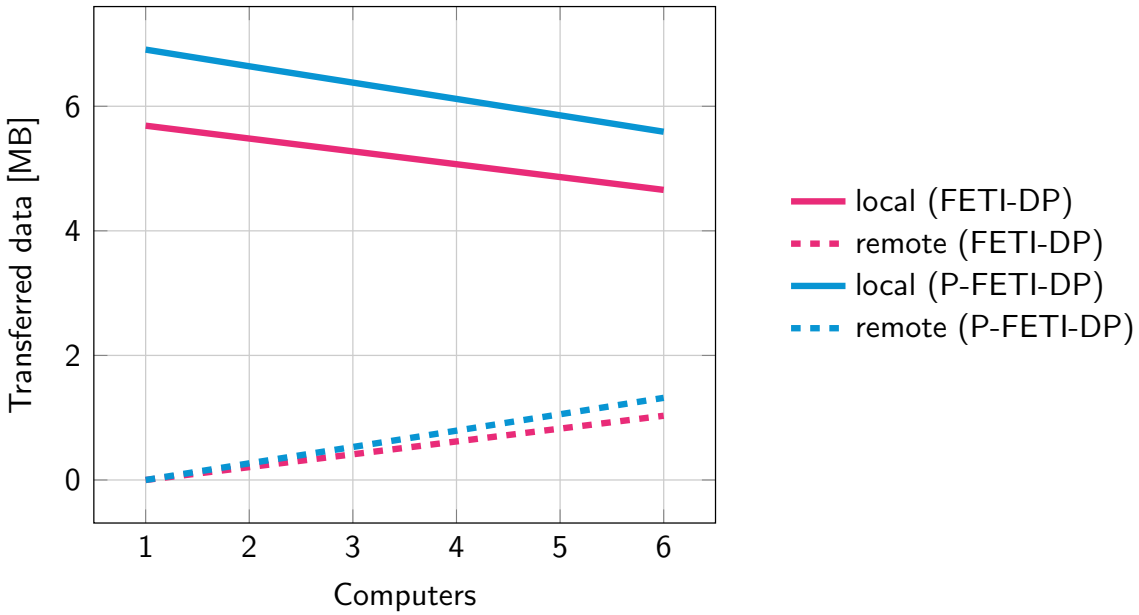
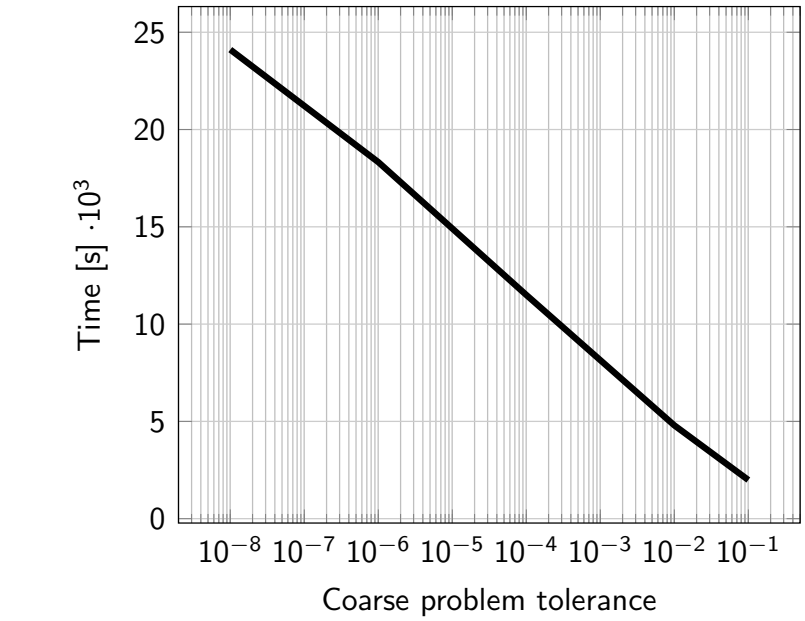


Figure 6.8: Test case 1. Data transfers per application of equation (5.70) (P-FETI-DP) or (5.100) (FETI-DP). Local data transfers happen between subdomains on the same computer, while remote data transfers between different computers.

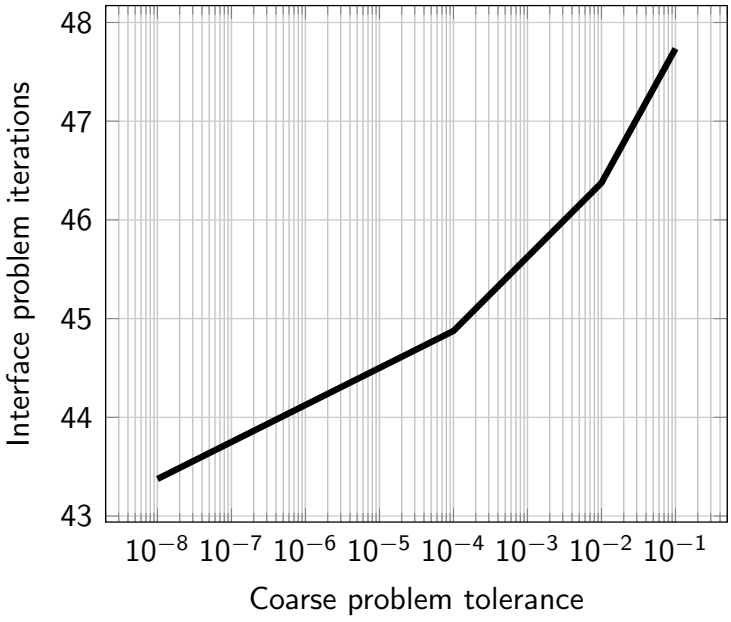
6.2.6 Coarse problem solution strategies

Moreover, three different strategies for solving the coarse problem of P-FETI-DP and P-FETI-DP-I are considered. The first strategy solves the global version of the coarse problem on 1 computer using a direct method and broadcasts the result to all other computers, as explained in section 5.5.2.2.2. This requires extra memory in 1 computer and data transfers after the solution of the coarse problem. The second strategy solves the global version of the coarse problem on all computers using a direct method, as explained in section 5.5.2.2.3. It requires extra memory from all computers, but leads to better load balancing. The third strategy solves the distributed version of coarse problem on all computers using an iterative method (PCG), as explained in section 5.5.2.2.1. This approach has minimum memory requirements but is usually slower, since direct methods are better suited for multiple linear systems with the same matrix but different RHS vectors.

Fig. 6.9a depicts the total time required for the solution phase of P-FETI-DP-I when the coarse problem is solved using PCG (the third strategy) with various convergence tolerances. A mesh of $(72 \times 36 \times 72)$ elements (616,115 global DOFs on average) and $(12 \times 6 \times 12)$ subdomains (13,190 corner DOFs on average) is used. For the same case, Fig. 6.9b shows the average iterations required by the interface-problem PCG to converge to a tolerance of $1E-7$. It can be observed that relaxing the coarse-problem PCG tolerance, slightly decreases the convergence rate of the interface-problem PCG, but the overall computing time



(a)



(b)

Figure 6.9: Test case 1. Coarse problem of P-FETI-DP-I is solved with PCG. a) Solution time for various tolerances of the coarse-problem PCG and b) iterations of the interface-problem PCG.

is significantly improved. However, increasing the coarse-problem PCG tolerance beyond $1E-1$ will cause the interface-problem PCG to not converge at all.

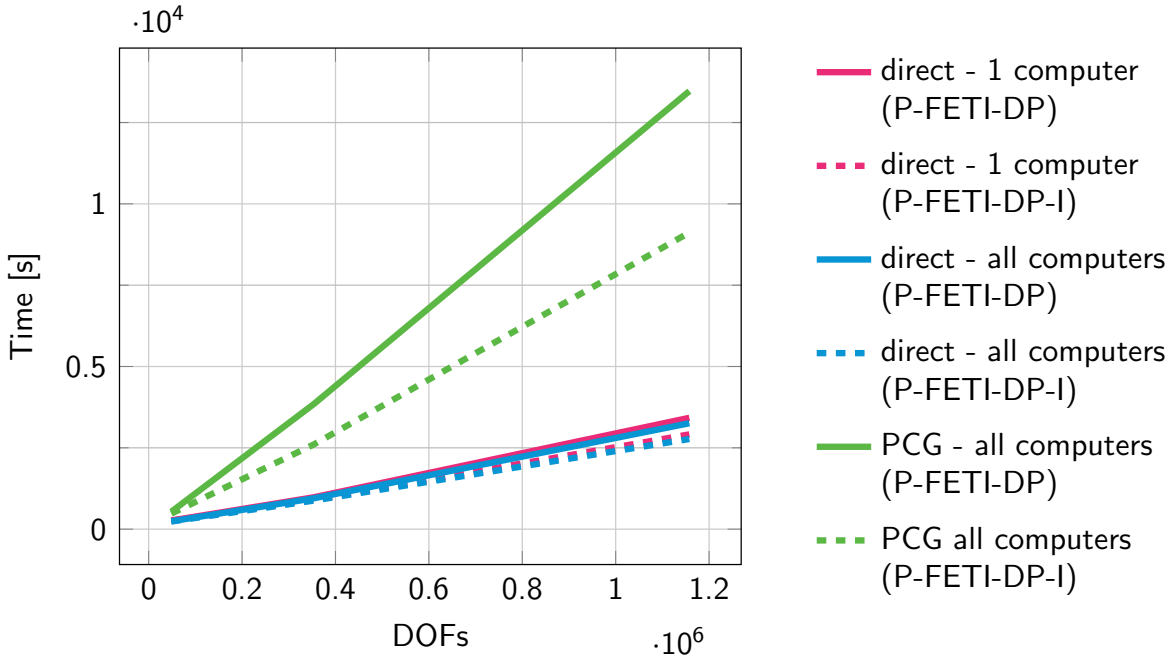


Figure 6.10: Test case 1. Solution time of P-FETI-DP with different coarse problem solution strategies.

Finally, the performances of P-FETI-DP and P-FETI-DP-I are compared, when using each of the three coarse-problem solution strategies. A subdomain-element size ratio of $H/h = 5$ and some of the meshes of table 6.1 are used. The third coarse-problem solution strategy, a tolerance of $1E-1$ is used for the coarse-problem PCG, since it was shown to be optimal for this problem in Fig. 6.9a. The total time required for the solution phase of the XFEM analysis is illustrated in Fig. 6.10. It can be observed that the difference between the first two strategies, which use a direct method, is very small, while the third strategy, which uses an iterative method, is considerably slower, which counter-balances the advantage of reduced memory requirements.

6.3 Test case 2: 4-point bending beam

6.3.1 Problem description

The second test case involves a crack propagating in a beam supported at three points and loaded at a fourth point, as illustrated in figure 6.11. The material properties are $E = 3 \cdot 10^7\ N/mm^2$, $v = 0.3$ and the applied load is $F = 1000\ N$. The dimensions of

the beam, the placement of supports and load and the initial configuration of the crack surface are shown in figure 6.11. Similarly to the first test case, the crack propagates in a quasi-static manner with a constant increment $da = 8\ mm$, until it reaches the boundary of the domain and collapse occurs after 13 propagation steps. At each propagation step, XFEM analysis is performed using a uniform mesh of hexahedral elements with 8 nodes. The whole quasi-static crack propagation is repeated for various mesh densities and for each solver. The resulting crack path after 13 propagation steps is depicted in figure 6.12a, while the nodes enriched with Heaviside and crack tip functions can be observed in figure 6.12b. The number of DOFs varies per propagation step, since new enriched DOFs are introduced as the crack propagates. Therefore, the initial and final number of DOFs for each mesh are listed in table 6.4.

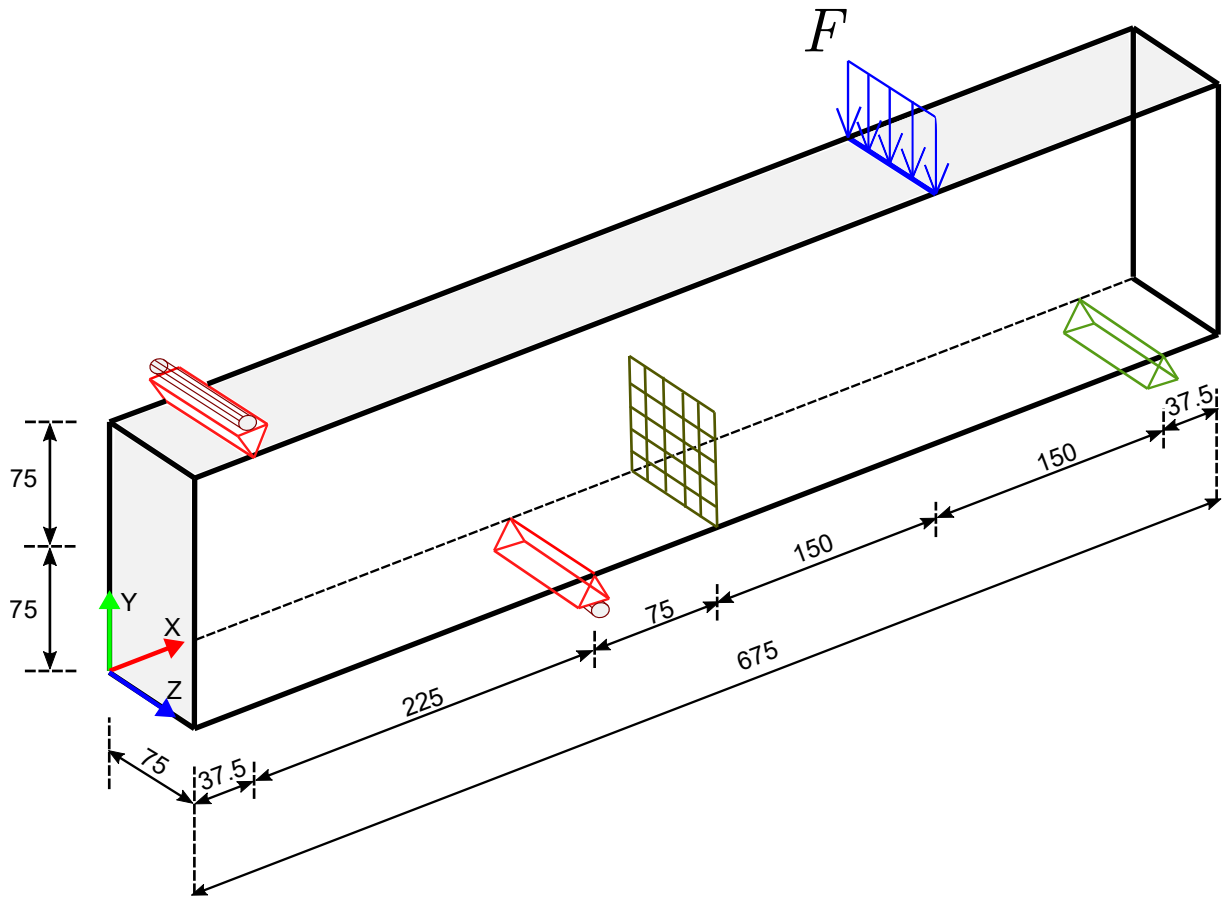
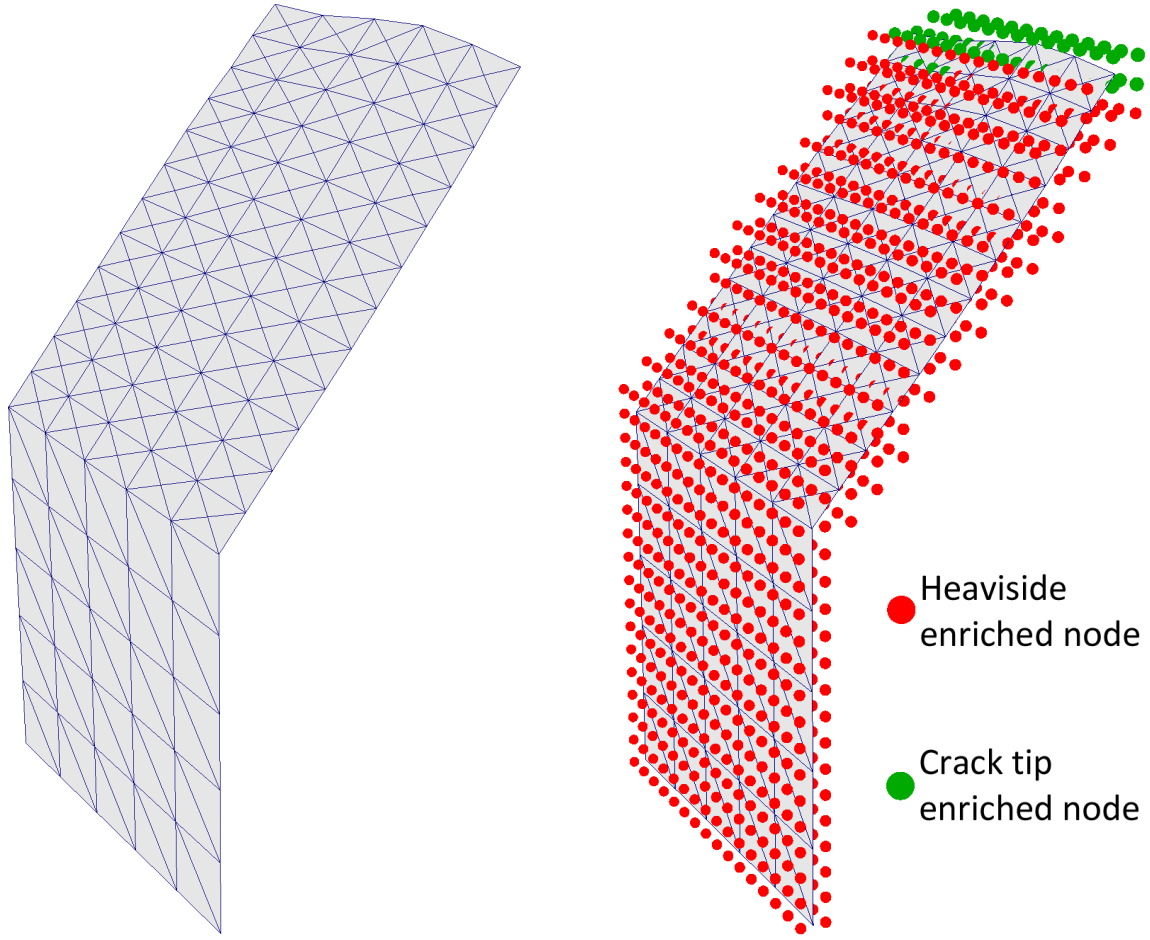


Figure 6.11: Test case 2. 4-point bending beam test case. Geometry, boundary conditions and initial configuration of the crack surface (dimensions in mm).



(a) (b)

Figure 6.12: Test case 2. (a) Crack path and (b) enriched nodes, at the last propagation step. The crack starts propagating from x = 337.5 at step = 0.

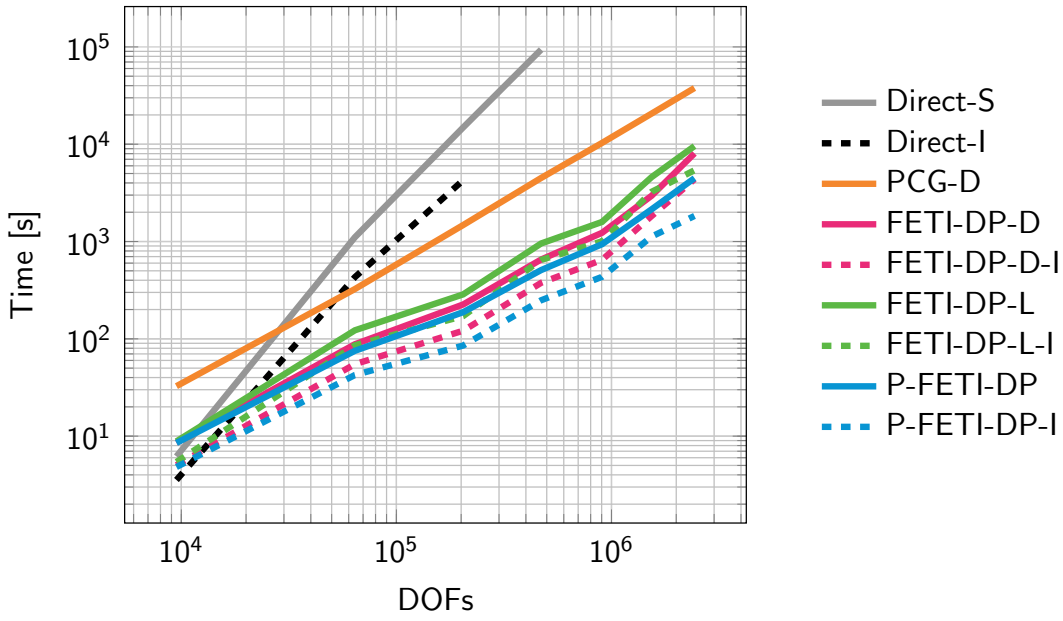
Mesh	DOFs at first step	DOFs at last step	Average
$45 \times 10 \times 5$	9,492	9,816	9,666
$90 \times 20 \times 10$	64,130	65,384	64,775
$135 \times 30 \times 15$	204,336	206,880	205,617
$180 \times 40 \times 20$	470,820	475,419	473,093
$225 \times 50 \times 25$	903,812	910,832	907,286
$270 \times 60 \times 30$	1,537,228	1,548,295	1,543,366
$315 \times 70 \times 35$	2,431,872	2,444,940	2,438,435

Table 6.4: Test case 2. Number of DOFs per mesh.

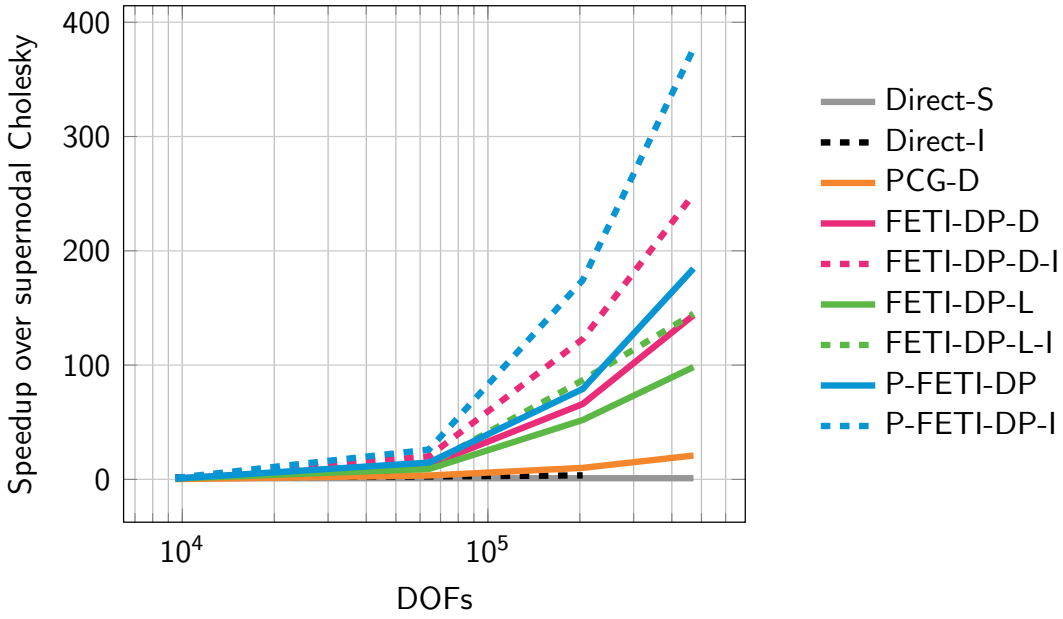
6.3.2 Comparison of various solvers

The total time required by each solver for the solution phases of the whole analysis can be seen in figure 6.13a and the corresponding speedup ($speedup = t\,(\text{Direct-S}) \;/\; t\,(\text{solver})$) of the solvers with respect to the default direct solver (Direct-S) in figure 6.13b. In both figures, the x-axis corresponds to the initial number of DOFs for the meshes listed in table 6.4. The supernodal Cholesky solver can be used for meshes up to $4.73 \cdot 10^5$ DOFs only, while incremental Cholesky runs out of memory even sooner, at approximately $2.5 \cdot 10^5$ DOFs. As discussed in the previous test case, the presence of enriched DOFs, which are active in a few crack propagation steps and inactive in the rest, deteriorates the effectiveness of the AMD reordering, thus increasing the bandwidth of the stiffness matrix and the memory requirements in the incremental Cholesky solver. However, contrary to the previous test case, incremental Cholesky offers a performance improvement over the default supernodal Cholesky solver, which increases as the mesh is refined, similarly to the results in 2D problems reported in Pais et al. (2012).

Nevertheless, the improvement of FETI-DP and P-FETI-DP solvers is superior, especially for larger problems. In the finest mesh where comparison with the Direct-I solver is possible, the speedups are 66 (FETI-DP-D), 52 (FETI-DP-L), 79 (P-FETI-DP), 117 (FETI-DP-D-I), 83 (FETI-DP-L-I) and 171 (P-FETI-DP-I), compared to 3.5 (Direct-I). As the number of DOFs increases, the speedup over the supernodal Cholesky becomes even greater. The maximum speedups that are recorded, due to the memory limitations of Direct-S, are 144 (FETI-DP-D), 98 (FETI-DP-L), 184 (P-FETI-DP), 249 (FETI-DP-D-I), 145 (FETI-DP-L-I) and 377 (P-FETI-DP-I). The maximum reported speedup of PCG-D is 20 at the finest mesh investigated, which is 21 times slower than P-FETI-DP-I. The memory requirements of FETI-DP, P-FETI-DP and PCG solvers are also far lower than the supernodal Cholesky solver, and can solve problems with more than $2.5 \cdot 10^6$ DOFs, even with one processor.



(a)



(b)

Figure 6.13: Test case 2. Performance comparison of the solvers for the 4-point bending beam: (a) Computing time (in seconds). (b) Speedup of the solvers relative to supernodal Cholesky solver.

6.3.3 Comparison of reanalysis features

Next, the effectiveness of the reanalysis techniques of the proposed DDM solvers and the incremental Cholesky solver of Pais et al. (2012) are compared. By examining figure 6.2b and figure 6.12b, it can be observed that the crack front is more extensive in the first test case, resulting in an increased number of crack tip enriched DOFs. Therefore, between two successive crack propagation steps, substantially more DOFs are modified in the first test case. Figure 6.14 illustrates the percentage of total DOFs that are modified between two crack propagation steps, because enriched DOFs are added or removed. The x-axis corresponds to the initial number of total DOFs for the meshes listed in table 6.4. When the incremental Cholesky solver updates the factorized stiffness matrix, the number of rows that need to be add or deleted, because they correspond to these modified DOFs, is higher in the first test case than in the second one. This results in worse performance than factorizing the whole matrix, while in the second test case, incremental Cholesky outperforms the standard supernodal Cholesky solver, due to a very low percentage of modified DOFs.

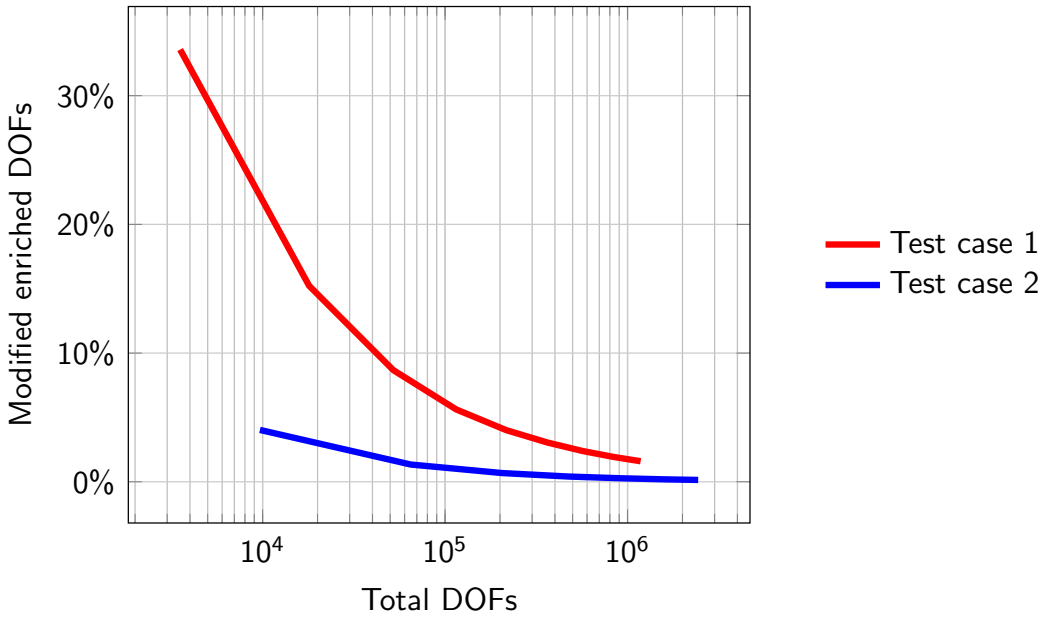


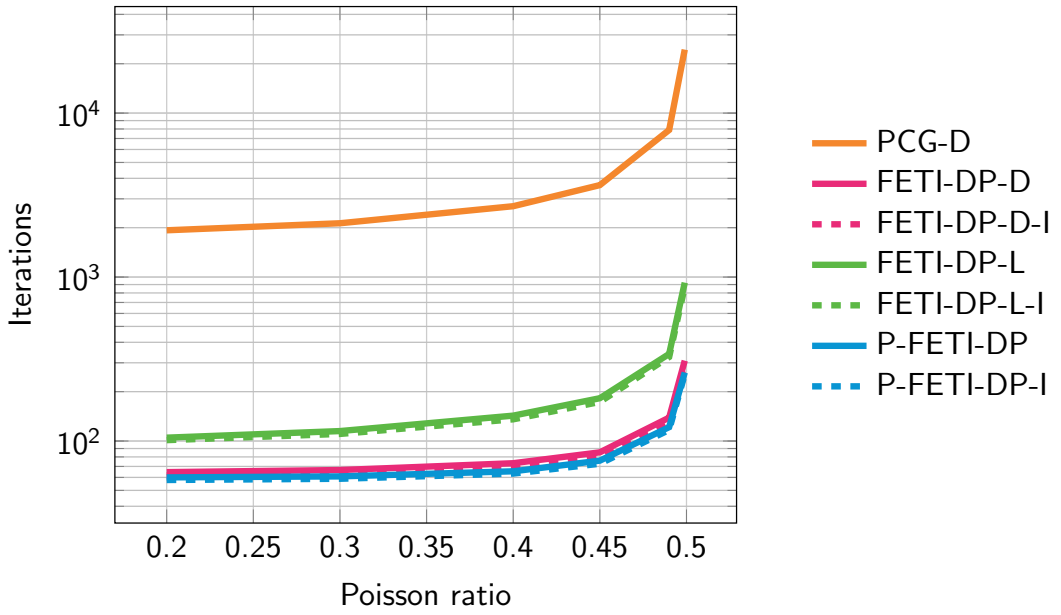
Figure 6.14: Percentage of enriched DOFs, newly added or deleted between propagation steps, to total DOFs.

In contrast, the effectiveness of reusing data from previous propagation steps in the proposed FETI-DP and P-FETI-DP solvers does not depend on a low percentage of modified DOFs. In both numberical examples, the FETI-DP-D-I, FETI-DP-L-I and P-FETI-DP-I versions are roughly 2 times faster than FETI-DP-D, FETI-DP-L and P-FETI-DP, respectively. The reanalysis technique involves identifying and reusing previous subdomain-level matrices, thus avoiding repeated operations, as well as reducing the iterations needed for

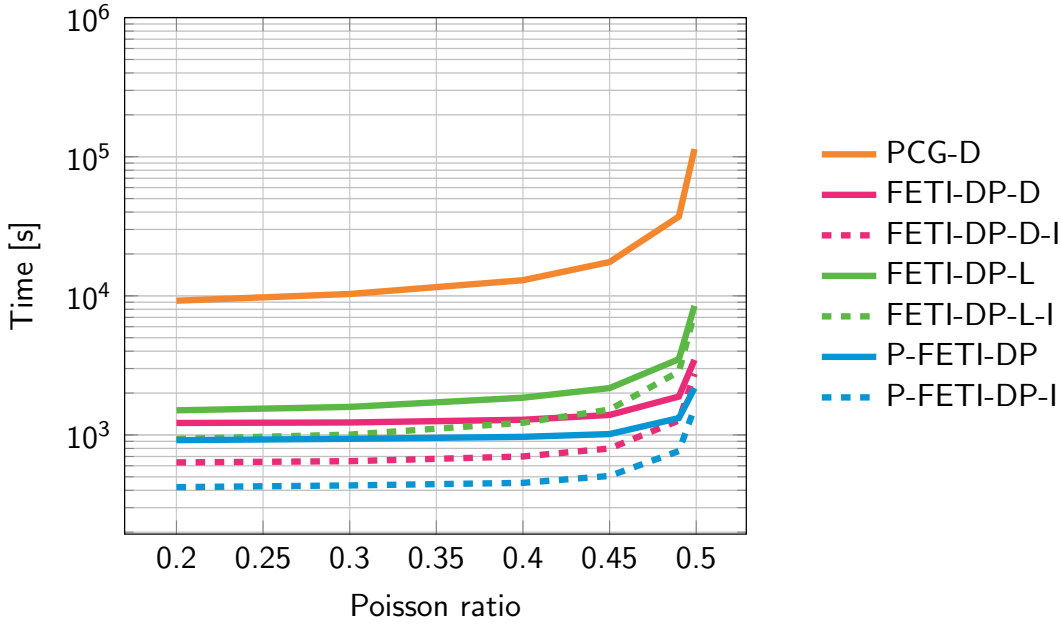
the interface problem, by using its solution during the previous propagation step to start the iterations with a better initial guess. These operations have no additional cost as opposed to the row-add and row-delete operations of the incremental Cholesky solver, which end up being more time consuming than factorizing the whole matrix in the test case 1. Additionally, in the incremental Cholesky solver, the AMD reordering algorithm is applied only at the first propagation step and needs to account for all possibly enriched DOFs, thus deteriorating the quality of the DOF ordering and increasing the stiffness matrix bandwidth. On the other hand, in the proposed solvers, AMD is applied at each propagation step and can optimize the DOF ordering for each step independently from the previous ones.

6.3.4 Ill-conditioning caused by high Poisson ratio

In the following, the sensitivity of the proposed solvers to the Poisson ratio v is investigated. Materials with increased Poisson ratio result in very ill-conditioned linear systems. While this does not affect dramatically the performance of direct solvers, apart from their accuracy in extreme cases, iterative and domain decomposition solvers exhibit substantial sensitivity in their convergence rate. Figure 6.15 illustrates the performance of PCG-D and the proposed domain decomposition solvers for various values of the Poisson ratio v. The $(225 \times 50 \times 25)$ mesh was used with $903,812 \div 910,832$ DOFs. In figure 6.15a the number of iterations are plotted for each case. It is evident that the proposed FETI-DP and especially P-FETI-DP solvers are less sensitive to the ellipticity of problem at hand than the PCG solver. For the highest Poisson ratio v = 0.499, the P-FETI-DP-I solver is up to 73 times faster than PCG-D and 1.8 times faster then FETI-DP-D-I.



(a)

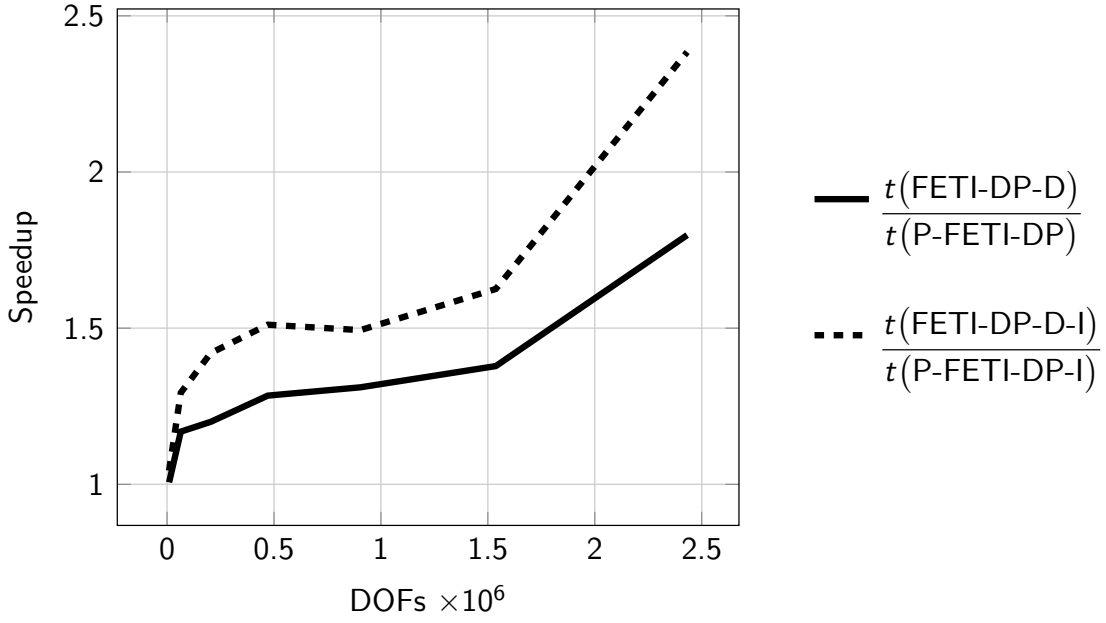


(b)

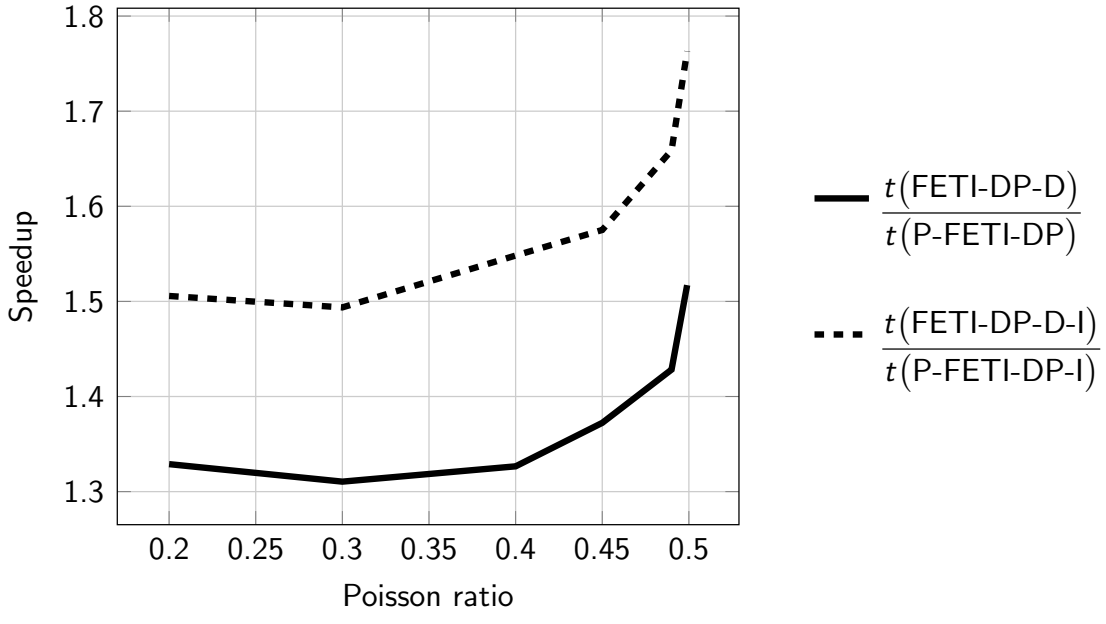
Figure 6.15: Test case 2. Convergence rate vs Poisson ratio. (a) Required iterations. (b) Time (in seconds) required for the solution phase of the analysis.

6.3.5 FETI-DP vs P-FETI-DP

Finally, the proposed DDM solvers are compared to each other in more detail, with respect to their convergence rate and overall computing time. In all examples, FETI-DP-L is significantly less efficient than FETI-DP-D and P-FETI-DP. The reduced operations for the lumped preconditioner described in equation (5.40) cannot counter-balance the increase in iterations caused by ignoring the stiffness of internal DOFs $\boldsymbol{K}_{ii}^{s}$. Furthermore, Dirichlet-preconditioned FETI-DP is outperformed by P-FETI-DP in all tests performed. Although the amount of work per iteration is the same, P-FETI-DP always requires less iterations. Figures (6.16a),(6.16b) show the speedup of P-FETI-DP with respect to Dirichlet-preconditioned FETI-DP, namely $speedup = t\,(\text{FETI-DP-D}) \;/\; t\,(\text{P-FETI-DP})$. Figure (6.16a) clarifies that P-FETI-DP is up to 2.4 times faster than Dirichlet-preconditioned FETI-DP and this ratio tends to progressively increase as the mesh is refined. The same trend is observed in figure (6.16b), where P-FETI-DP becomes progressively more efficient than Dirichlet-preconditioned FETI-DP for higher values of the Poisson ratio.



(a)



(b)

Figure 6.16: Test case 2. Speedup of P-FETI-DP solver over Dirichlet-preconditioned FETI-DP solver. (a) For various meshes and $v = 0.3$. (b) For mesh $225 \times 50 \times 25$ and various poisson ratios.

6.3.6 Parallel scalability investigation

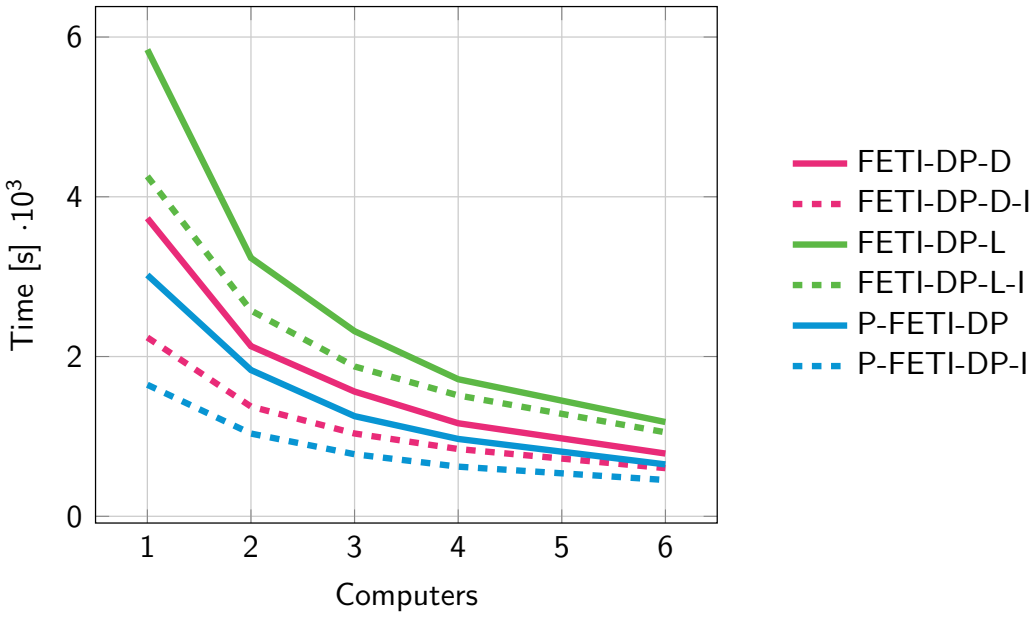
This section investigates the performance of the proposed DDM solvers when executed on the HPC system described in section 6.1. A mesh with $(252 \times 56 \times 28)$ elements is partitioned into $(36 \times 8 \times 4)$ subdomains (1152 in total). The number of subdomain DOFs, global DOFs, boundary DOFs (size of P-FETI-DP's interface problem), corner DOFs (size of coarse problem) and Lagrange multipliers (size of FETI-DP's interface problem) is listed in table 6.3. Since new enriched DOFs are introduced at each crack propagation step, the minimum, maximum and average values are shown. This problem is then solved by each solver on a cluster computing environment, which consists of 1,2,4 or 6 computers, with the specifications given in section 6.1. The 1152 subdomains are evenly divided among the available computers in each case, to ensure balanced loads. Furthermore, the communication needed between the computers is minimized, by minimizing the boundary DOFs that belong to subdomains allocated to different computers and maximizing the boundary DOFs that belong to subdomains allocated to the same machine. The solution of the coarse problem of FETI-DP and P-FETI-DP is performed according to section 5.5.2.2.3

	Min	**Max**	**Average**
Subdomain DOFs	1,512	2,424	1,550
Global DOFs	1,260,572	1,269,011	1,264,822
Boundary DOFs	407,731	410,488	409,080
Corner DOFs	7,836	10,593	9,185
Lagrange multipliers	631,765	631,765	631,765

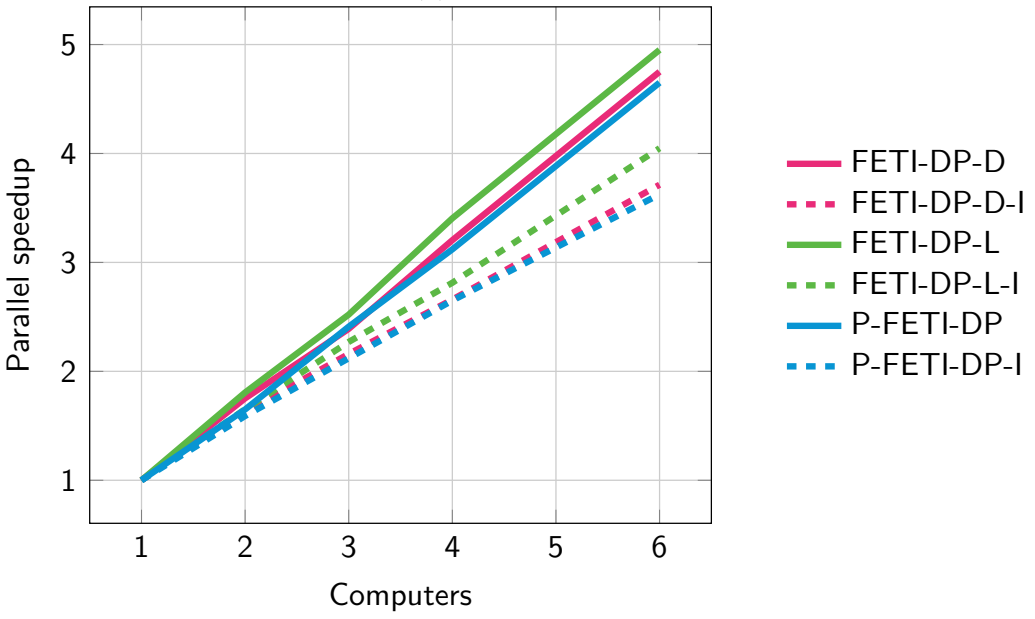
Table 6.5: Test case 2. Number of DOFs for the case of $(252 \times 56 \times 28)$ elements and $(36 \times 8 \times 4)$ subdomains.

Fig. 6.17a depicts the time required for the solution phase of the XFEM analysis by each of the solvers. As previously, P-FETI-DP is faster than FETI-DP-D and FETI-DP-L and the improved versions P-FETI-DP-I, FETI-DP-D-I, FETI-DP-L-I are even faster. However, as more resources (computers) are added, the relative performance differences between the 6 solvers become less pronounced. The parallel speedup, namely $speedup = \dfrac{t(1 \text{ computers})}{t(\text{n computers})}$ is shown in Fig. 6.17b. It can be observed that all solvers scale well, when more computers are added to the system.

In order to quantify the communication between neighboring subdomains and their corresponding computers, Fig. 6.8 presents the data transferred in MB. It can be observed that *remote* data transfers (between subdomains of different computers) are significantly lower than *local* data transfers (between subdomains of the same computer) in this application. Additionally, FETI-DP requires transferring less data than P-FETI-DP, which can be attributed to the removal of enriched DOFs from the interface-problem of FETI-DP (see section 5.4.2), but not from the interface problem of P-FETI-DP.



(a)



(b)

Figure 6.17: Test case 2. Parallel scalability analysis. Constant elements ($252 \times 56 \times 28$) and subdomains ($36 \times 8 \times 4$), variable number of computers. (a) Time (in seconds) required for the solution phase (b) Parallel speedup.

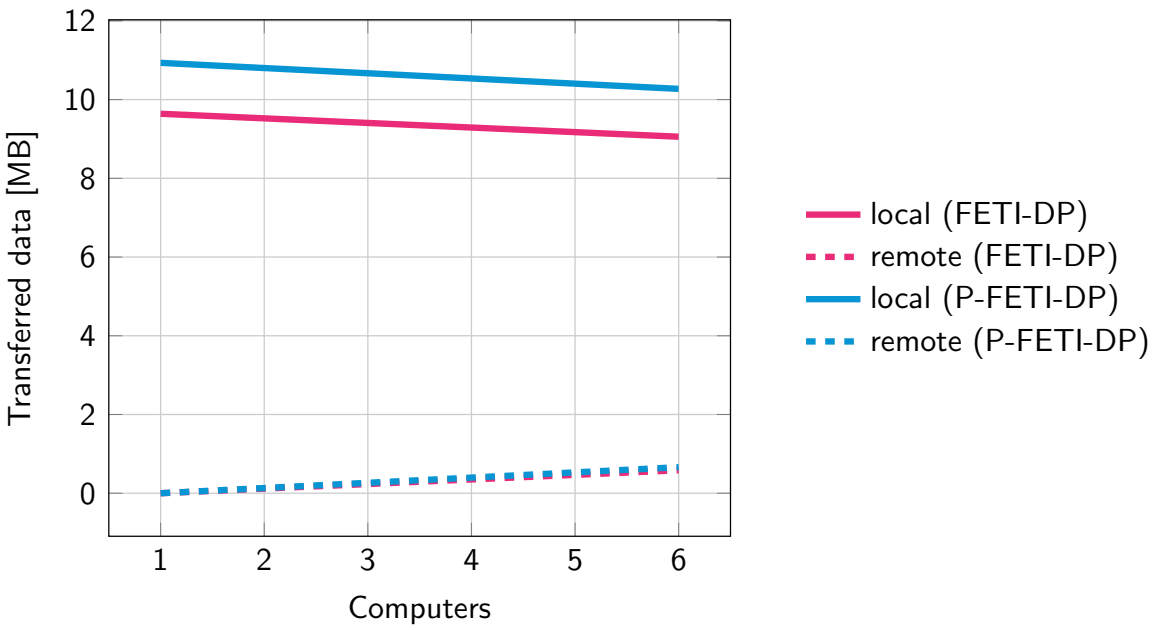
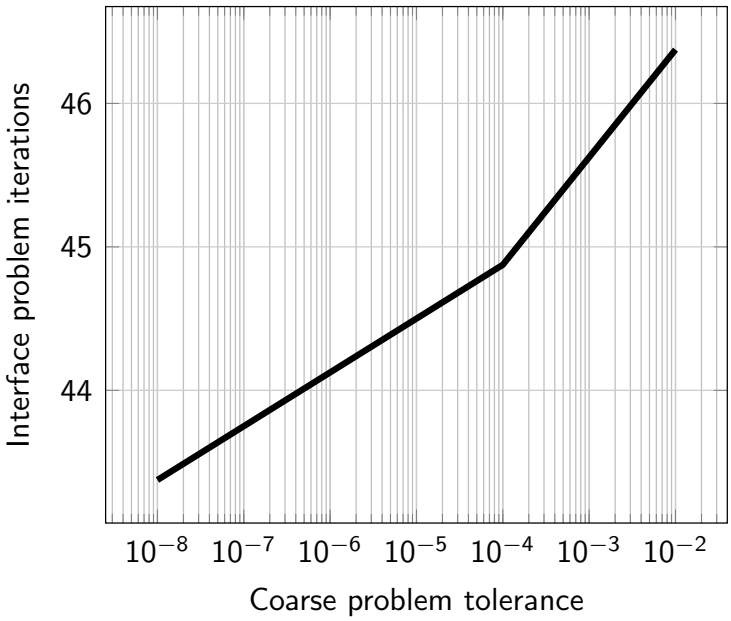


Figure 6.18: Test case 2. Data transfers per application of equation (5.70) (P-FETI-DP) or (5.100) (FETI-DP). Local data transfers happen between subdomains on the same computer, while remote data transfers between different computers.

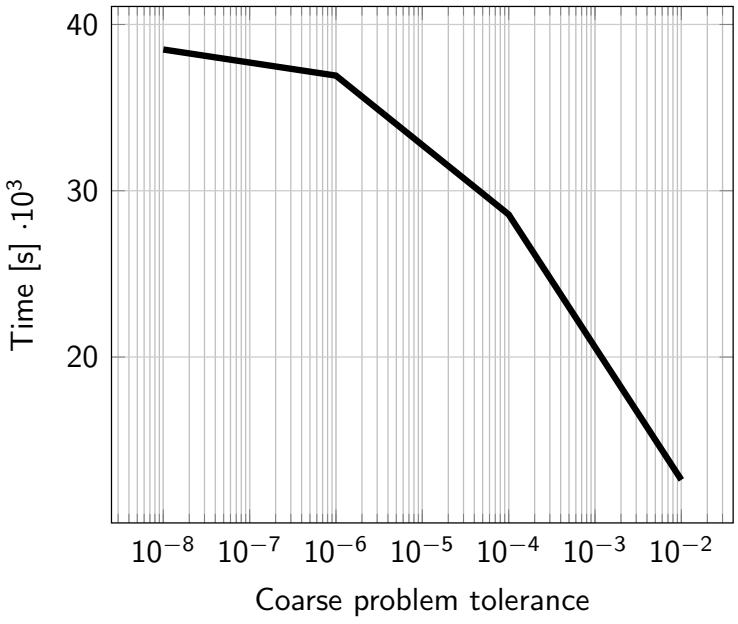
6.3.7 Coarse problem solution strategies

Moreover, the solution of the coarse problem of P-FETI-DP and P-FETI-DP-I is investigated, when using the three different strategies for of sections 5.5.2.2.2, 5.5.2.2.3 and 5.5.2.2.1. Fig. 6.19b depicts the total time required for the solution phase of P-FETI-DP-I when the coarse problem is solved using PCG (the third strategy) with various convergence tolerances. A mesh of $(216 \times 48 \times 24)$ elements (805,162 global DOFs on average) and $(36 \times 8 \times 4)$ subdomains (8,244 corner DOFs on average) is used. For the same case, Fig. 6.19a shows the average iterations required by the interface-problem PCG to converge to a tolerance of $1E-7$. It can be observed that relaxing the coarse-problem PCG tolerance, slightly decreases the convergence rate of the interface-problem PCG, but the overall computing time is significantly improved. However, increasing the coarse-problem PCG tolerance beyond $1E-2$ will cause the interface-problem PCG to not converge at all.

Finally, the performances of P-FETI-DP and P-FETI-DP-I are compared, when using each of the three coarse-problem solution strategies. A subdomain-element size ratio of $H/h = 5$ and some of the meshes of table 6.4 are used. The third coarse-problem solution strategy, a tolerance of $1E-2$ is used for the coarse-problem PCG, since it was shown to be optimal for this problem in Fig. 6.19b. The total time required for the solution phase of the XFEM analysis is illustrated in Fig. 6.20. It can be observed that the difference between the



(a)



(b)

Figure 6.19: Test case 2. Coarse problem of P-FETI-DP-I is solved with PCG. a) Solution time for various tolerances of the coarse-problem PCG and b) iterations of the interface-problem PCG.

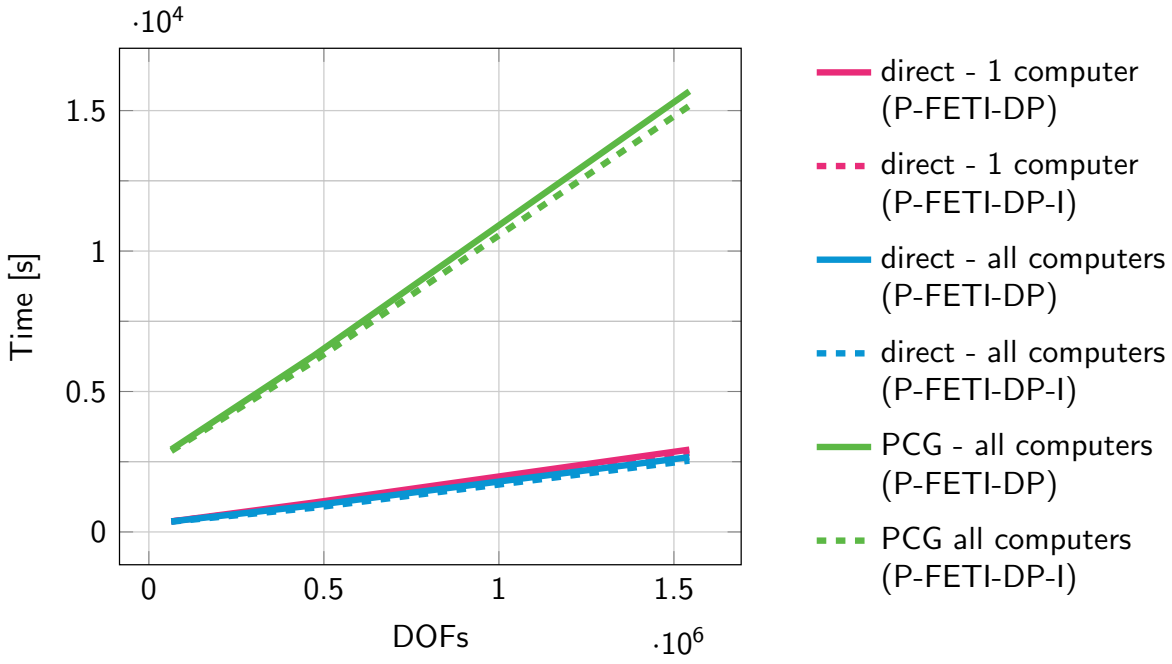


Figure 6.20: Test case 2. Solution time of P-FETI-DP with different coarse problem solution strategies.

first two strategies, which use a direct method, is very small, while the third strategy, which uses an iterative method, is considerably slower, which counter-balances the advantage of reduced memory requirements.

6.4 Conclusions

In the 3D examples investigated, both FETI-DP and P-FETI-DP are faster than optimally implemented standard iterative (73 times faster) and direct solvers (833 times faster). They are also much more insensitive to ill-conditioned problems than iterative solvers. Comparison with the incremental Cholesky solver developed in Pais et al., 2012 specifically for XFEM crack propagation, proves that the proposed methods are significantly faster, require less memory and use a more robust reanalysis approach. Between the two proposed solvers, P-FETI-DP is overall faster than FETI-DP, but both exhibit numerical scalability with respect to the number of subdomains and the problem size. Their efficiency is further improved, when they are executed in high performance computing systems, such as computer clusters.

Chapter 7

Summary - Innovation of thesis

This thesis presents a two-scale numerical framework for conductive heat transfer in nano-composites with complex geometries and thermal resistance along the interfaces between different materials. This numerical model is based the extended finite element method (XFEM), which enriches the polynomial approximation space of FEM with discontinuous functions, in order to capture the jump in the temperature field across material interfaces. Thus, new basis functions, enriched with the Heaviside function, and corresponding enriched freedom degrees are added to the approximation space. In the proposed XFEM formulation, novel junction functions are also introduced for the enrichment of nodes in finite elements, where more than two material interfaces coincide. The interface thermal resistance is taken into account through its effect on the the temperature jump across material interfaces, which is calculated using the Heaviside and junction enrichments.

This XFEM formulation is then coupled with the level set method (LSM), in order to represent the geometry of the material interfaces. Specifically, a double-mesh LSM approach has been developed, which uses two different meshes for XFEM and LSM operations. The XFEM mesh is fine enough to achieve the desired accuracy for the analysis, but coarse enough to not redundantly increase the computational effort. The LSM mesh is much finer, in order to capture the complex geometry of the interfaces between the matrix material and the inclusions. Yet, the fine LSM mesh derives from the coarser XFEM mesh at a subset of its nodes, so that they geometric operations on the coarse mesh needed by XFEM can be performed on the fine mesh instead.

The combined XFEM-LSM model is used for simulating heat transfer in composite materials, consisting of carbon nanotubes (CNT) embedded into a polymer matrix. Using computational homogenization, the macroscopic effective conductivity of the material was estimated using reference volume elements (RVE) in the micro-scale. The thermal resistance of the interfaces between CNTs and polymers is generally unknown and was inferred by calibrating the developed model using experimentally measured values of the macroscopic conductivity for various material configurations. Apart from polymer-CNT nano-composites, conductive heat transfer in materials with polycrystalline structure was also modeled and validated against results from the literature.

Furthermore, the solution of algebraic equations resulting from XFEM has been addressed. Specifically, solvers based on the domain decomposition methods (DDM) FETI-DP and P-FETI-DP have been developed for modeling crack propagation problems with XFEM. In these DDM solvers, the domain is partitioned into multiple subdomains, which can be processed in parallel, in order to reduce the computational effort and memory requirements. However, mechanisms are developed in subdomains that are completely intersected by cracks, which causes their stiffness matrices to become singular. In this thesis, the aforementioned mechanisms were attributed to the discontinuous enrichment functions used by XFEM to capture the displacement field jump across the crack, as well as their corresponding enriched freedom degrees. The proposed solvers restore the invertibility of subdomain matrices, by transferring a subset of these enriched freedom degrees to the coarse problem of FETI-DP and P-FETI-DP.

Another difficulty posed by XFEM is the ill-conditioning of the stiffness matrix, due to enrichment functions that model the singular stress field around the crack front. As a result, iterative solution methods, including those employed by FETI-DP and P-FETI-DP internally, are inefficient for the resulting linear systems, since their converge rate is low. In this thesis, the XFEM-related ill-conditioning has been completely eliminated by further modifying the coarse problem of the proposed DDM solvers. Additionally, optimized versions of FETI-DP and P-FETI-DP were developed for the case of brittle crack propagation. These solvers reuse the solution of the interface problem, the stiffness matrices and Schur complements during one propagation step, in order to reduce the solver iterations and computational effort required for the next step.

Subsequently, an implementation of the proposed FETI-DP and P-FETI-DP solvers was developed for high performance computing systems, specifically computer clusters. Computer clusters are distributed memory environments, consisting of readily available computers linked via a local area network (LAN) to produce a low-cost, yet powerful system. To that end, the original equations of both solvers were replaced with equivalent ones, which avoid global-level operations in favor of exchanges between neighboring subdomains. These enable an implementation with more efficient communication operations, since the amount of data transferred between computers is reduced and distributed more evenly. Finally, the performance of the proposed DDM solvers was investigated in 3D crack propagation problems, where both FETI-DP and P-FETI-DP were found to be significantly more efficient than well-known direct and iterative solvers, as well as a solver that was developed specifically for XFEM crack propagation problems. Moreover, they scale well with respect to the number of subdomains and the problem size.

Below are some different avenues for future work in the topics addressed in this thesis:

- The XFEM-LSM model of chapter 2 can be extended to different types of differential equations, such as elasticity and electrical conduction, as well as coupled problems, such as thermomechanical coupling. In fact, the extension to pure elasticity problems has already been successful after the completion of this thesis.

- Using the numerical model of chapter 2 in the context of structural topology optimization. The proposed XFEM-LSM approach is an attractive choice for topology optimization, since it operates on fixed meshes, while the boundary between the material phases is smooth and can be moved according to the optimization rules. Various interface behaviors can be considered, such as cohesive, namely the primarily field is discontinuous, as in chapter 2, or coherent, namely the first derivative field is discontinuous.

- Extension of the DDM solvers of chapter 5 to more complicated fracture problems than brittle crack propagation under Linear Elastic Fracture Mechanics. Dynamic crack propagation, cohesive cracks, elasto-plastic and ductile materials, which introduce material and geometric non-linearities, are of particular interest. Moreover, problems with heterogeneous materials and multiple interacting or branching cracks should be considered. In the cases above, new problem-specific enrichment functions would be introduced in the XFEM approximation. The proposed coarse problem of section 5.4.2 needs to take into account the corresponding enriched DOFs, in order to avoid singular matrices and ill-conditioning. Furthermore, the optimizations of section 5.4.3 may require significant alterations or not be possible at all, outside brittle crack propagation.

- Improved numerical scalability of the proposed DDM solvers. Specifically, the combination of FETI-DP and P-FETI-DP with artificial intelligence and multigrid methods is already under investigation, in order to handle any parts of the algorithms that are not fully scalable yet.

- Optimized parallel implementation and increase of the parallel speedup of the DDM solvers. The main obstacle consists of time wasted during data transfers between subdomains that belong to different computers. To this end, multiple network topologies (e.g. cartesian) should be explored for the local area network connecting the individual machines of the computer cluster. In addition, the utilization of the MPI library can always become more efficient, e.g. by using batch data transfers more frequently. Finally, the proposed DDM solvers should be made as easy to use as possible, by employing adaptive load balancing techniques.

References

Abdi, M., Ashcroft, I., & Wildman, R. (2017). Topology optimization of geometrically nonlinear structures using an evolutionary optimization method. *Engineering Optimization*, *54*(1), 1850–1870.

Ajorloo, M., Fasihi, M., Ohshima, M., & Taki, K. (2019). How are the thermal properties of polypropylene/graphene nanoplatelet composites affected by polymer chain configuration and size of nanofiller? *Materials & Design*, *181*, 108068. https://doi.org/https://doi.org/10.1016/j.matdes.2019.108068

Amestoy, P. R., Davis, T. A., & Duff, I. S. (2004). Algorithm 837: AMD, an approximate minimum degree ordering algorithm. *ACM Transactions on Mathematical Software*, *30*(3), 381–388.

Bakalakos, S., Kalogeris, I., & Papadopoulos, V. (2020). An extended finite element method formulation for modeling multi-phase boundary interactions in steady state heat conduction problems. *Composite Structures*, *258*, Article 113202. https://doi.org/https://doi.org/10.1016/j.compstruct.2020.113202

Bakalakos, S., Kalogeris, I., Papadopoulos, V., Papadrakakis, M., Maroulas, P., Dragatogiannis, D. A., & Charitidis, C. A. (2022). An integrated XFEM modeling with experimental measurements for optimizing thermal conductivity in carbon nanotube reinforced polyethylene. *Modelling and Simulation in Materials Science and Engineering*, *30*(2). https://doi.org/10.1088/1361-651X/ac4899

Bansal, M., Singh, I., Mishra, B., & Bordas, S. (2019). A parallel and efficient multi-split XFEM for 3-d analysis of heterogeneous materials. *Computer Methods in Applied Mechanics and Engineering*, *347*, 365–401.

Bechet, E., Minnebo, H., Moës, N., & Burgardt, B. (2005). Improved implementation and robustness study of the X-FEM for stress analysis around cracks. *International Journal for Numerical Methods in Engineering*, *64*(8), 1033–1056.

Beese, S., Loehnert, S., & Wriggers, P. (2018). 3D ductile crack propagation within a polycrystalline microstructure using XFEM. *Computational Mechanics*, *61*, 71–88.

Belytschko, T., & Black, T. (1999). Elastic crack growth in finite elements with minimal remeshing. *International Journal for Numerical Methods in Engineering*, *45*(5), 601–620.

Benvenuti, E. (2014). XFEM with equivalent eigenstrain for matrix–inclusion interfaces. *Computational Mechanics*, *53*, 893–908.

Berber, S., Kwon, Y.-K., & Tománek, D. (2000). Unusually high thermal conductivity of carbon nanotubes. *Physical Review Letters*, *84*(20), 4613–4616.

Bogdanor, M. J., Oskay, C., & Clay, S. B. (2015). Multiscale modeling of failure in composites under model parameter uncertainty. *Computational Mechanics*, *56*(3), 389–404.

Bogdanor, M. J., Mahadevan, S., & Oskay, C. (2013). Uncertainty quantification in damage modeling of heterogeneous materials. *International Journal for Multiscale Computational Engineering*, *11*(3), 289–307.

Carlborg, C. F., Shiomi, J., & Maruyama, S. (2008). Thermal boundary resistance between single-walled carbon nanotubes and surrounding matrices. *Physical Review B*, *78*, 205406. https://doi.org/10.1103/PhysRevB.78.205406

Chalopin, Y., Volz, S., & Mingo, N. (2009). Upper bound to the thermal conductivity of carbon nanotube pellets. *Journal of Applied Physics*, *105*(8), 084301. https://doi.org/10.1063/1.3088924

Che, J., Çagin, T., & Goddard, W. A. (2000). Thermal conductivity of carbon nanotubes. *Nanotechnology*, *11*(2), 65–69.

Che, J., Wu, K., Lin, Y., Wang, K., & Fu, Q. (2017). Largely improved thermal conductivity of HDPE/expanded graphite/carbon nanotubes ternary composites via filler network-network synergy. *Composites Part A: Applied Science and Manufacturing*, *99*, 32–40. https://doi.org/https://doi.org/10.1016/j.compositesa.2017.04.001

Chen, H., Ginzburg, V. V., Yang, J., Yang, Y., Liu, W., Huang, Y., Du, L., & Chen, B. (2016). Thermal conductivity of polymer-based composites: Fundamentals and applications. *Progress in Polymer Science*, *59*, 41–85. https://doi.org/https://doi.org/10.1016/j.progpolymsci.2016.03.001

Chen, X., & Cai, X.-C. (2022). A recycling preconditioning method with auxiliary tip subspace for elastic crack propagation simulation using XFEM. *Journal of Computational Physics*, *452*, Article 110910.

Chen, Y., Davis, T. A., Hager, W. W., & Rajamanickam, S. (2008). Algorithm 887: CHOLMOD, supernodal sparse Cholesky factorization and update/downdate. *ACM Transactions on Mathematical Software*, *35*(3), Article 22.

Chew, P. L. (1989). Constrained delaunay triangulations. *Algorithmica*, *4*(1), 97–108.

Chopp, D. L., & Sukumar, N. (2003). Fatigue crack propagation of multiple coplanar cracks with the coupled extended finite element/fast marching method. *International Journal of Engineering Science*, *41*(8), 845–869.

Cuthill, E., & McKee, J. (1969). Reducing the bandwidth of sparse symmetric matrices. *Proceedings of the 1969 ACM national conference*, *1*, 157–172.

Dassault Systemes. (2020). Abaqus FEA. https://www.3ds.com/products-services/simulia/products/abaqus/

Daux, C., Moës, N. S., Dolbow, J., Sukumar, N., & Belytschko, T. (2000). Arbitrary branched and intersecting cracks with the extended finite element method. *International Journal for Numerical Methods in Engineering*, *48*(12), 1741–1760.

Davis, T. A. (2022). Suitesparse. https://people.engr.tamu.edu/davis/suitesparse.html

Davis, T. A., Gilbert, J. R., Larimore, S. I., & Ng, E. G. (2004). Algorithm 836: COLAMD, a column approximate minimum degree ordering algorithm. *ACM Transactions on Mathematical Software*, *30*(3), 377–380.

Duflot, M. (2008). The extended finite element method in thermoelastic fracture mechanics. *International Journal for Numerical Methods in Engineering*, *74*, 827–847.

Elguedj, T., Gravouil, A., & A., C. (2006). Appropriate extended functions for X-FEM simulation of plastic fracture mechanics. *Computer Methods in Applied Mechanics and Engineering*, *195*(7), 501–515.

Farhat, C., Lesoinne, M., & Pierson, K. (2000). A scalable dual-primal domain decomposition method. *Numerical Linear Algebra with Applications*, *7*, 687–714.

Farhat, C., & Roux, F. X. (1991). A method of finite element tearing and interconnecting and its parallel solution algorithm. *International Journal for Numerical Methods in Engineering*, *32*(6), 1205–1227.

Farhat, C., & Roux, F. X. (1994). Implicit parallel processing in structural mechanics. *Computational Mechanics Advances*, *2*(1), 1–124.

Feng, S. Z., & Han, X. (2019). A novel multi-grid based reanalysis approach for efficient prediction of fatigue crack propagation. *Computer Methods in Applied Mechanics and Engineering*, *353*, 107–122.

Fragakis, Y., & Papadrakakis, M. (2003). The mosaic of high performance domain decomposition methods for structural mechanics: Formulation, interrelation and numerical efficiency of primal and dual methods. *Computer Methods in Applied Mechanics and Engineering*, *192*(35), 3799–3830.

Fries, T. P., & Baydoun, M. (2012). Crack propagation with the XFEM and a hybrid explicit-implicit crack description. *International Journal for Numerical Methods in Engineering*, *89*(12), 1527–1558.

Geers, M. G. D., Kouznetsova, V. G., & Brekelmans, W. A. M. (2010). Multi-scale computational homogenization: Trends and challenges. *Journal of Computational and Applied Mathematics*, *234*(7), 2175–2182.

George, A. (1973). Nested dissection of a regular finite element mesh. *SIAM Journal on Numerical Analysis*, *10*(2), 345–363.

Georgioudakis, M., Lagaros, N. D., & Papadrakakis, M. (2017). Probabilistic shape design optimization of structural components under fatigue. *Computers & Structures*, *182*, 252–266.

Gerstenberger, A., & Tuminaro, R. S. (2013). An algebraic multigrid approach to solve XFEM based fracture problems. *International Journal for Numerical Methods in Engineering*, *94(3)*, 248–272.

Gerstenberger, A., & Wall, W. (2008). An extended finite element method/lagrange multiplier based approach for fluid-structure interaction. *Computer Methods in Applied Mechanics and Engineering*, *197*, 1699–1714.

Gojny, F. H., Wichmann, M. H., Fiedler, B., Kinloch, I. A., Bauhofer, W., Windle, A. H., & Schulte, K. (2006). Evaluation and identification of electrical and thermal conduction mechanisms in carbon nanotube/epoxy composites. *Polymer*, *47*(6), 2036–2045.

Golias, N. A., & Dutton, R. W. (1997). Delaunay triangulation and 3d adaptive mesh generation. *Finite Elements in Analysis and Design*, *25*(3), 331–341.

Gravouil, A., Moës, N., & Belytschko, T. (2002). Non-planar 3D crack growth by the extended finite element and level sets—Part II: Level set update. *International Journal for Numerical Methods in Engineering*, *53*(11), 2569–2586.

Gravouil, A., Rannou, J., & Baietto, M.-C. (2008). A local multi-grid X-FEM approach for 3D fatigue crack growth. *International Journal of Material Forming*, *1*, 1103–1106.

Gurtin, M. E. (1995). The nature of configurational forces. *Archive for Rational Mechanics and Analysis*, *131*(1), 67–100.

Hiriyur, B., Tuminaro, R. S., Waisman, H., Boman, E. G., & Keyes, D. (2012). A quasi-algebraic multigrid approach to fracture problems based on extended finite elements. *SIAM Journal on Scientific Computing*, *34*(2).

Hussain, A. R. J., Alahyari, A. A., Eastman, S. A., Thibaud-Erkey, C., Johnston, S., & Sobkowicz, M. J. (2017). Review of polymers for heat exchange applications: Factors concerning thermal conductivity. *Applied Thermal Engineering*, *113*, 1118–1127.

Intel Corporation. (2022). Intel (R) oneAPI Math Kernel Library. https://www.intel.com/content/www/us/en/developer/tools/oneapi/onemkl.html

Kalthoff, J. F., & Winkler, S. (1987). Failure mode transition at high rates of shear loading. *Impact Loading and Dynamic Behavior of Materials*, *1*, 185–195.

Kamiński, M., & Ostrowski, P. (2021). Homogenization of heat transfer in fibrous composite with stochastic interface defects. *Composite Structures*, *261*, 113555. https://doi.org/https://doi.org/10.1016/j.compstruct.2021.113555

Khalkhali, M., Rajabpour, A., & Khoeini, F. (2019). Thermal transport across grain boundaries in polycrystalline silicene: A multiscale modeling. *Scientific Reports*, *9*(1), 5684.

Khoei, A. R., Biabanaki, S. O. R., & Anahid, M. (2009). A lagrangian-extended finite-element method in modeling large-plasticity deformations and contact problems. *International Journal of Mechanical Sciences*, *51*, 384–401.

Konstantopoulos, G., Maroulas, P., Dragatogiannis, D. A., Koutsoumpis, S., Kyritsis, A., & Charitidis, C. A. (2021). The effect of interfacial resistance and crystallinity on heat transfer mechanism in carbon nanotube reinforced polyethylene. *Materials & Design*, *199*, 109420. https://doi.org/https://doi.org/10.1016/j.matdes.2020.109420

Kumar, S., Alam, M. A., & Murthy, J. Y. (2007). Effect of percolation on thermal transport in nanotube composites. *Applied Physics Letters*, *90*(10), 104105.

Kumar, S., & Murthy, J. Y. (2009). Interfacial thermal transport between nanotubes. *Journal of Applied Physics*, *106*(8), 084302. https://doi.org/10.1063/1.3245388

Lang, C., Makhija, D., Doostan, A., & Maute, K. (2014). A simple and efficient preconditioning scheme for heaviside enriched XFEM. *Computational Mechanics*, *54*(5), 1357–1374.

Laschet, G., Apel, M., Wipperfürth, J., Hopmann, C., Spekowius, M., & Spina, R. (2017). Effective thermal properties of an isotactic polypropylene (α-ipp) injection moulded part by a multiscale approach. *Materialwissenschaft und Werkstofftechnik*, *48*(12), 1213–1219.

Liao, Q., Liu, Z., Liu, W., Deng, C., & Yang, N. (2015). Extremely high thermal conductivity of aligned carbon nanotube-polyethylene composites. *Scientific Reports*, *5*.

Maier, C., & Calafut, T. (1998). *Polypropylene: The definitive user's guide and databook*. Norwich: William Andrew.

Marconnet, A. M., Panzer, M. A., & Goodson, K. E. (2013). Thermal conduction phenomena in carbon nanotubes and related nanostructured materials. *Reviews of Modern Physics*, *85*, 1295–1326.

Marcos-Gómez, D., Ching-Lloyd, J., Elizalde, M. R., Clegg, W., & Molina-Aldareguía, J. (2010). Predicting the thermal conductivity of composite materials with imperfect interfaces. *Composites Science and Technology*, *70*, 2276–2283.

Menk, A., & Bordas, S. (2011). A robust preconditioning technique for the extended finite element method. *International Journal for Numerical Methods in Engineering*, *85*, 1609–1632.

Metsis, P., & Papadrakakis, M. (2012). Overlapping and non-overlapping domain decomposition methods for large-scale meshless EFG simulations. *Computer Methods in Applied Mechanics and Engineering*, *229-232*, 128–141.

MGroup. (2022). MSolve. https://github.com/mgroupntua/MSolve.Core

Miehe, C., & Koch, A. (2002). Computational micro-to-macro transitions of discretized microstructures undergoing small strains. *Archive of Applied Mechanics*, *72*, 300–317.

Moës, N., & Belytschko, T. (2002). Extended finite element method for cohesive crack growth. *Engineering Fracture Mechanics*, *69*, 813–833.

Moës, N., Cloire, M., Cartraud, P., & Remacle, J.-F. (2003). A computational approach to handle complex microstructure geometries. *Computer Methods in Applied Mechanics and Engineering*, *192*(28), 3163–3177.

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

Moisala, A., Li, Q., Kinloch, I., & Windle, A. (2006). Thermal and electrical conductivity of single- and multi-walled carbon nanotube-epoxy composites. *Composites Science and Technology*, *66*(10), 1285–1288.

Mortazavi, B., Poetschke, M., & Cuniberti, G. (2014). Multiscale modeling of thermal conductivity of polycrystalline graphene sheets. *Nanoscale*, *6*(6), 3344–3352.

Mortazavi, B., Quey, R., Ostadhossein, A., Villani, A., Moulin, N., van Duin, A. C., & Rabczuk, T. (2017). Strong thermal transport along polycrystalline transition metal dichalcogenides revealed by multiscale modeling for mos2. *Applied Materials Today*, *7*, 67–76.

Osher, S., & Sethian, J. A. (1988). Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations. *Journal of Computational Physics*, *79*(1), 12–49.

Ozisik, M. N. (1993). *Heat conduction*. John Wiley & Sons.

Pais, M. J., Yeralan, S. N., Davis, T. A., & Kim, N. H. (2012). An exact reanalysis algorithm using incremental Cholesky factorization and its application to crack growth modeling. *International Journal for Numerical Methods in Engineering*, *91*(12), 1358–1364.

Papadopoulos, V., & Impraimakis, M. (2017). Multiscale modeling of carbon nanotube reinforced concrete. *Composite Structures*, *182*, 251–260.

Papadopoulos, V., Seventekidis, P., & Sotiropoulos, G. (2017). Stochastic multiscale modeling of graphene reinforced composites. *Engineering Structures*, *145*, 176–189.

Pop, E., Mann, D., Wang, Q., Goodson, K., & Dai, H. (2006). Thermal conductance of an individual single-wall carbon nanotube above room temperature. *Nano letters*, *61*, 96–100.

Pyrialakos, S., Kalogeris, I., Sotiropoulos, G., & Papadopoulos, V. (2021). A neural network-aided Bayesian identification framework for multiscale modeling of nanocomposites. *Computer Methods in Applied Mechanics and Engineering*, *384*, 113937. https://doi.org/https://doi.org/10.1016/j.cma.2021.113937

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

Saha, S. K., & Shi, L. (2007). Molecular dynamics simulation of thermal transport at a nanometer scale constriction in silicon. *Journal of Applied Physics*, *101*(7), 074304. https://doi.org/10.1063/1.2715488

Savvas, D., Stefanou, G., Papadrakakis, M., & Deodatis, G. (2014). Homogenization of random heterogeneous media with inclusions of arbitrary shape modeled by XFEM. *Computational Mechanics*, *54*, 1221–1235.

Savvas, D., Papaioannou, I., & Stefanou, G. (2020). Bayesian identification and model comparison for random property fields derived from material microstructure. *Computer Methods in Applied Mechanics and Engineering*, *365*, 113026.

Savvas, D., & Papadopoulos, V. (2014). Nonlinear multiscale homogenization of carbon nanotube reinforced composites with interfacial slippage. *International Journal for Multiscale Computational Engineering*, *12*(4), 271–289.

Seidel, G. D., & Puydupin-Jamin, A.-S. (2011). Analysis of clustering, interphase region, and orientation effects on the electrical conductivity of carbon nanotube–polymer nanocomposites via computational micromechanics. *Mechanics of Materials*, *43*(12), 755–774.

Shewchuk, J. R. (2008). Constrained delaunay triangulations. *Discrete & Computational Geometry*, *39*(1), 580–637.

Singh, I. V., Tanaka, M., & Endo, M. (2007). Effect of interface on the thermal conductivity of carbon nanotube composites. *International Journal of Thermal Sciences*, *46*(9), 842–847.

Song, W., Krishnaswamy, V., & Pucha, R. V. (2016). Computational homogenization in RVE models with material periodic conditions for CNT polymer composites. *Composite Structures*, *137*, 9–17. https://doi.org/https://doi.org/10.1016/j.compstruct.2015.11.013

Stavroulakis, G., Tsapetis, D., & Papadrakakis, M. (2012). Non-overlapping domain decomposition solution schemes for structural mechanics isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, *341*, 695–717.

Stolarska, M., & Chopp, D. L. (2003). Modeling thermal fatigue cracking in integrated circuits by level sets and the extended finite element method. *International Journal of Engineering Science*, *41*(20), 2381–2410.

Stolarska, M., Chopp, D. L., Moës, N., & Belytschko, T. (2001). Modelling crack growth by level sets in the extended finite element method. *International Journal for Numerical Methods in Engineering*, *51*(8), 943–960.

Sukumar, N., Chopp, D. L., Moës, N., & Belytschko, T. (2001). Modeling holes and inclusions by level sets in the extended finite-element method. *Computer Methods in Applied Mechanics and Engineering*, *190*(46-47), 6183–6200.

Talischi, C., Paulino, G. H., Pereira, A., & Menezes, I. F. (2012). PolyMesher: a general-purpose mesh generator for polygonal elements written in Matlab. *Structural and Multidisciplinary Optimization*, *45*(3), 309–328.

Ventura, G., Budyn, E., & Belytschko, T. (2003). Vector level sets for description of propagating cracks in finite elements. *International Journal for Numerical Methods in Engineering*, *58*(10), 1571–1592.

Villanueva, C. H., & Maute, K. (2014). Density and level set-XFEM schemes for topology optimization of 3-D structures. *Computational Mechanics*, *54*(1), 133–150.

Waisman, H., & Berger-Vergiat, L. (2013). An adaptive domain decomposition preconditioner for crack propagation problems modeled by XFEM. *International Journal for Multiscale Computational Engineering*, *11*(6), 633–654.

Wang, R., Xie, C., Gou, B., Xu, H., Luo, S., Zhou, J., & Zeng, L. (2020). Epoxy nanocomposites with high thermal conductivity and low loss factor: Realize 3D thermal conductivity network at low content through core-shell structure and micro-nano technology. *Polymer Testing*, *89*, 106574.

Wu, T., Temizer, I., & Wriggers, P. (2013). Computational thermal homogenization of concrete. *Cement and Concrete Composites*, *35*(1), 59–70.

Wyart, E., Duflot, M., Coulon, D., Martiny, P., Pardoen, T., Remacle, J.-F., & Lani, F. (2008). Substructuring FE-XFE approaches applied to three-dimensional crack propagation. *Journal of Computational and Applied Mathematics*, *215*(2), 626–638.

Xie, H. (2007). Thermal and electrical transport properties of a self-organized carbon nanotube pellet. *Journal of Materials Science*, *42*. https://doi.org/https://doi.org/10.1007/s10853-007-1707-6

Yunsheng, X., Gunawidjaja, R., & Beckry, A.-M. (2006). Thermal behavior of single-walled carbon nanotube polymer–matrix composites. *Composites Part A: Applied Science and Manufacturing*, *37*(1), 114–121.

Yvonnet, J., He, Q., Zhu, Q., & Shao, J. (2011). A general and efficient computational procedure for modelling the kapitza thermal resistance based on XFEM. *Computational Materials Science*, *50*(4), 1220–1224.

Yvonnet, J., Quang, H., & He, Q. (2008). An XFEM/level set approach to modelling surface/interface effects and to computing the size-dependent effective properties of nanocomposites. *Computational Mechanics*, *42*, 119–131.

Zhang, G., Xia, Y., Wang, H., Tao, Y., Tao, G., Tu, S., & Wu, H. (2010). A percolation model of thermal conductivity for filled polymer composites. *Journal of Composite Materials*, *44*(8), 963–970. https://doi.org/10.1177/0021998309349690

Zhang, S., Minus, M. L., Zhu, L., Wong, C.-P., & Kumar, S. (2008). Polymer transcrystallinity induced by carbon nanotubes. *Polymer*, *49*(5), 1356–1364.

Zhang, W.-b., Zhang, Z.-x., Yang, J.-h., Huang, T., Zhang, N., Zheng, X.-t., Wang, Y., & Zhou, Z.-w. (2015). Largely enhanced thermal conductivity of poly(vinylidene fluoride)/carbon nanotube composites achieved by adding graphene oxide. *Carbon*, *90*, 242–254. https://doi.org/https://doi.org/10.1016/j.carbon.2015.04.040

Zhang, Y., Hughes, T. J. R., & Bajaj, C. L. (2010). An automatic 3D mesh generation method for domains with multiple materials. *Computer Methods in Applied Mechanics and Engineering*, *199*(5), 405–415.

Zhong, H., & Lukes, J. R. (2006). Interfacial thermal resistance between carbon nanotubes: Molecular dynamics simulations and analytical thermal modeling. *Physical Review B*, *74*, 125403. https://doi.org/10.1103/PhysRevB.74.125403

Appendix A

Element types

A.1 Isoparametric mapping

In this dissertation, isoparametric elements are employed to discretize the domain and the interfaces between subdomains, such as material interfaces. In the isoparametric formulation, apart from the *global coordinate system* $(x,\ y,\ z)$, a secondary one is defined: the *natural coordinate system* $(\xi,\ \eta,\ \zeta)$, which is also called *element-local coordinate system*. Each element type has a specific position is the natural system. Mapping a point from the natural to the global system is done using the Lagrange polynomial shape functions, which are defined as functions of the natural coordinates $(\xi,\ \eta,\ \zeta)$:

$$\boldsymbol{x} = \sum_{k=1}^{n_{ne}} N_k\left(\boldsymbol{\xi}\right) \boldsymbol{x}_k \tag{A.1}$$

where n_{ne} is the number of nodes of the element and $\boldsymbol{x}_k = (x_k,\ y_k,\ z_k)$ are the coordinates of node k in the global system. The Jacobian matrix of this isoparametric mapping from the natural to the global system is in 2D problems

$$\boldsymbol{J}_{NG}\left(\boldsymbol{\xi}\right) = \begin{bmatrix} \dfrac{\partial x}{\partial \xi} & \dfrac{\partial y}{\partial \xi} \\ \dfrac{\partial x}{\partial \eta} & \dfrac{\partial y}{\partial \eta} \end{bmatrix} = \begin{bmatrix} \displaystyle\sum_{k=1}^{n_{ne}} \frac{\partial N_k\left(\boldsymbol{\xi}\right)}{\partial \xi} x_k & \displaystyle\sum_{k=1}^{n_{ne}} \frac{\partial N_k\left(\boldsymbol{\xi}\right)}{\partial \xi} y_k \\ \displaystyle\sum_{k=1}^{n_{ne}} \frac{\partial N_k\left(\boldsymbol{\xi}\right)}{\partial \eta} x_k & \displaystyle\sum_{k=1}^{n_{ne}} \frac{\partial N_k\left(\boldsymbol{\xi}\right)}{\partial \eta} y_k \end{bmatrix} \tag{A.2}$$

and in 3D problems

$$\begin{aligned}\boldsymbol{J}_{NG}(\boldsymbol{\xi}) &= \begin{bmatrix} \dfrac{\partial x}{\partial \xi} & \dfrac{\partial y}{\partial \xi} & \dfrac{\partial z}{\partial \xi} \\ \dfrac{\partial x}{\partial \eta} & \dfrac{\partial y}{\partial \eta} & \dfrac{\partial z}{\partial \eta} \\ \dfrac{\partial x}{\partial \zeta} & \dfrac{\partial y}{\partial \zeta} & \dfrac{\partial z}{\partial \zeta} \end{bmatrix} \\ &= \begin{bmatrix} \displaystyle\sum_{k=1}^{n_{ne}} \frac{\partial N_k(\boldsymbol{\xi})}{\partial \xi} x_k & \displaystyle\sum_{k=1}^{n_{ne}} \frac{\partial N_k(\boldsymbol{\xi})}{\partial \xi} y_k & \displaystyle\sum_{k=1}^{n_{ne}} \frac{\partial N_k(\boldsymbol{\xi})}{\partial \xi} z_k \\ \displaystyle\sum_{k=1}^{n_{ne}} \frac{\partial N_k(\boldsymbol{\xi})}{\partial \eta} x_k & \displaystyle\sum_{k=1}^{n_{ne}} \frac{\partial N_k(\boldsymbol{\xi})}{\partial \eta} y_k & \displaystyle\sum_{k=1}^{n_{ne}} \frac{\partial N_k(\boldsymbol{\xi})}{\partial \eta} z_k \\ \displaystyle\sum_{k=1}^{n_{ne}} \frac{\partial N_k(\boldsymbol{\xi})}{\partial \zeta} x_k & \displaystyle\sum_{k=1}^{n_{ne}} \frac{\partial N_k(\boldsymbol{\xi})}{\partial \zeta} y_k & \displaystyle\sum_{k=1}^{n_{ne}} \frac{\partial N_k(\boldsymbol{\xi})}{\partial \zeta} z_k \end{bmatrix} \end{aligned} \tag{A.3}$$

The derivatives of a vector function $\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{f}(\boldsymbol{x}(\boldsymbol{\xi})) = \boldsymbol{f}(\boldsymbol{\xi})$ with respect to the global coordinates, as functions of the natural coordinates, can be calculated using the chain rule, which in 2D problems becomes

$$\nabla_{\boldsymbol{x}}\, \boldsymbol{f}(\boldsymbol{\xi}) = \boldsymbol{J}_{NG}^{-1} \cdot \nabla_{\boldsymbol{\xi}}\, \boldsymbol{f}(\boldsymbol{\xi}) \iff \begin{bmatrix} \dfrac{\partial \boldsymbol{f}(\boldsymbol{\xi})}{\partial x} \\ \dfrac{\partial \boldsymbol{f}(\boldsymbol{\xi})}{\partial y} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial \xi}{\partial x} & \dfrac{\partial \eta}{\partial x} \\ \dfrac{\partial \xi}{\partial y} & \dfrac{\partial \eta}{\partial y} \end{bmatrix} \cdot \begin{bmatrix} \dfrac{\partial \boldsymbol{f}(\boldsymbol{\xi})}{\partial \xi} \\ \dfrac{\partial \boldsymbol{f}(\boldsymbol{\xi})}{\partial \eta} \end{bmatrix} \tag{A.4}$$

and in 3D problems

$$\nabla_{\boldsymbol{x}}\, \boldsymbol{f}(\boldsymbol{\xi}) = \boldsymbol{J}_{NG}^{-1} \cdot \nabla_{\boldsymbol{\xi}}\, \boldsymbol{f}(\boldsymbol{\xi}) \iff \begin{bmatrix} \dfrac{\partial \boldsymbol{f}(\boldsymbol{\xi})}{\partial x} \\ \dfrac{\partial \boldsymbol{f}(\boldsymbol{\xi})}{\partial y} \\ \dfrac{\partial \boldsymbol{f}(\boldsymbol{\xi})}{\partial z} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial \xi}{\partial x} & \dfrac{\partial \eta}{\partial x} & \dfrac{\partial \zeta}{\partial x} \\ \dfrac{\partial \xi}{\partial y} & \dfrac{\partial \eta}{\partial y} & \dfrac{\partial \zeta}{\partial y} \\ \dfrac{\partial \xi}{\partial z} & \dfrac{\partial \eta}{\partial z} & \dfrac{\partial \zeta}{\partial z} \end{bmatrix} \cdot \begin{bmatrix} \dfrac{\partial \boldsymbol{f}(\boldsymbol{\xi})}{\partial \xi} \\ \dfrac{\partial \boldsymbol{f}(\boldsymbol{\xi})}{\partial \eta} \\ \dfrac{\partial \boldsymbol{f}(\boldsymbol{\xi})}{\partial \zeta} \end{bmatrix} \tag{A.5}$$

A 1D element with 2 nodes is depicted in Fig. A.1. Its shape functions are

$$\begin{aligned} N_1(\boldsymbol{\xi}) &= 0.5(1-\xi) \\ N_2(\boldsymbol{\xi}) &= 0.5(1+\xi) \end{aligned} \tag{A.6}$$

Figure A.1: Isoparametric 1D element with 2 nodes.

Node	ξ	η
P_1	–1	–1
P_2	+1	–1
P_3	+1	+1
P_4	–1	+1

Table A.1: Nodal coordinates of a quadrilateral element with 4 nodes.

A quadrilateral element with 4 nodes is depicted in Fig. A.2. The natural coordinates of its nodes are given in table A.1. Its shape functions are

$$\begin{aligned} N_1(\boldsymbol{\xi}) &= 0.25(1-\xi)(1-\eta) \\ N_2(\boldsymbol{\xi}) &= 0.25(1+\xi)(1-\eta) \\ N_3(\boldsymbol{\xi}) &= 0.25(1+\xi)(1+\eta) \\ N_4(\boldsymbol{\xi}) &= 0.25(1-\xi)(1+\eta) \end{aligned} \tag{A.7}$$

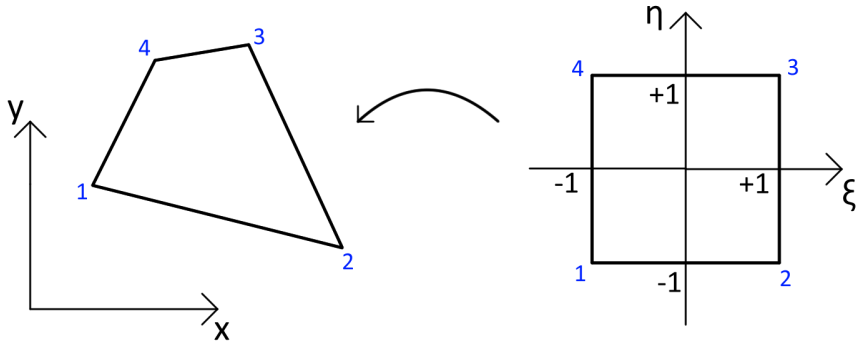


Figure A.2: Isoparametric quadrilateral element with 4 nodes.

A triangular element with 3 nodes is depicted in Fig. A.2. Its shape functions are

$$\begin{aligned} N_1(\boldsymbol{\xi}) &= 1-\xi-\eta \\ N_2(\boldsymbol{\xi}) &= \xi \\ N_3(\boldsymbol{\xi}) &= \eta \end{aligned} \tag{A.8}$$

and their derivatives are

$$\begin{aligned} \frac{\partial N_1(\boldsymbol{\xi})}{\partial \xi} &= -1 \quad \frac{\partial N_1(\boldsymbol{\xi})}{\partial \eta} = -1 \\ \frac{\partial N_2(\boldsymbol{\xi})}{\partial \xi} &= +1 \quad \frac{\partial N_2(\boldsymbol{\xi})}{\partial \eta} = 0 \\ \frac{\partial N_3(\boldsymbol{\xi})}{\partial \xi} &= 0 \quad \frac{\partial N_3(\boldsymbol{\xi})}{\partial \eta} = +1 \end{aligned} \tag{A.9}$$

which are constant, therefore the Jacobian matrix of equation (A.2) is also constant

$$\begin{aligned} \boldsymbol{J}_{NG} = \boldsymbol{J}_{NG}(\boldsymbol{\xi}) &= \begin{bmatrix} \sum_{k=1}^{3} \frac{\partial N_k(\boldsymbol{\xi})}{\partial \xi} x_k & \sum_{k=1}^{3} \frac{\partial N_k(\boldsymbol{\xi})}{\partial \xi} y_k \\ \sum_{k=1}^{3} \frac{\partial N_k(\boldsymbol{\xi})}{\partial \eta} x_k & \sum_{k=1}^{3} \frac{\partial N_k(\boldsymbol{\xi})}{\partial \eta} y_k \end{bmatrix} \\ &= \begin{bmatrix} -1x_1 + 1x_2 + 0x_3 & -1y_1 + 1y_2 + 0y_3 \\ -1x_1 + 0x_2 + 1x_3 & -1y_1 + 0y_2 + 1y_3 \end{bmatrix} = \begin{bmatrix} x_2 - x_1 & y_2 - y_1 \\ x_3 - x_1 & y_3 - y_1 \end{bmatrix} \end{aligned} \tag{A.10}$$

which is a constant. Actually the determinant of $\boldsymbol{J}_{NG}$ is related to the area A_{tri} of the triangular element $(\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3)$ in the global cartesian system

$$\begin{aligned} \det(\boldsymbol{J}_{NG}) = \det\left(\boldsymbol{J}_{NG}^T\right) &= \det\left(\begin{bmatrix} x_2 - x_1 & x_3 - x_1 \\ y_2 - y_1 & y_3 - y_1 \end{bmatrix}\right) \\ &= \|(\boldsymbol{x}_2 - \boldsymbol{x}_1) \times (\boldsymbol{x}_3 - \boldsymbol{x}_1)\| = 2\ A_{tri} \end{aligned} \tag{A.11}$$

since the norm of the cross product $(\boldsymbol{x}_2 - \boldsymbol{x}_1) \times (\boldsymbol{x}_3 - \boldsymbol{x}_1)$ is the area of a parallelogram that shares two sides $\boldsymbol{x}_2 - \boldsymbol{x}_1$, $\boldsymbol{x}_3 - \boldsymbol{x}_1$ with the triangle.

A hexahedral element with 8 nodes is depicted in Fig. A.4. The natural coordinates of its nodes are given in table A.2. Its shape functions are

$$\begin{aligned} N_1(\boldsymbol{\xi}) &= 0.125(1-\xi)(1-\eta)(1-\zeta) \\ N_2(\boldsymbol{\xi}) &= 0.125(1+\xi)(1-\eta)(1-\zeta) \\ N_3(\boldsymbol{\xi}) &= 0.125(1+\xi)(1+\eta)(1-\zeta) \\ N_4(\boldsymbol{\xi}) &= 0.125(1-\xi)(1+\eta)(1-\zeta) \\ N_5(\boldsymbol{\xi}) &= 0.125(1-\xi)(1-\eta)(1+\zeta) \\ N_5(\boldsymbol{\xi}) &= 0.125(1+\xi)(1-\eta)(1+\zeta) \\ N_7(\boldsymbol{\xi}) &= 0.125(1+\xi)(1+\eta)(1+\zeta) \\ N_8(\boldsymbol{\xi}) &= 0.125(1-\xi)(1+\eta)(1+\zeta) \end{aligned} \tag{A.12}$$

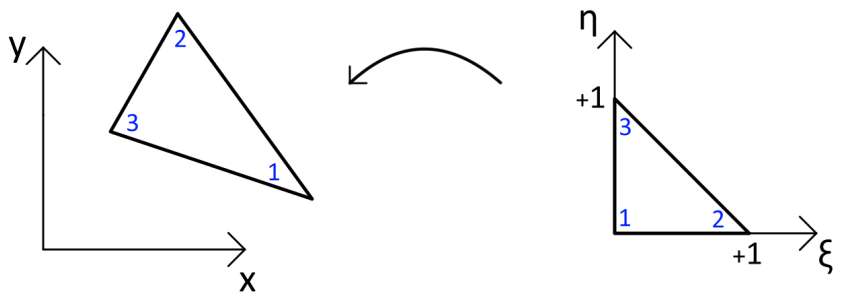


Figure A.3: Isoparametric triangular element with 3 nodes.

Node	ξ	η	ζ
P_1	-1	-1	-1
P_2	$+1$	-1	-1
P_3	$+1$	$+1$	-1
P_4	-1	$+1$	-1
P_5	-1	-1	$+1$
P_6	$+1$	-1	$+1$
P_7	$+1$	$+1$	$+1$
P_8	-1	$+1$	$+1$

Table A.2: Nodal coordinates of a hexahedral element with 8 nodes.

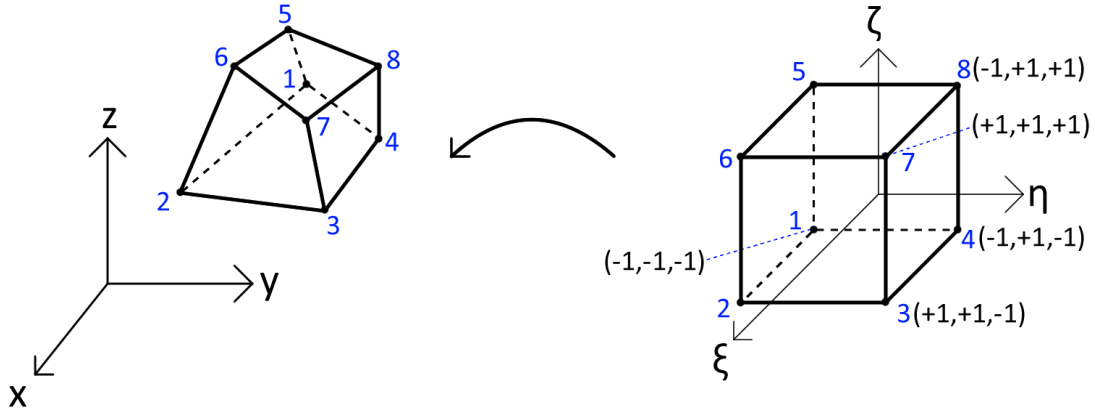


Figure A.4: Isoparametric hexahedral element with 8 nodes.

A tetrahedral element with 4 nodes is depicted in Fig. A.4. Its shape functions are

$$\begin{aligned} N_1(\boldsymbol{\xi}) &= 1-\xi-\eta-\zeta \\ N_2(\boldsymbol{\xi}) &= \xi \\ N_3(\boldsymbol{\xi}) &= \eta \\ N_4(\boldsymbol{\xi}) &= \zeta \end{aligned} \tag{A.13}$$

and their derivatives are

$$\begin{array}{lll} \dfrac{\partial N_1(\boldsymbol{\xi})}{\partial \xi} = -1 & \dfrac{\partial N_1(\boldsymbol{\xi})}{\partial \eta} = -1 & \dfrac{\partial N_1(\boldsymbol{\xi})}{\partial \zeta} = -1 \\ \dfrac{\partial N_2(\boldsymbol{\xi})}{\partial \xi} = +1 & \dfrac{\partial N_2(\boldsymbol{\xi})}{\partial \eta} = 0 & \dfrac{\partial N_2(\boldsymbol{\xi})}{\partial \zeta} = 0 \\ \dfrac{\partial N_3(\boldsymbol{\xi})}{\partial \xi} = 0 & \dfrac{\partial N_3(\boldsymbol{\xi})}{\partial \eta} = +1 & \dfrac{\partial N_3(\boldsymbol{\xi})}{\partial \zeta} = 0 \\ \dfrac{\partial N_4(\boldsymbol{\xi})}{\partial \xi} = 0 & \dfrac{\partial N_4(\boldsymbol{\xi})}{\partial \eta} = +1 & \dfrac{\partial N_4(\boldsymbol{\xi})}{\partial \zeta} = +1 \end{array} \tag{A.14}$$

which are constant, therefore the Jacobian matrix of equation (A.2) is also constant

$$\begin{aligned} \boldsymbol{J}_{NG} = \boldsymbol{J}_{NG}(\boldsymbol{\xi}) &= \begin{bmatrix} \sum_{k=1}^{4} \dfrac{\partial N_k(\boldsymbol{\xi})}{\partial \xi} x_k & \sum_{k=1}^{4} \dfrac{\partial N_k(\boldsymbol{\xi})}{\partial \xi} y_k & \sum_{k=1}^{4} \dfrac{\partial N_k(\boldsymbol{\xi})}{\partial \xi} z_k \\ \sum_{k=1}^{4} \dfrac{\partial N_k(\boldsymbol{\xi})}{\partial \eta} x_k & \sum_{k=1}^{4} \dfrac{\partial N_k(\boldsymbol{\xi})}{\partial \eta} y_k & \sum_{k=1}^{4} \dfrac{\partial N_k(\boldsymbol{\xi})}{\partial \eta} z_k \\ \sum_{k=1}^{4} \dfrac{\partial N_k(\boldsymbol{\xi})}{\partial \zeta} x_k & \sum_{k=1}^{4} \dfrac{\partial N_k(\boldsymbol{\xi})}{\partial \zeta} y_k & \sum_{k=1}^{4} \dfrac{\partial N_k(\boldsymbol{\xi})}{\partial \zeta} z_k \end{bmatrix} \\ &= \begin{bmatrix} -1x_1 + 1x_2 + 0x_3 + 0x_4 & -1y_1 + 1y_2 + 0y_3 + 0y_4 & -1z_1 + 1z_2 + 0z_3 + 0z_4 \\ -1x_1 + 0x_2 + 1x_3 + 0x_4 & -1y_1 + 0y_2 + 1y_3 + 0y_4 & -1z_1 + 0z_2 + 1z_3 + 0z_4 \\ -1x_1 + 0x_2 + 0x_3 + 1x_4 & -1y_1 + 0y_2 + 0y_3 + 1y_4 & -1z_1 + 0z_2 + 0z_3 + 1z_4 \end{bmatrix} \\ &= \begin{bmatrix} x_2 - x_1 & y_2 - y_1 & z_2 - z_1 \\ x_3 - x_1 & y_3 - y_1 & z_3 - z_1 \\ x_4 - x_1 & y_4 - y_1 & z_4 - z_1 \end{bmatrix} \end{aligned} \tag{A.15}$$

which is a constant. Actually the determinant of $\boldsymbol{J}_{NG}$ is related to the volume V_{tet} of the tetrahedral element $(\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, \boldsymbol{x}_4))$ in the global cartesian system

$$\begin{aligned} \det(\boldsymbol{J}_{NG}) = \det\left(\boldsymbol{J}_{NG}^T\right) &= \det\left(\begin{bmatrix} x_2 - x_1 & x_3 - x_1 & x_4 - x_1 \\ y_2 - y_1 & y_3 - y_1 & y_4 - y_1 \\ z_2 - z_1 & z_3 - z_1 & z_4 - z_1 \end{bmatrix}\right) \\ &= \left|(\boldsymbol{x}_4 - \boldsymbol{x}_1) \cdot \left((\boldsymbol{x}_2 - \boldsymbol{x}_1) \times (\boldsymbol{x}_3 - \boldsymbol{x}_1)\right)\right| = 6\ V_{tet} \end{aligned} \tag{A.16}$$

since the absolute value of the triple scalar product $(\boldsymbol{x}_2 - \boldsymbol{x}_1) \times (\boldsymbol{x}_3 - \boldsymbol{x}_1)$ is the volume of a parallelepiped that shares three sides $\boldsymbol{x}_2 - \boldsymbol{x}_1$, $\boldsymbol{x}_3 - \boldsymbol{x}_1$, $\boldsymbol{x}_4 - \boldsymbol{x}_1$ with the tetrahedron.

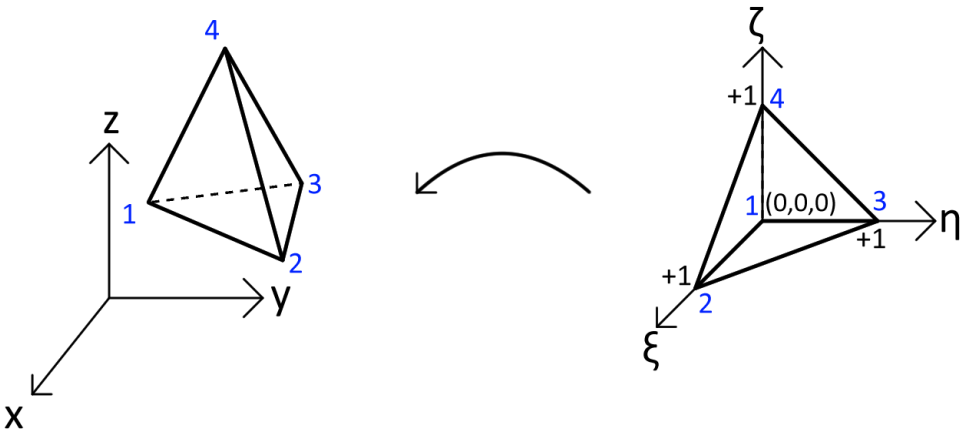


Figure A.5: Isoparametric tetrahedral element with 4 nodes.

A.2 Integration

A main advantage of isoparametric elements is the ease of integration. The following equations will focus on scalar functions, but integrating vector and matrix functions follows the same logic. Any integral of a function $f(\boldsymbol{x}) = f(\boldsymbol{x}(\boldsymbol{\xi}))$ over an element Ω_e, can be converted to the natural system of the element:

$$\begin{aligned} \int_{\Omega_e} f(\boldsymbol{x})\, d\Omega &= \int_{\widetilde{\Omega}_e} f(\boldsymbol{x}(\boldsymbol{\xi}))\ \det(\boldsymbol{J}_{NG}(\boldsymbol{\xi}))\, d\xi d\eta \quad \text{in 2D} \\ \int_{\Omega_e} f(\boldsymbol{x})\, d\Omega &= \int_{\widetilde{\Omega}_e} f(\boldsymbol{x}(\boldsymbol{\xi})) \det(\boldsymbol{J}_{NG}(\boldsymbol{\xi}))\, d\xi d\eta d\zeta \quad \text{in 3D} \end{aligned} \tag{A.17}$$

where $\widetilde{\Omega}_e$ is the surface (in 2D problems) or space (in 3D problems) occupied by the isoparametric element in its natural coordinate system and $\det(\boldsymbol{J}_{NG}(\boldsymbol{\xi}))$ is the determinant of the Jacobian matrix of the isoparametric mapping. These integrals can be calculated by using numerical integration and specifically, Gaussian-Legendre quadrature, which guarantees exact integration of polynomials with the minimum number of required integration points:

$$\int_{\Omega} f(\boldsymbol{x})\, d\Omega = \sum_{p=1}^{n_{GP}} f(\boldsymbol{\xi}_p)\ \det(\boldsymbol{J}_{NG}(\boldsymbol{\xi}_p))\ w_p \tag{A.18}$$

where $\boldsymbol{\xi}_p$ are the coordinates in the natural system of certain *integration points*, where the function is evaluated. Each of the n_{GP} integration points corresponds to a weight coefficient w_p. In order to exactly integrate polynomials of higher order, Gauss-Legendre needs to increase the number of integration points. The coordinates and weights for integrating over the various elements used in this dissertation are listed in tables A.3 - A.7.

Points	ξ	weight
1	0.0	2.0
2	-0.5773502691896257	1.0
	+0.5773502691896257	1.0
3	-0.7745966692414834	0.5555555555555556
	0.0	0.8888888888888889
	+0.7745966692414834	0.5555555555555556
4	-0.8611363115940526	0.3478548451374538
	-0.3399810435848563	0.6521451548625461
	+0.3399810435848563	0.6521451548625461
	+0.8611363115940526	0.3478548451374538

Table A.3: Gauss-Legendre quadrature for 1D elements.

Points	ξ	η	weight
1	0.0	0.0	2.0
4	-0.5773502691896257	-0.5773502691896257	1.0
	-0.5773502691896257	+0.5773502691896257	1.0
	+0.5773502691896257	-0.5773502691896257	1.0
	+0.5773502691896257	+0.5773502691896257	1.0
9	-0.7745966692414834	-0.7745966692414834	0.308641975308642
	-0.7745966692414834	0.0	0.4938271604938272
	-0.7745966692414834	+0.7745966692414834	0.308641975308642
	0.0	-0.7745966692414834	0.4938271604938272
	0.0	0.0	0.7901234567901235
	0.0	+0.7745966692414834	0.4938271604938272
	+0.7745966692414834	-0.7745966692414834	0.30864197530864
	+0.7745966692414834	0.0	0.4938271604938272
	+0.7745966692414834	+0.7745966692414834	0.30864197530864

Table A.4: Gauss-Legendre quadrature for quadrilateral elements.

Points	ξ	η	weight
1	0.3333333333333333	0.3333333333333333	0.5
3	0.6666666666666667	0.1666666666666667	0.1666666666666667
	0.1666666666666667	0.6666666666666667	0.1666666666666667
	0.1666666666666667	0.1666666666666667	0.1666666666666667
4	0.3333333333333333	0.3333333333333333	-0.28125
	0.2	0.2	0.2604166666666667
	0.2	0.6	0.2604166666666667
	0.6	0.2	0.2604166666666667
6	0.44594849091597	0.44594849091597	0.111690794839005
	0.44594849091597	0.10810301816807	0.111690794839005
	0.10810301816807	0.44594849091597	0.111690794839005
	0.09157621350977	0.09157621350977	0.054975871827660
	0.09157621350977	0.81684757298046	0.054975871827660
	0.81684757298046	0.09157621350977	0.054975871827660

Table A.5: Gauss-Legendre quadrature for triangular elements.

Points	ξ	η	ζ	weight
1	0.0	0.0	0.0	2.0
8	-0.5773502691896257	-0.5773502691896257	-0.5773502691896257	1.0
	-0.5773502691896257	-0.5773502691896257	+0.5773502691896257	1.0
	-0.5773502691896257	+0.5773502691896257	-0.5773502691896257	1.0
	-0.5773502691896257	+0.5773502691896257	+0.5773502691896257	1.0
	+0.5773502691896257	-0.5773502691896257	-0.5773502691896257	1.0
	+0.5773502691896257	-0.5773502691896257	+0.5773502691896257	1.0
	+0.5773502691896257	+0.5773502691896257	-0.5773502691896257	1.0
	+0.5773502691896257	+0.5773502691896257	+0.5773502691896257	1.0
27	-0.7745966692414834	-0.7745966692414834	-0.7745966692414834	0.1714677640603567
	-0.7745966692414834	-0.7745966692414834	0.0	0.2743484224965702
	-0.7745966692414834	-0.7745966692414834	+0.7745966692414834	0.1714677640603567
	-0.7745966692414834	0.0	-0.7745966692414834	0.2743484224965702
	-0.7745966692414834	0.0	0.0	0.4389574759945131
	-0.7745966692414834	0.0	+0.7745966692414834	0.2743484224965702
	-0.7745966692414834	+0.7745966692414834	-0.7745966692414834	0.1714677640603567
	-0.7745966692414834	+0.7745966692414834	0.0	0.2743484224965702
	-0.7745966692414834	+0.7745966692414834	+0.7745966692414834	0.1714677640603567
	0.0	-0.7745966692414834	-0.7745966692414834	0.2743484224965702
	0.0	-0.7745966692414834	0.0	0.4389574759945131
	0.0	-0.7745966692414834	+0.7745966692414834	0.2743484224965702
	0.0	0.0	-0.7745966692414834	0.4389574759945131
	0.0	0.0	0.0	0.7023319615912209
	0.0	0.0	+0.7745966692414834	0.4389574759945131
	0.0	+0.7745966692414834	-0.7745966692414834	0.2743484224965702
	0.0	+0.7745966692414834	0.0	0.4389574759945131
	0.0	+0.7745966692414834	+0.7745966692414834	0.2743484224965702
	+0.7745966692414834	-0.7745966692414834	-0.7745966692414834	0.1714677640603567
	+0.7745966692414834	-0.7745966692414834	0.0	0.2743484224965702
	+0.7745966692414834	-0.7745966692414834	+0.7745966692414834	0.1714677640603567
	+0.7745966692414834	0.0	-0.7745966692414834	0.2743484224965702
	+0.7745966692414834	0.0	0.0	0.4389574759945131
	+0.7745966692414834	0.0	+0.7745966692414834	0.2743484224965702
	+0.7745966692414834	+0.7745966692414834	-0.7745966692414834	0.1714677640603567
	+0.7745966692414834	+0.7745966692414834	0.0	0.2743484224965702
	+0.7745966692414834	+0.7745966692414834	+0.7745966692414834	0.1714677640603567

Table A.6: Gauss-Legendre quadrature for hexahedral elements.

Points	ξ	η	ζ	weight
1	0.25	0.25	0.25	0.166666666666667
4	0.138196601125011	0.138196601125011	0.138196601125011	0.041666666666667
	0.138196601125011	0.138196601125011	0.585410196624969	0.041666666666667
	0.138196601125011	0.585410196624969	0.138196601125011	0.041666666666667
	0.585410196624969	0.138196601125011	0.138196601125011	0.041666666666667
5	0.250000000000000	0.250000000000000	0.250000000000000	-0.133333333333333
	0.166666666666667	0.166666666666667	0.166666666666667	0.075000000000000
	0.166666666666667	0.166666666666667	0.500000000000000	0.075000000000000
	0.166666666666667	0.500000000000000	0.166666666666667	0.075000000000000
	0.500000000000000	0.166666666666667	0.166666666666667	0.075000000000000
15	0.250000000000000	0.250000000000000	0.250000000000000	0.019753086419753
	0.319793627829630	0.319793627829630	0.319793627829630	0.011511367871045
	0.319793627829630	0.319793627829630	0.040619116511110	0.011511367871045
	0.319793627829630	0.040619116511110	0.319793627829630	0.011511367871045
	0.040619116511110	0.319793627829630	0.319793627829630	0.011511367871045
	0.091971078052723	0.091971078052723	0.091971078052723	0.011989513963170
	0.091971078052723	0.091971078052723	0.724086765841831	0.011989513963170
	0.091971078052723	0.724086765841831	0.091971078052723	0.011989513963170
	0.724086765841831	0.091971078052723	0.091971078052723	0.011989513963170
	0.056350832689629	0.056350832689629	0.443649167310371	0.008818342151675
	0.056350832689629	0.443649167310371	0.056350832689629	0.008818342151675
	0.443649167310371	0.056350832689629	0.056350832689629	0.008818342151675
	0.056350832689629	0.443649167310371	0.443649167310371	0.008818342151675
	0.443649167310371	0.056350832689629	0.443649167310371	0.008818342151675
	0.443649167310371	0.443649167310371	0.056350832689629	0.008818342151675

Table A.7: Gauss-Legendre quadrature for tetrahedral elements.