

POLITECNICO DI MILANO

Scuola di Ingegneria Industriale e dell'Informazione

Corso di Laurea Magistrale in Ingegneria Aeronautica - Strutture



Modellazione strutturale multi-livello utilizzando un codice FEM commerciale

Relatore: Prof. MARCO MORANDINI

Correlatore: Prof. RICCARDO VESCOVINI

Tesi di Laurea di:
GIANLUCA FRANZINI
Matr. 841873

Anno Accademico 2016-2017

Sommario

La tesi tratta lo sviluppo di una metodologia di analisi strutturale multi-livello utilizzando un codice ad elementi finiti commerciale.

La tecnica global/local sviluppata si propone di studiare il comportamento complessivo della struttura, tenendo conto di fenomeni non-lineari che si verificano in una regione ristretta, attraverso l'utilizzo di due modelli separati con differenti livelli di dettaglio: (1) modello globale dell'intera struttura, relativamente semplice, (2) modello locale contenente la descrizione dettagliata del materiale della regione considerata.

La presenza del livello locale consente di aggiungere alla modellazione dei dettagli che nel livello superiore non sono presenti ed è in grado di aggiornare gli elementi globali per tenere conto degli effetti locali.

Le analisi sui rispettivi modelli sono eseguite separatamente e consecutivamente, realizzando un processo iterativo di scambio di informazioni nelle due direzioni.

Il codice a elementi finiti utilizzato è Abaqus. In particolare, per la realizzazione della procedura global/local si è fatto utilizzo delle User Subroutine.

Il metodo è stato applicato ad una piastra forata, attraverso la modellazione manuale dei 2 modelli: il modello locale contiene la descrizione dei fenomeni plastici nella regione situata in prossimità della discontinuità geometrica (plasticità di von Mises), invece la restante regione è considerato elastica.

La metodologia rappresenta un primo passo verso un'implementazione global-local più completa in grado di modellare automaticamente le regioni critiche della struttura ed essere utilizzata in applicazioni complesse, per lo studio di diversi problemi, per esempio il danneggiamento progressivo, la plasticità, le fratture.

Parole chiave: Metodo Global/Local; plasticità di von Mises; analisi ad elementi finiti; User Subroutine

Abstract

The thesis deals with the development of a multi-level structural analysis methodology using a commercial FEM code.

The global/local method is proposed to investigate the overall behaviour of the structure as well as local nonlinearities occur within a small area, using two separated models with different level fidelity: (1) relatively simple global model of the entire structure, (2) local model with material's detailed description.

The local level allows to add the details that are not included in the higher level and to update the global elements due to local effects.

The analysis of the models are performed separately and consequently, in an iterative process, with two-way information exchange between the models.

The FE software used is Abaqus. In particular, the global/local procedure is realized by the coding of the User Subroutine.

The method is applied to a holed plate, through the manual modelling of the two models: the local model contains the plastic effects due to the geometric discontinuity (von Mises plasticity), instead the remaining part of the structure is considered elastic.

The methodology is a first step to the development of a complete global-local method that allow the automatic modelling of the structure's critical areas in complex applications to analyze different problems, for example the progressive damages, plasticity, fractures.

Keywords: Global/Local method; von Mises plasticity; finite element analysis; User Subroutine

Ringraziamenti

Desidero ringraziare i miei supervisor prof. Marco Morandini e prof. Riccardo Vescovini, che mi hanno assistito, spronato durante tutta l'attività di Tesi. Hanno saputo mettere a disposizione la loro esperienza e disponibilità in questo progetto, guidandomi durante le fasi di lavoro, attraverso consigli e critiche costruttive che ne farò tesoro per il futuro.

Ringrazio i miei genitori che mi hanno sostenuto in questi anni, non facendomi mai mancare il loro supporto con la loro presenza o con una parola di incoraggiamento.

Ringrazio i miei nonni che in questo momento saranno fieri di me.

Ringrazio tutti i compagni di università che negli anni hanno vissuto con me quest'esperienza di studio.

Ringrazio i miei amici, che rappresentano una parte importante della mia vita.

Indice

Elenco delle figure	ix
Elenco delle tabelle	xi
1 Introduzione	1
1.1 Metodi one-way	2
1.2 Metodi two-way	3
1.3 Approcci esistenti	5
1.4 Obiettivo della tesi	6
1.5 Struttura della tesi	6
2 Le Subroutine di Abaqus	7
2.1 Descrizione delle Subroutine:	
UMAT, DISP, URDIFL	8
2.1.1 UMAT	8
2.1.2 DISP	8
2.1.3 URDFIL	9
2.2 Funzionalità esterne: scambio dati server-client	9
3 Sviluppo del metodo Global/Local	11
3.1 Il metodo tramite schema a blocchi	12
3.2 Limitazioni di utilizzo	15
3.3 Implementazione del metodo	15
3.3.1 Introduzione del modello costitutivo elastico: UMAT	15
3.3.2 Spostamenti imposti: DISP	17
3.3.3 Lettura dei risultati: URDFIL	19
3.3.4 Modello Globale-Locale: piastra quadrata isotropa	19
3.3.5 Modello Globale-Locale: piastra forata isotropa	20
4 Risultati	25
4.1 Validazione delle procedure del metodo	25
4.1.1 Validazione DISP e URDFIL	25
4.1.2 Validazione modello di piastra quadrata isotropa	28
4.2 Validazione del modello elasto-plastico	28
5 Conclusioni	35
5.1 Prospettive future	35
A Implementazione Subroutine	37
A.1 Struttura generale Subroutine	37
A.2 Struttura generale scambio dati server-client	38
A.3 UMAT: modello costitutivo elastico	39
A.4 DISP: imposizione spostamenti nodali	40
A.5 URDFIL: lettura dei risultati	40
A.6 Modello globale-locale: piastra quadrata isotropa	41
A.7 Modello globale-locale: piastra forata isotropa modello elasto-plastico	45
Bibliografia	63

Elenco delle figure

1.1	Procedura iterativa di correzione globale non-intrusive.	3
1.2	Schema della procedura two-way descritta in [15].	4
2.1	Scambio di informazioni utilizzando il modello di messaggio REQ/REP.	10
3.1	Scambio di informazioni tra il modello globale e il modello locale.	11
3.2	Schema a blocchi del metodo global/local	13
3.3	Funzionalità richieste dalle analisi nei 2 livelli della simulazione.	14
3.4	Modellazione a 1 e 2 elementi	17
3.5	Rappresentazione piastra quadrata isotropa.	18
3.6	Modellazione global/local piastra quadrata isotropa	20
3.7	Modellazione global-local piastra forata.	20
3.8	Curva SIGMA-EPLAS con legge di incrudimento lineare a tratti.	22
3.9	Return mapping con incrudimento isotropo.	24
4.1	Confronto risultati, sforzo S_x , ELASTIC-DISP	26
4.2	Nodi utilizzati per il confronto degli spostamenti.	26
4.3	Risultati piastra quadrata, sforzo S_x , global/local	28
4.4	Risultati piastra forata, deformazione S_x , global/local	30
4.5	Risultati piastra forata PLASTIC, sforzo S_x	31
4.6	Risultati piastra forata, deformazione S_y , global/local	31
4.7	Risultati piastra forata PLASTIC, sforzo S_y	31
4.8	Risultati piastra forata, deformazione S_{xy} , global/local	32
4.9	Risultati piastra forata PLASTIC, sforzo S_{xy}	32
4.10	Risultati piastra forata, deformazione E_x , global/local	32
4.11	Risultati piastra forata PLASTIC, deformazione E_x	33
4.12	Risultati piastra forata, deformazione E_y , global/local	33
4.13	Risultati piastra forata PLASTIC, deformazione E_y	33
4.14	Risultati piastra forata, deformazione E_{xy} , global/local	34
4.15	Risultati piastra forata PLASTIC, deformazione E_{xy}	34
4.16	Risultati piastra forata, PEEQ	34

Elenco delle tabelle

3.1	Associazione Subroutine con le funzionalità richieste dal metodo.	15
3.2	Variabili in ingresso e uscita UMAT	16
3.3	Variabili in ingresso e uscita DISP	17
3.4	Proprietà elastiche di un acciaio dolce.	22
3.5	Tabella dei valori SIGMA e EPLAS di incrudimento.	22
4.1	File utilizzati nel comando delle analisi.	25
4.2	Tabella spostamenti nodali Ux modello ELASTIC.	27
4.3	Tabella dei risultati ottenuti tramite URDFIL e mostrati a schermo	27
4.4	File e procedure della simulazione global/local piastra quadrata isotropa.	28
4.5	File.o e procedure associate.	29
4.6	Tempi di analisi.	29
4.7	Ordine risultati piastra forata.	29

Capitolo 1

Introduzione

Gli ultimi decenni hanno visto un largo sviluppo dei metodi di analisi global-local, spinti dall'esigenza dei progettisti di ridurre i tempi computazionali e i costi delle simulazioni numeriche, caratterizzate da gradi di libertà sempre più elevati.

L'idea alla base di questi metodi è quella di coniugare l'utilizzo di 2 modelli per la simulazione numerica di una struttura, caratterizzati da un differente livello di dettaglio:

1. **modello globale:** costituito da una modellazione a basso costo computazionale, in grado di descriverne il comportamento complessivo.
2. **modello locale:** contenente informazioni più dettagliate, in grado di descrivere più accuratamente una determinata regione.

Questa metodologia è in grado di risolvere i problemi strutturali presenti a livello industriale, soprattutto in campo aeronautico, dove i fenomeni non-lineari si verificano in una regione ristretta, richiedente quindi un maggior livello di dettaglio.

Tale regione può risultare nota a priori in fase preliminare oppure essere riconosciuta in seguito, attraverso una modellazione automatizzata, tramite la verifica di determinate condizioni durante l'analisi.

Come descritto in [1], alcuni elementi comuni a questi problemi strutturali sono:

- **concentrazione di sforzo**
- **stato di sforzo 3D**
- **grandi deformazioni**
- **discontinuità locali (fori, intagli, fibre discontinue,...):**
- **non-linearità del materiale (plasticità, fratture,...)**

Questi elementi devono essere tenuti in considerazione durante la definizione dell'analisi, alcuni di essi possono non essere inclusi nel modello globale, ma sono sicuramente presenti nella modellazione locale.

Alcuni problemi che possono essere studiati attraverso questa metodologia sono:

- **analisi accurata di una regione locale:** in termini di spostamenti, deformazioni, sforzi, in presenza di discontinuità geometriche (come fori, intagli,...) che possono generare concentrazioni di sforzo.

La simulazione prevede la realizzazione del modello locale in prossimità della discontinuità, tramite un progressivo incremento degli elementi, rispetto alla restante regione, per ottenere una soluzione più regolare.

- **analisi dello stato di sforzo 3D:**

La simulazione prevede la transizione da un modello globale, che può essere modellato a piastre, ad un modello solido locale, tramite un accoppiamento opportuno delle condizioni al contorno.

- **analisi locale con descrizione dettagliata del comportamento non-lineare del materiale:**

La simulazione prevede l'analisi di fenomeni non-lineari legati alla natura del materiale, tramite un modello costitutivo locale dettagliato.

Tale metodologia permette, per esempio, lo studio elasto-plastico o l'introduzione del danneggiamento.

La lista citata rappresenta solo una parte delle possibili analisi che possono essere eseguite tramite questi metodi, infatti la loro capacità e le possibili applicazioni sono in continuo sviluppo, per esempio, nello studio di giunzioni bullonate o inchiodate, problemi di impatto tra 2 corpi, concentrazione di sforzi sulle superfici libere nei laminati, in strutture contenenti fratture.

Queste metodologie possono essere classificate in funzione delle capacità di comunicazione e scambio di informazioni tra i 2 livelli della struttura, come introdotto in [15]:

- **metodi one-way** la comunicazione avviene in una sola direzione:

globale \rightarrow locale

oppure

locale \rightarrow globale.

- **metodi two-way** lo scambio di informazioni avviene in entrambe le direzioni:

globale \rightarrow locale

+

locale \rightarrow globale.

Nella prima categoria sono contenuti tutti i metodi nei quali l'analisi segue una sola direzione, in maniera sequenziale la simulazione procede dal modello globale al modello locale o viceversa.

Nella seconda categoria invece sono contenuti tutti i metodi nei quali l'analisi segue un processo iterativo, di scambio di informazioni reciproco tra i due modelli, permettendo un'interazione durante la simulazione.

In questo modo lo scambio di informazioni avviene in entrambe le direzioni permettendo l'interazione tra gli effetti globali e locali.

1.1 Metodi one-way

In questa prima categoria rientrano i metodi che utilizzando la cosiddetta tecnica dello zooming che, come esposto da Noor [2], prevede il passaggio di informazioni in una direzione, da un modello globale grossolano, ad un modello locale fine.

Questa tecnica è un particolare tipo di sotto-strutturazione, che prevede la suddivisione della struttura in regioni più piccole e distinte interfacciate tra loro.

Tale possibilità viene sfruttata con l'ottica di differenziare la regione globale dalla regione localizzata, utilizzando una modellazione più o meno dettagliata a seconda della regione considerata.

Le informazioni scambiate tra i modelli interessano gli spostamenti nodali globali che costituiscono le condizioni da imporre ai nodi al contorno della regione locale.

Tali nodi d'interfaccia possono essere coincidenti oppure richiedere l'interpolazione dei valori di spostamento.

Le analisi sui rispettivi modelli seguono un andamento sequenziale: una prima analisi è volta

alla determinazione degli spostamenti globali; successivamente i dati sono inviati al modello locale per effettuare la propria analisi e determinare i risultati.

Questa tecnica può essere utilizzata in caso di forte gradiente di sforzo locale, legato per esempio a delle discontinuità geometriche, per ottenere una descrizione più accurata dello stato di sforzo, come analizzato in [3].

Uno studio sullo stato di sforzo 3D di un laminato con un foro centrale è presentato in [4], attraverso l'analisi di un modello globale realizzato a piastre, interfacciato ad un modello solido locale per la descrizione della regione in prossimità del foro.

Questi metodi one-way presentano delle forti limitazioni, scambiando le informazioni in una sola direzione non permettono una corretta simulazione dell'influenza dei fenomeni locali sul modello globale e viceversa.

Tale mancanza può comportare degli errori legati alla definizione delle condizioni al contorno, che poi ovviamente si ripercuotono sull'analisi locale successiva.

Per superare tale situazione sono state introdotte delle metodologie che utilizzano delle correzioni globali, per tenere conto dei fenomeni locali, come descritto da Withcomb in [5], Mao in [6], attraverso una procedura iterativa.

Tra i metodi che utilizzano questa tecnica di correzione globale ci sono i metodi "non-intrusive" [7], la cui tecnica di funzionamento è brevemente riassunta in figura 1.1:

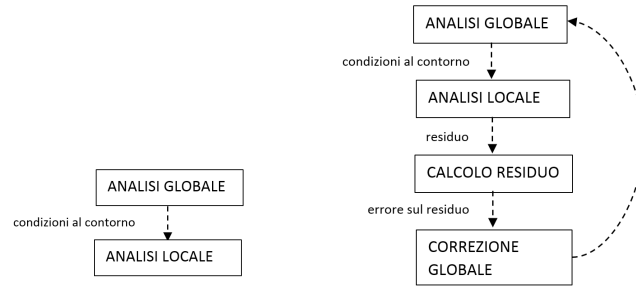


Figura 1.1: Metodo one-way (a sinistra); metodo one-way con correzione non-intrusiva (a destra).

Come osservato, il processo iterativo prevede una prima fase comune ai metodi one-way tradizionali, seguita dal calcolo del residuo all'interfaccia tra i 2 modelli.

La definizione di un errore permette di verificare quando l'analisi raggiunge una certa tolleranza terminando il processo iterativo di applicazione della correzione globale come carico aggiuntivo al contorno.

Tale metodo è utilizzato in [8], con lo studio di una struttura soggetta a fenomeni plastici locali, in [9] con l'accoppiamento 2D-3D per lo studio del comportamento meccanico di grandi strutture costituite dall'assemblaggio di pannelli sottili, con analisi dello stato di sforzo tridimensionale nelle regioni locali.

In [10], si presenta lo studio del comportamento di un laminato in materiale composito, con descrizione del danneggiamento del materiale nei vari livelli, prima del collasso.

Un aggiornamento dei metodi non-intrusivi è descritto in [11] con i recenti sviluppi negli algoritmi della metodologia.

1.2 Metodi two-way

Nella seconda categoria troviamo i metodi two-way che prevedono lo scambio di informazioni reciproco tra i 2 modelli, tra essi ci sono i metodi multiscale, che prevedono una separazione tra i livelli dettato da differenti scale dimensionali, sia dal punto di vista strutturale che fenomenologico, come avviene tipicamente nei materiali compositi caratterizzati da un materiale eterogeneo costituito da scale dimensionali significativamente differenti.

Al livello più alto il laminato, fino ad arrivare alle scale più piccole contenenti le fasi costituenti (fibre, matrice).

Ciascuna scala presenta i propri meccanismi di danno che vanno tenuti in considerazione, attraverso l'utilizzo di più modelli, in grado di descriverne il comportamento e trasferire le informazioni alla scala successiva.

In [12], sono presentate le caratteristiche principali di questi metodi, basati sull'utilizzo degli RVE (representative volume element) che sono gli elementi rappresentativi del comportamento eterogeneo al livello locale.

L'analisi segue 2 processi: (1) la localizzazione che introduce le condizioni al contorno, trasmesse dalla scala più alta a quella in successione più piccola, su ogni RVE, (2) l'omogeneizzazione che permette di ottenere un comportamento equivalente di ciascun RVE, per l'utilizzo nel livello superiore.

Un'applicazione di tale metodologia è descritta in [13], con l'analisi del comportamento meccanico di un osso umano, sottoposto alle condizioni di esercizio, realizzato attraverso 2 livelli: livello globale nel quale ciascuna struttura costituente l'osso è considerata omogenea e il livello locale per la descrizione eterogenea del tessuto osseo tramite RVE.

Il comportamento al livello globale è ottenuto, inviando le condizioni al contorno su ogni RVE, che incorporano la descrizione dei danneggiamenti, attraverso un processo di omogeneizzazione.

In [14] è proposto un metodo multiscale per la simulazione di fratture all'interno di una struttura con particolare interesse all'interazione tra micro e macro fratture.

Tra i metodi two-way rientrano anche altre metodologie dove la suddivisione dei modelli non utilizza una separazione di scala, ma è dettata dalle esigenze della simulazione.

Un esempio è mostrato in [15], con l'applicazione di un nuovo metodo global-local per lo studio di pannelli rinforzati (tipici delle fusoliere).

Questo metodo prevede lo studio del danneggiamento progressivo del materiale, che può avvenire sui pannelli o sulle nervature, attraverso la modifica delle proprietà elastiche del modello globale, utilizzando dei test sul modello locale per la determinazione le proprietà equivalenti.

Il modello locale contiene la descrizione dettagliata dei danneggiamenti che si possono sviluppare nel materiale.

Lo schema di funzionamento della metodologia contenuta in [15] è rappresentato in figura 1.2.

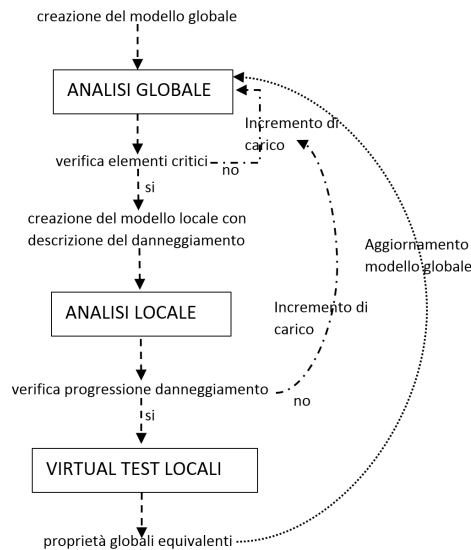


Figura 1.2: Schema della procedura two-way descritta in [15].

Il funzionamento del metodo, mostrato nella figura 1.2, prevede un'analisi globale costituita da elementi di piastra, seguita dalla verifica di eventuali elementi critici tramite criteri di resistenza (come per esempio Hashin) e, in funzione dell'esito, la creazione di un modello locale solido.

L'esito negativo delle verifiche determina l'incremento del carico applicato.

La verifica positiva comporta la creazione del modello locale e la successiva analisi e i virtual test, che vengono effettuati per l'aggiornamento delle proprietà degli elementi globali risultati critici, che possono andare incontro a danneggiamenti e/o rotture.

I test prevedono carichi uniformi sugli elementi solidi locali.

La fase di aggiornamento del modello globale richiede alcune iterazioni per verificare che la modifica apportata non comporti un'ulteriore progressione dei danneggiamenti.

In [16] è descritta la continuazione del lavoro precedente con uno studio mirato sul miglioramento della definizione delle dimensioni del modello locale, in particolare per pannelli con più di un rinforzo, al fine di ottenere una simulazione più vicina alla realtà.

Un'ulteriore applicazione two-way la ritroviamo in [17], con l'impiego della PFA (Progressive Failure Analysis) per lo studio del comportamento di una giunzione discontinua, tra 2 pannelli.

1.3 Approcci esistenti

Tra gli approcci global-local finora introdotti solo alcuni permettono di ottenere una modellazione multi-livello utilizzabile attraverso un codice FEM commerciale.

I metodi multiscale, descritti in [13] e [14], richiedono la modifica del codice sorgente, questo non permette l'applicazione immediata tramite un software ad elementi finiti standard.

Inoltre le analisi sui rispettivi modelli vengono eseguite in parallelo richiedendo la definizione delle regioni locali, tramite RVE, in fase preliminare, non adattandosi alla formazione di nuove aree critiche.

I metodi 'non-intrusive', con la metodologia descritta in [7], consentono tale utilizzo, la loro caratteristica è quella di superare le limitazioni delle analisi standard attraverso un'analisi computazionalmente efficiente ottenuta interfacciando 2 livelli tramite una correzione globale.

Una modellazione multi-livello completa dove è possibile aggiornare il livello globale per tenere conto degli effetti locali attraverso un processo iterativo e sequenziale delle analisi è sviluppata in [15], essa prevede la comunicazione tra i modelli costitutivi dei rispettivi livelli. Questa metodologia consente l'utilizzo attraverso un codice FEM tradizionale ed è specifica per l'utilizzo in strutture in materiale composito con la simulazione di danneggiamenti progressivi fino a rottura.

Come in altre metodologie i 2 livelli sono interfacciati tramite gli spostamenti al contorno da imporre al livello locale per tenere conto degli effetti globali.

In questo approccio le analisi sono eseguite in successione e questo consente la definizione delle regioni critiche successivamente all'analisi globale, permettendo anche l'automatizzazione della procedura.

Manca tuttavia in letteratura una metodologia generale in grado di implementare una modellazione multi-livello attraverso un software commerciale, che permetta all'utente di introdurre un modello costitutivo locale, in funzione delle proprie esigenze, per ottenere la descrizione degli effetti locali richiesti dalla simulazione, per esempio fenomeni plastici, fratture, danneggiamenti.

1.4 Obiettivo della tesi

L'obiettivo di questo lavoro di tesi riguarda l'implementazione di una metodologia generale di analisi strutturale multi-livello attraverso l'utilizzo del software ad elementi finiti Abaqus, in grado di simulare l'interazione tra gli effetti globali e i fenomeni locali attraverso lo scambio di informazioni tra i 2 livelli della struttura.

Gli effetti globali interagiscono con il livello locale tramite gli spostamenti al contorno, l'interazione inversa invece è ottenuta aggiornando le proprietà degli elementi globali.

Questa metodologia presenta caratteristiche generali permettendo di modellare il comportamento della regione locale in funzione delle richieste dell'utente.

Le funzionalità dell'analisi sono ottenute attraverso la programmazione delle User Subroutine di Abaqus, in linguaggio Fortran.

La metodologia presenta molte parti da sviluppare manualmente, mostrando alcune limitazioni ma altrettante potenzialità di funzionamento e di aggiornamenti futuri.

1.5 Struttura della tesi

L'organizzazione della struttura della tesi segue un andamento progressivo degli argomenti, con l'obiettivo di addentrarsi nella realizzazione del metodo, attraverso una conoscenza graduale di tutti gli strumenti utili alla sua comprensione.

Concluso questo primo capitolo introduttivo, segue il capitolo 2 che introduce le User Subroutine di Abaqus, che sono fondamentali per l'implementazione del metodo.

Troviamo in questa sezione la descrizione generale e le funzioni di ciascuna procedura che trova applicazione nella simulazione.

Il capitolo 3 invece è dedicato alla descrizione e implementazione del metodo global/local, con attenzione alla metodologia di analisi e ai passaggi seguiti per la realizzazione.

Esso contiene uno schema a blocchi, per una migliore comprensione d'insieme della simulazione.

Segue il capitolo 4, dedicato ai risultati, dove il metodo viene validato utilizzando opportune analisi comparative, con relativa descrizione.

In questa fase vengono sfruttate opportune rappresentazione grafiche, tabelle e modelli per ottenere la comparazione.

Infine il capitolo 5, dedicato alle conclusioni, con una riflessione sui possibili miglioramenti e propositi per il futuro, con la prospettiva di ampliamento e automazione del metodo.

Capitolo 2

Le Subroutine di Abaqus

Questo capitolo è dedicato alla descrizione delle User Subroutine disponibili in Abaqus, contenute nella sezione "Abaqus User Subroutine Reference Guide" del manuale [18].

L'utente può modificare le analisi standard disponibili, al fine di ottenere un metodologia di calcolo specifico per le proprie esigenze, attraverso l'utilizzo e la programmazione nella simulazione, in linguaggio Fortran, delle User Subroutine.

La loro funzione è quella di restituire i risultati richiesti (output) tramite l'elaborazione degli ingressi noti (input), nelle operazioni programmate.

Queste procedure possono essere chiamate all'interno di altre procedure o programmi attraverso l'istruzione "call nome_subroutine(lista_argomenti)", in modo da suddividere le azioni previste: il nome_subroutine costituisce il nominativo della procedura interpellata e la lista_argomenti contiene tutte le variabili in entrata ed in uscita dalla procedura stessa.

Nella programmazione il linguaggio Fortran prevede la possibilità di utilizzo di 2 tipologie di istruzioni:

- **eseguibili:** istruzioni che il calcolatore deve eseguire (somma, moltiplicazione,...).
- **non eseguibili:** istruzioni che il calcolatore non esegue ma che sono fondamentali per il corretto completamento del programma (dichiarazione delle variabili, tipologie e dimensioni).

La struttura generale di una Subroutine è costituita da 3 parte fondamentali, come si può osservare nell'appendice A.1-Struttura generale subroutine, contenente anche un semplice esempio di una procedura che esegue la somma tra 2 numeri (calcolo_somma.f):

- **parte dichiarativa:** si trova all'inizio della procedura e contiene tutte le istruzioni non eseguibili, necessarie a dichiarare il nome del procedura e le variabili utilizzate, che come descritto in precedenza costituiscono gli input e output.

Le variabili sono dichiarate formalmente nella lista_argomenti e sono seguite dalla descrizione dettagliata delle rispettive tipologie e dimensioni.

Nell'esempio calcolo_somma.f le variabili utilizzate sono numero_1, numero_2 e somma, dichiarate come scalari reali.

Le prime 2 variabili sono input necessari della subroutine somma che viene interpellata per elaborare il risultato.

- **parte esecutiva:** contenente tutte le istruzioni eseguibili, che complessivamente determinano le funzioni della procedura.

In calcolo_somma.f la parte esecutiva contiene l'istruzione di chiamata alla subroutine somma che esegue l'operazione di addizione richiesta e il print della somma ottenuta.

- **parte conclusiva:** contenente l'istruzione di chiusura del programma.

È importante sottolineare che le variabili durante la chiamata possono avere nominativi differenti all'interno della propria procedura ma identica tipologia e dimensione, mantenendo la dovuta coerenza nelle dichiarazioni e utilizzo.

2.1 Descrizione delle Subroutine: UMAT, DISP, URDIFL

Segue ora la descrizione delle User Subroutine fondamentali per il completamento del metodo global/local, focalizzandosi sulle caratteristiche e sulle funzioni specifiche.

Il relativo ruolo all'interno della procedura di analisi è rimandato al capitolo 3.

Il codice di ciascuna Subroutine è invece contenuto nelle sottosezioni dell'appendice A.

2.1.1 UMAT

La Subroutine UMAT è dedicata alla definizione del modello costitutivo del materiale.

Il legame costitutivo rappresenta la relazione matematica tra gli sforzi e le deformazioni ed è una delle 3 relazioni importanti per la formulazione del problema strutturale, insieme alle relazioni cinematiche e alle equazioni di equilibrio.

Viene chiamato in causa in tutti i punti di integrazione di ciascun elemento della modellazione, rientrando nel calcolo della soluzione strutturale.

La programmazione di questa Subroutine passa attraverso la definizione di 3 elementi fondamentali:

- **Vettore di sforzo:** costituito dalle componenti di sforzo.
Questo vettore ha dimensioni diverse a seconda della tipologia di problema da analizzare, per esempio è costituito da 3 componenti in un problema caratterizzato da uno stato di sforzo piano $\{\sigma_{xx}, \sigma_{yy}, \tau_{xy}\}^T$.
Considerando invece uno stato di sforzo completo contiene tutte le 6 componenti: $\{\sigma_{xx}, \sigma_{yy}, \sigma_{zz}, \tau_{yz}, \tau_{zx}, \tau_{xy}\}^T$.
- **Tensore elastico:** costituisce la matrice Jacobiana del materiale $[\frac{\partial \Delta \sigma}{\partial \Delta \epsilon}]$.
Ha dimensioni legate al problema, per esempio 3X3 in un problema piano, dove la componente (I,J) della matrice fornisce la variazione i-esima di sforzo legata alla j-esima variazione di deformazione, tramite la relazione matematica: $\Delta \sigma_I = \frac{\partial \Delta \sigma}{\partial \Delta \epsilon} \cdot \Delta \epsilon_J$
- **Variabili di stato:** possono essere presenti nel modello e in tal caso costituiscono le variabili dipendenti dalla soluzione.
Un esempio di variabili di stato sono le deformazioni plastiche equivalenti ottenute a seguito dello snervamento del materiale, oppure parametri di danneggiamento in materiali compositi.
Esse costituiscono delle variabili importanti nella definizione corrente del materiale.

Abaqus, durante il calcolo della soluzione, utilizza questa Subroutine per ottenere i valori aggiornati delle variabili appena descritte, fornendo in ingresso delle informazioni, che verranno descritte nella fase di implementazione del metodo, necessarie all'elaborazione dei risultati.

Questi 3 elementi vanno quindi programmati opportunamente all'interno della procedura, per l'implementazione del modello costitutivo scelto per la simulazione (per esempio elastico, elasto-plastico, con danneggiamento,...).

2.1.2 DISP

La Subroutine DISP è dedicata alla definizione degli spostamenti nodali imposti.

Essa infatti permette di imporre delle condizioni al contorno tramite spostamenti nodali,

con la possibilità di definire il numero e il valore delle componenti di spostamento (3 nel piano, 6 nello spazio) da applicare ad una determinata lista di nodi. La lista dei nodi va definita nel file di input, attraverso la keyword 'boundary, user' e risulta quindi fissata in fase iniziale.

2.1.3 URDFIL

La Subroutine URDFIL è dedicata alla lettura dei risultati.

Essa permette di accedere al file dei risultati (.fil) e di leggere tutti i valori ottenuti dall'analisi, per ciascun incremento di carico.

La procedura è chiamata ad ogni fine incremento e permette la lettura attraverso l'utilizzo di 2 Utility Routine, POSFIL e DBFILE:

- **POSFIL:** routine che permette di selezionare la posizione di inizio lettura dei file, tramite la definizione dell'identificativo dello step e dell'incremento. Senza nessuna specificazione sulla posizione la lettura procede progressivamente dal primo step per tutti gli incrementi e via dicendo per tutti gli step presenti.
- **DBFILE:** routine che permette la lettura dei risultati. Importante è la definizione della key, ovvero dell'identificativo numerico che è associato con la tipologia di risultato da leggere (come mostrato nella sezione 5.1.2 "Results file output format del manuale" [18]), per esempio l'identificativo 11 restituisce come risultato tutti gli sforzi di ciascun punto di integrazione dei vari elementi. Per il corretto utilizzo di questa Subroutine è fondamentale specificare esattamente nel file di input, sotto la keyword 'output', i risultati che si vogliono leggere tramite la procedura oltre a quelli che si vogliono visualizzare nella fase di post-processing, attraverso *NODE/EL FILE.

2.2 Funzionalità esterne: scambio dati server-client

Questa sezione introduce una funzionalità esterna che verrà richiesta dal metodo global/locale, che prevede lo scambio dati tra 2 programmi (tramite la comunicazione server-client). Questa comunicazione è possibile attraverso l'utilizzo di una binding library di Fortran 77, per ZeroMQ, che permette di utilizzare delle funzioni per il passaggio di informazioni tra 2 procedure Fortran.

ZeroMQ costituisce infatti una communication library basata sul passaggio asincrono dei messaggi tra server e client.

Come definito in [20], sono necessari alcuni file per la comunicazione, che verranno inseriti in una cartella, `f77_zmq`:

- **f77_zmq.h:** file che serve per definire tutte le costanti `zmq_*` utilizzate nelle varie funzionalità della comunicazione. Questo file va importato in ciascun programma interessato dalla comunicazione attraverso l'istruzione:
" include 'f77_zmq/f77_zmq.h' ".
- **libf77zmq.so** oppure **libf77zmq.a:** serve per definire la libreria, necessaria per la compilazione ed esecuzione della comunicazione.

La compilazione del programma è effettuata utilizzando il software `gfortran`, i comandi utilizzati sono inseriti in file `.sh`, per essere eseguiti direttamente nella shell del calcolatore.

Questi file .sh sono descritti in fase di implementazione, quando sarà necessaria la compilazione dei programmi sviluppati.

In appendice A.2 è contenuto l'esempio di 2 programmi, server.f e client.f, che si scambiano informazioni.

Tutti i passaggi necessari per creare ed eseguire il trasferimento dati sono descritti a seguire:

1. **definizione dell'address:** definisce l'indirizzo utilizzato per la creazione del collegamento tra i 2 programmi, importante che la porta scelta per la comunicazione sia identica.
2. **creazione del context:** rappresenta la scatola che durante la comunicazione viene riempita con i dati.
3. **socket:** questa funzione permette di associare ad una variabile, che per esempio in appendice A.2 è rappresentata da "local", il context con la fila dove i dati si incolonnano, utilizzando un opportuno parametro per indicare se si tratta di una fila di invio o di ricezione.
4. **definizione del modello di coppia di dati (invio-ricezione):** In questo lavoro considereremo la coppia REQ/REP, che corrisponde a richiesta/risposta, essa permette la comunicazione nelle 2 direzioni tra i 2 programmi.
La comunicazione utilizzando questa coppia di dati avviene tramite la seguente procedura:

- Il Client richiede i dati, REQ, il Server riceve la richiesta, la elabora e invia la risposta, REP.
- il Client riceve infine la risposta, come mostrato in figura 2.1.

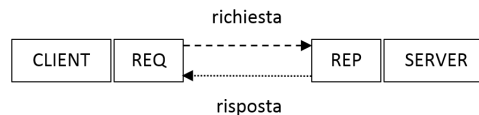


Figura 2.1: Scambio di informazioni utilizzando il modello di messaggio REQ/REP.

5. **creazione del legame tra socket e address:** tramite la variabile 'rc' possiamo collegare la coda di dati all'indirizzo scelto per lo scambio di informazioni.
6. **utilizzo delle funzioni send e recv:** infine si utilizzano le funzionalità che permettono di inviare e ricevere dati, ovvero send e recv.
Queste funzioni devono contenere la variabile local, oltre che la variabile da inviare, definita nella parte dichiarativa, ed il numero di byte da utilizzare per la comunicazione (per esempio 4 per la comunicazione di numeri interi, 8 per la doppia precisione). Importante durante la fase di implementazione dei 2 programmi che un comando send in server.f sia corrisposto ad un comando recv in client.f e viceversa.

Capitolo 3

Sviluppo del metodo Global/Local

La metodologia di analisi global/local viene implementata in questo capitolo attraverso la definizione di tutti i passaggi seguiti per la realizzazione delle funzionalità richieste. La simulazione multi-livello è ottenuta attraverso lo scambio di informazioni tra i 2 modelli che costituiscono la struttura, in una regione critica, come mostrato dalla figura 3.1.

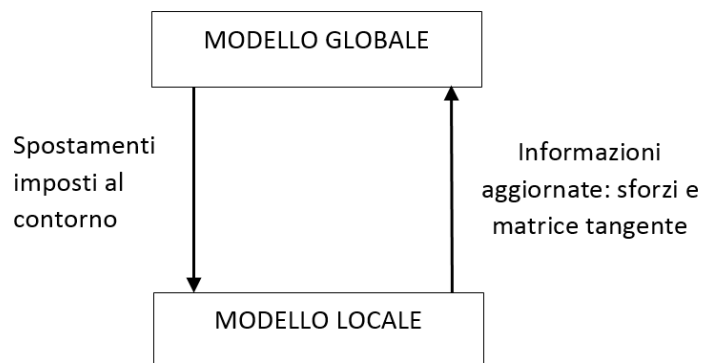


Figura 3.1: Scambio di informazioni tra il modello globale e il modello locale.

Questa interazione avviene tramite il passaggio degli spostamenti calcolati a livello globale e lo scambio di informazioni aggiornate tra i modelli costitutivi dei rispettivi livelli. Gli spostamenti vengono utilizzati dal livello locale come carichi imposti per risentire degli effetti globali, le informazioni aggiornate invece interessano la descrizione del materiale a livello globale, dove è presente un modello costitutivo 'fittizio' che necessita di ricevere l'aggiornamento delle sue variabili di sforzo e della matrice tangente per tenere conto degli effetti locali.

La sequenza delle operazioni all'interno della metodologia di analisi è la seguente:

1. **analisi globale:**

Il modello globale viene vincolato e caricato per effettuare l'analisi prevista sulla struttura.

2. **richiesta di informazioni aggiornate (locale → globale):**

Nella regione multi-livello, l'analisi a livello globale richiede dal modello costitutivo locale le informazioni aggiornate.

Questa richiesta avviene durante tutto l'incremento di carico, fino ad ottenere una condizione di equilibrio.

3. invio spostamenti nodali (globale \rightarrow locale):

Invio delle componenti di spostamento dei nodi al contorno, al termine di ciascun incremento di carico globale.

4. analisi locale:

Il modello locale riceve gli spostamenti e li utilizza come carichi imposti, necessari all'aggiornamento della configurazione di equilibrio all'incremento corrente e di salvataggio delle variabili di stato, che memorizzano la configurazione raggiunta dal materiale.

5. incremento di carico successivo:

Si procede al livello di carico successivo, ripartendo dal punto da 1, iterando per tutti gli n-incrementi definiti nel file di input del modello globale.

3.1 Il metodo tramite schema a blocchi

La figura 3.2 descrive il metodo tramite uno schema a blocchi, che permette una visione di insieme della metodologia con focalizzazione sui passaggi di informazioni che avvengono tra i 2 modelli e la sequenza delle operazioni.

Vi sono 2 fasi distinte: la fase preliminare e quella di analisi.

La fase preliminare consiste nella creazione manuale dei 2 modelli tramite la conoscenza della regione critica della struttura, da cui risulteranno note: la lista degli elementi e dei nodi al contorno.

La fase seguente è quella dell'analisi contenente le operazioni e le comunicazioni tra i 2 modelli tali da ottenere la simulazione global/local.

Le comunicazioni tra i livelli avvengono seguendo un preciso protocollo di comunicazione Server-Client (REQ-REP : richiesta-risposta), già introdotto in 2.2, ora specializzato alle richieste del metodo.

Sono descritte entrambe le comunicazioni, la prima operazione di ogni punto è relativa allo scambio di informazioni sugli spostamenti nodali, la seconda per la comunicazione tra i modelli costitutivi.

1. richiesta lista nodi o elementi:

Da parte del livello globale.

2. invio lista nodi o elementi:

Da parte del livello locale.

3. invio spostamenti o richiesta informazioni aggiornate

Il livello globale invia gli spostamenti dei nodi al contorno oppure richiede la comunicazione con il modello costitutivo locale, per gli elementi definiti nella lista.

4. ricezione spostamenti o invio informazioni aggiornate

Il livello locale riceve gli spostamenti nodali oppure invia le informazioni aggiornate dal proprio modello costitutivo.

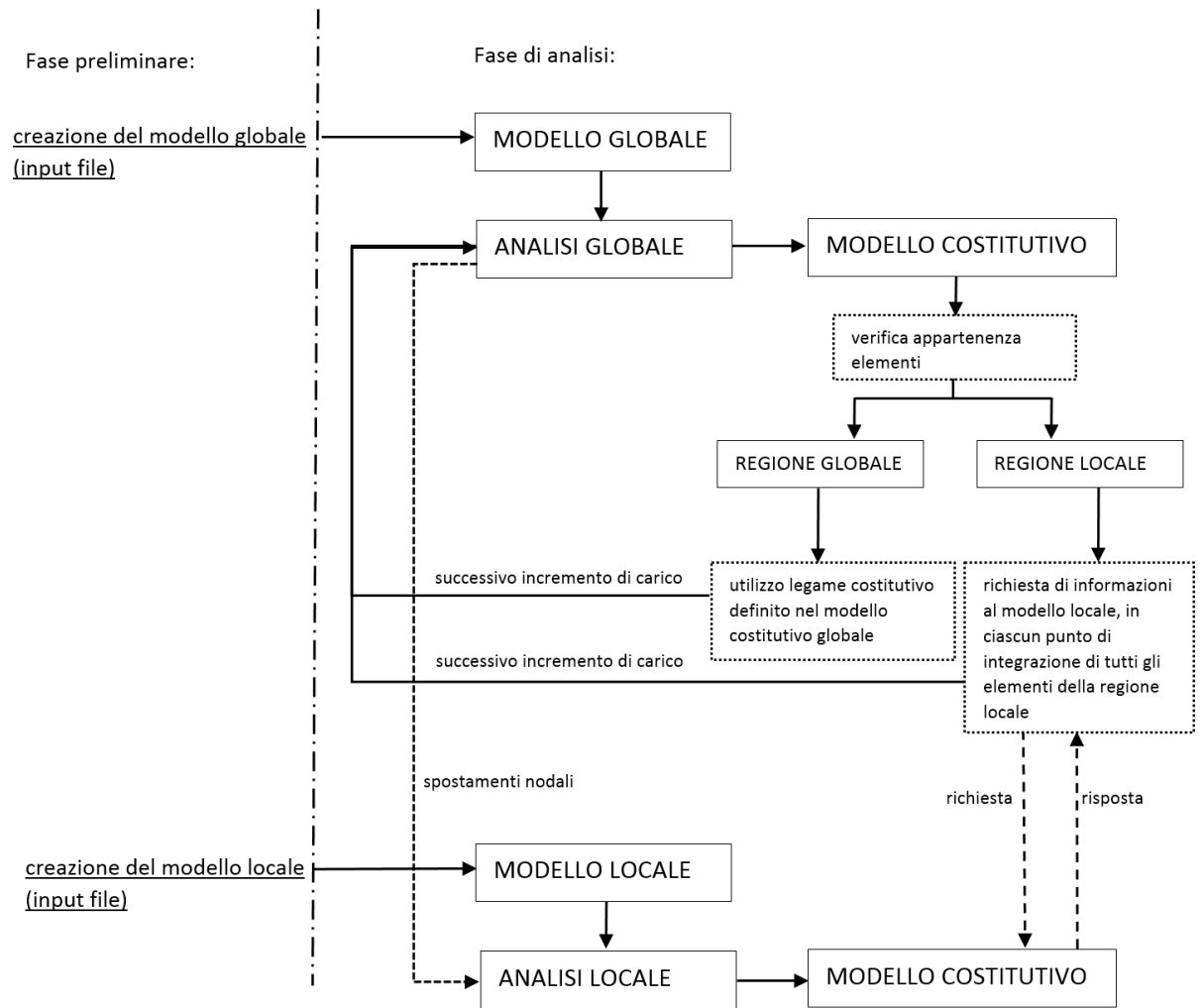


Figura 3.2: Schema a blocchi del metodo global/local

Possiamo riassumere le funzionalità richieste dal metodo a ciascun livello di analisi attraverso la figura 3.3, che introduce un identificativo, affianco a ciascuna funzione (per esempio G1), che verrà utilizzato nelle sezioni a seguire.

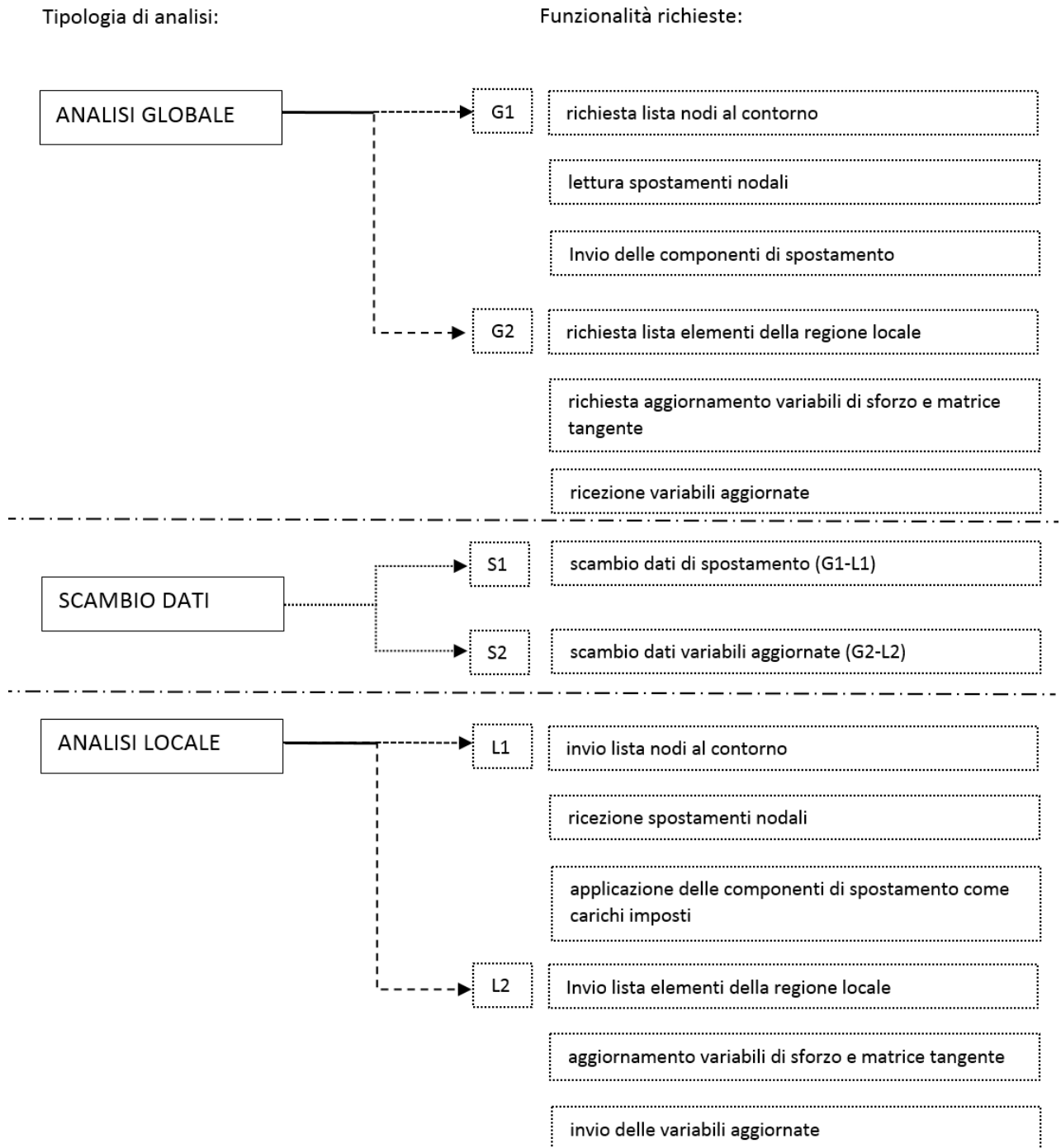


Figura 3.3: Funzionalità richieste dalle analisi nei 2 livelli della simulazione.

3.2 Limitazioni di utilizzo

La metodologia presenta alcune limitazioni di utilizzo che sono descritte a seguire.

Una limitazione riguarda l'incapacità del metodo di utilizzare una metodologia automatica per la modellazione della regione locale, a partire dagli elementi globali critici, attualmente questa fase richiede una modellazione manuale da parte dell'utente e specifica in funzione della struttura da analizzare.

L'interfaccia tra i 2 livelli è ottenuta tramite nodi coincidenti, questo comporta la lettura immediata degli spostamenti evitando l'operazione di interpolazione dei valori però non permette una diversa modellazione al contorno.

Ulteriore limitazione risiede nella comunicazione tra i modelli costitutivi, che viene effettuata direttamente elemento per elemento tra i 2 livelli, senza operazioni di equivalenza (omogenizzazione) intermedie, questo comporta che deve esserci una corrispondenza degli elementi dei 2 livelli interessati dallo scambio di informazioni.

Questo non esclude la possibilità di avere un livello globale a piastre ed modello locale ad elementi solidi, attraverso però una modellazione identica nella regione multi-livello e un'opportuna definizione delle condizioni al contorno.

Infine le analisi dei 2 livelli sono realizzate in sequenza e quindi caratterizzate dallo stesso numero di incrementi di carico, prefissati nel file di input globale.

Questa limitazione può essere superata fissando il numero di incrementi locali corrispondenti a ciascun incremento globale, per esempio un rapporto 2:1.

3.3 Implementazione del metodo

Segue l'implementazione del metodo, attraverso la programmazione delle Subroutine al fine di ottenere le funzionalità richieste dalla simulazione, seguendo l'associazione definita nella tabella 3.1.

Subroutine	Funzionalità richiesta
URDFIL_globale	G1
DISP_locale	L1
UMAT_globale	G2
URDFIL_locale + UMAT_locale	L2

Tabella 3.1: Associazione Subroutine con le funzionalità richieste dal metodo, la dicitura '_globale' o '_locale' indica il livello di appartenenza.

Nelle sezioni che seguono, si introducono le Subroutine e le funzioni di comunicazione richieste dalla metodologia, attraverso delle applicazioni pratiche.

Inizialmente si considerano esempi contenenti singole procedure, per passare all'aggiunta graduale di funzionalità.

L'applicazione della metodologia completa è testata su una piastra forata contenente tutte le funzionalità e comunicazioni richieste, con la modellazione locale tramite materiale elastoplastica (secondo von Mises).

La piastra forata permette l'applicazione del metodo con le limitazioni attuali permettendo di conoscere la regione critica della struttura ovvero in prossimità della discontinuità geometrica.

3.3.1 Introduzione del modello costitutivo elastico: UMAT

La Subroutine UMAT viene utilizzata in 2 semplici applicazioni, per la modellazione del legame costitutivo elastico isotropo, di 2 modelli costituiti rispettivamente da 1 e 2 elementi di piastra.

La Subroutine programmata si trova nell'appendice "A.3 UMAT: modello costitutivo elastico", contenente le variabili elencate nella tabella 3.2, suddivise in funzione della tipologia, input e output:

variabili INGRESSO	variabili USCITA
STRAN	DDSDDE(NTENS,NTENS)
DSTRAN	STRESS(NTENS)
TIME	STATEV(NSTATV)
DTIME	
CMNAME	
NDI	
NSHR	
NTENS	
NSTATV	
PROPS(NPROPS)	
NPROPS	
COORDS	
DROT(3,3)	
CELENT	
DFGRDO(3,3)	
DFGRD1(3,3)	
NOEL	
NPT	
LAYER	
KSPT	
JSTEP	
KINC	

Tabella 3.2: Variabili in ingresso e uscita UMAT

Le variabili nella colonna di sinistra vengono fornite dal software di analisi e possono essere utilizzate dall'utente per definire i 3 elementi fondamentali della procedura, elencati nella colonna di destra.

Nella programmazione sono state utilizzate solo alcune variabili in ingresso (la descrizione completa è rimandata alla sezione "1.1.44 UMAT", del manuale in [18]) :

- **DSTRAN(NTENS):** Incremento di deformazione.
- **NTENS:** Dimensione dei vettori di sforzo e deformazione.
- **NSTATV:** Numero di variabili di stato, definito nella keyword *DEPVAR.
- **PROPS(NPROPS):** Valore delle costanti del materiale, definite nella keyword *USER MATERIAL.
- **NPROPS:** Numero di costanti del materiale.

Nella parte esecutiva le proprietà elastiche, contenute in PROPS(NPROPS), vengono rinominate opportunamente per essere utilizzate nella procedura, in E (modulo elastico), G (modulo di taglio) e ν (coefficiente di Poisson).

Viene creata poi la matrice tangente DDSDDE(NTENS,NTENS) di dimensioni 3X3, per

uno stato di sforzo piano, di componenti:

$$\begin{bmatrix} \frac{E}{1-\nu^2} & \frac{E\nu}{1-\nu^2} & 0 \\ \frac{E\nu}{1-\nu^2} & \frac{E}{1-\nu^2} & 0 \\ 0 & 0 & G \end{bmatrix}$$

Questa matrice è utilizzata per l'aggiornamento delle componenti di sforzo, a seguito di un incremento di deformazione DSTRAN, che vengono salvate nelle variabili di stato.

Questa programmazione è identica per i 2 esempi, il modello a 2 elementi è utilizzato per ottenere una modellazione diversa degli elementi, rispettivamente tramite UMAT e tramite *ELASTIC, verificando questa possibilità da utilizzare in seguito all'interno del metodo global/local.

Entrambi i modelli sono sottoposti a trazione uniforme e vincolati isostaticamente, attraverso un vincolo cerniera-carrello, come mostra la figura 3.4:

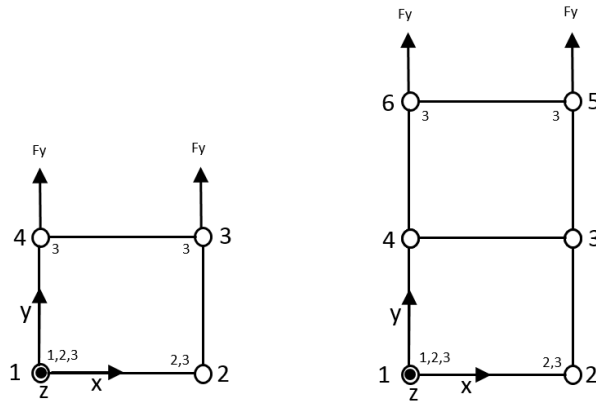


Figura 3.4: Modello ad 1 elemento (a sinistra), modello a 2 elementi (a destra).

I numeri esterni rappresentano l'identificativo dei nodi, quelli interni sono i vincoli imposti, invece il carico applicato F_y è rappresentato dalle frecce in direzione y .

3.3.2 Spostamenti imposti: DISP

La Subroutine DISP, viene applicata per l'imposizione di condizioni al contorno, tramite spostamenti nodali in direzione x , su di una piastra quadrata isotropa.

La procedura necessita della definizione della lista dei nodi e gdl, sul quale imporre le condizioni, che vanno definiti nel file di input, nella keyword *BOUNDARY, USER.

La Subroutine programmata si trova nell'appendice "A.4 DISP: imposizione spostamenti nodali", contenente le variabili elencate a seguire:

variabili INGRESSO	variabili USCITA
KSTEP	U
KINC	
TIME	
NODE	
NOEL	
JDOF	
COORDS	

Tabella 3.3: Variabili in ingresso e uscita DISP

Solo alcune di esse trovano applicazione nella procedura (la descrizione completa è rimandata alla sezione "1.1.4 DISP", del manuale in [18]) :

- **TIME:** Ha dimensioni 3, TIME(1) è il valore corrente del tempo nello step, TIME(2) è il valore corrente del tempo totale e TIME(3) è il valore corrente dell'incremento temporale.
- **NODE:** Numero identificativo del nodo.
- **JDOF:** Grado di libertà.

L'output da definire nella parte esecutiva della procedura è il valore delle componenti di spostamento:

- **U:** U ha dimensione 3, U(1) rappresenta lo spostamento e le altre 2 componenti sono rispettivamente a velocità e accelerazione.
Le componenti di spostamento da definire sono 6 (3 traslazioni e 3 rotazioni).

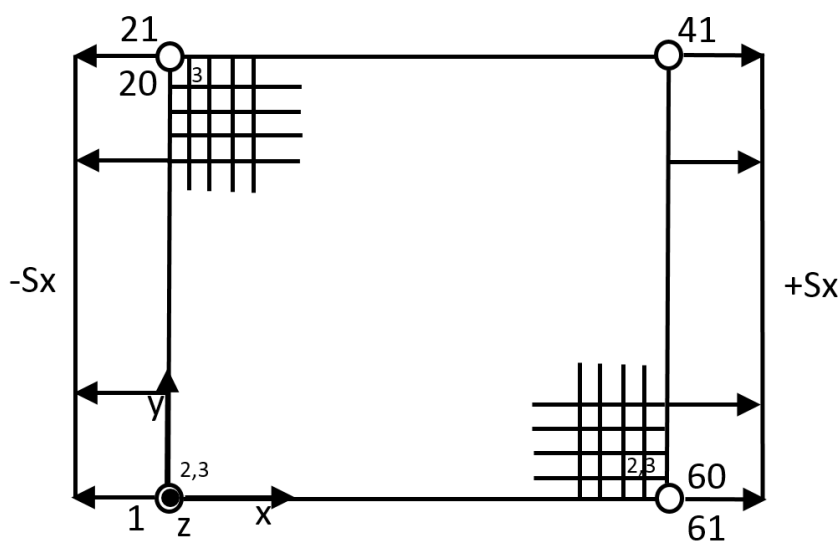


Figura 3.5: Rappresentazione piastra quadrata isotropa.

La piastra è modellata con 20×20 elementi, figura 3.5, è sottoposta ad un carico di trazione uniaassiale applicato ai due estremi (nodi da 1 a 21 con carico $-S_x$, nodi 41 a 61 con carico $+S_x$), attraverso l'utilizzo di 2 cicli 'if' per associare ai nodi il rispettivo carico.

Il valore del carico è incrementato progressivamente, ad ogni load step, utilizzando come coefficiente moltiplicativo il 'TIME(2)'.

La struttura è vincolata isostaticamente su 3 nodi estremi (1,21,61), tale da rimuovere tutti i gdl rigidi.

3.3.3 Lettura dei risultati: URDFIL

La Subroutine URDFIL, viene utilizzata per la lettura dei risultati, applicata al modello di piastra precedentemente descritto, figura 3.5.

La programmazione della procedura è contenuta nell'appendice "A.5 URDFIL: lettura dei risultati" e passa attraverso l'utilizzo di 2 utility routine:

- **POSFIL:** interpellata tramite 'CALL POSFIL(NSTEP,NINC,ARRAY,JRCD)', permette di posizionarsi nel file dei risultati tramite la definizione delle variabili in ingresso: NSTEP (step desiderato) e NINC (incremento desiderato).
Esse possono essere inserite dall'utente, per specificare il posizionamento per la lettura in un determinato istante della simulazione, oppure lasciare vuoto tale campo, permettendo il posizionamento incremento dopo incremento.
Le variabili restituite in uscita sono: ARRAY (file contenente le informazioni del record 2000, codifica per indicare i dati relativi allo step, all'inizio dell'incremento), JRCD (restituisce un valore nullo se l'incremento specificato è stato trovato, altrimenti restituisce 1).
- **DBFILE:** interpellata tramite 'CALL DBFILE(LOP,ARRAY,JRCD)', permette la lettura dei risultati dal file .fil.
La variabile in ingresso è: LOP (per indicare l'operazione di lettura da eseguire nella routine).
Vengono restituite in uscita le variabili: ARRAY (contenente i risultati) e JRCD (valore sempre nullo tranne quando viene letto un indicatore di fine file).

La programmazione procede con il posizionamento (POSFIL), lasciando liberi i campi di NSTEP e NINC per il posizionamento in tutti gli incrementi e step, seguito da un ciclo 'do', per la lettura (DBFILE), che termina attraverso un ciclo 'if () go to 110' di verifica della fine del file (JRCD=0).

Il ciclo 'do' permette la creazione di tutte le KEY, previste dai risultati richiesti in output e sfruttando un ciclo 'if' si verifica un determinato identificativo per passare alla lettura dei risultati (il numero 101 corrisponde agli spostamenti nodali).

Nel record 101 si trova in posizione 3 l'ID dei nodi seguito dai valori delle componenti di spostamento, per esempio in posizione 4 si legge lo spostamento in direzione x.

Le prime 2 posizioni di ogni record sono dedicate alla lunghezza del record stesso e alla rispettiva KEY.

Il risultato viene infine printato a schermo con l'ID del nodo corrispondente e il valore di spostamento.

3.3.4 Modello Globale-Locale: piastra quadrata isotropa

Questa sezione rappresenta la naturale continuazione del percorso di aggiunta di operazioni al metodo per ottenere le funzionalità richieste, introducendo la modellazione locale, al modello di piastra quadrata isotropa descritto nelle precedenti sezioni.

La modellazione completa della piastra è mostrata nella figura 3.6, contenente il livello globale (elementi in nero) ed il livello locale (elemento in verde).

In quest'implementazione, descritta nell'appendice A.6 "Modello globale-locale: piastra quadrata isotropa", i 2 livelli sono in grado di scambiarsi informazioni relative agli spostamenti nodali (6 componenti: u,v,w,rx,ry,rz), tramite la funzione S1 che permette la comunicazione tra le funzionalità G1 ed L1.

Nel modello globale è inserita la Subroutine URDFIL, per la lettura degli spostamenti nei

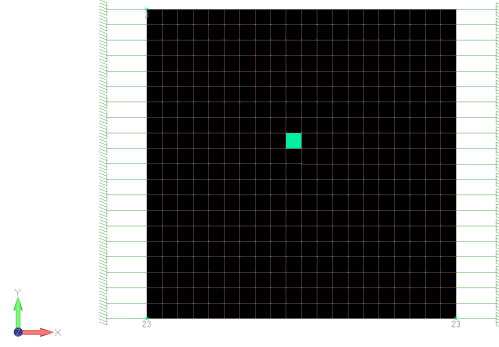


Figura 3.6: Modellazione global/local piastra quadrata isotropa (in verde l'elemento utilizzato per l'analisi locale).

nodi richiesti, alla fine di ogni incremento di carico, nel modello locale invece è contenuta la Subroutine DISP, per imporre gli spostamenti ricevuto come condizioni al contorno.

Il materiale di entrambi i modelli è descritto attraverso la keyword `*ELASTIC`, inserita nel rispettivo livello.

La procedura ottenuta finora rappresenta un metodo di analisi one-way, il passaggio finale è contenuto nella prossima sezione, con l'introduzione della comunicazione tra i modelli costitutivi per creare il ritorno delle informazioni locali aggiornate (two-way).

3.3.5 Modello Globale-Locale: piastra forata isotropa

L'implementazione completa della metodologia è testata su una piastra forata, rappresentata in figura 3.7, contenente il livello globale (a) ed il livello locale (b).

La piastra forata è modellata tramite la sola metà di destra, utilizzando i vincoli di simmetria e caricata a trazione uniforme.

Il modello locale, comprende la regione circostante il foro, caratterizzata da un forte gradiente di sforzo e nascita di fenomeni non-lineari, descritti nel comportamento costitutivo (UMAT locale) considerando un materiale elasto-plastico, la regione esterna contenuta nel modello globale è considerata elastica.

La programmazione completa è contenuta nell'appendice "A.7 Modello globale-locale: piastra forata isotropa modello elasto-plastico".

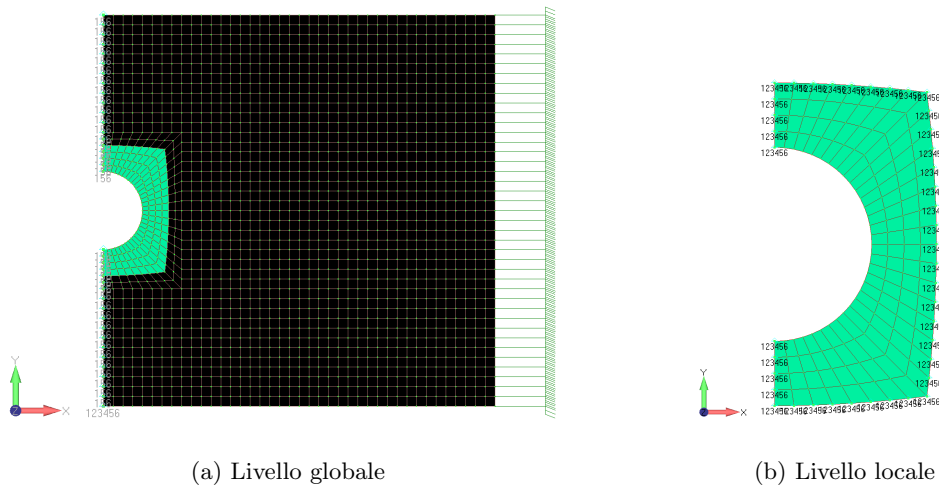


Figura 3.7: Modellazione global-local piastra forata.

Descrizione dell'implementazione

L'implementazione prevede le 2 comunicazioni richieste dal metodo, attraverso l'utilizzo di variabili globali, contenute in "MODULE CONN_DATA", per lo scambio di informazioni. Le funzionalità di comunicazione send-recv sono richiamate all'interno delle Subroutine attraverso l'utilizzo delle procedure esterne dei rispettivi livelli: livello globale attraverso "CALL CLIENT_CONN()", livello locale attraverso "CALL SERVER.CONN()", che utilizzando le variabili globali tramite "USE CONN_DATA" per creare una sola volta il collegamento.

In questo modo in sequenza le Subroutine dei rispettivi livelli possono comunicare per l'invio o la ricezione delle informazioni e con la possibilità di passare alla procedura successiva senza la creazione di ulteriori collegamenti o nuove variabili.

In questo modo è possibile interrompere una Subroutine globale, che comunica con una Subroutine locale, verificata la condizione di entrata nella successiva procedura locale o globale.

Questo permette di completare tutte le operazioni necessarie alla metodologia global/local, nella giusta sequenza.

Le interruzioni delle Subroutine previste dalla programmazione, durante l'esecuzione, sono:

- 1. interruzione URDFIL_locale:** Si verifica quando l'analisi globale è giunta ad una condizione di equilibrio e la UMAT_globale non necessita più di informazioni aggiornate, procedendo con la lettura dei risultati (URDFIL_globale).
Tale condizione è verificata nella URDFIL_locale se una variabile 'n_inc' inviata dalla URDFIL_globale è uguale a zero, permettendo la continuazione dell'analisi.
- 2. interruzione URDFIL_globale:** Si verifica quando l'analisi locale riceve le informazioni di spostamento nodali (in DISP_locale) e tramite la variabile 'continuo' concede il permesso alla URDFIL_globale di proseguire e terminare.
Il livello globale procede all'incremento di carico successivo restando in attesa del completamento dell'incremento locale.

La DISP_locale è programmata per imporre le condizioni al contorno, provenienti dal livello globale (URDFIL_globale), nei lati estremi della regione ad esclusione della semi-circonferenza del foro, che risulta scarica.

La UDFIL_locale riceve l'incremento deformativo globale e legge i risultati raggiunti dal livello locale nel precedente incremento di carico: le 5 variabili di stato (contenenti le deformazioni elastiche e la deformazione plastica equivalente) e le 3 componenti di sforzo.

Queste variabili vengono fornite in ingresso alla UMAT_locale, insieme alle proprietà elastiche del materiale e all'incremento deformativo globale (DSTRAN), per restituire gli sforzi e della matrice tangente ('sforzo' e 'k') da aggiornare nel livello globale.

La descrizione dei fenomeni non-lineari avviene nella UMAT_locale, realizzata per tenere conto di eventuali fenomeni plastici (secondo von Mises) per ottenere un modello costitutivo elasto-plastico.

Descrizione del materiale elasto-plastico

Il materiale utilizzato per la simulazione è contenuto in [22] e corrisponde ad un acciaio dolce (A-533, Grade B, Class-1 nuclear-pressure-vessel steel) con le proprietà elastiche elencate nella tabella 3.4.

La curva di incrudimento è rappresentata in figura 3.8, realizzata utilizzando i valori della tabella 3.5.

Proprietà	Valore
E	206.9 [GPa]
ν	0.29
σ_{y0}	450 [MPa]

Tabella 3.4: Proprietà elastiche di un acciaio dolce.

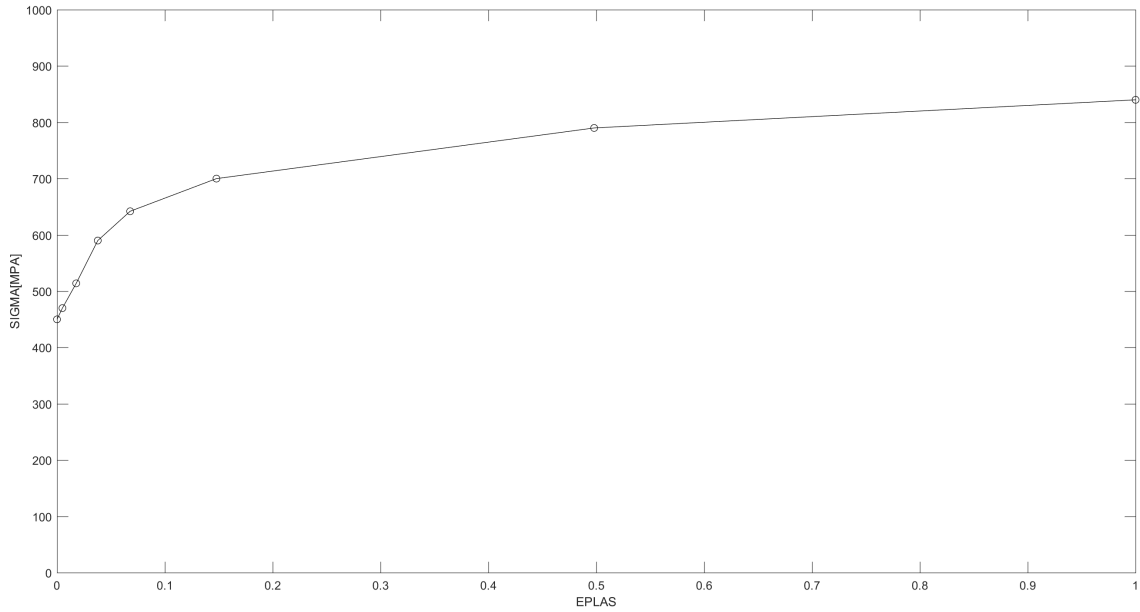


Figura 3.8: Curva SIGMA-EPLAS con legge di incrudimento lineare a tratti.

SIGMA	450	470	514	590	642	700	790	840
EPLAS	0	0.005	0.0178	0.0378	0.0678	0.1478	0.498	1

Tabella 3.5: Tabella dei valori SIGMA e EPLAS di incrudimento.

Plasticità di von Mises

Il modello elasto-plastico utilizzato è descritto in [21], nel capitolo 9 dedicato alla plasticità con stato di sforzo piano.

Questa metodologia utilizza il plane stress-projected plasticity model che consiste nel valutare le equazioni del problema attraverso le solo componenti di sforzo e deformazioni nel

piano, che risultano quindi le variabili primarie da calcolare.

Le variabili fuori dal piano sono dipendenti dalle componenti nel piano e possono essere calcolate a posteriori.

Questa modellazione è equivalente alla trattazione 3D del problema con l'aggiunta dei vincoli legati allo stato di sforzo piano:

$$\sigma_{13} = \sigma_{23} = \sigma_{33} = 0$$

Per ottenere le equazioni nel piano si devono determinare le espressioni delle componenti elastiche e plastiche fuori dal piano dipendenti dalle componenti primarie e toglierle dalle equazioni del problema.

Le deformazioni sono divise nella loro parte elastica e plastica: $\varepsilon = \varepsilon^e + \varepsilon^p$

La dipendenza delle variabili fuori dal piano risulta:

$$\begin{aligned} \varepsilon_{13}^e = \varepsilon_{23}^e = 0, \varepsilon_{33}^e &= -\frac{\nu}{1-\nu}(\varepsilon_{11}^e + \varepsilon_{22}^e) \\ \varepsilon_{13}^p &= \varepsilon_{23}^p = 0 \end{aligned}$$

Utilizzando la condizione di plasticità incomprimibile:

$$\varepsilon_{33}^p = -(\varepsilon_{11}^p + \varepsilon_{22}^p)$$

Si possono ottenere le relazioni complete delle variabili fuori dal piano, definite come segue:

$$\begin{aligned} \varepsilon_{13}(t) &= \varepsilon_{13}^e(t) + \varepsilon_{13}^p(t) = 0 \\ \varepsilon_{23}(t) &= \varepsilon_{23}^e(t) + \varepsilon_{23}^p(t) = 0 \\ \varepsilon_{33}(t) &= -\left[\frac{\nu}{1-\nu}(\varepsilon_{11}^e(t) + \varepsilon_{22}^e(t)) + \varepsilon_{11}^p(t) + \varepsilon_{22}^p(t)\right] \end{aligned}$$

Queste relazioni vengono tolte dalle equazioni del problema per ottenere le equazioni proiettate nel piano (notazione per componenti):

$$\begin{cases} \varepsilon_{\alpha\beta} = \varepsilon_{\alpha\beta}^e + \varepsilon_{\alpha\beta}^p \\ \sigma_{\alpha\beta} = D_{\alpha\beta\gamma\delta}^e \varepsilon_{\gamma\delta}^e \\ \Phi = \sqrt{\frac{3}{2}[\sigma_{\alpha\beta}\sigma_{\alpha\beta} - \frac{1}{3}(\sigma_{\alpha\alpha})^2]} - \sigma_y(\varepsilon^p) \\ \varepsilon_{\alpha\beta}^p = \gamma \sqrt{\frac{3}{2} \frac{s_{\alpha\beta}}{\sqrt{\sigma_{\gamma\delta}\sigma_{\gamma\delta} - \frac{1}{3}(\sigma_{\gamma\gamma})^2}}} \\ \varepsilon^p = \gamma \\ \gamma \geq 0, \Phi \leq 0, \gamma\Phi = 0 \end{cases}$$

Definite le equazioni del problema, l'algoritmo di integrazione parte con il calcolo dello stato elastico tentativo:

$$\begin{aligned} \varepsilon_{n+1}^{e,trial} &= \varepsilon_n^e + \Delta\varepsilon \\ \sigma_{n+1}^{trial} &= D^e \varepsilon_{n+1}^{e,trial} \\ \varepsilon_{n+1}^{p,trial} &= \varepsilon^p \end{aligned}$$

Dove il $\Delta\varepsilon$ è l'incremento deformativo (nel piano) nell'intervallo $[t_n, t_{n+1}]$.

A questo punto l'algoritmo calcola la funzione di snervamento Φ e verifica se $\Phi \leq 0$, ovvero in campo elastico, permettendo l'aggiornamento delle variabili:

$$(\cdot)_{n+1} = (\cdot)_{n+1}^{trial}$$

Se tale condizione non è verificata significa che il processo è in campo plastico ($\Phi > 0$) e richiede la risoluzione delle equazioni non-lineari, $\Phi(\Delta\gamma) = 0$, tramite il metodo di Newton-Raphson.

Si applica l'algoritmo di return mapping, che determina le variabili correttive tali da aggiornare la configurazione per riportarla sulla superficie di snervamento, considerando un incrudimento isotropo.

La figura 3.9 riassume il return mapping, con $\Delta\sigma^r$ che rappresenta la correzione di sforzo.

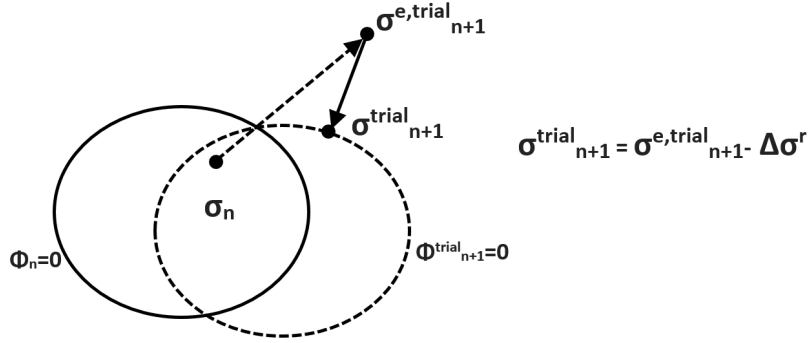


Figura 3.9: Return mapping con incrudimento isotropo.

Il problema termina con la determinazione della sola variabile incognita ovvero l'incremento del moltiplicatore plastico $\Delta\gamma$ tale per cui $|\tilde{\Phi}| \leq \epsilon_{tol}$.

La Subroutine AHARD all'interno del modello costitutivo è utilizzata per fornire i valori di SIGMA e il parametro HARD (pendenza del tratto di incrudimento), attraverso la lettura delle variabili in ingresso: la tabella del materiale (TABLE) e la deformata plastica equivalente raggiunta (EQPLAS), tali da determinare la posizione nella curva SIGMA-EPLAS.

È richiamata per ottenere la SIGMAY utilizzata per la verifica di snervamento e in seguito durante le iterazioni di N-R per ottenere il parametro H.

Infine il modello costitutivo richiede l'aggiornamento della matrice elasto-plastica tangente, che avviene nell'ultima parte della procedura:

$$D^ep = \frac{d\sigma_{n+1}}{d\varepsilon_{n+1}} = \frac{d\sigma_{n+1}}{d\varepsilon_{n+1}^{trial}}$$

Con σ_{n+1} ottenuto dal return mapping.

Capitolo 4

Risultati

La fase di implementazione è validata in questa sezione, attraverso il confronto dei risultati ottenuti con delle opportune analisi comparative.

La validazione interessa ciascun passaggio seguito per lo sviluppo del metodo, tale da permettere la verifica della corretta programmazione delle procedure, fino ad arrivare alla validazione del modello completo applicato ad una piastra forata.

4.1 Validazione delle procedure del metodo

In questa sezione vengono verificate le procedure DISP, URDFIL, le analisi sono eseguite tramite i comandi da terminale, in ambiente Linux, come descritto nel manuale [19], nella sezione "3.2.2 Abaqus/Standard, Abaqus/Explicit, and Abaqus/CFD execution":

abaqus j=job_name, user=source_file interactive'.

Con la seguente sostituzione:

sostituzione	procedure utilizzate
job_name = nome_input_file (senza estensione .inp)	
source_file = nome_procedura.f	
	DISP.f
	URDFIL.f

Tabella 4.1: File utilizzati nel comando delle analisi.

4.1.1 Validazione DISP e URDFIL

La validazione delle procedure DISP e URDFIL viene eseguita confrontando i risultati ottenuti tramite la modellazione di una piastra quadrata isotropa contenente un modello costitutivo elastico, lega Al, sottoposta ai vincoli e carichi imposti di spostamento (tramite *BOUNDARY), suddivisi in 5 incrementi.

Dal file di input, il materiale ha le seguenti proprietà:

```
*MATERIAL, NAME=LegaAl
*ELASTIC, TYPE=ISOTROPIC
73000,0.3
```

I carichi e i vincoli sono:

```
*BOUNDARY
1,2,3
21,3
61,2,3
*BOUNDARY
N1,1,1,-200
N2,1,1,200
```

I vincoli sono applicati isostaticamente su 3 nodi estremi (1,21,61), N1 e N2 rappresentano rispettivamente la lista dei nodi dell'estremo sinistro e destro, 200 è il carico uniforme S_x di spostamento [mm].

La piastra ha dimensioni 100mm \times 100mm e spessore di 1mm, realizzata attraverso 20×20 elementi.

La figura 4.1 mostra il primo confronto con l'utilizzo della procedura DISP, in termini di sforzi in direzione x, che è la componente predominante della simulazione.

Tale comparazione è seguita da un'altra validazione con il confronto tra i risultati di spostamento letti sul modello tramite la procedura URDFIL, tabella 4.3 e i risultati ottenuti in fase di post-processing dal modello elastico, tabella 4.2, nei nodi evidenziati nella figura 4.2. Lista nodi: 27, 75, 176 \div 194.

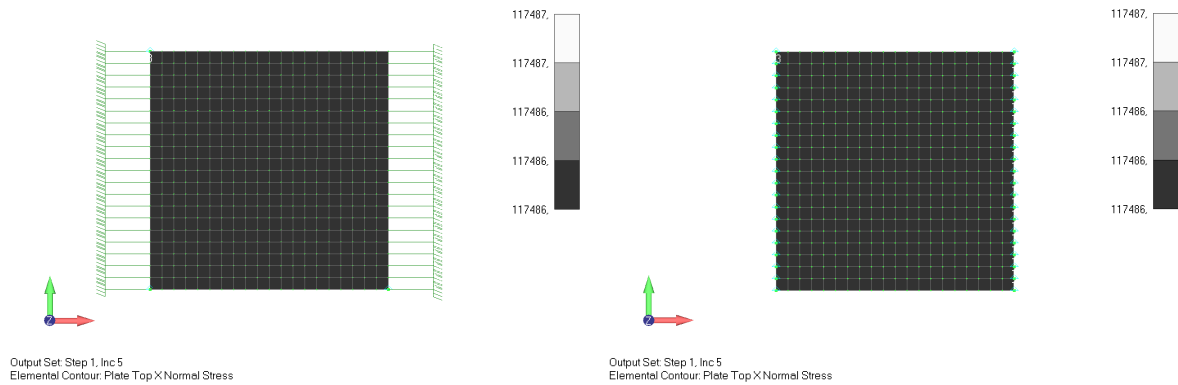


Figura 4.1: Confronto risultati, sforzo S_x : ELASTIC (a destra), DISP (a sinistra).

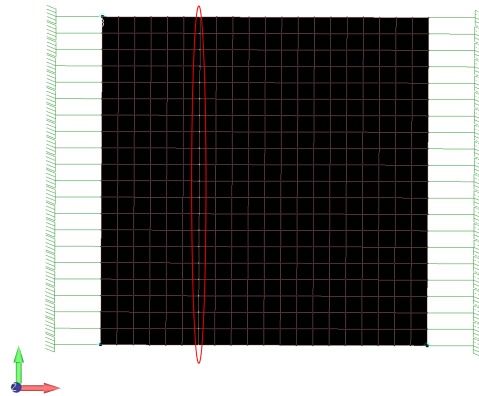


Figura 4.2: Nodi utilizzati per il confronto degli spostamenti.

ID	STEP_1 Ux [mm]	STEP_2 Ux [mm]	STEP_3 Ux [mm]	STEP_4 Ux [mm]	STEP_5 Ux [mm]
27	-16	-32	-48	-64	-80
75	-16	-32	-48	-64	-80
176	-16	-32	-48	-64	-80
...	-16	-32	-48	-64	-80
192	-16	-32	-48	-64	-80
193	-16	-32	-48	-64	-80
194	-16	-32	-48	-64	-80

Tabella 4.2: Tabella spostamenti nodali Ux modello ELASTIC.

NODO , U1, INC	27	-15.99	1
NODO , U1, INC	75	-15.99	1
NODO , U1, INC	176	-16	1
NODO , U1, INC	1
NODO , U1, INC	192	-16	1
NODO , U1, INC	193	-15.99	1
NODO , U1, INC	194	-16	1
NODO , U1, INC	27	-31.99	2
NODO , U1, INC	75	-31.99	2
NODO , U1, INC	176	-32	2
NODO , U1, INC	2
NODO , U1, INC	192	-32	2
NODO , U1, INC	193	-31.99	2
NODO , U1, INC	194	-32	2
NODO , U1, INC	27	-47.99	3
NODO , U1, INC	75	-47.99	3
NODO , U1, INC	176	-48	3
NODO , U1, INC	2
NODO , U1, INC	192	-48	3
NODO , U1, INC	193	-47.99	3
NODO , U1, INC	194	-48	3
NODO , U1, INC	27	-63.99	4
NODO , U1, INC	75	-63.99	4
NODO , U1, INC	176	-64	4
NODO , U1, INC	4
NODO , U1, INC	192	-64	4
NODO , U1, INC	193	-63.99	4
NODO , U1, INC	194	-64	4
NODO , U1, INC	27	-79.99	5
NODO , U1, INC	75	-79.99	5
NODO , U1, INC	176	-80	5
NODO , U1, INC	5
NODO , U1, INC	192	-80	5
NODO , U1, INC	193	-79.99	5
NODO , U1, INC	194	-80	5

Tabella 4.3: Tabella dei risultati ottenuti tramite URDFIL e mostrati a schermo

La verifica mostra che la programmazione delle procedure determina risultati coerenti con-

frontati con le analisi standard di Abaqus e permette la fase successiva di utilizzo di più procedure all'interno della simulazione.

4.1.2 Validazione modello di piastra quadrata isotropa

La validazione procede con la verifica della piastra quadrata isotropa con l'aggiunta del modello locale, le 2 analisi vengono eseguite utilizzando 2 terminali comandati separatamente, per l'esecuzione dei file di input e delle rispettive procedure, tabella 4.4.

input file	procedure associate
globale.inp	URDFIL.f
locale.inp	DISP.f

Tabella 4.4: File e procedure della simulazione global/local piastra quadrata isotropa.

I risultati ottenuti sul modello globale e su quello locale, in termini di sforzo S_x , figura 4.3, mostrano il corretto trasferimento e applicazione degli spostamenti imposti sul livello locale e risultai coerenti ad un'analisi standard.

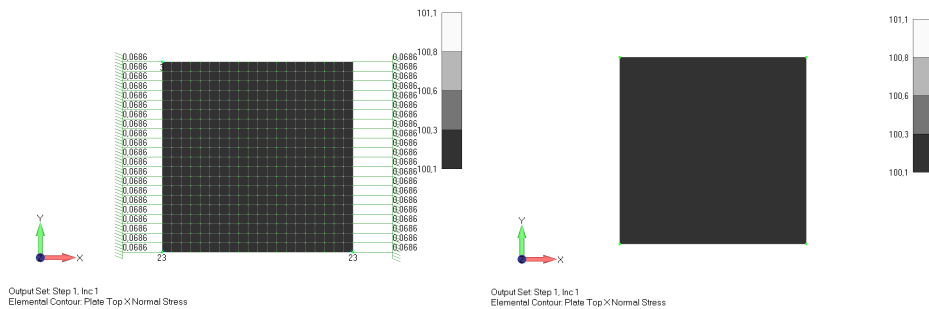


Figura 4.3: Risultati piastra quadrata, sforzo S_x : modello globale (a destra), modello locale (a sinistra).

4.2 Validazione del modello elasto-plastico

La validazione del modello completo applicato ad una piastra forata è ottenuta tramite il confronto dei risultati con un modello elasto-plastico utilizzando la modellazione di Abaqus *PLASTIC, che permette di introdurre la tabella 3.5, di incrudimento lineare a tratti.

```
*MATERIAL, NAME=AcciaioDolce
*ELASTIC, TYPE=ISOTROPIC
206900,0.29
*PLASTIC
450,0
470,0.005
514,0.0178
590,0.0378
642,0.0678
700,0.1478
790,0.498
840,1
```

L'analisi è ottenuta come per il caso precedente tramite 2 terminali, che lanciano rispettivamente l'analisi globale e locale.

Le procedure globali e locali vengono compilate in file oggetto .o attraverso l'utilizzo dei file .sh, che permettono di eseguire l'operazione direttamente nella shell del calcolatore.

file.o	procedure inserite
GLOBALE_Mises_plasticity	URDFIL.f, UMAT.f
LOCALE_Mises_plasticity	DISP.f, UMAT.f, URDFIL.f

Tabella 4.5: File.o e procedure associate.

I tempi di esecuzione delle analisi, sono descritti a seguire:

tipologia di analisi	riepilogo dei tempi di esecuzione	tempo (secondi)
analisi plastica standard	user time	12.980
	system time	2.4800
	total CPU time	15.460
	wallclock time	16
analisi globale	user time	14.130
	system time	5.6900
	total CPU time	19.820
	wallclock time	27
analisi locale	user time	3.4100
	system time	3.4400
	total CPU time	6.8500
	wallclock time	26

Tabella 4.6: Tempi di analisi.

I risultati ottenuti sono mostrati a partire dal confronto degli sforzi (S_x, S_y, S_{xy}) espressi in [MPa], procedendo con le deformazioni totali (E_x, E_y, E_{xy}) ed infine la deformazione plastica equivalente (PEEQ).

Il confronto è ottenuto seguendo l'ordine definito nella tabella 4.7.

ordine analisi	risultati	note
GLOBALE - LOCALE	Si	i=x,y,xy; analisi global-local
PLASTIC	Si	i=x,y,xy; analisi elasto-plastica standard
GLOBALE - LOCALE	Ei	i=x,y,xy; analisi global-local
PLASTIC	Ei	i=x,y,xy; analisi elasto-plastica standard
PLASTIC - LOCALE	PEEQ	

Tabella 4.7: Ordine risultati piastra forata.

I risultati mostrano che la metodologia global-local è fedele nella descrizione del comportamento elasto-plastico della struttura portando agli stessi risultati di un'analisi standard, come mostrato nelle figure 4.4÷4.9 (sforzi) e nelle figure 4.10÷4.15 (deformazioni).

La dimensione della regione locale è stata scelta correttamente riuscendo a ottenere i risultati attesi, legati alla forte concentrazione di sforzo dovuta alla discontinuità geometrica, in termini di sforzo, deformazione e plasticità.

L'ultima figura infatti mostra la deformazione plastica equivalente (PEEQ) contenuta nella quinta componente delle variabili di stato del modello locale, confrontata con i risultati dell'analisi standard, che mette in luce come la dimensione della regione locale sia adeguata per l'analisi dei fenomeni plastici.

Essi rimangono confinati in tale regione permettendo la modellazione elastica nella restante parte della struttura, ovviamente se tale condizione non è verificata e la diffusione della plasticità supera tale regione, le ipotesi di utilizzo del metodo perdono di validità.

Il riepilogo dei tempi di esecuzione, tabella 4.6, ha però fatto emergere una differenza importante in termini di velocità di analisi rispetto all'utilizzo tradizionale.

Questo è legato al costo computazionale delle operazioni di comunicazione multi-livello, non previste dalla simulazione standard, infatti i tempi di utilizzo della CPU sono comparabili, tale differenza risulterebbe meno importante in problemi di dimensioni maggiori dove il tempo della CPU diventa maggiormente predominante.

Da tale tabella si osserva inoltre che l'analisi locale sembrerebbe terminare prima di quella globale, in realtà la conclusione avviene quasi in contemporanea, questa differenza è dettata dall'aver lanciato 1 secondo dopo tale analisi rispetto a quella globale.

Nell'applicazione considerata dunque l'analisi standard risulta più efficace e semplice da utilizzare rispetto alla metodologia global-local che paga in termini di prontezza di esecuzione e facilità di implementazione.

Ciò che rende più appetibile tale metodologia rispetto a quella tradizionale deriva dalle potenzialità delle operazioni che possono essere eseguite, contrariamente all'analisi standard.

La modellazione multi-livello infatti studia nel dettaglio solo una porzione di struttura attraverso l'interazione tra più modelli per tenere conto degli effetti globali e locali, permettendo all'utente di modificare solamente la regione locale per aumentare o ridurre la descrizione della struttura.

Inoltre attraverso poche modifiche si può spostare la regione critica in un'altra porzione di struttura o aggiungere più regioni multi-livello.

Con questa metodologia la fase di post-processing è facilitata, attraverso i risultati del livello globale si ottiene il comportamento complessivo, se invece si vuole verificare la sola regione critica si può analizzare solo i risultati del modello locale.

La sequenzialità delle analisi sui rispettivi livelli inoltre permette un'implementazione automatizzata per la modellazione autonoma delle regioni critiche, ottenendo un metodo in grado di analizzare molteplici problemi dove tali regioni non sono note a priori:

- plasticità localizzate e/o diffuse;
- danneggiamenti progressivi;
- nascita e propagazione di fratture e propagazione.

Permette lo studio di analisi complesse che possono interessare anche la fase di collasso della struttura caratterizzata da forti non-linearità e interazioni multi-livello tra i fenomeni globali e locali.

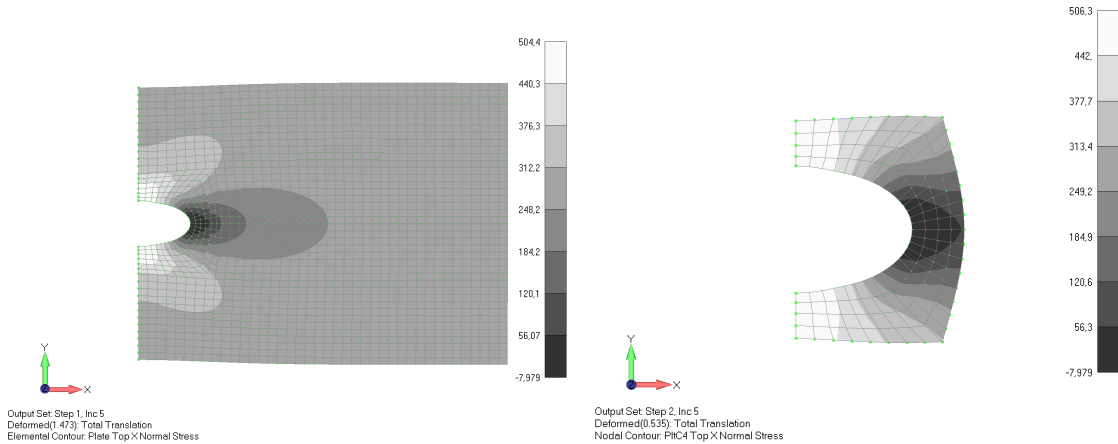


Figura 4.4: Risultati piastra forata, sforzo S_x : modello globale (a sinistra), modello locale (a destra).

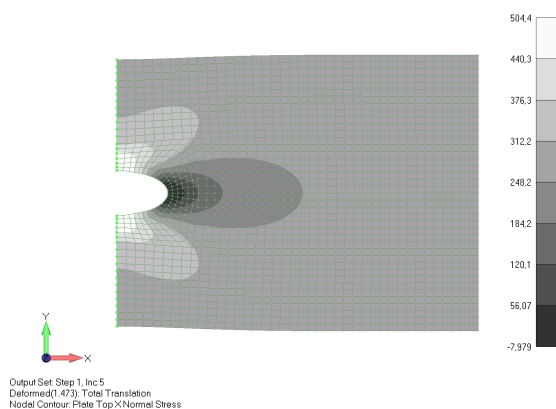


Figura 4.5: Risultati piastra forata PLASTIC, sforzo S_x .

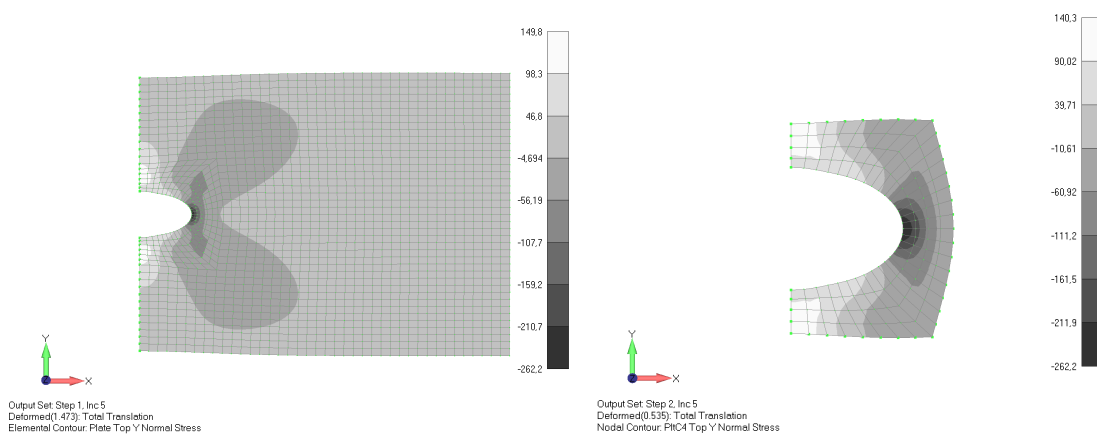


Figura 4.6: Risultati piastra forata, sforzo S_y : modello globale (a sinistra), modello locale (a destra).

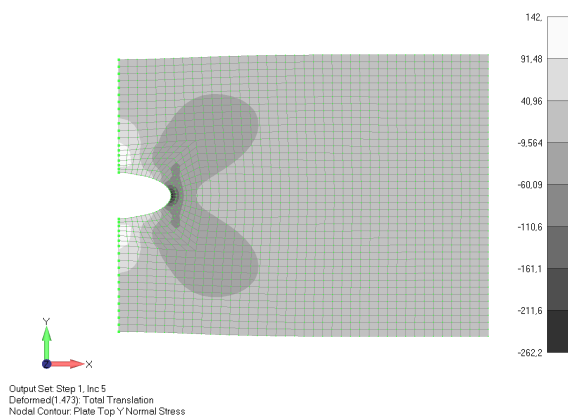


Figura 4.7: Risultati piastra forata PLASTIC, sforzo S_y .

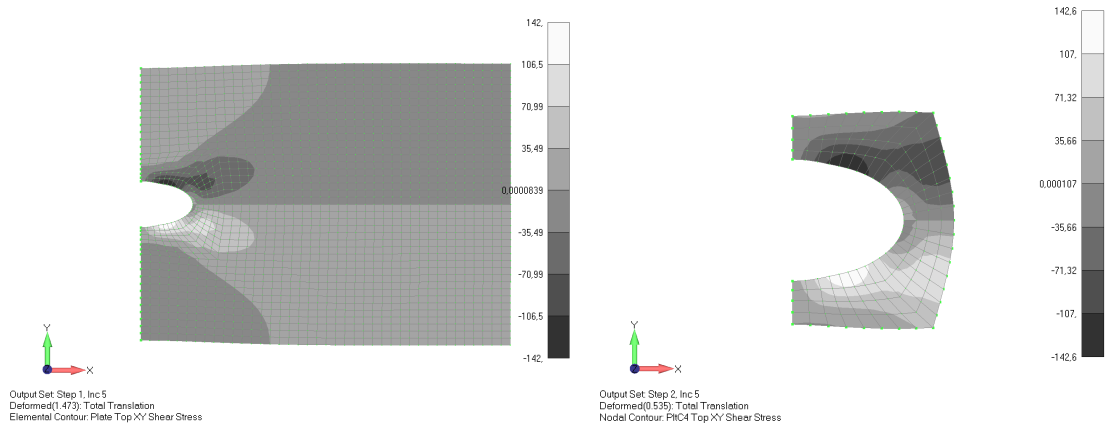


Figura 4.8: Risultati piastra forata, sforzo S_{xy} : modello globale (a sinistra), modello locale (a destra).

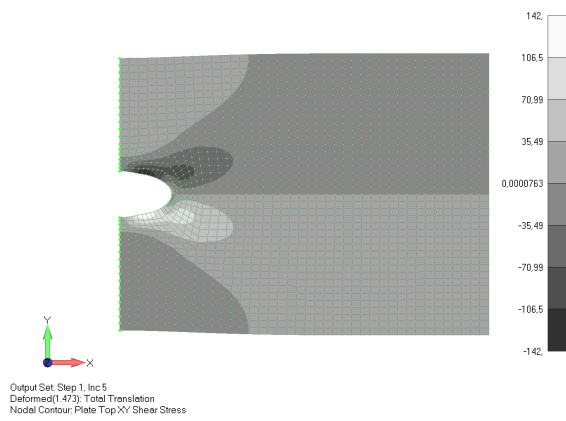


Figura 4.9: Risultati piastra forata PLASTIC, sforzo S_{xy} .

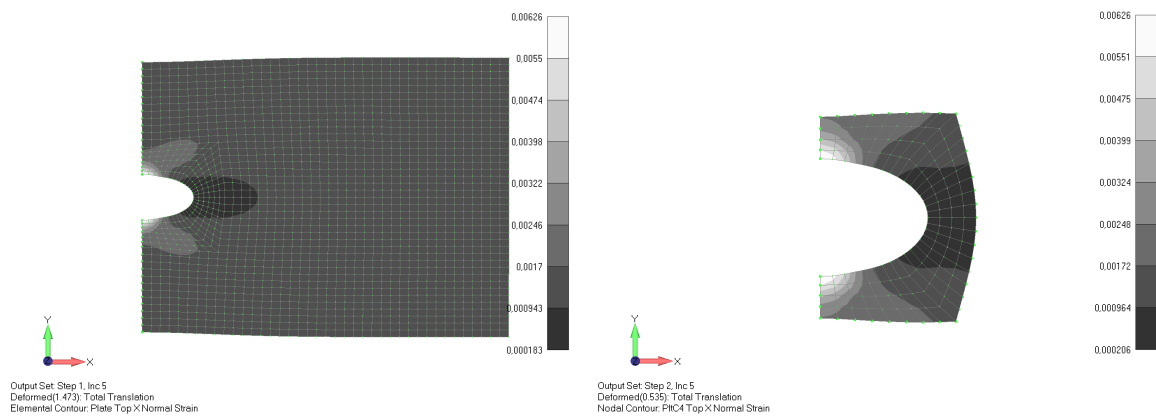


Figura 4.10: Risultati piastra forata, deformazione E_x : modello globale (a sinistra), modello locale (a destra).

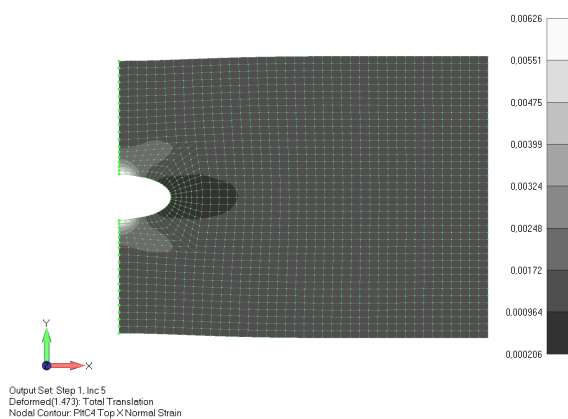


Figura 4.11: Risultati piastra forata PLASTIC, deformazione Ex.

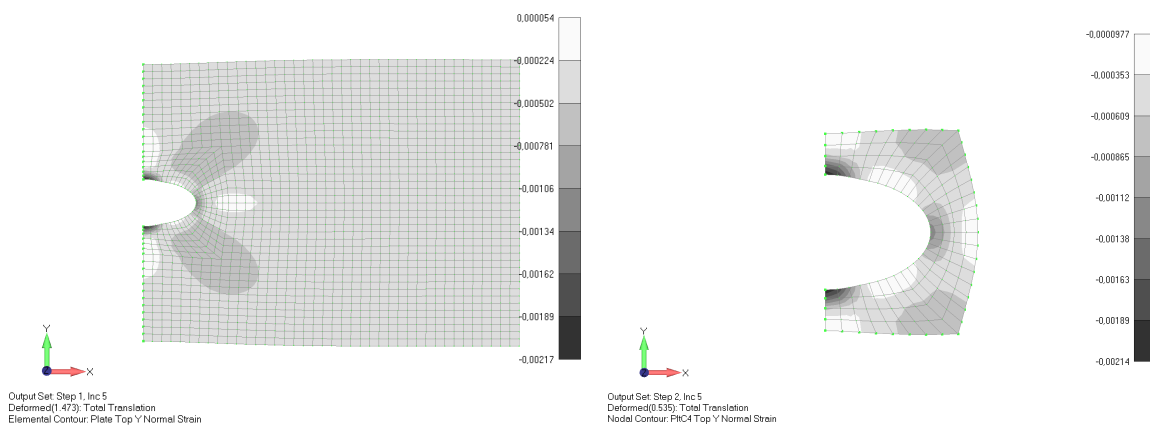


Figura 4.12: Risultati piastra forata, deformazione Ey: modello globale (a sinistra), modello locale (a destra).

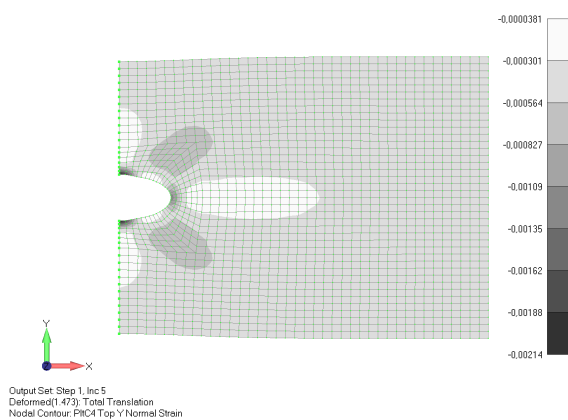


Figura 4.13: Risultati piastra forata PLASTIC, deformazione Ey.

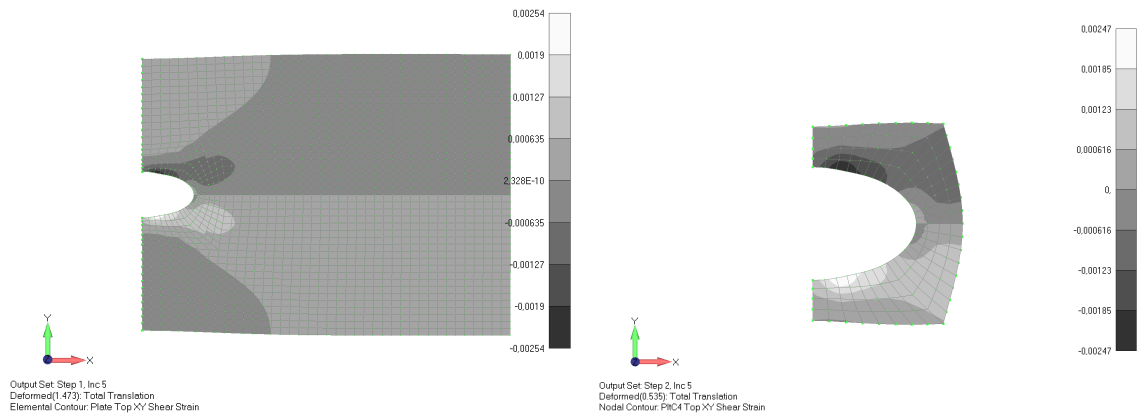


Figura 4.14: Risultati piastra forata, deformazione Exy: modello globale (a sinistra), modello locale (a destra).

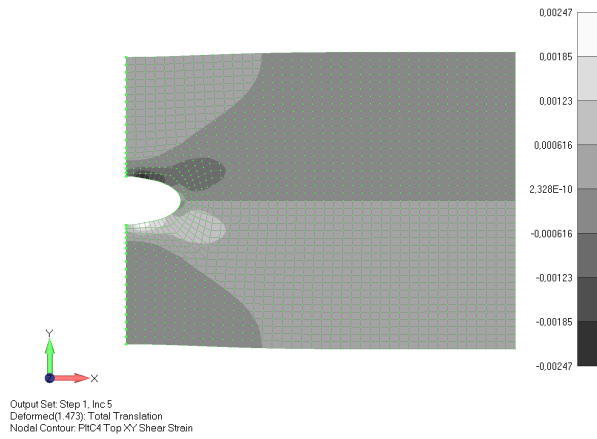


Figura 4.15: Risultati piastra forata PLASTIC, deformazione Exy.

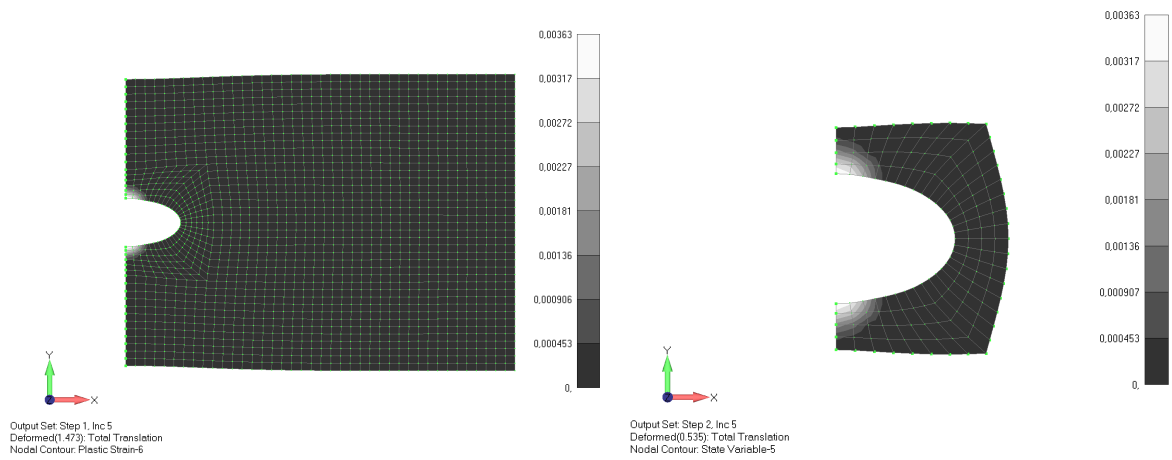


Figura 4.16: Risultati piastra forata, PEEQ: PLASTIC (a sinistra), modello locale (a destra).

Capitolo 5

Conclusioni

In questo lavoro di tesi è sviluppato un metodo di analisi strutturale multi-livello che permette di studiare il comportamento complessivo di una struttura con una descrizione dei fenomeni locali, attraverso l'interazione (two-way) di due modelli: (1) globale contenente l'intera struttura, (2) locale in una regione ristretta utilizzata per descrivere accuratamente il comportamento del materiale.

Il metodo è stato testato su una piastra forata e validato confrontando i risultati con un'analisi elasto-plastica standard, mettendo in luce le potenzialità e le limitazioni di utilizzo. Le potenzialità sono quelle di una metodologia in grado di studiare i fenomeni locali attraverso la descrizione del modello costitutivo del livello locale.

La fase di modellazione dei 2 livelli deve essere eseguita manualmente dall'utente, con la modellazione della regione locale, l'introduzione della lista dei nodi al contorno e degli elementi, una volta definita questa fase la metodologia è in grado di descrivere i fenomeni locali presi in considerazione e utilizzarli per creare l'interazione tra i livelli, aggiornando il modello globale durante l'analisi.

Attraverso la sola modifica del modello costitutivo locale si può modificare quindi la descrizione dei fenomeni da considerare (plasticità, frattura, danneggiamenti) in modo tale da poter analizzare diversi problemi.

Questa metodologia rappresenta una base di partenza per future implementazioni, partendo dalle funzioni di comunicazione introdotte, dall'approccio multi-livello seguito con le funzionalità introdotte nelle Subroutine.

Per quanto riguarda le limitazioni sono relative alla modellazione della regione locale e alla modalità di comunicazione tra i livelli:

- **metodo non automatizzato nella modellazione della regione locale**
- **nodi coincidenti al contorno tra i 2 modelli**
- **stesso numero di elementi e tipologia nella regione critica**
- **stesso numero di incrementi di carico nelle 2 analisi**

5.1 Prospettive future

Le limitazioni e le potenzialità mostrate sono uno stimolo per migliorare e ampliare la metodologia, con l'obiettivo di superare le attuali limitazioni.

Gli obiettivi futuri possono essere divisi in 2 fasi: (1) una prima fase di modifica della procedura per rendere automatica la modellazione delle regioni critiche, oltre alla definizione della lista dei nodi al contorno e degli elementi da aggiornare, (2) una seconda fase di applicazione del metodo a strutture aeronautiche complesse.

La modifica della procedura richiede una revisione dei protocolli di comunicazione tra le Subroutine, in quanto sarà il livello globale a contenere direttamente le informazioni sugli elementi e nodi al contorno necessari per la modellazione locale e lo scambio di informazioni.

Questa realizzazione potrebbe richiedere la programmazione delle analisi in ambiente Python, attraverso opportuni script all'interno del processo iterativo.

In questo processo può rientrare l'implementazione di una procedura per l'omogenizzazione degli effetti locali sul modello globale, superando l'attuale limitazione sul numero e tipologia degli elementi.

Il metodo va in seguito validato con opportune analisi comparative e infine può essere applicato a strutture aeronautiche complesse, come per esempio:

- **pannelli alari rinforzati**
- **longheroni con fori di alleggerimento**
- **giunzioni tra pannelli**
- **pannelli di fusoliera rinforzati**

Appendice A

Implementazione Subroutine

A.1 Struttura generale Subroutine

La struttura generale di una Subroutine è la seguente:

```
SUBROUTINE nome_subroutine(lista_argumenti)
C      .....
C      sezione dichiarativa delle variabili
C      .....
C      sezione esecutiva che definisce la funzione della procedura
C      .....
END SUBROUTINE
```

Segue ora l'esempio di una semplice procedura, calcolo_somma.f, che ha il compito di calcolare la somma tra 2 numeri e stampare il risultato a schermo, attraverso l'utilizzo di una subroutine esterna per effettuare l'operazione richiesta:

```
PROGRAM calcolo_somma
C      .....
C      sezione dichiarativa
C      real    ::    numero_1, numero_2 ! variabili scalari reali
C      real    ::    somma    ! variabile scalare reale
C      .....
C      sezione esecutiva
C      numero_1=2
C      numero_1=5
C      CALL somma(numero_1, numero_2, somma)
C      stampa a schermo del risultato della somma
C      print*, 'risultato=', somma
END PROGRAM
```

```
SUBROUTINE somma(addendo_1, addendo_2, risultato)
C      .....
C      sezione dichiarativa
C      real    ::    addendo_1, addendo_2 ! variabili scalari reali
C      real    ::    risultato    ! variabile scalare reale
C      .....
C      sezione esecutiva
C      risultato = addendo_1 + addendo_2
C      .....
END SUBROUTINE
```

A.2 Struttura generale scambio dati server-client

Segue la struttura generale di 2 programmi, server.f e client.f, che comunicano tra loro scambiandosi numeri interi.

Questa sottosezione si focalizza sugli elementi e le istruzioni essenziali per la comunicazione:

```
PROGRAM client
C   sezione dichiarativa
      INCLUDE 'f77_zmq.h'
      INTEGER(ZMQ_PTR) :: context
      INTEGER(ZMQ_PTR) : local
      CHARACTER*(64)  :: address
      INTEGER :: rc
      INTEGER :: send
      INTEGER :: recv
      address = 'tcp://localhost:5555'
C   .....
C   definizione collegamento
      context = f77_zmq_ctx_new()
      local = f77_zmq_socket(context, ZMQ_REQ)
      rc = f77_zmq_connect(local, address)
C   sezione esecutiva
      send=1
      rc = f77_zmq_send(local, send, 4, 0)
      print *, rc
      rc = f77_zmq_recv(local, recv, 4, 0)
      print *, rc
C   .....
END PROGRAM
```

```
PROGRAM server
C   sezione dichiarativa
      INCLUDE 'f77_zmq.h'
      INTEGER(ZMQ_PTR) :: context
      INTEGER(ZMQ_PTR) : local
      CHARACTER*(64)  :: address
      INTEGER :: rc
      INTEGER :: send
      INTEGER :: recv
      address = 'tcp://*:5555'
C   .....
C   definizione collegamento
      context = f77_zmq_ctx_new()
      local = f77_zmq_socket(context, ZMQ_REP)
      rc = f77_zmq_connect(local, address)
C   sezione esecutiva
      send=2
      rc = f77_zmq_recv(local, send, 4, 0)
      print *, rc
      rc = f77_zmq_send(local, recv, 4, 0)
      print *, rc
C   .....
END PROGRAM
```

A.3 UMAT: modello costitutivo elastico

Subroutine UMAT con modellazione elastica:

```

SUBROUTINE UMAT(STRESS,STATEV,DDSDDE,SSE,SPD,SCD,
1      RPL,DDSDDT,DRPLDE,DRPLDT,
2      STRAN,DSTRAN,TIME,DTIME,TEMP,DTEMP,PREDEF,DPRED,CMNAME,
3      NDI,NSHR,NTENS,NSTATV,PROPS,NPROPS,COORDS,DROT,PNEWDT,
4      CELENT,DFGRD0,DFGRD1,NOEL,NPT,LAYER,KSPT,KSTEP,KINC)
C
  INCLUDE 'ABA.PARAM.INC'
C
  CHARACTER*80 CMNAME
  DIMENSION STRESS(NTENS),STATEV(NSTATV),
1      DDSDE(NTENS,NTENS),
2      DDSDDT(NTENS),DRPLDE(NTENS),
3      STRAN(NTENS),DSTRAN(NTENS),TIME(2),PREDEF(1),DPRED(1),
4      PROPS(NPROPS),COORDS(3),DROT(3,3),DFGRD0(3,3),DFGRD1(3,3)
  REAL      ::      E, NU
C
C      Caratteristiche elastiche del materiale e matrice elastica 3X3
C
  E = PROPS(1)           !E
  NU = PROPS(2)          !NU
  G = E / (2 * (1 + NU)) !G
C
  DO I = 1, NTENS
    DO J = 1, NTENS
      DDSDE(I,J)=0.0
    END DO
  END DO
  ATEMP = 1 - (NU**2)
  DDSDE(1,1) = E / ATEMP
  DDSDE(2,2) = DDSDE(1,1)
  DDSDE(3,3) = G
  DDSDE(1,2) = (E * NU) / ATEMP
  DDSDE(2,1) = DDSDE(1,2)
C
C      Calcolo sforzo nel vettore e salvataggio nelle variabili di stato
C
  DO I = 1, NTENS
    DO J = 1, NTENS
      STRESS(J)=STRESS(J)+DDSDE(J,I)*DSTRAN(I)
    END DO
  END DO
C
  DO I = 1, NTENS
    STATEV(I)= STRESS(I)
  END DO
C
  RETURN
END
```

A.4 DISP: imposizione spostamenti nodali

Subroutine DISP di imposizione di spostamenti nodali.

```

SUBROUTINE DISP(U,KSTEP,KINC,TIME,NODE,NOEL,JDOF,COORDS)
C
  INCLUDE 'ABA.PARAM.INC'
C
  DIMENSION U(3),TIME(3),COORDS(3)
C
  Imposizione degli spostamenti nodali
C
    if ((NODE.GE.1).AND.(NODE.LE.21)) then
      U(1) = -TIME(2)*200
      print *, 'NODO,U(1),TEMPO-TOT', NODE, U(1), TIME(2)
    end if
    if ((NODE.GE.41).AND.(NODE.LE.61)) then
      U(1) = TIME(2)*200
      print *, 'NODO,U(1),TEMPO-TOT', NODE, U(1), TIME(2)
    end if
C
  RETURN
END

```

A.5 URDFIL: lettura dei risultati

Subroutine URDFIL di lettura dei risultati dal file .fil:

```

SUBROUTINE URDFIL(LSTOP,LOVRWRT,KSTEP,KINC,DTIME,TIME)
C
  INCLUDE 'ABA.PARAM.INC'
C
  DIMENSION ARRAY(513),JRRAY(NPRECD,513),TIME(2)
  EQUIVALENCE (ARRAY(1),JRRAY(1,1))
C
  Lettura degli spostamenti nodali
C
  CALL POSFIL(KSTEP,KINC,ARRAY,JRCD)
  do K1=1,999999
    CALL DBFILE(0,ARRAY,JRCD)
    if (JRCD.NE.0) go to 110
    KEY = JRRAY(1,2)
C
C  Identificativo 101 corrispondente ai risultati nodali di spostamento
C
    if (KEY.EQ.101) then
      NODE=JRRAY(1,3)
      u=ARRAY(4)
      print *, 'NODO,U1,INC', NODE, u, KINC
    end if
  end do
110  continue
C
  RETURN
END

```

A.6 Modello globale-locale: piastra quadrata isotropa

Subroutine URDFIL contenuta nell'analisi globale:

```
SUBROUTINE URDFIL(LSTOP,LOVRWRT,KSTEP,KINC,DTIME,TIME)
C
      INCLUDE 'aba_param_dp.inc'
      INCLUDE 'f77_zmq/f77_zmq.h'
C
      DIMENSION ARRAY(513),JRRAY(NPRECD,513),TIME(2)
      EQUIVALENCE (ARRAY(1),JRRAY(1,1))
C
C
C  definizione variabili
C
C
      integer(ZMQ_PTR), save :: context
      integer(ZMQ_PTR), save :: local
      character*64, save :: address
      logical, save :: already_called=.FALSE.
      integer :: rc
      double precision :: richiesta_nodi
      double precision :: r1,r2,r3,r4,r5
      double precision :: nodi(4)
      double precision :: u(4),v(4),w(4)
      double precision :: rx(4),ry(4),rz(4)
C
C
C  connessione tra i 2 modelli (GLOBALE-LOCALE)
C
C
      if (.not. already_called) then
        address = 'tcp://localhost:5556'
        context = f77_zmq_ctx_new ()
        local = f77_zmq_socket (context, ZMQ_REQ)
        rc = f77_zmq_connect (local, address)
        already_called = .TRUE.
        write(*,*) 'CLSIAMO_CONNESSI,rc=', rc
      endif
C
C
C  ricezione richiesta spostamenti nodali
C
C
      richiesta_nodi = 1
      rc = f77_zmq_send (local, richiesta_nodi, 8, 0)
      print *, 'rc=', rc
      rc = f77_zmq_recv (local, nodi, 8*size(nodi), 0)
      print *, 'rc=', rc
      print *, 'nodi=', nodi
C
C
C  lettura spostamenti nodali
C
C
```

```

do K1=1,999999
  CALL DBFILE(0,ARRAY,JRCD)
  if (JRCD.NE.O) go to 110
  KEY = JRRAY(1,2)
  if (KEY.EQ.101) then
    do i=1, 4
      NODE=JRRAY(1,3)
      if (NODE.EQ.nodi(i)) then
        u(i)=ARRAY(4)
        v(i)=ARRAY(5)
        w(i)=ARRAY(6)
        rx(i)=ARRAY(7)
        ry(i)=ARRAY(8)
        rz(i)=ARRAY(9)
      end if
    end do
  end if
end do
110 continue
C-----
C
C invio spostamenti nodali
C
C-----
rc = f77_zmq_send (local, u, 8*size(u), 0)
  print *, 'rc=', rc
  print *, 'u=', u
rc = f77_zmq_recv (local, r1, 8, 0)
  print *, 'rc=', rc
rc = f77_zmq_send (local, v, 8*size(v), 0)
  print *, 'rc=', rc
  print *, 'v=', v
rc = f77_zmq_recv (local, r2, 8, 0)
  print *, 'rc=', rc
rc = f77_zmq_send (local, w, 8*size(w), 0)
  print *, 'rc=', rc
  print *, 'w=', w
rc = f77_zmq_recv (local, r3, 8, 0)
  print *, 'rc=', rc
rc = f77_zmq_send (local, rx, 8*size(rx), 0)
  print *, 'rc=', rc
  print *, 'rx=', rx
rc = f77_zmq_recv (local, r4, 8, 0)
  print *, 'rc=', rc
rc = f77_zmq_send (local, ry, 8*size(ry), 0)
  print *, 'rc=', rc
  print *, 'ry=', ry
rc = f77_zmq_recv (local, r5, 8, 0)
  print *, 'rc=', rc
rc = f77_zmq_send (local, rz, 8*size(rz), 0)
  print *, 'rc=', rc
  print *, 'rz=', rz
C-----
RETURN
END

```

Subroutine DISP contenuta nell'analisi locale:

```
SUBROUTINE DISP (U,KSTEP,KINC,TIME,NODE,NOEL,JDOF,COORDS)
C
      INCLUDE 'aba_param_dp.inc'
      INCLUDE 'f77_zmq/f77_zmq.h'
C
      CHARACTER*80 CMNAME
      DIMENSION U(3),TIME(3),COORDS(3)
C
C
C  definizione variabili
C
C
C
      integer                ::      index
      integer(ZMQ_PTR), save ::      context
      integer(ZMQ_PTR), save ::      local
      character*(64), save  ::      address
      integer                ::      rc
      double precision      ::      richiesta_nodi
      double precision      ::      r1,r2,r3,r4,r5
      double precision, save ::      nodi(4)
      double precision, save ::      uu(4),vv(4),ww(4)
      double precision, save ::      rx(4),ry(4),rz(4)
C
C
C  connessione tra i 2 modelli (LOCALE-GLOBALE)
C
C
C
      address = 'tcp://*:5556'
      context = f77_zmq_ctx_new ()
      local   = f77_zmq_socket (context, ZMQ_REP)
      rc      = f77_zmq_bind (local, address)
      write(*,*) 'CLSIAMO CONNESSI,rc=', rc
C
C
C  richiesta spostamenti nodali
C
C
C
      do i=1, 4
        nodi(i) = 0
      end do
      nodi(1)=243
      nodi(2)=244
      nodi(3)=262
      nodi(4)=263
      if (rc.EQ.-1) go to 110
      rc = f77_zmq_recv (local, richiesta_nodi, 8, 0)
      print *, 'rc=', rc
      rc = f77_zmq_send (local, nodi, 8*size(nodi), 0)
      print *, 'rc=', rc
C
C
C  spostamenti nodali dal modello GLOBALE
C
C
```

```

r1=2
r2=3
r3=4
r4=5
r5=6
rc = f77_zmq_recv (local , uu, 8*size(uu), 0)
  print *, 'rc=', rc
  print *, 'u=', uu
rc = f77_zmq_send (local , r1, 8, 0)
  print *, 'rc=', rc
rc = f77_zmq_recv (local , vv, 8*size(vv), 0)
  print *, 'rc=', rc
  print *, 'v=', vv
rc = f77_zmq_send (local , r2, 8, 0)
  print *, 'rc=', rc
rc = f77_zmq_recv (local , ww, 8*size(ww), 0)
  print *, 'rc=', rc
  print *, 'w=', ww
rc = f77_zmq_send (local , r3, 8, 0)
  print *, 'rc=', rc
rc = f77_zmq_recv (local , rx, 8*size(rx), 0)
  print *, 'rc=', rc
  print *, 'rx=', rx
rc = f77_zmq_send (local , r4, 8, 0)
  print *, 'rc=', rc
rc = f77_zmq_recv (local , ry, 8*size(ry), 0)
  print *, 'rc=', rc
  print *, 'ry=', ry
rc = f77_zmq_send (local , r5, 8, 0)
  print *, 'rc=', rc
rc = f77_zmq_recv (local , rz, 8*size(rz), 0)
  print *, 'rc=', rc
  print *, 'rz=', rz
C
110 continue
  do i=1, 4
    if (NODE.EQ.nodi(i)) then
      select case(JDOF)
        case(1)
          U(1) = uu(i)
        case(2)
          U(1) = vv(i)
        case(3)
          U(1) = ww(i)
        case(4)
          U(1) = rx(i)
        case(5)
          U(1) = ry(i)
        case(6)
          U(1) = rz(i)
      end select
    end if
  end do
C
RETURN
END

```

A.7 Modello globale-locale: piastra forata isotropa modello elasto-plastico

```

C*****
C
C Modello Globale
C
C*****
C-----
C
C Modulo per la definizione delle variabili globali
C (context, local)
C per
C la connessione Globale-Locale
C
C-----
C
C MODULE CONN.DATA
C INCLUDE 'f77_zmq/f77_zmq.h'
C   integer (ZMQ_PTR), save :: context
C   integer (ZMQ_PTR), save :: local
C END MODULE
C-----
C
C Subroutine per la connessione Globale-Locale
C lato Globale (Client)
C
C-----
C
C SUBROUTINE CLIENT_CONN()
C USE CONN.DATA
C   character*64, save :: address
C   logical, save :: already_called = .FALSE.
C   integer :: rc
C   if (.not. already_called) then
C     address = 'tcp://localhost:5556'
C     context = f77_zmq_ctx_new ()
C     local = f77_zmq_socket (context, ZMQ_REQ)
C     rc = f77_zmq_connect (local, address)
C     already_called = .TRUE.
C     write(*,*) 'CLSIAMO CONNESSI,rc=', rc
C   end if
C END SUBROUTINE
C-----
C
C UMAT: legame costitutivo del modello Globale
C con richiesta DDSDE e STRESS corrente dal
C modello Locale
C
C-----
C
C SUBROUTINE UMAT(STRESS,STATEV,DDSDE,SSE,SPD,SCD,
1   RPL,DDSDDT,DRPLDE,DRPLDT,
2   STRAN,DSTRAN,TIME,DTIME,TEMP,DTEMP,PREDEF,DPRED,CMNAME,
3   NDI,NSHR,NTENS,NSTATV,PROPS,NPROPS,COORDS,DROT,PNEWDT,
4   CELENT,DFGRD0,DFGRD1,NOEL,NPT,LAYER,KSPT,KSTEP,KINC)
C

```

```

USE CONN_DATA
INCLUDE 'aba_param_dp.inc'

C
CHARACTER*80 CMNAME
DIMENSION STRESS(NTENS),STATEV(NSTATV),
1     DDSdde(NTENS,NTENS),
2     DDSDDT(NTENS),DRPLDE(NTENS),
3     STRAN(NTENS),DSTRAN(NTENS),TIME(2),PREDEF(1),DPRED(1),
4     PROPS(NPROPS),COORDS(3),DROT(3,3),DFGRD0(3,3),DFGRD1(3,3)

DIMENSION STRANT(NTENS)
DIMENSION DDFDE(3),DDMDE(3),DCDDF(3,3),DCDDM(3,3)

C
C
C Definizione variabili
C
C
integer :: elemento_ricevuto
integer :: elemento_globale
integer :: pti_integrazione_ricevuto
integer :: pti_integrazione
integer :: n_inc_ricevuto
integer :: n_inc
integer :: richiesta_invio_sforzi
integer :: richiesta_invio_ddsdde
double precision :: d_stran(3), sforzo(3), k(3,3)

C
C
C Connessione tra i modelli (GLOBALE-LOCALE)
C
C
CALL CLIENT_CONN()

C
C
C Caratteristiche elastiche del materiale
C e
C richiesta DDSdde e STRESS corrente
C dal modello Locale
C
C
elemento_globale = NOEL
pti_integrazione = NPT
n_inc = KINC
d_stran = DSTRAN
invio_sforzi = 1
invio_ddsdde = 2
rc = f77_zmq_send(local,n_inc,4,0)
C      print *, 'rc=', rc
rc = f77_zmq_recv(local,n_inc_ricevuto,4,0)
C      print *, 'rc=', rc
rc = f77_zmq_send(local,elemento_globale,4,0)
C      print *, 'rc=', rc
rc = f77_zmq_recv(local,elemento_ricevuto,4,0)
C      print *, 'rc=', rc
rc = f77_zmq_send(local,pti_integrazione,4,0)
C      print *, 'rc=', rc

```

A.7. MODELLO GLOBALE-LOCALE: PIASTRA FORATA ISOTROPA MODELLO ELASTO-PLASTICO

```

rc = f77_zmq_recv (local, pti_integrazione_ricevuto, 4, 0)
C      print *, 'rc=', rc
rc = f77_zmq_send (local, d_stran, 8*size(d_stran), 0)
C      print *, 'rc=', rc
rc = f77_zmq_recv (local, dstan_ricevuto, 4, 0)
C      print *, 'rc=', rc
rc = f77_zmq_send (local, richiesta_invio_sforzi, 4, 0)
C      print *, 'rc=', rc
rc = f77_zmq_recv (local, sforzo, 8*size(sforzo), 0)
C      print *, 'rc=', rc
rc = f77_zmq_send (local, richiesta_invio_ddsde, 4, 0)
C      print *, 'rc=', rc
rc = f77_zmq_recv (local, k, 8*size(k), 0)
C      print *, 'rc=', rc
C
C
C Matrice elastica 3X3, ricevuta dal modello locale
C
C
C
      DDSDE(1,1) = k(1,1)
      DDSDE(1,2) = k(1,2)
      DDSDE(1,3) = k(1,3)
      DDSDE(2,1) = k(2,1)
      DDSDE(2,2) = k(2,2)
      DDSDE(2,3) = k(2,3)
      DDSDE(3,1) = k(3,1)
      DDSDE(3,2) = k(3,2)
      DDSDE(3,3) = k(3,3)
C
C
C Salvataggio sforzo aggiornato, ricevuto dal modello
C locale
C
C
C
      STRESS(1)=sforzo(1)
      STRESS(2)=sforzo(2)
      STRESS(3)=sforzo(3)
C
C      RETURN
C      END
C
C
C URDFIL: lettura spostamenti al contorno e invio
C al modello Locale
C
C
C
      SUBROUTINE URDFIL(LSTOP,LOVRWRT,KSTEP,KINC,DTIME,TIME)
C
C      USE CONN_DATA
C      INCLUDE 'aba_param_dp.inc'
C
C      DIMENSION ARRAY(513),JRRAY(NPRECD,513),TIME(2)
C      EQUIVALENCE (ARRAY(1),JRRAY(1,1))
C
C
C Definizione variabili

```

```

C
C
      integer          :: risposta_fine
      integer          :: continuo
      integer          :: richiesta_continuazione
      integer          :: permesso_proseguì
      integer          :: fine_urdfil_locale
      integer          :: richiesta_nodi
      integer          :: r1,r2,r3,r4,r5
      integer          :: nodi(41)
      double precision :: t
      double precision, save :: u(41),v(41),w(41)
      double precision, save :: rx(41),ry(41),rz(41)
C
C
C Connessione tra i modelli (GLOBALE-LOCALE)
C
C
      CALL CLIENT_CONN()
C
C
C Ricezione richiesta spostamenti nodali, nodi
C e
C lettura spostamenti nodali
C
C
      richiesta_nodi = 1
      richiesta_continuazione = 2
      risposta_fine = KINC
      fine_urdfil_locale = 0
C
      rc = f77_zmq_send (local, fine_urdfil_locale, 4, 0)
      print *, 'rc=', rc
      rc = f77_zmq_recv (local, permesso_proseguì, 4, 0)
      print *, 'rc=', rc
      print *, 'KINC_globale=', risposta_fine
C
      rc = f77_zmq_send (local, risposta_fine, 4, 0)
      print *, 'rc=', rc
      rc = f77_zmq_recv (local, permesso_proseguì, 4, 0)
      print *, 'rc=', rc
      rc = f77_zmq_send (local, richiesta_nodi, 8, 0)
      print *, 'rc=', rc
      rc = f77_zmq_recv (local, nodi, 4*size(nodi), 0)
      print *, 'rc=', rc
C
      CALL POSFIL(KSTEP,KINC,ARRAY,JRCD)
      print *, 'KSTEP_globale, KINC_globale', KSTEP, KINC
C
      do K1=1,999999
      CALL DBFILE(0,ARRAY,JRCD)
      if (JRCD.NE.0) go to 110
      KEY = JRRAY(1,2)
      if (KEY.EQ.101) then
      NODE=JRRAY(1,3)
      do i=1, 41

```

```

        if (NODE.EQ.nodi(i)) then
            u(i)=ARRAY(4)
            v(i)=ARRAY(5)
            w(i)=ARRAY(6)
            rx(i)=ARRAY(7)
            ry(i)=ARRAY(8)
            rz(i)=ARRAY(9)
        end if
    end do
end if
end do
110    continue
C
C
C invio spostamenti nodali
C
C
C
C      rc = f77_zmq_send (local, u, 8*size(u), 0)
C      print *, 'rc=', rc
C      print *, 'u=', u
C      rc = f77_zmq_recv (local, r1, 4, 0)
C      print *, 'rc=', rc
C      rc = f77_zmq_send (local, v, 8*size(v), 0)
C      print *, 'rc=', rc
C      print *, 'v=', v
C      rc = f77_zmq_recv (local, r2, 4, 0)
C      print *, 'rc=', rc
C      rc = f77_zmq_send (local, w, 8*size(w), 0)
C      print *, 'rc=', rc
C      print *, 'w=', w
C      rc = f77_zmq_recv (local, r3, 4, 0)
C      print *, 'rc=', rc
C      rc = f77_zmq_send (local, rx, 8*size(rx), 0)
C      print *, 'rc=', rc
C      print *, 'rx=', rx
C      rc = f77_zmq_recv (local, r4, 4, 0)
C      print *, 'rc=', rc
C      rc = f77_zmq_send (local, ry, 8*size(ry), 0)
C      print *, 'rc=', rc
C      print *, 'ry=', ry
C      rc = f77_zmq_recv (local, r5, 4, 0)
C      print *, 'rc=', rc
C      rc = f77_zmq_send (local, rz, 8*size(rz), 0)
C      print *, 'rc=', rc
C      print *, 'rz=', rz
C      rc = f77_zmq_recv (local, continuo, 4, 0)
C      print *, 'rc=', rc
C
RETURN
END

C*****
C
C Modello Locale
C
C*****

```



```

        integer          :: continuo
        integer          :: n_inc
        integer          :: n_inc_ricevuto
        integer          :: elemento_globale
        integer          :: elemento_ricevuto
        integer          :: pti_integrazione
        integer          :: pti_integrazione_ricevuto
        integer          :: dstran_ricevuto
        integer          :: richiesta_invio_sforzi
        integer          :: richiesta_invio_ddsdde
        integer          :: c_1, c_2, c_3, c_4
        double precision :: DDSDE(3,3),STRESS(3),DSTRAN(3)
        double precision :: STATEV(5)
        double precision :: sforzo_1(1280),sforzo_2(1280)
        double precision :: sforzo_3(1280)
        double precision :: statev_1(1280),statev_2(1280)
        double precision :: statev_3(1280)
        double precision :: statev_4(1280),statev_5(1280)
        double precision :: elementi(128)
        double precision :: PROPS(18)
C_____
C
C CALL Subroutine CLIENT_CONN per la
C connessione tra i modelli (GLOBALE-LOCALE)
C usando variabili globali (context, address, local)
C_____
C          CALL SERVER_CONN()
C_____
C
C Ricezione richiesta DDSDE e STRESS dal globale
C e
C invio DDSDE e STRESS corrente
C_____
C          do i=1, 41
C             elementi(i) = 0
C          end do
C
C Lista elementi della regione locale
C
C             elementi(1)=3103
C          .....
C             elementi(128)=3294
C
C if (KINC.EQ.5) go to 150
C
C          continuo = 1
C          n_inc_ricevuto = 2
C          elemento_ricevuto = 3
C          pti_integrazione_ricevuto = 4
C          dstran_ricevuto = 5
C
C          rc = f77_zmq_recv (local, n_inc, 4, 0)
C          print *, 'rc=', rc
C          rc = f77_zmq_send (local, n_inc_ricevuto, 4, 0)

```

```

C          print *, 'rc=', rc
          d=0
          e=0
          print *, 'KSTEP, KINC', KSTEP, KINC
          CALL POSFIL(KSTEP, KINC, ARRAY, JRCD)
C
do K1=1,999999
  CALL DBFILE(0, ARRAY, JRCD)
  if (JRCD.NE.0) go to 110
  KEY = JRRAY(1,2)
  if (KEY.EQ.5) then
    e=e+1
    statev_1(e) = ARRAY(3)
    statev_2(e) = ARRAY(4)
    statev_3(e) = ARRAY(5)
    statev_4(e) = ARRAY(6)
    statev_5(e) = ARRAY(7)
  end if
  if (KEY.EQ.11) then
    d=d+1
    sforzo_1(d) = ARRAY(3)
    sforzo_2(d) = ARRAY(4)
    sforzo_3(d) = ARRAY(5)
  end if
end do
110      continue
C
DO WHILE (n_inc.NE.0)
C
  if (.not.salto) then
    rc = f77_zmq_recv (local, n_inc, 4, 0)
    print *, 'rc=', rc
    rc = f77_zmq_send (local, n_inc_ricevuto, 4, 0)
    print *, 'rc=', rc
    salto=.FALSE.
  end if
C
  if (n_inc.EQ.0) go to 130
  rc = f77_zmq_recv (local, elemento_globale, 4, 0)
  print *, 'rc=', rc
  rc = f77_zmq_send (local, elemento_ricevuto, 4, 0)
  print *, 'rc=', rc

  rc = f77_zmq_recv (local, pti_integrazione, 4, 0)
  print *, 'rc=', rc
  rc = f77_zmq_send (local, pti_integrazione_ricevuto, 4, 0)
  print *, 'rc=', rc
  rc = f77_zmq_recv (local, DSTRAN, 8*size(DSTRAN), 0)
  print *, 'rc=', rc
  rc = f77_zmq_send (local, dstran_ricevuto, 4, 0)
  print *, 'rc=', rc
C
  do i=1, 128
    if (elemento_globale.EQ.elementi(i)) then
      select case(pti_integrazione)
        case(1)

```

```

c_1=249+(i*8)
STRESS(1)=sforzo_1(c_1)
STRESS(2)=sforzo_2(c_1)
STRESS(3)=sforzo_3(c_1)
STATEV(1)=statev_1(c_1)
STATEV(2)=statev_2(c_1)
STATEV(3)=statev_3(c_1)
STATEV(4)=statev_4(c_1)
STATEV(5)=statev_5(c_1)
case(2)
c_2=251+(i*8)
STRESS(1)=sforzo_1(c_2)
STRESS(2)=sforzo_2(c_2)
STRESS(3)=sforzo_3(c_2)
STATEV(1)=statev_1(c_2)
STATEV(2)=statev_2(c_2)
STATEV(3)=statev_3(c_2)
STATEV(4)=statev_4(c_2)
STATEV(5)=statev_5(c_2)
case(3)
c_3=253+(i*8)
STRESS(1)=sforzo_1(c_3)
STRESS(2)=sforzo_2(c_3)
STRESS(3)=sforzo_3(c_3)
STATEV(1)=statev_1(c_3)
STATEV(2)=statev_2(c_3)
STATEV(3)=statev_3(c_3)
STATEV(4)=statev_4(c_3)
STATEV(5)=statev_5(c_3)
case(4)
c_4=255+(i*8)
STRESS(1)=sforzo_1(c_4)
STRESS(2)=sforzo_2(c_4)
STRESS(3)=sforzo_3(c_4)
STATEV(1)=statev_1(c_4)
STATEV(2)=statev_2(c_4)
STATEV(3)=statev_3(c_4)
STATEV(4)=statev_4(c_4)
STATEV(5)=statev_5(c_4)
end select
end if
end do
C
C
C Caratteristiche elastiche locali
C
C
C
PROPS(1) = 206900      !E
PROPS(2) = 0.29       !NU
PROPS(3) = 450        !SIGMAY_0
PROPS(4) = 0          !EPLAST_0
PROPS(5) = 470        !SIGMAY_1
PROPS(6) = 0.005      !EPLAST_1
PROPS(7) = 514        !SIGMAY_2
PROPS(8) = 0.0178     !EPLAST_2
PROPS(9) = 590        !SIGMAY_3

```

```

        PROPS(10) = 0.0378      !EPLAST_3
        PROPS(11) = 642        !SIGMAY_4
        PROPS(12) = 0.0678      !EPLAST_4
        PROPS(13) = 700        !SIGMAY_3
        PROPS(14) = 0.1478      !EPLAST_3
        PROPS(15) = 790        !SIGMAY_4
        PROPS(16) = 0.498      !EPLAST_4
        PROPS(17) = 840        !SIGMAY_4
        PROPS(18) = 1          !EPLAST_4
        NPROPS = 18
        NTENS = 3
        NSTATV = 5

C
        CALL UMAT(STRESS,STATEV,DDSDDE,SSE,SPD,SCD,
1      RPL,DDSDDT,DRPLDE,DRPLDT,
2      STRAN,DSTRAN,TIME,DTIME,TEMP,DTEMP,PRED,DPRED,CMNAME,
3      NDI,NSHR,NTENS,NSTATV,PROPS,NPROPS,COORDS,DROT,PNEWDT,
4      CELENT,DFGRD0,DFGRD1,NOEL,NPT,LAYER,KSPT,KSTEP,KINC)

C
        rc = f77_zmq_recv (local, richiesta_invio_sforzi, 4, 0)
C
        print *, 'rc=', rc
        rc = f77_zmq_send (local, STRESS, 8*size(STRESS), 0)
C
        print *, 'rc=', rc
        rc = f77_zmq_recv (local, richiesta_invio_ddsde, 4, 0)
C
        print *, 'rc=', rc
        rc = f77_zmq_send (local, DDSDDE, 8*size(DDSDDE), 0)
C
        print *, 'rc=', rc
C
C
130      continue
C
        salto=.FALSE.
        END DO
        salto=.TRUE.

C
150      continue
C
        RETURN
        END

C
C
C UMAT: legame costitutivo elasto-plastico del modello Locale
C      (codice derivato da: EA de Souza Neto, Peric,
C      DRJ Owen. Computational methods for plasticity - theory
C      and applications, Wiley, 2008)
C
C
C
C
C
        SUBROUTINE UMAT(STRESS,STATEV,DDSDDE,SSE,SPD,SCD,
1      RPL,DDSDDT,DRPLDE,DRPLDT,
2      STRAN,DSTRAN,TIME,DTIME,TEMP,DTEMP,PRED,DPRED,CMNAME,
3      NDI,NSHR,NTENS,NSTATV,PROPS,NPROPS,COORDS,DROT,PNEWDT,
4      CELENT,DFGRD0,DFGRD1,NOEL,NPT,LAYER,KSPT,KSTEP,KINC)

C
        INCLUDE 'aba_param_dp.inc'

C
        CHARACTER*80 CMNAME

```

A.7. MODELLO GLOBALE-LOCALE: PIASTRA FORATA ISOTROPA MODELLO ELASTO-PLASTICO

```

      DIMENSION STRESS(NTENS) ,STATEV(NSTATV) ,
1      DDSDE(NTENS,NTENS) ,
2      DDSDDT(NTENS) ,DRPLDE(NTENS) ,
3      STRAN(NTENS) ,DSTRAN(NTENS) ,TIME(2) ,PREDEF(1) ,DPRED(1) ,
4      PROPS(NPROPS) ,COORDS(3) ,DROT(3,3) ,DFGRD0(3,3) ,DFGRD1(3,3)

      DIMENSION STRANT(NTENS)
      DIMENSION DDFDE(3) , DDMDE(3) , DCDDF(3,3) , DCDDM(3,3)
C
      DIMENSION FOID(4,4) ,SOID(4) ,VECN(3)
      DIMENSION STRAT(3) , STREST(3) , EET(3) , TABLE(2,8)
      REAL      :: E, NU
      DATA
1      R0      ,RP5      ,R1      ,R2      ,R3      ,R4      ,R6      ,TOLER /
2      0.0D0,0.5D0,1.0D0,2.0D0,3.0D0,4.0D0,6.0D0,1.D-06/
      DATA MXITER / 10 /
C Local arrays
      DATA
1      FOID(1,1) ,FOID(1,2) ,FOID(1,3) ,FOID(1,4) /
2      1.0D0      ,0.0D0      ,0.0D0      ,0.0D0      /
3      FOID(2,1) ,FOID(2,2) ,FOID(2,3) ,FOID(2,4) /
4      0.0D0      ,1.0D0      ,0.0D0      ,0.0D0      /
5      FOID(3,1) ,FOID(3,2) ,FOID(3,3) ,FOID(3,4) /
6      0.0D0      ,0.0D0      ,0.5D0      ,0.0D0      /
7      FOID(4,1) ,FOID(4,2) ,FOID(4,3) ,FOID(4,4) /
8      0.0D0      ,0.0D0      ,0.0D0      ,1.0D0      /
      DATA
1      SOID(1)      ,SOID(2)      ,SOID(3)      ,SOID(4)      /
2      1.0D0      ,1.0D0      ,0.0D0      ,1.0D0      /
C
C
C -----
C      UMAT FOR ISOTROPIC ELASTICITY AND ISOTROPIC PLASTICITY
C      FOR PLANE STRESS PROBLEM
C -----
C      PROPS(1) - E
C      PROPS(2) - NU
C      PROPS(3) - SIGMAY
C      STATEV(1) - EEL1
C      STATEV(2) - EEL2
C      STATEV(3) - EEL3
C      STATEV(4) - EEL12
C      STATEV(5) - EEQPLAS
C      STRAT - EEL + DELTA.EEL
C -----
C      internal variables
C      DGAMA = R0              !delta gamma
C      IFPLAS = 0
C      EPBARN = STATEV(5)      !plastic strain
C -----
C
C      elastic porperties
C -----
C      E = PROPS(1)              !E
C      NU = PROPS(2)              !NU
C      G = E / (R2 * (R1 + NU))  !G

```

```

BULK = E / (R3*(R1-R2*NU))      !BULK (K)
R2G = R2*G                      !2G
R4G = R4*G                      !4G
R1D3=R1/R3                      !1/3
R1D6=R1/R6                      !1/6
R2D3=R2*R1D3                    !2/3
SQR2D3=SQR(R2D3)              !SQRT(2/3)
R4GD3=R4G*R1D3                  !(4/3)G
STRAT(1)= STATEV(1)+DSTRAN(1)
STRAT(2)= STATEV(2)+DSTRAN(2)
STRAT(3)= STATEV(3)+DSTRAN(3)
C
C
C   Volumetric strain
C
C
C   FACTOR=R2G/(BULK+R4GD3)
C   EEV = (STRAT(1) + STRAT(2))*FACTOR
C
C
C   Elastic trial deviatoric strain
C   Elastic trial stress components
C
C
C   EEVD3 = EEV/R3
C   EET(1) = STRAT(1)-EEVD3
C   EET(2) = STRAT(2)-EEVD3
C   EET(3) = STRAT(3)*RP5
C   PT = BULK*EEV
C   STREST(1) = R2G*EET(1)+PT
C   STREST(2) = R2G*EET(2)+PT
C   STREST(3) = R2G*EET(3)
C
C
C   Hardening curve
C
C
C   TABLE(1,1)=PROPS(3)
C   TABLE(1,2)=PROPS(5)
C   TABLE(1,3)=PROPS(7)
C   TABLE(1,4)=PROPS(9)
C   TABLE(1,5)=PROPS(11)
C   TABLE(2,1)=PROPS(4)
C   TABLE(2,2)=PROPS(6)
C   TABLE(2,3)=PROPS(8)
C   TABLE(2,4)=PROPS(10)
C   TABLE(2,5)=PROPS(12)
C   NVALUE=NPROPS/2-1
C   CALL AHARD(SIGMAY,HARD,EPBARN,TABLE,NVALUE)
C
C
C   Compute yield function value as trial state
C
C
C   A1=(STREST(1)+STREST(2))*(STREST(1)+STREST(2))
C   A2=(STREST(2)-STREST(1))*(STREST(2)-STREST(1))

```

A.7. MODELLO GLOBALE-LOCALE: PIASTRA FORATA ISOTROPA MODELLO ELASTO-PLASTICO

```

A3=STREST(3)*STREST(3)
XI=R1D6*A1+RP5*A2+R2*A3
PHI=RP5*XI-R1D3*SIGMAY*SIGMAY
C
C
C      Determine if actively yielding
C
C
C      IF (PHI/SIGMAY.GT.TOLER)THEN
C
C
C      Solve Newton iteration
C
C
C      IFPLAS=1
C      EPBAR=EPBARN
C      SQRTEXI=SQRT(XI)
C      B1=R1
C      B2=R1
C      FMODU=E/(R3*(R1-NU))
C      DO NRITER=1,MXITER
C Compute residual derivative
C      CALL AHARD(SIGMAY,HARD,EPBAR,TABLE,NVALUE)
C      DXI=-A1*FMODU/(R3*B1*B1*B1)-R2G*(A2+R4*A3)/(B2*B2*B2)
C      HBAR=R2*SIGMAY*HARD*SQR2D3*(SQRTEXI+DGAMA*DXI/(R2*SQRTEXI))
C      DPHI=RP5*DXI-R1D3*HBAR
C Compute Newton-Raphson increment and update equation variable DGAMA
C      DGAMA=DGAMA-PHI/DPHI
C Compute new residual (yield function value)
C      B1=R1+FMODU*DGAMA
C      B2=R1+R2G*DGAMA
C      XI=R1D6*A1/(B1*B1)+(RP5*A2+R2*A3)/(B2*B2)
C      SQRTEXI=SQRT(XI)
C      EPBAR=EPBARN+DGAMA*SQR2D3*SQRTEXI
C      CALL AHARD(SIGMAY,HARD,EPBAR,TABLE,NVALUE)
C      PHI=RP5*XI-R1D3*SIGMAY*SIGMAY
C Check for convergence
C      RESNOR=ABS(PHI/SIGMAY)
C      IF (RESNOR.LE.TOLER)THEN
C Update accumulated plastic strain
C      STATEV(5)=EPBAR
C Update stress components:  sigma := A sigma^trial
C      ASTAR1=R3*(R1-NU)/(R3*(R1-NU)+E*DGAMA)
C      ASTAR2=R1/(R1+R2G*DGAMA)
C      A11=RP5*(ASTAR1+ASTAR2)
C      A22=A11
C      A12=RP5*(ASTAR1-ASTAR2)
C      A21=A12
C      A33=ASTAR2
C      STRESS(1)=A11*STREST(1)+A12*STREST(2)
C      STRESS(2)=A21*STREST(1)+A22*STREST(2)
C      STRESS(3)=A33*STREST(3)
C Compute corresponding elastic (engineering) strain components
C      FACTG=R1/R2G
C      P=R1D3*(STRESS(1)+STRESS(2))
C      EEV=P/BULK

```

```

        EEVD3=R1D3*EEV
        STATEV(1)=FACTG*(R2D3*STRESS(1)-R1D3*STRESS(2))+EEVD3
        STATEV(2)=FACTG*(R2D3*STRESS(2)-R1D3*STRESS(1))+EEVD3
        STATEV(3)=FACTG*STRESS(3)*R2
        STATEV(4)=-NU/(R1-NU)*(STATEV(1)+STATEV(2))
        GOTO 999
    ENDIF
END DO
ELSE
C Elastic step: Update stress using linear elastic law
C -----
        STRESS(1)=STREST(1)
        STRESS(2)=STREST(2)
        STRESS(3)=STREST(3)
C Elastic engineering strain
        STATEV(1)=STRAT(1)
        STATEV(2)=STRAT(2)
        STATEV(3)=STRAT(3)
        STATEV(4)=-NU/(R1-NU)*(STRAT(1)+STRAT(2))
    ENDIF
999      CONTINUE
C -----
        IF (IFPLAS.EQ.1) THEN
C -----
C Compute elastoplastic consistent tangent modulus
C -----
C Compute XI
        XI=R2D3*(STRESS(1)**2+STRESS(2)**2-STRESS(1)*STRESS(2))
        XI=XI+R2*STRESS(3)*STRESS(3)
        CALL AHARD(SIGMAY,HARD,STATEV(5),PROPS(3),NVALUE)
C Matrix E components
        ESTAR1=R3*E/(R3*(R1-NU)+E*DGAMA)
        ESTAR2=R2G/(R1+R2G*DGAMA)
        ESTAR3=G/(R1+R2G*DGAMA)
        E11=RP5*(ESTAR1+ESTAR2)
        E22=E11
        E12=RP5*(ESTAR1-ESTAR2)
        E33=ESTAR3
C Components of the matrix product EP
        EPSTA1=R1D3*ESTAR1
        EPSTA2=ESTAR2
        EPSTA3=EPSTA2
        EP11=RP5*(EPSTA1+EPSTA2)
        EP22=EP11
        EP12=RP5*(EPSTA1-EPSTA2)
        EP21=EP12
        EP33=EPSTA3
C Vector n
        VECN(1)=EP11*STRESS(1)+EP12*STRESS(2)
        VECN(2)=EP21*STRESS(1)+EP22*STRESS(2)
        VECN(3)=EP33*STRESS(3)
C Scalar alpha
        DENOM1=STRESS(1)*(R2D3*VECN(1)-R1D3*VECN(2))+
1          STRESS(2)*(R2D3*VECN(2)-R1D3*VECN(1))+
2          STRESS(3)*R2*VECN(3)
        DENOM2=R2*XI*HARD/(R3-R2*HARD*DGAMA)

```

A.7. MODELLO GLOBALE-LOCALE: PIASTRA FORATA ISOTROPA MODELLO ELASTO-PLASTICO

```

        ALPHA=R1 / (DENOM1+DENOM2)
C -----
        DDSDE(1,1)=E11-ALPHA*VECN(1)*VECN(1)
        DDSDE(1,2)=E12-ALPHA*VECN(1)*VECN(2)
        DDSDE(1,3)=-ALPHA*VECN(1)*VECN(3)
        DDSDE(2,1)=DDSDE(1,2)
        DDSDE(2,2)=E22-ALPHA*VECN(2)*VECN(2)
        DDSDE(2,3)=-ALPHA*VECN(2)*VECN(3)
        DDSDE(3,1)=DDSDE(1,3)
        DDSDE(3,2)=DDSDE(2,3)
        DDSDE(3,3)=E33-ALPHA*VECN(3)*VECN(3)
        ELSE
C Compute plane stress elasticity matrix
C -----
        R4GD3=R4*G/R3
        FACTOR=(BULK-R2G/R3)*(R2G/(BULK+R4GD3))
        DO I=1,NTENS
            DO J=I,NTENS
                DDSDE(I,J)=R2G*FOID(I,J)+FACTOR*SOID(I)*SOID(J)
            END DO
        END DO
C lower triangle
        DO J=1,NTENS-1
            DO I=J+1,NTENS
                DDSDE(I,J)=DDSDE(J,I)
            END DO
        END DO
C      print *, DDSDE
C      print *, STRESS
        ENDIF
        RETURN
        END

C
        SUBROUTINE AHARD(SIGMAY,HARD,EQPLAS,TABLE,NVALUE)
C
        INCLUDE 'aba_param_dp.inc'
        DIMENSION TABLE(2,NVALUE)
C
C      Set yield stres to last value of the table , hardening to zero
        SIGMAY=TABLE(1,NVALUE)
        HARD=0.0
C
C      If more than one entry , search table
C
        IF(NVALUE.GT.1) THEN
            DO K1=1,NVALUE-1
                EQPL1=TABLE(2,K1+1)
                IF(EQPLAS.LT.EQPL1) THEN
                    EQPL0=TABLE(2,K1)
C                    IF(EQPL1.LE.EQPL0) THEN
C                        WRITE(6,1)
C                        FORMAT(/,30X,'***ERROR--PLASTIC STRAIN MUST BE ',
C                        1      'ENTERED IN ASCENDING ORDER')
C                        CALL XIT
                    ENDIF
                END DO
            END IF
        END IF

```

```

C      Current yield stress and hardening
C
      DEQPL=EQPL1-EQPL0
      SYIEL0=TABLE(1,K1)
      SYIEL1=TABLE(1,K1+1)
      DSYIEL=SYIEL1-SYIEL0
      HARD=DSYIEL/DEQPL
      SIGMAY=SYIEL0+(EQPLAS-EQPL0)*HARD
      GOTO 20
    ENDIF
  END DO
20    CONTINUE
    ENDIF
    RETURN
    END

```

```

C
C
C DISP: ricezione delle condizioni al contorno dal modello
C Globale (spostamenti imposti) e imposizione di tali
C condizioni
C
C

```

```

      SUBROUTINE DISP(U,KSTEP,KINC,TIME,NODE,NOEL,JDOF,COORDS)
C
      USE CONN_DATA
      INCLUDE 'aba_param_dp.inc'
C
      CHARACTER*80 CMNAME
      DIMENSION U(3),TIME(3),COORDS(3)
      DIMENSION uuu(41),vvv(41),www(41)
      DIMENSION rrx(41),rry(41),rrz(41)

```

```

C
C
C Definizione variabili
C
C

```

```

      logical, save      :: chiamata_finita=.TRUE.
      integer            :: richiesta_continuazione
      integer, save      :: continuo
      integer, save      :: risposta_fine
      integer            :: permesso_proseguì
      integer            :: richiesta_nodi
      integer            :: r1,r2,r3,r4,r5
      integer, save      :: nodi(41)
      double precision, save :: uu(41),vv(41),ww(41)
      double precision, save :: rx(41),ry(41),rz(41)

```

```

C
C
C Connessione tra i 2 modelli (LOCALE-GLOBALE)
C
C

```

```

      CALL SERVER_CONN()

```

```

C
C
C Permesso di continuazione
C

```

```

C-----
      continuo = 1
      if ((KINC.NE.risposta_fine).AND.(KINC.NE.1)) then
        chiamata_finita = .TRUE.
      end if
      if (.not. chiamata_finita) go to 110
C-----
C
C Definizione nodi
C e
C richiesta spostamenti nodali
C
C-----
      do i=1, 41
        nodi(i) = 0
      end do
C
C Lista nodi del contorno locale
C
      nodi(1)=3261
      .....
      nodi(41)=3361
C
      print *, 'NODI', nodi
      print *, 'KSTEP_locale', 'KINC_locale', KSTEP, KINC
      permesso_proseguì=2
      rc = f77_zmq_recv (local, risposta_fine, 4, 0)
C      print *, 'rc=', rc
      print *, 'risposta_fine(KINC_globale)=', risposta_fine
      rc = f77_zmq_send (local, permesso_proseguì, 4, 0)
C      print *, 'rc=', rc
C-----
C
C Ricezione spostamenti nodali dal modello GLOBALE
C
C-----
      r1=3
      r2=4
      r3=5
      r4=6
      r5=7
      rc = f77_zmq_recv (local, richiesta_nodi, 4, 0)
C      print *, 'rc=', rc
      rc = f77_zmq_send (local, nodi, 4*size(nodi), 0)
C      print *, 'rc=', rc
      rc = f77_zmq_recv (local, uu, 8*size(uu), 0)
C      print *, 'rc=', rc
      print *, 'u=', uu
      uuu=uu
      rc = f77_zmq_send (local, r1, 4, 0)
C      print *, 'rc=', rc
      rc = f77_zmq_recv (local, vv, 8*size(vv), 0)
C      print *, 'rc=', rc
      print *, 'v=', vv
      vvv=vv
      rc = f77_zmq_send (local, r2, 4, 0)

```

```

C      print *, 'rc=', rc
      rc = f77_zmq_recv (local, ww, 8*size(ww), 0)
C      print *, 'rc=', rc
      print *, 'w=', ww
      www=ww
      rc = f77_zmq_send (local, r3, 4, 0)
C      print *, 'rc=', rc
      rc = f77_zmq_recv (local, rx, 8*size(rx), 0)
C      print *, 'rc=', rc
      print *, 'rx=', rx
      rrx=rx
      rc = f77_zmq_send (local, r4, 4, 0)
C      print *, 'rc=', rc
      rc = f77_zmq_recv (local, ry, 8*size(ry), 0)
C      print *, 'rc=', rc
      print *, 'ry=', ry
      rry=ry
      rc = f77_zmq_send (local, r5, 4, 0)
C      print *, 'rc=', rc
      rc = f77_zmq_recv (local, rz, 8*size(rz), 0)
C      print *, 'rc=', rc
      print *, 'rz=', rz
      rrz=rz
      rc = f77_zmq_send (local, continuo, 4, 0)
C      print *, 'rc=', rc
      chiamata_finita=.FALSE.

C_____
C
C Spostamenti imposti
C
C_____
110  continue
C      print *, 'NODI', NODE, JDOF
      do i=1, 41
        if (NODE.EQ.nodi(i)) then
          select case(JDOF)
            case(1)
              U(1) = uu(i)
            case(2)
              U(1) = vv(i)
            case(3)
              U(1) = ww(i)
            case(4)
              U(1) = rx(i)
            case(5)
              U(1) = ry(i)
            case(6)
              U(1) = rz(i)
          end select
        end if
      end do

C_____
      RETURN
      END

```

Bibliografia

- [1] J.N. Reddy, (1989). On computational schemes for global-local stress analysis. NASA, Langley research center, Computational methods for structural mechanics and dynamics; part 1, p 123-134.
- [2] A.K. Noor, (1986). Global-local methodologies and their application to nonlinear analysis. Finite elements in analysis and design; volume 2, issue 4, pages 333-346.
- [3] N.G. Cormier, B.S. Smallwood, G.B. Sinclair, G. Meda, (1999). Aggressive submodeling of stress concentrations. International journal for numerical methods in engineering; volume 46, issue 6.
- [4] D.M.M. Thompson, (1990) Two-dimensional to three-dimensional global/local finite element analysis of laminated composites in compression. Thesis submitted to the faculty of the Virginia Polytechnic Institute and State University.
- [5] J.D Whitcomb, (1991). Iterative global-local finite element analysis. Computers & Structures; volume 40, issue 4, pages 1027-1031.
- [6] K.M. Mao, C.T. Sun, (1991). A refined global-local finite element analysis method. International journal for numerical methods in engineering; volume 32, issue 1.
- [7] O. Allix, L. Gendre, P. Gosselet, G. Guguin, (2011). Non-intrusive coupling: an attempt to merge industrial and research software capabilities. Recent developments and innovative applications in computational mechanics; pages 125-133.
- [8] L. Gendre, O. Allix, P. Gosselet, F. Comte, (2009). Non-intrusive and exact global/local techniques for structural problems with local plasticity. Computational Mechanics; volume 44, issue 2, pages 233-245.
- [9] G. Guguin, O. Allix, P. Gosselet, S. Guinard, (2014). Nonintrusive coupling of 3D and 2D laminated composite models based on finite element 3D recovery. International journal for numerical methods in engineering; volume 98, issue 5.
- [10] F. Daghighi, P. Ladevèze, (2012). A micro-meso computational strategy for the prediction of the damage and failure of laminates. Composite Structures; volume 94, issue 12, pages 3644-3653.
- [11] M. Duval, J.C. Passieux, M. Salaün, S. Guinard, (2016). Non-intrusive coupling: recent advances and scalable nonlinear domain decomposition. Archives of computational methods in engineering; volume 23, issue 1, pages 17-38.
- [12] P. Ladeveze, D. Neron, (2010). Multiscale methods. Encyclopedia of aerospace engineering.
- [13] T. You, Y.R. Kim, T. Park, (2017). Two-way coupled multiscale model for predicting mechanical behavior of bone subjected to viscoelastic deformation and fracture damage. Journal of engineering materials and technology, volume 139, issue 2.

- [14] S. Loehnert, T. Belytschko, (2007). A multiscale projection method for macro/micro-crack simulations. *International journal for numerical methods in engineering*; volume 71, issue 12.
- [15] S. Huehne, J. Reinoso, E. Jansen, A. Rolfes, (2016). A two-way loose coupling procedure for investigating the buckling and damage behaviour of stiffened composite panels. *Composite Structures*; volume 136, pages 513-525.
- [16] M. Akterskaia, E. Jansen, S. Hühne, R. Rolfes, (2018). Efficient progressive failure analysis of multi-stringer stiffened composite panels through a two-way loose coupling global-local approach. *Composite Structures*; volume 183, pages 137-145.
- [17] G. Labeas, S. Belesis, I. Diamantakos, K. Tserpes, (2012). Adaptive progressive damage modeling for large-scale composite structures. *International journal of damage mechanics*; volume 21 issue 3, pages 441-462.
- [18] Abaqus 2016 Online Documentation.
- [19] J.S. Chapman, (2000). *Fortran 90/95 guida alla programmazione*. McGraw-Hill.
- [20] https://github.com/zeromq/f77_zmq.
- [21] EA de Souza Neto, D. Peric, D.R.J Owen, (2008). *Computational methods for plasticity - theory and applications*. Wiley.
- [22] J.O. Hallquist, (1979). *NIKE2D: an implicit, finite-deformation, finite-element code for analyzing the static and dynamic response of two-dimensional solids*. National technical information service.
- [23] N.S. Ottosen, M. Ristinmaa, (2005). *The mechanics of constitutive modeling*. Elsevier.