Politecnico di Torino

Politecnico di Torino

Corso di Laurea Magistrale in Ingegneria Aerospaziale
A.A. 2022/2023
Sessione di Laurea Marzo/Aprile 2023

Sviluppo e applicazione di un metodo meshless GFDM per la simulazione del processo di stampa 3D FDM

Tesi di Laurea Magistrale

Relatore:
Prof. Enrico Zappino

Candidato:
Andrea Pavan

Abstract

IT

OpenGFDM è un nuovo software open-source per la simulazione termica e meccanica basato su un metodo meshless alle differenze finite generalizzate. Questo lavoro di tesi si propone di descriverne le caratteristiche tecniche e le potenzialità, focalizzandosi in particolare sull'applicazione alla simulazione di processo della stampa 3D FDM.

Dopo una breve descrizione delle fenomenologie in gioco viene sviluppato un modello fisico e matematico. Le equazioni sono quindi discretizzate applicando il metodo GFDM e risolte numericamente. L'implementazione fa uso dei più recenti sviluppi in ambito software quali Julia ed Electron con lo scopo di realizzare un'applicazione moderna ed intuitiva senza compromettere prestazioni e scalabilità.

EN

OpenGFDM is a novel open-source software for thermal and mechanical simulation based on a meshless Generalized Finite Differences Method. The aim of this thesis work is to describe its technical merits and its potential, with a particular focus on the application to the 3D printing process simulation.

After a brief overview on the acting phenomena, a mathematical and physical model is developed. Then, the governing equations are discretized using the GFDM formulation and solved numerically. The implementation makes use of the latest software developments such as Julia and Electron to build a modern and easy-to-use app without compromising performance or scalability.

Indice

Capitolo 1

Introduzione

La cosiddetta "stampa 3D a filamento" è la più semplice e diffusa tecnica di additive manufacturing per materiali polimerici. Un filamento di materiale termoplastico è scaldato ad alta temperatura da un ugello e depositato su un piatto. L'oggetto è costruito layer su layer muovendo l'ugello secondo un percorso prestabilito.

Il workflow di tale processo è piuttosto semplice: dopo aver disegnato un modello CAD 3D, esso viene tessellato ed esportato in formato STL per la fase di slicing. Qui traiettoria e parametri di stampa vengono generati automaticamente a partire da geometria, qualità desiderata e una moltitudine di parametri. L'output è una sequenza di istruzioni in formato GCODE specifica per la macchina.

Anche disponendo di una stampante performante, il raggiungimento di una buona qualità di stampa non è affatto banale e comporta la taratura di decine di parametri. Tipicamente questo viene fatto in modo empirico, e a seconda dell'esperienza dell'utente i risultati possono essere più o meno buoni. Chiaramente diversi materiali richiedono profili di stampa differenti, così come variazioni allo spessore dei layer o alla velocità di stampa. In casi particolari è possibile che si debba definire un profilo ad-hoc per la stampa di un solo oggetto o addirittura per porzioni di esso.

E anche con un profilo ottimizzato e una stampa andata a buon fine, non è detto che il pezzo abbia le caratteristiche dimensionali e prestazionali attese. Fenomeni come warping, shrinking, delaminazione e sviluppo di stress termici localizzati possono infatti pregiudicare l'accuratezza dell'oggetto e le sue proprietà meccaniche.

In questo contesto disporre di uno strumento per simulare il processo di stampa in modo virtuale può ridurre sensibilmente il numero di pezzi non conformi e le risorse necessarie per le varie prove, sia in termini di materiale sia in termini di tempo. Ovviamente l'attività di simulazione non va a rimpiazzare l'indagine empirica, ma può affiancarla e potenziarla. Trattandosi di strumenti piuttosto onerosi in termini di costo computazionale, potrebbe non avere senso impiegarli per tutte le stampe, bensì solo per quelle più critiche, per i componenti più sollecitati, oppure per studiare più in dettaglio l'effetto dei parametri di stampa.

In prospettiva, un impiego esteso di software di simulazione di processo potrebbe fornire la capacità di realizzare pezzi e componenti funzionali al primo tentativo di stampa, senza aggiustamenti successivi. Un altro ambito di applicazione potrebbe riguardare la validazione del processo di fabbricazione per parti certificate.

Molte sono le soluzioni commerciali disponibili, ma quasi tutte dedicate alla simulazione della stampa di parti in metallo e rivolte all'industria. Sono pochi i software per la tecnologia Fused Deposition Modeling (FDM) per parti in plastica e in pratica nessuno di essi accessibile ad utenti amatoriali.

OpenGFDM ha l'ambizione di colmare questo vuoto: uno strumento open-source, dotato di interfaccia grafica moderna e accattivante, semplice da usare anche per i non addetti ai lavori, meshless, scalabile dal PC di casa al cloud computing, accessibile.

La formulazione meshless aiuta da questo punto di vista fornendo flessibilità e semplicità. Ragionare in termini di pointcloud evita completamente le problematiche relative alla generazione della mesh, anche se a prezzo di un costo computazionale maggiore. Con la potenza di calcolo oggi disponibile non è un problema insormontabile e può essere considerata una controindicazione accettabile, anche perché la quantità di lavoro che deve apportare l'utente si riduce notevolmente.

Del resto la simulazione di un processo di stampa 3D deve tenere conto di un'ampia varietà di fenomeni termici, meccanici e termomeccanici all'interno di una geometria che cresce nel tempo e che si auto-interseca. L'attività di meshing risulterebbe molto onerosa e complessa, quindi è naturale che la scelta ricada su un approccio meshless. Questi metodi inoltre hanno una formulazione tipicamente più semplice rispetto ai classici FEM/FVM, per cui si sposano bene con la tendenza ad impiegare nelle simulazione dei modelli sempre più generici e meno restrittivi, anche multifisici.

1.1 Organizzazione tesi

Questo elaborato vuole descrivere le motivazioni alla base di OpenGFDM, descriverne lo sviluppo e commentare i risultati raggiunti. A tale scopo esso è organizzato come segue:

- Nel resto di questo capitolo introduttivo viene fatta una breve panoramica sul mercato dei software di simulazione, sui benefici offerti dall'approccio open-source e sulle caratteristiche di OpenGFDM;
- Il Capitolo 2 inizia con una breve descrizione delle tecnologie di AM, soffermandosi sulle caratteristiche della FDM e sui fenomeni fisici in gioco durante il processo di stampa. Tenendo in mente le esigenze di natura computazionale si propongono un modello fisico-matematico e una procedura di simulazione. Vengono derivate le equazioni di governo e le relative condizioni al contorno;
- Il Capitolo 3 è incentrato sul metodo numerico GFDM: derivazione matematica, proprietà, confronto con altri metodi e infine applicazione alle equazioni di governo per scrivere il sistema discretizzato da risolvere;
- Nel Capitolo 4 si descrivono le attività di sviluppo vere e proprie, dagli script dimostrativi all'applicativo finale. Particolare attenzione è posta sui problemi di implementazione e sulle difficoltà riscontrate;
- Infine nel Capitolo 5 vengono presentati e commentati i risultati raggiunti, le potenzialità e i possibili miglioramenti;

Per la stesura delle parti più tecniche vengono privilegiate citazioni open-access, se disponibili.

1.2 Software open-source

Con l'avanzare delle capacità di calcolo negli ultimi decenni le tecniche di simulazione numerica sono diventate sempre più diffuse e utilizzate in ambito ingegneristico. Oramai in tutte le aree per lo sviluppo di un prodotto si fa un massiccio uso di strumenti FEM e/o CFD avanzati, e nel breve termine ci si attende una crescente adozione. Il mercato tuttavia è relativamente piccolo e dominato da soluzioni commerciali proprietarie il cui modello di business è fortemente ancorato alla semplice vendita di licenze più che ai servizi accessori. In assenza di incentivi, non sorprende che negli anni non siano state sviluppate soluzioni **open-source** diffuse come lo è Tensorflow nel settore AI.

Questo però va a limitare l'utilizzatore finale. Se per un'attività commerciale i costi di licenza possono definirsi contenuti se comparati con i costi del personale specializzato, invece per l'utente amatoriale risultano quasi sempre eccessivi. Altre criticità riguardano l'obbligo di legarsi ad una piattaforma proprietaria, con relative restrizioni e rischio di vendor lock-in.

I software open-source non soffrono questi problemi e anzi presentano ulteriori vantaggi:

- nessuna limitazione su funzionalità o dimensione del problema legata al tipo di licenza;
- possibilità di esaminare nel dettaglio il modello fisico-matematico e l'implementazione;
- nessuna funzione di smorzamento nascosta o limiter segreto. Solitamente i codici commerciali cercano di fornire un qualche risultato anche in presenza di errori e/o mancata convergenza, mentre la maggior parte dei software open-source semplicemente smette di funzionare;
- libertà di adattare il codice alle proprie esigenze e di implementare nuovi modelli;
- possibilità di compilare il codice in modo ottimizzato anche su macchine non certificate;
- completo controllo sulla distribuzione e sugli aggiornamenti, anche nel caso di abbandono o fallimento della casa madre;
- accessibili ad utenti singoli quali studenti, hobbisti, disoccupati;

Altri benefici legati all'open-source riguardano l'ambiente di sviluppo collaborativo, una community potenzialmente molto vasta e la possibilità di integrare al meglio i propri tool interni con i codici di simulazione.

Al giorno d'oggi sono disponibili moltissimi codici di Computer Aided Engineering (CAE) open-source, ma per la maggior parte dei casi si tratta di progetti accademici o personali. Molto validi da un punto di vista tecnico, ma difficili da usare ed estremamente specializzati. Spesso la documentazione è carente e non c'è un workflow completo; a titolo di esempio, il software CFD open-source OpenFOAM include ottimi solutori, ma la generazione della mesh è scadente. Per un utilizzo commerciale è di fatto necessario usare software di meshing proprietari.

La difficoltà d'uso fa desistere dall'adozione non soltanto l'utente singolo ma anche molte attività economiche, perché si traduce in una minore produttività e in time-to-market più lunghi, senza contare il minor numero di addetti capaci ad usare tali strumenti.

Solo ultimamente si può osservare un'accelerazione nella proposta di soluzioni open-source da parte dei produttori. Si citano ad esempio Altair Radioss e Simscale (che impiega OpenFOAM e Code_Aster), che hanno scelto la via del codice aperto per esigenze fondamentalmente differenti. Da una parte il produttore può rilasciare il suo software proprietario con licenza aperta con lo scopo di velocizzarne lo sviluppo collaborando con istituti, università e in generale

chiunque sia interessato.[1] In questo modo l'applicativo riceve i più recenti sviluppi nei più svariati settori dell'industria e può affrontare i radicali cambiamenti oggi in atto.

Un'altra ragione per rilasciare soluzioni open-source è la crescente tendenza verso il cloud computing.[2] Da questo punto di vista il trend è allineato a quello di altre aree tech quali l'AI. Con la pandemia COVID-19 infatti le piattaforme cloud hanno garantito una soluzione concreta ed economicamente sostenibile al bisogno di molte realtà di continuare ad operare, senza che queste debbano dover mettere in piedi in poco tempo una nuova infrastruttura complessa e costosa.

In ambito stampa 3D tutti questi problemi e tendenze sono particolamente accentuati. La simulazione di processo è una tecnologia che si sposa bene con le innovazioni rese possibili dall'additive manufacturing (AM) e come detto offre benefici tangibili, eppure è relegata a poche soluzioni commerciali non accessibili al grande pubblico.

1.3 Mercato software simulazione

Le proiezioni vedono il mercato dei software di simulazione sempre più rilevante e con una crescita sostenuta, ma ancora relativamente piccolo. Al 2021 infatti era valutato[3] 13.25 miliardi USD a livello globale con una crescita stimata del 13.5% dal 2022 al 2030. A titolo di confronto, il mercato AI è un ordine di grandezza più grande,[4] 136.55 miliardi USD nel 2022 e con una previsione di crescita del 37.3% nel periodo 2023-2030.

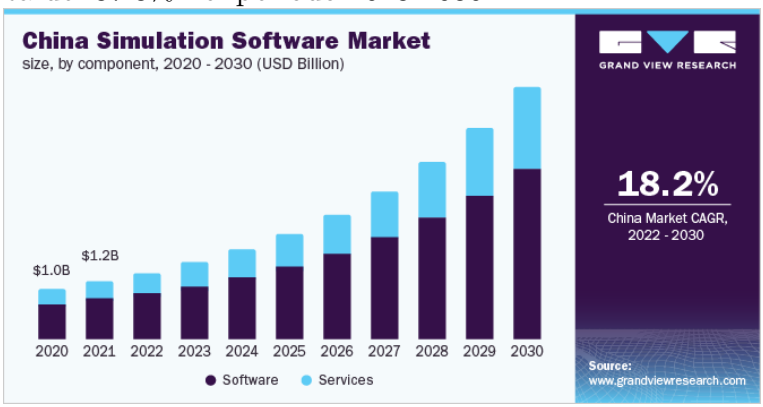


Figura 1.1: Previsioni di crescita del mercato dei software di simulazione al 2030 (immagine dal report GVR-3-68038-751-3)

Il mercato nel 2021 è dominato da multinazionali statunitensi ed europee. Il modello di business preponderante prevede la vendita delle licenze software e la fornitura di servizi di supporto e consulenza. Al contrario di quanto avviene nel resto del settore tech, ben il 70.5% delle entrate proviene dalla vendita licenze.

[1] *Industry-Proven Altair Radioss Finite Element Analysis Solver Now Available as Open-Source Solution.* Altair. URL: `https://www.altair.com/newsroom/news-releases/industry-proven-altair-radioss-finite-element-analysis-solver-now-available-as-open-source-solution`.

[2] Meg Jenkins. *The Past, Present, and Future of Cloud Computing and CFD.* Simscale. URL: `https://www.simscale.com/blog/future-of-cloud-computing-cfd-engineers/`.

[3] *Simulation Software Market Size, Share & Trends Analysis Report.* Rapp. tecn. GVR-3-68038-751-3. Grand View Research, 2021.

[4] *Artificial Intelligence Market Size, Share & Trends Analysis Report.* Rapp. tecn. GVR-1-68038-955-5. Grand View Research, 2022.

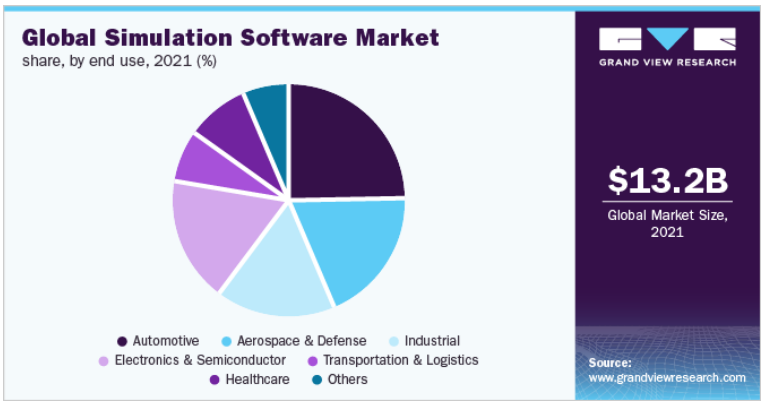


Figura 1.2: Segmenti del mercato dei software di simulazione (immagine dal report GVR-3-68038-751-3)

Per quanto riguarda gli utilizzatori, il settore dell'automotive è quello che genera la maggior parte delle entrate (circa il 24.7%) seguito dall'aerospaziale e difesa e poi dalle altre attività industriali. Entrambi i segmenti ci si attende essere in crescita dati gli ingenti investimenti su veicoli elettrici ed autonomi, e nella difesa.
Automotive e aerospazio dal punto di vista dei software di simulazione sono settori piuttosto simili. Essi devono far riferimento a normative piuttosto restrittive che richiedono un certo grado di affidabilità e retrocompatibilità.

1.4 OpenGFDM

OpenGFDM si propone di risolvere i problemi elencati in precedenza dei più comuni software open-source di simulazione. La difficoltà d'uso è certamente l'ostacolo maggiore alla diffusione, a cominciare dall'installazione: l'utente deve poter scaricare ed eseguire l'applicativo senza lunghe e noiose procedure di compilazione. È utile disporre di un'interfaccia grafica semplice e user-friendly che permetta di eseguire le simulazioni in pochi click ed estrarre i risultati di cui necessita con altrettanta semplicità.

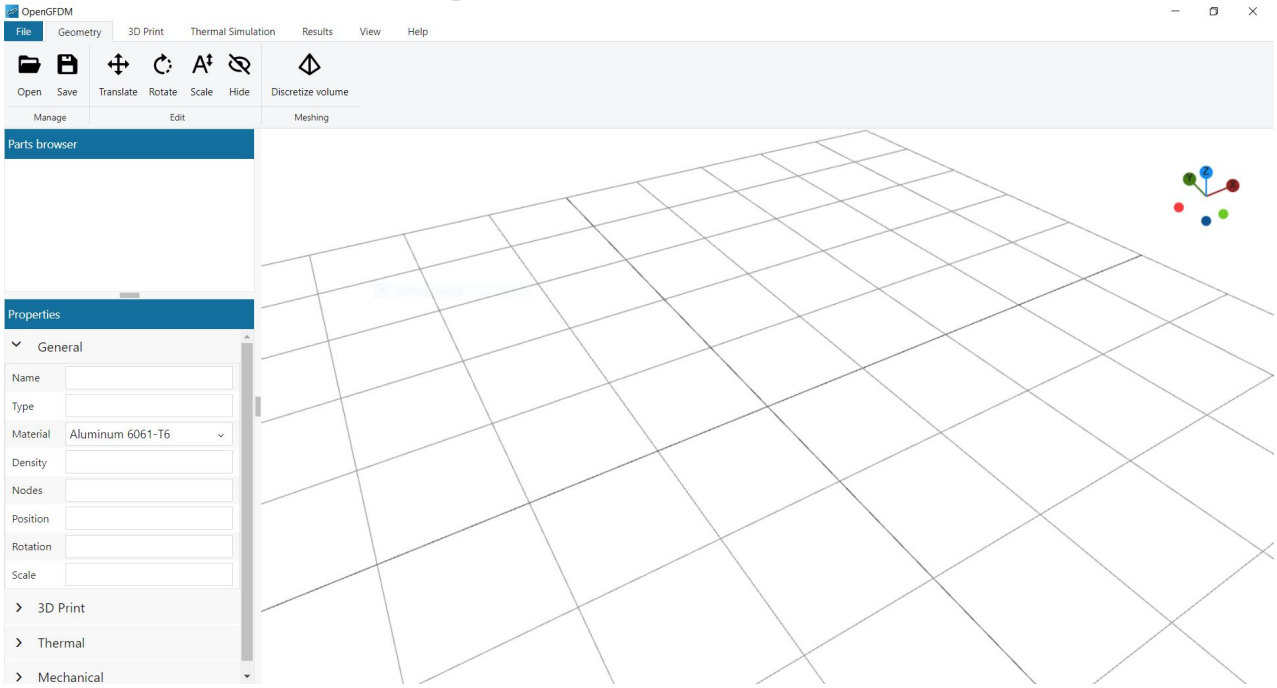
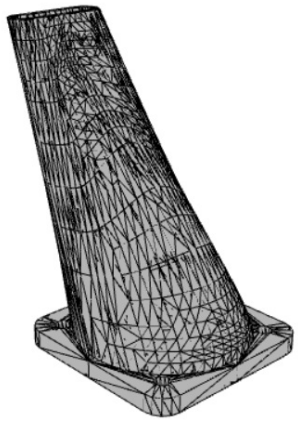
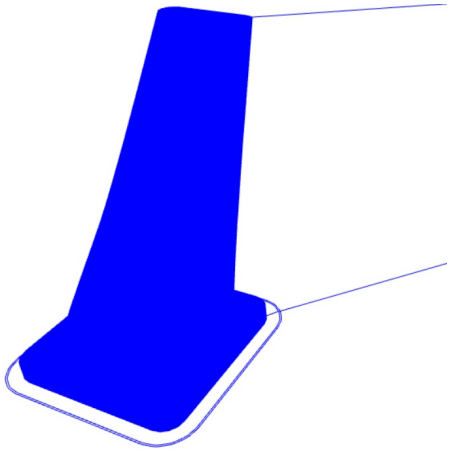


Figura 1.3: Screenshot di OpenGFDM all'avvio

Molto importante ovviamente è la compatibilità con formati file esistenti, sia in input sia in output. L'utilizzatore deve poter usare il suo software CAD preferito per modellare la geometria del sistema da simulare e, una volta ottenuti i risultati, essere in grado di utilizzarli nel modo in cui gli è più comodo. Per non complicare eccessivamente il lavoro, ci si è concentrati sulla compatibilità con i formati STL per la geometria 3D, GCODE per la simulazione del processo di stampa 3D e CSV per l'esportazione dei risultati. Per come è strutturato, il software è facilmente interfacciabile con un qualunque linguaggio di programmazione tramite messaggi JSON via Websocket. Per maggiori informazioni si veda il paragrafo 4.1 dedicato all'architettura del software.





Coerentemente con lo spirito "open" del progetto, si è deciso di rendere l'interfaccia grafica accessibile ad utenti con disabilità visive. Tra le accortezze tenute in conto vi sono:

- scalabilità coerente con lo scale factor dello schermo;
- impiego di palette di colori "sicuri" di default;
- rendere tutti i controlli raggiungibili con la sola tastiera o con il solo tocco;
- compatibilità out-of-the-box con i più comuni software di screen-reader come Microsoft Narrator su Windows e Orca su Ubuntu Linux;

Invece per quanto riguarda i software di ingrandimento la compatibilità è solitamente garantita senza alcun intervento.
Ovviamente tutto ciò va ad influenzare l'ambiente grafico, ma non per questo è necessario compromettere l'usabilità per la maggior parte degli utenti.

Ricapitolando, OpenGFDM si caratterizza per:

- Metodo numerico meshless basato sulle differenze finite generalizzate (GFDM);
- Solutori disponibili: termico, meccanico, termo-elastico;
- Interfaccia grafica moderna e user-friendly;
- Free and Open-Source Software (FOSS);
- Compatibilità con i formati STL e GCODE;
- Architettura modulare e modificabile a piacimento;
- Frontend sviluppato con tecnologie web + Electron, backend sviluppato in Julia;
- Multipiattaforma, funziona su Windows, Linux, MacOS;
- Portatile, nessuna installazione richiesta;
- Accessibile a persone con disabilità visive;

Come detto in precedenza, i grandi player nell'automotive e nell'aerospaziale possono avere esigenze di retrocompatibilità con codici precedenti. OpenGFDM quindi non è rivolto a questo tipo di utenza, bensì alle realtà amatoriali, singoli hobbisti, studenti ed eventualmente a quei professionisti e piccole imprese in cui l'attività di simulazione non fa parte del core business e i costi di licenza non sono economicamente giustificati.

Capitolo 2

Simulazione di processo

Quello dell'additive manufacturing è un mondo molto variegato e comprende processi di fabbricazione molto diversi fra loro. Fino al decennio scorso la fabbricazione additiva era limitata alla prototipazione rapida oppure alla produzione di strumenti e/o consumabili di altri processi di fabbricazione. Ad oggi invece le macchine hanno raggiunto un certo grado di maturità e diffusione, complici il sempre più basso costo di accesso e i campi di applicazione sempre più ampi. In molti casi è possibile parlare di un vero e proprio mezzo di produzione per piccoli volumi, sovrapponibile alla tradizionale manifattura sottrattiva.
Rispetto alle lavorazioni per asportazione di truciolo, le tecnologie di additive manufacturing presentano i seguenti vantaggi:

- Libertà molto più ampia nelle geometrie realizzabili;
- Scarti di produzione molto limitati, con conseguente riduzione dei costi;
- Possibilità di realizzare un componente in un numero minore di pezzi, se non addirittura in una sola stampa, con benefici sui costi di manodopera e assemblaggio;
- Possibilità di modificare il disegno del pezzo senza dover attrezzare in modo diverso la macchina;
- Abbassamento del time-to-market grazie alle più rapide fasi di ingegnerizzazione e progettazione;

Gli svantaggi invece vedono:

- Un certo grado di porosità e anisotropia che portano a proprietà meccaniche inferiori;
- Volume di stampa limitato;
- Tempi di produzione molto alti;
- Costo del materiali molto più elevato e dalla scelta limitata;

Detto questo, i risultati migliori molte volte si ottengono affiancando la produzione AM a quella tradizionale, specialmente nel caso di materiali metallici che spesso richiedono un post-processing corposo.
Tra gli svantaggi è possibile inserire anche la maggiore pericolosità intrinseca dei processi. Sono richiesti livelli di sicurezza operativa più alti che precludono l'impiego di molte tecnologie in ambito domestico e alzano i costi in ambito industriale. Si citano ad esempio le criticità connesse con l'impiego di gas argon (che pur non essendo tossico tende a ristagnare in un ambiente chiuso rendendolo asfittico) e dell'azionamento di laser ad altissima potenza. Ovviamente non tutte le tecnologie ne sono affette, e anche all'interno di una categoria molto dipende dalla singola macchina, che può risentirne in modo diverso e presentare requisiti più o meno stringenti.

In questo capitolo viene sviluppato un modello fisico-matematico del processo di stampa 3D FDM considerando le fenomenologie più importanti. Si propone quindi un approccio idoneo alla simulazione di processo e, per concludere, vengono sviluppate le equazioni di governo.

2.1 Panoramica delle tecnologie AM

L'idea comune alle varie tecnologie di Additive Manufacturing è quella di realizzare un oggetto tridimensionale sviluppandolo layer su layer, di spessore più o meno variabile. A seconda del materiale e dei requisiti di qualità, costo, prestazione desiderati il processo può avvenire con modalità molto diverse: fondendo/ammorbidendo il materiale, curandolo mediante stimoli esterni, legandolo ad agenti consumabili o una combinazione di questi.
Data l'ampia varietà di tecnologie presenti e in sviluppo, le varie normative (es: ISO 17296-2:2015) propongono una classificazione nelle seguenti 7 categorie:

- Fotopolimerizzazione;
- Estrusione
- Material jetting
- Binder jetting
- Powder bed fusion
- Direct Energy Deposition
- Produzione di oggetti laminati

Ciascuna di esse viene brevemente passata in rassegna esaminandone le principali caratteristiche, i vantaggi offerti e gli svantaggi.

Nei processi di **fotopolimerizzazione** una resina fotosensibile reagisce chimicamente solidificandosi quando esposta alla luce di una sorgente laser/lampada UV (cura). La modellazione della forma dell'oggetto avviene tipicamente azionando un sistema ottico con una lente per ridirigere il fascio luminoso, come visibile dall'immagine sottostante.[1] La più diffusa tecnica di fotopolimerizzazione è senza dubbio la stereolitografia (Stereolithography, SLA), caratterizzata da una eccellente accuratezza dimensionale e finitura superficiale. Le maggiori criticità riguardano i materiali, limitati nella scelta e spesso aventi un certo grado di tossicità.

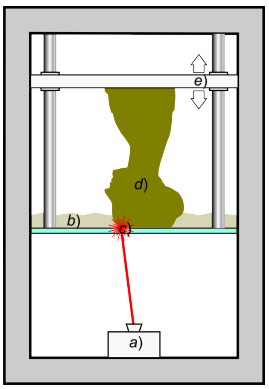


Figura 2.1: Illustrazione funzionamento macchina SLA (immagine da riferimento)

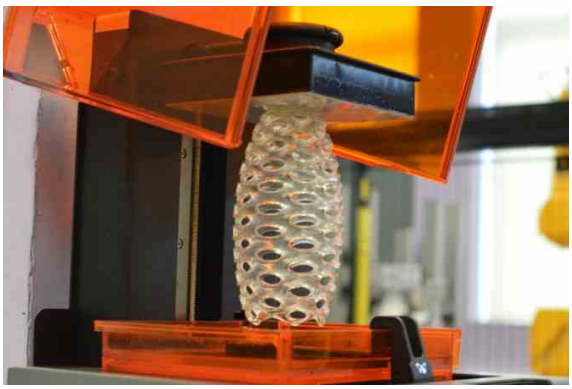


Figura 2.2: Stampa SLA di un oggetto 3D (immagine da re-fream.eu)

La fabbricazione per **estrusione** di materiali polimerici (Fused Deposition Modeling, FDM) è la più diffusa in ambito hobbistico. Come accennato in precedenza, un filamento viene scaldato e fuso per essere forzato attraverso un ugello circolare e depositato su un piatto. I vantaggi sono evidenti: semplicità del processo, bassi costi e ampia scelta dei materiali sono i fattori che più hanno contribuito alla diffusione. Gli svantaggi invece riguardano la necessità di supporti per angoli di overhang eccessivi, l'anisotropia delle proprietà meccaniche, l'accuratezza dimensionale limitata dalla cinematica della macchina, la finitura "a gradini".

[1] R. Scopigno et al. "Digital Fabrication Techniques for Cultural Heritage: A Survey". In: *Computer Graphics Forum* 36.1 (2017), pp. 6–21. DOI: https://doi.org/10.1111/cgf.12781. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.12781.

Oltre ai materiali termoplastici, varianti appositamente sviluppate permettono di stampare compositi (es: CFRP), cementi, ceramiche/metalli (mediante binding, ma richiede un opportuno post-processing) e addirittura cibo (es: cioccolata) e materiali biologici.
Questo lavoro di tesi si focalizza sulla tecnologia FDM per i materiali termoplastici, pertanto il processo verrà descritto più in dettaglio nel prossimo paragrafo.

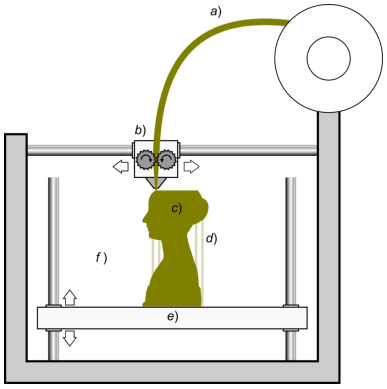


Figura 2.3: Illustrazione funzionamento macchina FDM (immagine da riferimento)

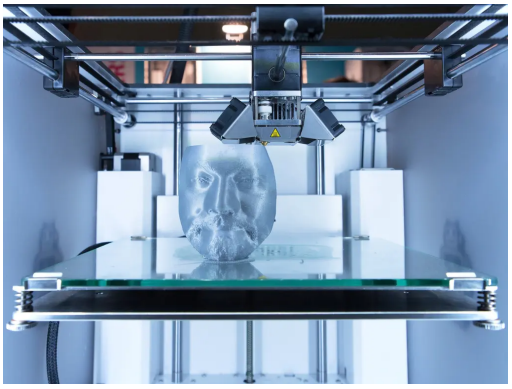


Figura 2.4: Stampa FDM su macchina consumer (immagine da sd3d.com)

La tecnica del **material jetting** consiste nello spruzzare un materiale fotosensibile in piccole goccioline (droplets) sul piatto di stampa per poi curarlo con una lampada UV. Il processo richiede un supporto, tipicamente un gel, stampato contemporaneamente al pezzo da un secondo ugello e rimosso in fase di post-processing.
Grazie allo spruzzo la tecnologia è in grado di stampare layer molto sottili con ottimo livello di dettaglio, buona accuratezza e qualità superficiale. Essa è infatti adatta alla realizzazione di stampi e pattern[2] per processi di manufacturing tradizionali, tra cui l'investment casting.
Tra gli svantaggi della tecnica svettano i costi molto elevati. Inoltre i fotopolimeri potrebbero mostrare un decadimento delle proprietà meccaniche nel tempo.
Rispetto alla stereolitografia è possibile stampare un oggetto con materiali e colori diversi, in modo più veloce e produttivo. Alcune macchine permettono di stampare anche metalli e ceramiche sotto forma di nanoparticelle caricate all'interno delle droplet liquide; facendo evaporare la fase liquida rimane la parte stampata in metallo/ceramica.

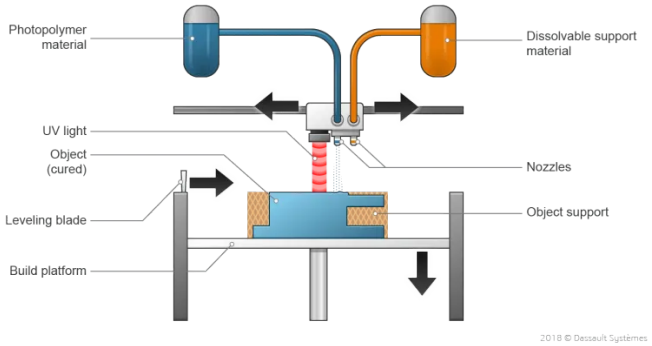


Figura 2.5: Illustrazione funzionamento macchina MJ (immagine da riferimento)

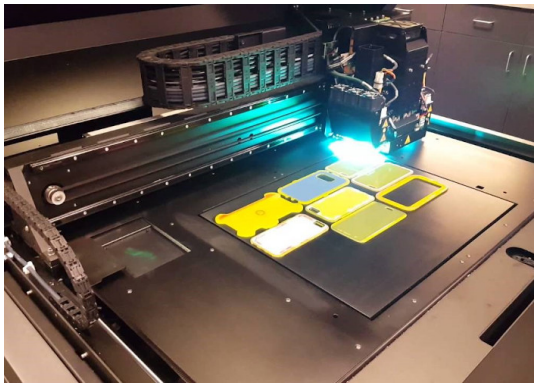


Figura 2.6: Foto stampa MJ (immagine da theadditivemanufacturing.com)

[2] *Introduction to Material Jetting — MJ, NPJ, DOD*. Dassault Systemes. URL: `https://www.3ds.com/make/guide/process/material-jetting`.

Il processo di **binder jetting** invece vede la presenza di un letto di polvere che, strato dopo strato, viene fatto agglomerare da un legante (binder) spruzzato da un ugello.
All'inizio di ciascun layer il piatto si abbassa e nuove particelle di materiale plastico, metallico e/o ceramico vengono sovrapposte alla stampa corrente. Un rullo (o una spazzola) si occupa di distribuirle nel modo più uniforme possibile creando una superficie piana, che funge anche da supporto per i layer successivi.
A questo punto l'iniezione del legante può avvenire ad opera di un set di ugelli. A seconda dei volumi e della velocità desiderati è possibile installare più testine che operano in parallelo.
La tecnologia quindi presenta diversi vantaggi: grandi volumi, ampia scelta di materiali e colori, può realizzare più parti sovrapposte in un'unica stampa. Di contro, la presenza del legante peggiora le proprietà meccaniche e la porosità, e richiede un pesante post-processing.

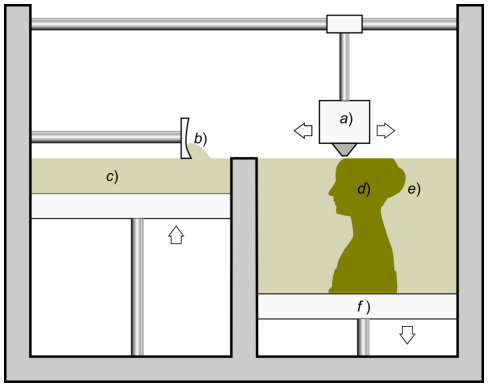


Figura 2.7: Illustrazione funzionamento macchina BJ (immagine da riferimento)

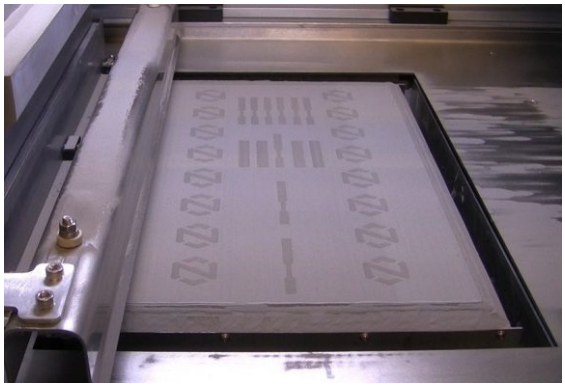


Figura 2.8: Foto stampa BJ (immagine da theadditivemanufacturing.com)

Anche le tecnologie **Powder Bed Fusion** (PBF) si basano su un letto di polvere distribuito ad ogni layer da un rullo. Qui però le particelle vengono fuse o sinterizzate da una sorgente laser (Selective Laser Sintering, SLS) o di elettroni (Electron Beam Melting, EBM) che scansionano l'intero piatto. I materiali processabili sono molteplici: polimeri, compositi, metalli, ceramiche. Si citano a titolo esemplificativo i poliammidi (nylon), la lega di alluminio AlSi10Mg e la lega di titanio Ti6Al4V.

Le tecniche laser a sinterizzazione (SLS) sono costruttivamente analoghe al binder jetting, ma l'agglomerazione delle polveri avviene ad opera di un laser pulsato che le riscalda ad alta temperatura. Un sistema ottico redirige il fascio luminoso sul percorso di stampa desiderato. Per effetto dei forti gradienti termici durante la stampa il pezzo deve essere fissato al piatto mediante dei supporti. A tale proposito un adeguato preriscaldamento del volume di stampa può minimizzare gli stress termici, nonché la potenza richiesta dal laser.
La camera inoltre deve essere riempita con un gas inerte (azoto, argon) per evitare ossidazioni o l'assorbimento di umidità da parte delle polveri. Di conseguenza è possibile operare con materiali altamente reattivi e leghe esotiche, difficilmente processabili con tecniche di manifattura tradizionali, purché abbiano un buon assorbimento laser e una ragionevole temperatura di fusione.[3]
Se il materiale non riesce ad arrivare ad una fusione completa ma solo parziale (sinterizzazione), il pezzo risultante avrà un elevato grado di porosità e richiederà un adeguato post-processing per ripristinare le proprietà meccaniche attese. Al contrario, le macchine che arrivano a

[3]Chaolin Tan et al. "Selective laser melting of high-performance pure tungsten: parameter design, densification behavior and mechanical properties". In: *Science and Technology of Advanced Materials* 19.1 (2018), pp. 370–380. URL: https://doi.org/10.1080/14686996.2018.1455154.

fusione completa (Selective Laser Melting, SLM) producono oggetti con densità molto vicina al 100%. In ogni caso un trattamento successivo alla fase di stampa è sempre necessario, per rimuovere i supporti, eliminare le tensioni interne, migliorare la finitura superficiale e le tolleranze geometriche.

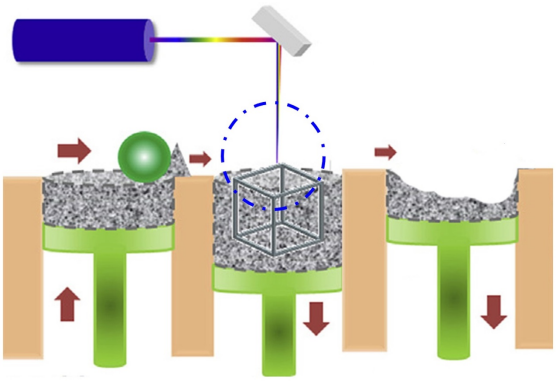


Figura 2.9: Illustrazione funzionamento macchina SLM (immagine da riferimento)

Figura 2.10: Foto stampa SLM (immagine da voestalpine.com)

Le tecniche EBM al posto del laser usano un fascio di elettroni generato da un filamento di tungsteno ad altissima temperatura. Tale fascio viene accelerato, deviato e focalizzato usando campi magnetici finemente controllati migliaia di volte al secondo. Esso infine viaggia attraverso la camera sotto vuoto spinto e impatta sul letto di polvere riscaldandolo. Il materiale viene completamente fuso, pertanto l'oggetto stampato avrà una porosità estremamente bassa, una densità molto prossima al 100% e proprietà meccaniche vicine a quelle nominali. La necessità di operare sotto vuoto inoltre previene i fenomeni di ossidazione e di contaminazione delle polveri senza usare gas inerti. Rispetto alle tecniche laser l'EBM permette di stampare più facilmente leghe con un alto punto di fusione, è generalmente più veloce ma meno accurato e con una qualità superficiale peggiore.

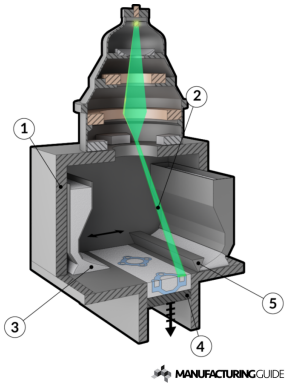


Figura 2.11: Schema funzionamento EBM (immagine da manufacturingguide.com)

Figura 2.12: Foto stampa EBM (immagine da ge.com)

La fabbricazione tramite **Direct Energy Depositon** (DED) consiste nel depositare sul piatto di stampa il materiale metallico sotto forma di filamento o di polvere. Il calore necessario a fonderlo può provenire da un laser, da un fascio di elettroni oppure da una torcia al plasma simile a quelle usate in saldatura. Le macchine DED quindi hanno molte somiglianze con le classiche stampanti ad estrusione. Infatti anche qui la testina di stampa può essere mossa da dei carrelli oppure da un braccio robotico.

La stampa avviene in un ambiente controllato: atmosfera inerte oppure vuoto spinto per macchine a fasci di elettroni. In alternativa è possibile espellere gas inerte da un ugello, che è pratica comune nella saldatura.
I vantaggi di questa tecnica sono la porosità praticamente assente, la possibilità di depositare materiale su un pezzo già esistente (ad esempio per riparazioni), l'ampia disponibilità e scelta dei materiali, pochissimi scarti (il volume di stampa non è riempito da polveri).
Tra gli svantaggi invece ci sono la fisica molto complessa, i costi fissi molto elevati, la bassa qualità superficiale e la povera accuratezza dimensionale. È però possibile integrare molto spesso dei tool di manifattura sottrattiva (es: frese) all'interno della macchina DED così da combinare il meglio delle tecnologie in un approccio ibrido.[4]

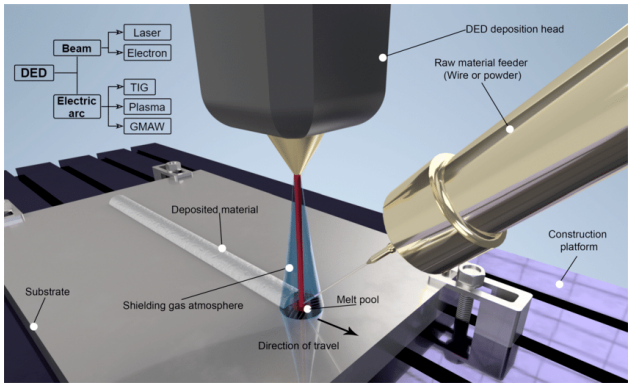


Figura 2.13: Illustrazione funzionamento macchina DED (immagine da riferimento)

Figura 2.14: Foto stampa DED (immagine da 3d-prints.com)

La **produzione di oggetti laminati** (Laminated Object Manufacturing, LOM) invece utilizza fogli di carta o plastica sovrapposti l'uno con l'altro, tagliati da un laser e fatti aderire da un rullo riscaldato. La tecnica quindi è estremamente veloce, economica e scalabile, però l'oggetto stampato ha una forte connotazione a layer ed è limitato nelle proprietà meccaniche e nella durabilità dal bonding tra strati.

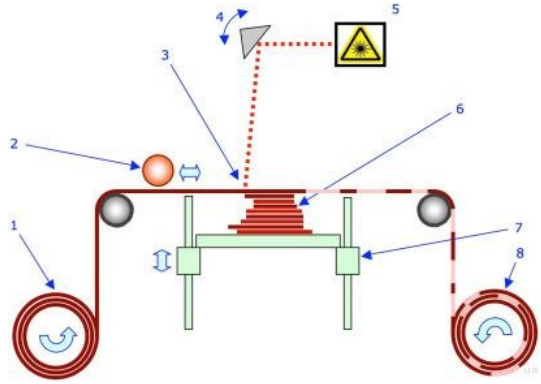


Figura 2.15: Illustrazione funzionamento LOM (immagine da theadditivemanufacturing.com)

Figura 2.16: Foto stampa LOM (immagine da theadditivemanufacturing.com)

[4]José Luis Dávila et al. "Hybrid manufacturing: a review of the synergy between directed energy deposition and subtractive processes". In: *The International Journal of Advanced Manufacturing Technology* 110 (2020), pp. 3377 –3390. URL: https://doi.org/10.1007/s00170-020-06062-7.

2.2 Tecnologia FDM

In questo lavoro l'attenzione è rivolta alla comune stampa 3D in plastica mediante estrusione di filamento (Fused Deposition Modeling, FDM). Essa è caratterizzata da bassi costi, sia della macchina sia del materiale, da una elevata area di stampa, da un basso livello di rischio per la sicurezza, da una certa semplicità strutturale.
Come accennato una stampante 3D FDM estrude il materiale termoplastico da un ugello riscaldato su un piatto di stampa seguendo una traiettoria preimpostata. Nella maggior parte dei casi il materiale è fornito sotto forma di bobine di filamento con diametro 1.75 mm oppure 2.85 mm, ma esistono macchine che possono lavorare direttamente con i pellet.

Figura 2.17: Stampante Prusa MK4, una delle più diffuse in ambito consumer (immagine da prusa3d.com)

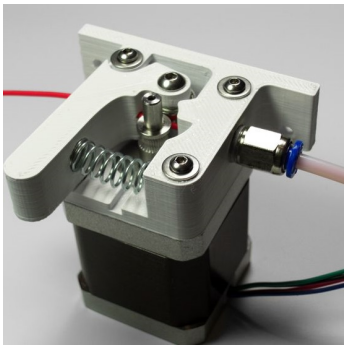


Figura 2.18: Foto di estrusore con drive gear (immagine da bernis-simple-bowden-extruder.com)

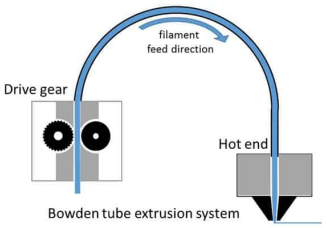


Figura 2.19: Schema funzionale di un sistema di tipo bowden (immagine da manufactur3dmag.com)

Il filamento è trascinato dal gruppo estrusore (drive gear), costituito da un motore stepper e da uno o più ingranaggi dentati detti hobb. Esso deve vincere gli attriti interni, muovere la giusta portata di materiale e garantire una sufficiente precisione e ripetibilità nelle fasi di ritrazione. Fondamentali sono il disegno dell'hobb e la regolazione della pressione con il filamento per non compromettere la qualità di stampa o deformare il materiale (grinding). È pratica comune infatti ricorrere più e più volte alla ritrazione durante i passaggi rapidi, in cui è necessario spostare la testina di stampa da una parte all'altra del piatto di stampa senza estrudere materiale. Se le impostazioni sono troppo aggressive o l'hardware non tarato bene, l'hobb può raschiare e assottigliare il filamento.

Una volta spinto dal gruppo estrusore il materiale raggiunge l'hotend per essere scaldato. Sono possibili diverse soluzioni costruttive, ciascuna con i propri vantaggi e svantaggi, ma le più comuni sono il sistema bowden e il sistema diretto.

Nei sistemi **bowden** si interpone una guaina flessibile tra gruppo estrusore e hotend, tipicamente un tubo di PTFE a basso attrito. Questo consente di fissare il drive gear sul telaio della stampante e di avere una massa in movimento estremamente ridotta (poche decine di grammi dell'hotend), a vantaggio di vibrazioni e velocità di stampa. È da apprezzare anche la possibilità di allontanare i motori e l'elettronica dalle zone più calde della macchine. Di contro, i maggiori attriti si traducono in tempi di risposta più lenti e peggiorano la precisione e il controllo dell'estrusione. Inoltre, non tutti i materiali possono essere utilizzati: quelli abrasivi hanno una pessima scorrevolezza a contatto con il PTFE e sono solitamente troppo rigidi per seguire la curvatura del tubo, mentre i materiali flessibili tendono ad ingobbarsi nel momento in cui vengono fatti entrare nell'ugello. Tali problemi diventano sempre più accentuati quanto più lungo è il tubo.

Figura 2.20: Esempio estrusore di tipo bowden (immagine da e3d-online.com)

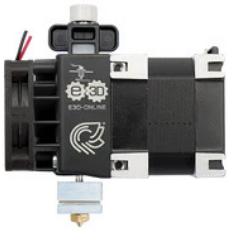


Figura 2.21: Esempio estrusore di tipo diretto (immagine da e3d-online.com)

Negli estrusori **diretti** invece l'ugello è posizionato immediatamente dopo l'uscita del drive gear. I vantaggi sono molteplici: totale controllo del processo di estrusione, velocità di ritrazione, completa compatibilità dei materiali e bassa potenza richiesta al motore. Sono anche più agevoli le operazioni di ispezione e di rimozione di eventuali ostruzioni.
Di contro ovviamente c'è la massa in movimento, con ripercussioni su velocità, accuratezza e qualità di stampa. Un problema tipico dei sistemi direct infatti è la presenza di ondulazioni (ringing) e perimetri non lisci nei pezzi stampati.

Passando all'**hotend**, esso ha la funzione di portare il materiale alla temperatura desiderata e di depositarlo correttamente sul piatto. Il filamento per prima cosa passa per un tubicino di acciaio inossidabile detto heat-break che ha lo scopo di isolare la zona calda e di confinare la fase di riscaldamento nel più breve tempo e nel più piccolo spazio possibile. Il materiale in questo stato infatti presenta problematiche di controllo e gestione rispetto al filamento solido che portano facilmente a surriscaldamenti o ostruzioni. Si capisce quindi che è necessario mantenere l'heat-break a temperatura ambiente raffreddandolo attivamente con un heat-sink e una ventola, durante la stampa ma anche a fine processo. Il ruolo della ventilazione quindi è essenziale per la buona riuscita del processo. Per gli stessi motivi prima di "staccare la spina" è buona pratica estrarre il filamento dall'hotend.

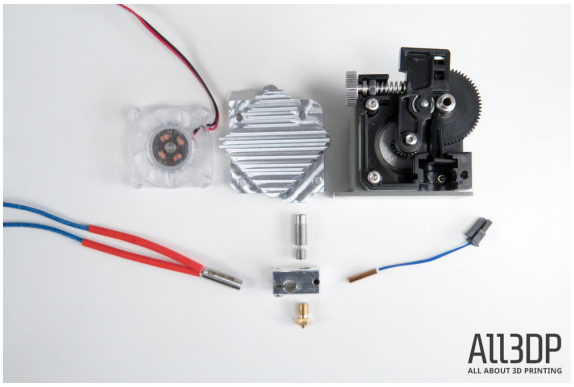


Figura 2.22: Vista esplosa componenti estrusore (immagine da all3dp.com)

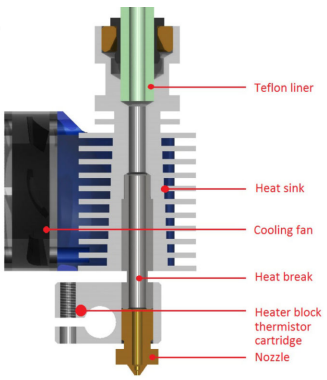


Figura 2.23: Schema costruttivo di un heatend (immagine da zen3d.hu)

Il blocco dove avviene la fusione è detto heat-block ed è solitamente fatto di un materiale conduttore quali alluminio o rame. Esso funge anche da elemento di supporto meccanico e di collegamento tra heat-break, ugello, riscaldatore elettrico e sonda per la temperatura (termistore o termocoppia).

Figura 2.24: Heat-block e alcuni nozzle per portate elevate (immagine da e3d-online.com)

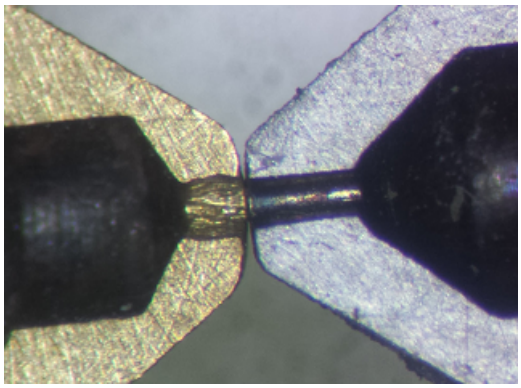


Figura 2.25: Confronto tra nozzle in ottone e in acciaio (immagine da e3d-online.com)

Il componente più critico dell'hotend è senza dubbio il **nozzle**, ossia l'ugello vero e proprio. Qui il filamento passa dai 1.75 mm o 2.85 mm di diametro ai 0.4 mm estrusi. Un corretto disegno della cross-section è essenziale per evitare ostruzioni e minimizzare le perdite di pressione.
Cambiare il nozzle può essere un modo molto semplice ed economico per aggiustare le caratteristiche della macchina. Se è richiesto un lavoro di precisione si possono montare ugelli più piccoli, fino ad un diametro di 0.1 mm, a scapito però di tempi di stampa che esplodono e di un rischio maggiorato di clogging.
Se invece l'obiettivo è ridurre i tempi di stampa conviene orientarsi su diametri più larghi, anche superiori al millimetro. Attenzione però che diametri troppo grandi richiedono una melt zone più ampia per portare in temperatura le portate maggiorate, ossia heat-block più lunghi con conseguenze negative sul controllo dell'estrusione.
Solitamente il materiale scelto per fabbricare un ugello è l'ottone per le buone proprietà di conducibilità termica e di lavorazione. I filamenti più abrasivi però tendono ad usurarlo molto velocemente e ad allargarne il diametro, come visibile dall'immagine soprastante, perciò per tali materiali sono stati sviluppati appositi ugelli in acciaio inossidabile o al carburo di tungsteno.

2.2.1 Processo di stampa e materiali

Come detto, il pezzo in tre dimensioni viene realizzato strato su strato estrudendo il materiale da un ugello in movimento. Serve quindi muovere l'hotend su tre assi indipendenti (x, y, z). Tipicamente questi assi sono ortogonali tra loro e azionati in modo molto diverso: i movimenti sull'asse z in genere sono molto lenti e piccoli, dell'ordine dei 0.1 mm, per cui l'azionamento è solitamente affidato a viti filettate oppure a viti a ricircolo di sfere. Invece i movimenti sul piano x-y sono tendenzialmente più veloci e ampi, per cui è preferibile usare un sistema di cinghie e pulegge.
In ogni caso, qualunque sia la configurazione meccanica, una stampante a tre assi richiede tre motori elettrici DC stepper, più un motore aggiuntivo per ciascun gruppo estrusore presente nella macchina. La scelta più comune in ambito consumer ricade sui motori NEMA17 con precisione di posizionamento di 1.8° oppure 0.9°, più che sufficiente per una macchina hobbistica e molto economici.

Le istruzioni GCODE per generare il pezzo quindi consistono in una lunga lista di spostamenti per ciascun asse e vengono tradotte dal firmware della scheda elettronica della macchina in movimenti rotatori del corrispondente motore.

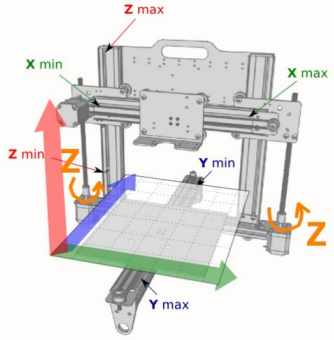


Figura 2.26: Illustrazione assi cartesiani (immagine da samanthavanrijs.nl)

```
;FLAVOR:Marlin
;TIME:1120
;Filament used: 0.087281m
;Layer height: 0.1
;MINX:65.2
;MINY:99.135
;MINZ:0.3
;MAXX:134.8
;MAXY:100
;MAXZ:20
;Generated with Cura_SteamEngine 5.2.1
M104 S200
M105
M109 S200
M82 ;absolute extrusion mode
G28 ;Home
G1 Z15.0 F6000 ;Move the platform down 15mm
;Prime the extruder
G92 E0
G1 F200 E3
G92 E0
G92 E0
G92 E0
```

Figura 2.27: Esempio istruzioni GCODE generate con il software di slicing Cura

Come già accennato, il workflow per realizzare un componente mediante tecnologia FDM si articola nei seguenti punti:

- Modellazione CAD della geometria solida;
- Slicing, ossia generazione delle istruzioni GCODE a partire dalla geometria CAD;
- Stampa del pezzo, seguendo le istruzioni GCODE;
- Post-processing, ad esempio per rimuovere supporti o migliorare la finitura superficiale;

Oltre alle caratteristiche meccaniche della macchina, anche i parametri usati nella fase di slicing hanno una forte influenza sulla qualità di stampa. Delle centinaia di regolazioni possibili, le più influenti sul risultato sono: percentuale infill, pattern infill, altezza layer, temperatura.

Ovviamente anche il materiale ha un ruolo preponderante sul risultato, non solo per le proprietà meccaniche ma anche per la facilità con cui può essere stampato. Tra i filamenti di uso più comune si trovano:

- Il PLA è il materiale "di default" per l'uso amatoriale; molto economico e semplice da stampare per via della bassa temperatura di estrusione e dell'ottima adesione tra layer, anche in assenza di piatto riscaldato. È riciclabile e biodegradabile sotto opportune condizioni, però ha una bassa resistenza al calore e alla radiazione solare.
- La plastica ABS è una delle prime ad essere state usate in ambito stampa 3D, e continua ad essere molto comune anche oggigiorno per via dei suoi costi molto bassi, della sua leggerezza, delle proprietà termiche e meccaniche molto buone. In fase di post-processing è comune far esporre il pezzo a vapori di acetone per migliorarne l'estetica. Tra gli svantaggi si citano la tendenza molto marcata al warping, la forte contrazione delle dimensioni una volta raffreddato e l'odore sgradevole in fase di stampa (con emissione di particelle nocive per l'uomo).
- Il PETG è un polimero derivato dal comune PET appositamente per la stampa 3D; risulta infatti molto facile da stampare, ha buone proprietà meccaniche e un basso tasso di restringimento. Tra gli altri pregi vi sono la durabilità, la resistenza chimica e la riciclabilità. Gli svantaggi invece riguardano il costo più alto e la densità ρ elevata, inoltre soffre molto il fenomeno dello stringing e la qualità scadente in bridging.
- I filamenti di Nylon ad oggi sono più diffusi in ambito industriale che in ambito consumer. Infatti, anche se offrono buone proprietà meccaniche, di deformazione e di resistenza al calore, purtroppo la stampa risulta spesso molto difficoltosa a causa della forte igroscopicità e della tendenza estrema al warping;
- L'ASA ha proprietà simili all'ABS ed è particolarmente indicato per pezzi e/o componenti da impiegare all'aperto grazie alla sua eccellente resistenza alle radiazioni UV. Offre anche una bassa tendenza al warping, però è piuttosto costoso e richiede un'alta temperatura di estrusione;

Tendenzialmente i materiali migliori sono quelli "carbon-filled", ossia filamenti rinforzati con fibre di carbonio triturate per un contenuto in volume fino al 20%. Pur ereditando i parametri di stampa del filamento base, le proprietà meccaniche sono di gran lunga superiori, sono più facili da stampare, soffrono poco il warping e il restringimento. Gli svantaggi risiedono nel costo molto più alto e nel fatto che sono molto abrasivi per l'ugello. Inoltre soffrono frequentemente problemi di clogging.

2.3 Fenomenologia

I materiali termoplastici possono essere considerati isotropi nelle loro proprietà. Ciononostante, la costruzione a layer sovrapposti fa sì che il pezzo finale sia fortemente anisotropo e mostri comportamenti differenti a seconda dell'orientazione. In particolare, la rigidezza e la resistenza in direzione z risultano ridotte in modo significativo, almeno del $20-30\%$. Del resto per avere rottura in questo caso è sufficiente delaminare due qualsiasi layer, senza arrivare al cedimento del materiale.

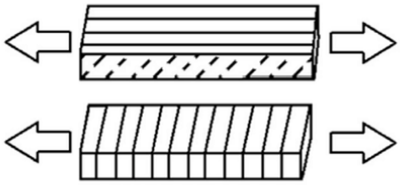


Figura 2.28: Illustrazione problematica anisotropia (immagine da 3dprint.org)

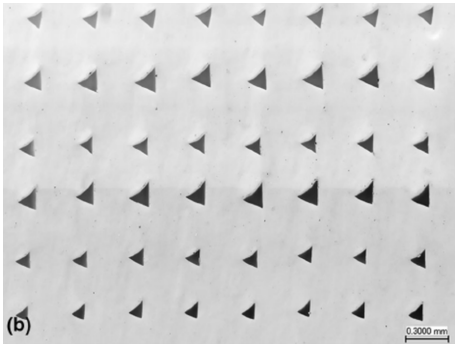


Figura 2.29: Vuoti interni nel processo FDM (immagine da 3dprint.org)

Il processo di stampa infatti introduce dei difetti e la presenza di vuoti all'interno della parte. Anche con un infill pieno, infatti, la deposizione di sezioni "oblong" arrotondate alle estremità rende geometricamente impossibile il riempimento completo: $A_{void} = h^2 - \pi\,(h/2)^2$.
Detto ciò è comunque conveniente cercare di riempirli il più possibile, ad esempio impostando in fase di slicing un overlap factor che va a ridurre lo spacing tra una passata e l'altra.
Da questo punto di vista potrebbe sembrare che ridurre l'altezza dei layer h porti a migliori proprietà meccaniche. In realtà ciò non avviene sempre: il processo di stampa è molto complesso e ci sono molti fattori che influenzano il risultato finale.

Questo lavoro infatti si pone l'obiettivo di realizzare un software in grado di prevedere tali fattori a partire da equazioni di governo quanto più generiche possibile.

Tra le principali fenomenologie agenti durante il processo di stampa vi sono:

- Liquefazione e deposizione ad opera dell'hotend;
- Effetto della temperatura sulle proprietà chimiche e fisiche del materiale (fisica dei polimeri), ad esempio il degrado termico della struttura molecolare;
- Generazione di forti gradienti termici e trasferimento di calore per conduzione interna;
- Effetto del raffreddamento dovuto al piatto, all'aria circostante e all'eventuale ventilazione;
- Cristallizzazione del materiale, che influenza ed è influenzata dal trasferimento termico e dalle proprietà termofisiche;

- Bonding con il materiale vicino quando $T > T_g$ con un'opportuna cinetica;
- Contrazione del materiale con il raffreddamento e formazione di stress residui, che possono indurre warping e/o distorsioni a seconda della situazione contingente;

Tutti questi fenomeni agenti dipendono fortemente dai parametri di stampa. In questo lavoro ovviamente non è possibile modellarli tutti in dettaglio, per cui ci si limita a considerare i contributi più importanti, tenendo d'occhio l'onere computazionale.

2.4 Simulazione di processo

Come accennato nel capitolo introduttivo la maggior parte dei software per la simulazione di processo in ambito Additive Manufacturing è perlopiù rivolta all'industria e alle tecnologie per materiali metallici.
Detto questo in letteratura sono presenti lavori incentrati sulla simulazione della stampa 3D FDM. Khanafer et al[5] ad esempio impiegano il metodo agli elementi finiti (Finite Element Method, FEM) e seguendo le istruzioni GCODE attivano i vari elementi che discretizzano il volume stampato. Essi però si limitano a simulare la storia termica del pezzo, pur introducendo degli elementi di nonlinearità, ma mancano di tenere conto della cinetica di cristallizzazione e della contrazione del materiale.
Altri lavori invece si soffermano su uno o sull'altro aspetto. In certi casi addirittura si approssima il materiale termoplastico con un fluido newtoniano molto viscoso e si fa evolvere la simulazione con un codice CFD accoppiato ad un algoritmo di element activation. Comminal et al[6] giustificano queste ipotesi molto semplificative con il loro obiettivo di studiare esclusivamente l'effetto del gap nozzle-sottostrato e della velocità di estrusione sulla geometria della forma estrusa.

In questo lavoro si vuole sviluppare un modello termico e termo-meccanico lineare per poi risolverlo con un metodo meshless basato sulle differenze finite generalizzate.
L'approccio meshless in particolare si sposa bene con la simulazione di processo: la gestione di un dominio di dimensioni crescenti è del tutto banale, non richiede algoritmi di attivazione delle celle e individua automaticamente quali nodi sono interni e quali appartengono ad un contorno. Di conseguenza riconosce automaticamente la presenza di vuoti e assegna le condizioni al contorno. Non è richiesto alcun modello di degradazione dei materiali, necessario invece nel FEM per tenere conto dei vuoti.

Il modello fisico proposto consiste in un modello termico lineare e in un modello termomeccanico disaccoppiato, anch'esso lineare e lanciato come post-processing di quello termico. Non viene considerata la cinetica di cristallizzazione e si assumono le proprietà del materiali costanti non dipendenti con la temperatura. Usando i valori a temperatura ambiente infatti l'analisi sarà in vantaggio di sicurezza riguardo gli stress indotti.
In questo lavoro non è di interesse simulare l'hotend e la fase di deposizione. Il materiale è assunto di forma ad oblong estruso e compare improvvisamente a tratti con temperatura $T = T_{extr}$ e si raffredda seguendo le equazioni di governo.

[5] Khalil Khanafer et al. "Thermal analysis of fused deposition modeling process based finite element method: Simulation and parametric study". In: *Numerical Heat Transfer, Part A: Applications* 81 (2022), pp. 94 –118. DOI: 10.1080/10407782.2022.2038972.

[6] Raphaël Comminal et al. "Numerical modeling of the strand deposition flow in extrusion-based additive manufacturing". In: *Additive Manufacturing* 20 (2018), pp. 68–76. URL: https://doi.org/10.1016/j.addma.2017.12.013.

L'obiettivo della simulazione di processo del resto è prevedere le proprietà del pezzo stampato ed evitare che la stampa fallisca. Per entrambi gli scopi è sufficiente seguire il raffreddamento, ciò che avviene prima della deposizione su un materiale liquido ha poca se non nessuna influenza, in particolare sulle tensioni indotte.
Al contrario, la dinamica della deposizione influisce fortemente sulla forma geometrica del materiale estruso. Per parametri di stampa non troppo estremi però la forma non si discosta eccessivamente dall'oblong estruso, in compenso l'onere computazionale è significativamente diverso per la risoluzione spaziale e le scale temporali da risolvere.

2.5 Equazioni di governo - bilancio termico

Si consideri la generica equazione di bilancio energetico in forma differenziale per un solido:

$$\frac{\partial \left(\rho c_p T\right)}{\partial t} = \nabla \cdot \left(k \nabla T\right) + Q_v$$

Dove ρ è la densità del materiale, c_p è il calore specifico (a pressione costante), k è il coefficiente di conduzione termica. Invece Q_v è un termine sorgente, positivo se genera calore.
Nell'ipotesi di materiale lineare, isotropo e omogeneo su tutto il dominio $k = cost$, $\rho = cost$, $c_p = cost$ si ottiene la nota equazione del calore:

$$\frac{\partial T}{\partial t} = \frac{k}{\rho c_p} \nabla^2 T + q_v$$

In cui ∇^2 è l'operatore laplaciano, a conferma del comportamento diffusivo del modello. In tre dimensioni si ha:

$$\frac{\partial T}{\partial t} = \frac{k}{\rho c_p} \left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} + \frac{\partial^2 T}{\partial z^2} \right) + q_v$$

Si ricorda inoltre che i flussi di calore sono proporzionali ai gradienti di temperatura e possono essere calcolati in post-processing semplicemente scrivendo le definizioni:

$$q_x = -k \frac{\partial T}{\partial x} \qquad q_y = -k \frac{\partial T}{\partial y} \qquad q_z = -k \frac{\partial T}{\partial z}$$

Per tenere conto della temperatura del piatto si impone una condizione al contorno di Dirichlet su tutti i nodi con $z = 0$:

$$T = T_{bed}$$

Tutti gli altri nodi al contorno invece sono a contatto con l'aria e risentono di un flusso di calore convettivo, qui modellato con una condizione al contorno di Robin:

$$-k \frac{\partial T}{\partial n} = h_{conv} \left(T - T_{amb}\right)$$

Dove n è la direzione normale uscente. h_{conv} è il coefficiente di convezione e vale circa $2 - 3$ nel caso di convezione naturale e arriva a 20 nel caso di convezione forzata. Più in generale, le condizioni al contorno possono essere scritte nella forma:

$$g_1 T + g_2 \frac{\partial T}{\partial n} = g_3 \quad \rightarrow \quad g_1 T + g_2 \left(n_x \frac{\partial T}{\partial x} + n_y \frac{\partial T}{\partial y} + n_z \frac{\partial T}{\partial z} \right) = g_3$$

Con coefficienti g_1, g_2, g_3 opportunamente scelti per ciascun nodo al contorno.

2.6 Equazioni di governo - equilibrio strutturale

Nell'ambito della teoria dell'elasticità lineare le equazioni di governo sono date dalle equazioni di equilibrio, dalle relazioni di congruenza e dalle equazioni costitutive:

$$\rho\frac{\partial^2\vec{u}}{\partial t^2} = \nabla\cdot\sigma + \vec{F} \qquad \varepsilon = \frac{1}{2}\left(\nabla\vec{u} + \nabla\vec{u}^\top\right) \qquad \sigma = C\varepsilon$$

In particolare, per un materiale elastico le equazioni costitutive si scrivono come segue:

$$\begin{Bmatrix}\sigma_{xx}\\ \sigma_{yy}\\ \sigma_{zz}\\ \tau_{yz}\\ \tau_{xz}\\ \tau_{xy}\end{Bmatrix} = \frac{E}{(1+\nu)(1-2\nu)}\begin{bmatrix}1-\nu & \nu & \nu & & & \\ \nu & 1-\nu & \nu & & & \\ \nu & \nu & 1-\nu & & & \\ & & & \frac{1-2\nu}{2} & & \\ & & & & \frac{1-2\nu}{2} & \\ & & & & & \frac{1-2\nu}{2}\end{bmatrix}\begin{Bmatrix}\varepsilon_{xx}\\ \varepsilon_{yy}\\ \varepsilon_{zz}\\ \gamma_{yz}\\ \gamma_{xz}\\ \gamma_{xy}\end{Bmatrix}$$

Così facendo le equazioni di equilibrio strutturale per una formulazione agli spostamenti si riducono ad un sistema di tre equazioni differenziali alle derivate parziali (PDE) di secondo ordine nel tempo (equazioni di Navier-Cauchy):

$$\rho\frac{\partial^2\vec{u}}{\partial t^2} = \nabla\cdot\sigma + \vec{F} \;\rightarrow\; \begin{cases}\rho\frac{\partial^2 u}{\partial t^2} = \mu\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}+\frac{\partial^2 w}{\partial z^2}\right) + (\lambda+\mu)\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 v}{\partial x\partial y}+\frac{\partial^2 w}{\partial x\partial z}\right) + F_x\\ \rho\frac{\partial^2 v}{\partial t^2} = \mu\left(\frac{\partial^2 v}{\partial x^2}+\frac{\partial^2 v}{\partial y^2}+\frac{\partial^2 v}{\partial z^2}\right) + (\lambda+\mu)\left(\frac{\partial^2 u}{\partial x\partial y}+\frac{\partial^2 v}{\partial y^2}+\frac{\partial^2 w}{\partial y\partial z}\right) + F_y\\ \rho\frac{\partial^2 w}{\partial t^2} = \mu\left(\frac{\partial^2 w}{\partial x^2}+\frac{\partial^2 w}{\partial y^2}+\frac{\partial^2 w}{\partial z^2}\right) + (\lambda+\mu)\left(\frac{\partial^2 u}{\partial x\partial z}+\frac{\partial^2 v}{\partial y\partial z}+\frac{\partial^2 w}{\partial z^2}\right) + F_z\end{cases}$$

Per quanto riguarda le condizioni al contorno, invece, per ciascun nodo vanno imposte tre equazioni. Ad esempio, sui nodi a contatto con il piatto si impone un incastro perfetto mediante le seguenti condizioni di Dirichlet:

$$u = 0 \qquad v = 0 \qquad w = 0$$

I carichi esterni invece vanno imposti con una condizione al contorno naturale specificando una trazione $\vec{t} = \sigma\vec{n}$. In due dimensioni ad esempio si ha:

$$t_x = \sigma_{xx}n_x + \tau_{xy}n_y = n_x\frac{E}{1-\nu^2}\frac{\partial u}{\partial x} + n_x\frac{E\nu}{1-\nu^2}\frac{\partial v}{\partial y} + n_y\frac{E}{2(1+\nu)}\left(\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\right)$$

$$t_y = \tau_{xy}n_x + \sigma_{yy}n_y = n_x\frac{E}{2(1+\nu)}\left(\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\right) + n_y\frac{E\nu}{1-\nu^2}\frac{\partial u}{\partial x} + n_y\frac{E}{1-\nu^2}\frac{\partial v}{\partial y}$$

Dove t_x, t_y sono le trazioni specificate dall'utente. Più in generale le condizioni al contorno si possono scrivere come:

$$\begin{cases} g_{1u}u + g_{2u}n_x\left[(2\mu+\lambda)\frac{\partial u}{\partial x} + \lambda\frac{\partial v}{\partial y} + \lambda\frac{\partial w}{\partial z}\right] + g_{2u}n_y\mu\left(\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\right) + g_{2u}n_z\mu\left(\frac{\partial u}{\partial z}+\frac{\partial w}{\partial x}\right) = g_{3u}\\ g_{1v}v + g_{2v}n_x\mu\left(\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\right) + g_{2v}n_y\left[\lambda\frac{\partial u}{\partial x} + (2\mu+\lambda)\frac{\partial v}{\partial y} + \lambda\frac{\partial w}{\partial z}\right] + g_{2v}n_z\mu\left(\frac{\partial v}{\partial z}+\frac{\partial w}{\partial y}\right) = g_{3v}\\ g_{1w}w + g_{2w}n_x\mu\left(\frac{\partial u}{\partial z}+\frac{\partial w}{\partial x}\right) + g_{2w}n_y\mu\left(\frac{\partial v}{\partial z}+\frac{\partial w}{\partial y}\right) + g_{2w}n_z\left[\lambda\frac{\partial u}{\partial x} + \lambda\frac{\partial v}{\partial y} + (2\mu+\lambda)\frac{\partial w}{\partial z}\right] = g_{3w}\end{cases}$$

Così facendo si possono anche mettere fondazioni elastiche con condizioni di Robin, con coefficienti di rigidezza differenti nelle tre direzioni x,y,z, semplicemente espandendo la scrittura e facendo comparire termini u, v, w su ciascuna riga.

In fase di post-processing può essere utile calcolare le tensioni usando le equazioni costitutive. Da queste è immediato calcolare la tensione di von Mises rappresentativa dello stato tensionale:

$$\sigma_v = \sqrt{\sigma_{xx}^2 + \sigma_{yy}^2 - \sigma_{xx}\sigma_{yy} + 3\tau_{xy}}$$

In due dimensioni nell'ipotesi di formulazione plane stress.

2.7 Equazioni termo-meccaniche

Durante il processo di stampa il materiale è sottoposto ad una sequenza di cicli termici di intensità non trascurabile. Quando scaldato e raffreddato esso si espande o contrae, e può sviluppare degli stress termici indesiderati. A stampa completata questi stress permangono nel pezzo come stress residui e possono alterarne le proprietà meccaniche e la durabilità.
In questo lavoro per semplicità vengono considerati i soli stress termici indotti dall'espansione termica. Non vengono modellati altri fenomeni, ad esempio gli shock termici.
Così facendo si ottiene una semplice formulazione lineare del tipo $\sigma = E\alpha(T - T_{ref})$, dove E è il modulo di Young del materiale (in Pa), α è il coefficiente di espansione termica (in K^{-1}), T_{ref} è la temperatura di riferimento rispetto alla quale si assegna allungamento nullo. Trattandosi di un modello lineare questi coefficienti sono ipotizzati costanti e non variano con la temperatura.
Estendendo questa formulazione all'interno delle equazioni costitutive di un materiale omogeneo isotropo si perviene a:

$$\begin{Bmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{zz} \\ \tau_{yz} \\ \tau_{xz} \\ \tau_{xy} \end{Bmatrix} = \frac{E}{(1+\nu)(1-2\nu)} \begin{bmatrix} 1-\nu & \nu & \nu & & & \\ \nu & 1-\nu & \nu & & & \\ \nu & \nu & 1-\nu & & & \\ & & & \frac{1-2\nu}{2} & & \\ & & & & \frac{1-2\nu}{2} & \\ & & & & & \frac{1-2\nu}{2} \end{bmatrix} \begin{Bmatrix} \varepsilon_{xx} - \alpha\Delta T \\ \varepsilon_{yy} - \alpha\Delta T \\ \varepsilon_{zz} - \alpha\Delta T \\ \gamma_{yz} \\ \gamma_{xz} \\ \gamma_{xy} \end{Bmatrix}$$

Tali equazioni costitutive vanno sostituite nelle equazioni di equilibrio $\rho\vec{u}_{tt} = \nabla \cdot \sigma + \vec{F}$ con lo scopo di ricavare le equazioni di governo nelle sole incognite di spostamento u, v, w.

2.7.1 Formulazione 2D plane stress

Prima di affrontare il problema 3D generico è utile risolvere casi 2D più semplici, possibilmente con soluzione analitica. Molti di questi presentano uno stato piano di tensione (plane stress):

$$\begin{Bmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \tau_{xy} \end{Bmatrix} = \frac{E}{1-\nu^2} \begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & \frac{1-\nu}{2} \end{bmatrix} \begin{Bmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \gamma_{xy} \end{Bmatrix} - \frac{E\alpha}{1-\nu}(T - T_{ref}) \begin{Bmatrix} 1 \\ 1 \\ 0 \end{Bmatrix}$$

Si ricorda che le deformazioni sono legate agli spostamenti mediante le relazioni di congruenza:

$$\varepsilon_{xx} = \frac{\partial u}{\partial x} \qquad \varepsilon_{yy} = \frac{\partial v}{\partial y} \qquad \gamma_{xy} = \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}$$

Da cui si ricava che:

$$\rho\frac{\partial^2 u}{\partial t^2} = \frac{\partial \sigma_{xx}}{\partial x} + \frac{\partial \tau_{xy}}{\partial y} + F_x = \frac{E}{1-\nu^2}\frac{\partial^2 u}{\partial x^2} + \frac{E\nu}{1-\nu^2}\frac{\partial^2 v}{\partial x \partial y} + \frac{E}{2(1+\nu)}\left(\frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 v}{\partial x \partial y}\right) + F_x - \frac{E\alpha}{1-\nu}\frac{\partial T}{\partial x}$$

$$\rho\frac{\partial^2 v}{\partial t^2} = \frac{\partial \tau_{xy}}{\partial x} + \frac{\partial \sigma_{yy}}{\partial y} + F_y = \frac{E}{2(1+\nu)}\left(\frac{\partial^2 u}{\partial x \partial y} + \frac{\partial^2 v}{\partial x^2}\right) + \frac{E\nu}{1-\nu^2}\frac{\partial^2 u}{\partial x \partial y} + \frac{E}{1-\nu^2}\frac{\partial^2 v}{\partial y^2} + F_y - \frac{E\alpha}{1-\nu}\frac{\partial T}{\partial y}$$

Che sono le equazioni di Navier-Cauchy con un carico volumetrico aggiuntivo.
Invece, per quanto riguarda le condizioni al contorno naturali $\vec{t} = \sigma\vec{n}$:

$$t_x = \sigma_{xx}n_x + \tau_{xy}n_y = n_x\frac{E}{1-\nu^2}\frac{\partial u}{\partial x} + n_x\frac{E\nu}{1-\nu^2}\frac{\partial v}{\partial y} + n_y\frac{E}{2(1+\nu)}\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right) - n_x\frac{E\alpha}{1-\nu}(T - T_{ref})$$

$$t_y = \tau_{xy}n_x + \sigma_{yy}n_y = n_x\frac{E}{2(1+\nu)}\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right) + n_y\frac{E\nu}{1-\nu^2}\frac{\partial u}{\partial x} + n_y\frac{E}{1-\nu^2}\frac{\partial v}{\partial y} - n_y\frac{E\alpha}{1-\nu}(T - T_{ref})$$

Dove t_x, t_y sono le trazioni specificate dall'utente.

2.7.2 Formulazione 3D

Introducendo le costanti di Lamè $\lambda = \frac{E\nu}{(1+\nu)(1-2\nu)}$ e $\mu = G = \frac{E}{2(1+\nu)}$ le equazioni costitutive si possono scrivere come segue:

$$\begin{Bmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{zz} \\ \tau_{yz} \\ \tau_{xz} \\ \tau_{xy} \end{Bmatrix} = \begin{bmatrix} 2\mu+\lambda & \lambda & \lambda & & & \\ \lambda & 2\mu+\lambda & \lambda & & & \\ \lambda & \lambda & 2\mu+\lambda & & & \\ & & & \mu & & \\ & & & & \mu & \\ & & & & & \mu \end{bmatrix} \begin{Bmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{zz} \\ \gamma_{yz} \\ \gamma_{xz} \\ \gamma_{xy} \end{Bmatrix} - \frac{E\alpha}{1-2\nu}\Delta T \begin{Bmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{Bmatrix}$$

Per cui le equazioni di equilibrio $\rho\frac{\partial^2\vec{u}}{\partial t^2} = \nabla\cdot\sigma + \vec{F}$ si sviluppano nel modo seguente:

$$\begin{cases} \rho\frac{\partial^2 u}{\partial t^2} = \mu\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 w}{\partial z^2}\right) + (\lambda+\mu)\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 v}{\partial x\partial y} + \frac{\partial^2 w}{\partial x\partial z}\right) + F_x + \frac{E\alpha}{1-2\nu}\frac{\partial T}{\partial x} \\ \rho\frac{\partial^2 v}{\partial t^2} = \mu\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} + \frac{\partial^2 v}{\partial z^2}\right) + (\lambda+\mu)\left(\frac{\partial^2 u}{\partial x\partial y} + \frac{\partial^2 v}{\partial y^2} + \frac{\partial^2 w}{\partial y\partial z}\right) + F_y + \frac{E\alpha}{1-2\nu}\frac{\partial T}{\partial y} \\ \rho\frac{\partial^2 w}{\partial t^2} = \mu\left(\frac{\partial^2 w}{\partial x^2} + \frac{\partial^2 w}{\partial y^2} + \frac{\partial^2 w}{\partial z^2}\right) + (\lambda+\mu)\left(\frac{\partial^2 u}{\partial x\partial z} + \frac{\partial^2 v}{\partial y\partial z} + \frac{\partial^2 w}{\partial z^2}\right) + F_z + \frac{E\alpha}{1-2\nu}\frac{\partial T}{\partial z} \end{cases}$$

Analogamente, le condizioni al contorno diventano:

$$\begin{cases} g_{1u}u + g_{2u}n_x\left[(2\mu+\lambda)\frac{\partial u}{\partial x} + \lambda\frac{\partial v}{\partial y} + \lambda\frac{\partial w}{\partial z} - \frac{E\alpha\Delta T}{1-2\nu}\right] + g_{2u}n_y\mu\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right) + g_{2u}n_z\mu\left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x}\right) = g_{3u} \\ g_{1v}v + g_{2v}n_x\mu\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right) + g_{2v}n_y\left[\lambda\frac{\partial u}{\partial x} + (2\mu+\lambda)\frac{\partial v}{\partial y} + \lambda\frac{\partial w}{\partial z} - \frac{E\alpha\Delta T}{1-2\nu}\right] + g_{2v}n_z\mu\left(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y}\right) = g_{3v} \\ g_{1w}w + g_{2w}n_x\mu\left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x}\right) + g_{2w}n_y\mu\left(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y}\right) + g_{2w}n_z\left[\lambda\frac{\partial u}{\partial x} + \lambda\frac{\partial v}{\partial y} + (2\mu+\lambda)\frac{\partial w}{\partial z} - \frac{E\alpha\Delta T}{1-2\nu}\right] = g_{3w} \end{cases}$$

Anche in questo caso si è in presenza di una formulazione identica a quella meccanica, ma con un carico volumetrico aggiuntivo e condizioni al contorno alterate.

Capitolo 3

Metodo GFDM

Nonostante i progressi degli ultimi decenni nel campo di generazione automatica della mesh, per casi complessi è ancora necessario intervenire manualmente. Tra questi si ritrovano le grandi deformazioni e i domini crescenti nel tempo tipici delle simulazioni di processo. In particolare nel campo della CFD, non è raro impiegare diverse giornate soltanto per generare la griglia di calcolo. Si tratta di un'attività dispendiosa, non sempre automatizzabile e che richiede molta esperienza.
Al contrario, i metodi meshless fanno uso solamente di una nuvola di punti/particelle (point-cloud) disposti in modo irregolare senza alcuna connettività. Gli algoritmi di generazione sono molto più semplici, efficienti e facilmente automatizzabili. Anche la modifica di una nuvola di punti è molto più semplice del remeshing di un intero dominio, il che porta all'ulteriore vantaggio di poter disporre di un efficiente sistema di raffinamento adattativo locale.

In questo capitolo viene presentato in dettaglio il funzionamento del metodo numerico e le sue caratteristiche in relazione agli approcci più comuni. Le equazioni di governo sono quindi discretizzate con l'obiettivo di ricavare il sistema di equazioni lineari da risolvere in Julia.
Verrà poi esaminata la convenienza dell'applicazione di alcuni dettagli/miglioramenti, come la decomposizione SVD o le varianti del metodo GFDM.

3.1 Differenze finite generalizzate

Si consideri una generica equazione differenziale alle derivate parziali (Partial Differential Equation, PDE) la cui soluzione incognita è $u(x, y)$ all'interno di un dominio Ω. Tale dominio è discretizzato mediante una nuvola di punti, e ciascun punto avrà un numero N di nodi vicini.

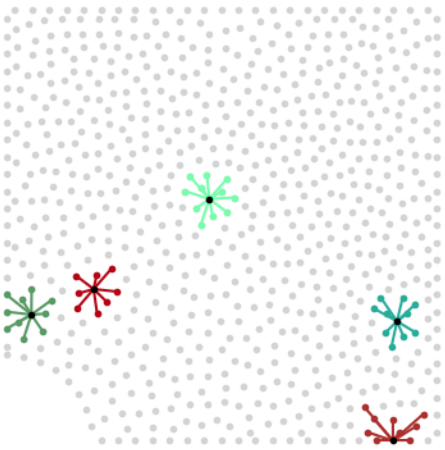


Figura 3.1: Esempio di pointcloud in un dominio 2D e connettività di alcuni nodi

Senza perdita di generalità è possibile limitarsi ad un dominio 2D e ad una sola equazione scalare.
Si consideri un generico nodo i con N nodi vicini. Sia u_0 il valore della funzione incognita in tale punto e u_j il valore nel j-esimo nodo vicino. Si scriva l'espansione in serie di Taylor per approssimare la funzione $u(x, y)$ nell'intorno del nodo i:

$$u_j = u_0 + ax_j + by_j + cx_j^2 + dy_j^2 + ex_jy_j$$

Dove (x_j, y_j, z_j) sono le coordinate del j-esimo nodo rispetto al nodo centrale i. a, b, c, d, e sono coefficienti incogniti. Così facendo si costruisce un'interpolazione polinomiale di secondo grado per ciascun nodo j.
Se si hanno esattamente 5 nodi vicini è possibile scrivere e mettere a sistema 5 equazioni e ricavare i coefficienti del polinomio:

$$\begin{bmatrix} x_1 & y_1 & x_1^2 & y_1^2 & x_1y_1 \\ x_2 & y_2 & x_2^2 & y_2^2 & x_2y_2 \\ x_3 & y_3 & x_3^2 & y_3^2 & x_3y_3 \\ x_4 & y_4 & x_4^2 & y_4^2 & x_4y_4 \\ x_5 & y_5 & x_5^2 & y_5^2 & x_5y_5 \end{bmatrix} \begin{Bmatrix} a \\ b \\ c \\ d \\ e \end{Bmatrix} = \begin{Bmatrix} u_1 - u_0 \\ u_2 - u_0 \\ u_3 - u_0 \\ u_4 - u_0 \\ u_5 - u_0 \end{Bmatrix}$$

Una volta noti i coefficienti è facile ricavare le derivate mediante delle semplici combinazioni lineari:

$$u(x, y) = u_0 + ax + by + cx^2 + dy^2 + exy$$

$$\rightarrow \frac{\partial u}{\partial x}(x, y) = a + 2cx + ey \quad \rightarrow \quad \left.\frac{\partial u}{\partial x}\right|_0 = a = \sum_{j=1}^{5} c_{1j} (u_j - u_0)$$

$$\rightarrow \frac{\partial u}{\partial y}(x, y) = b + 2dy + ex \quad \rightarrow \quad \left.\frac{\partial u}{\partial y}\right|_0 = b = \sum_{j=1}^{5} c_{2j} (u_j - u_0)$$

$$\rightarrow \frac{\partial^2 u}{\partial x^2}(x, y) = 2c \quad \rightarrow \quad \left.\frac{\partial^2 u}{\partial x^2}\right|_0 = 2c = 2\sum_{j=1}^{5} c_{3j} (u_j - u_0)$$

$$\rightarrow \frac{\partial^2 u}{\partial y^2}(x, y) = 2d \quad \rightarrow \quad \left.\frac{\partial^2 u}{\partial y^2}\right|_0 = 2d = 2\sum_{j=1}^{5} c_{4j} (u_j - u_0)$$

$$\rightarrow \frac{\partial^2 u}{\partial x \partial y}(x, y) = e \quad \rightarrow \quad \left.\frac{\partial^2 u}{\partial x \partial y}\right|_0 = e = \sum_{j=1}^{5} c_{5j} (u_j - u_0)$$

I coefficienti di tali combinazioni lineari prendono il nome di **stencil**.
Nota che quanto fatto finora è del tutto analogo al metodo delle differenze finite tradizionale (Finite Difference Method, FDM). Infatti il GFDM può essere visto come una generalizzazione al caso di nodi irregolari.
Se per un'approssimazione alle differenze finite equispaziate del II ordine i coefficienti sono $(-1/2, 0, +1/2)$ per la derivata prima e $(+1, -2, +1)$ per la derivata seconda, qui invece i coefficienti dipendono dalla posizione relativa dei nodi vicini (detti anche satelliti).

Ovviamente l'approccio appena descritto funziona bene solo se è possibile individuare esattamente 5 nodi vicini non allineati, altrimenti il problema non è ben posto. Tali nodi inoltre è bene che siano ben distribuiti attorno al punto centrale affinché l'approssimazione sia accurata.

Per superare il problema della scelta dei nodi è possibile ricorrere alla tecnica dei **minimi quadrati** (Least Squares, LS). In questa formulazione è possibile considerare un numero arbitrario N di nodi vicini per approssimare le derivate nel nodo centrale.

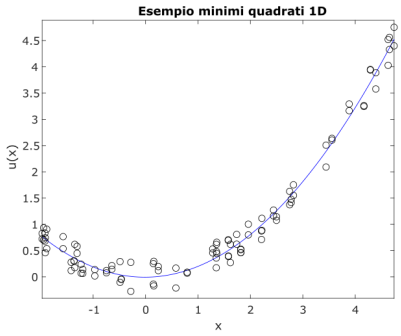


Figura 3.2: Esempio di approssimazione mediante minimi quadrati

Il polinomio di secondo grado in questo caso non interpola i valori u_j in tutti i nodi, anzi se ne discosta di un errore $e_j = u_0 + ax_j + by_j + cx_j^2 + dy_j^2 + ex_jy_j - u_j$. Si vuole perciò trovare quel polinomio che minimizza la somma dei quadrati di questi errori:

$$\min \sum_{j=1}^{N} w_j^2 e_j^2$$

Dove w_j sono dei pesi assegnati a ciascun nodo per rendere più influenti i nodi vicini e meno impattanti i nodi lontani. Tipicamente si sceglie una distribuzione gaussiana:

$$w_j = w(x_j, y_j) = \exp\left(-\alpha \frac{x_j^2 + y_j^2}{h^2}\right)$$

Dove α è una costante positiva, solitamente compresa tra 2 e 8, $x_j^2 + y_j^2$ è la distanza radiale dal nodo centrale al quadrato, mentre h è la cosiddetta **smoothing length** e definisce la dimensione del supporto/stella. Per il prosieguo di questo lavoro si prendono $\alpha = 6$ e h pari alla distanza del nodo più lontano.

Per ricavare i coefficienti a, b, c, d, e del polinomio si procede come mostrato da Benito et al.[1] Per prima cosa si scrivono in forma matriciale gli errori e_j commessi dall'approssimazione su ciascun nodo:

$$\begin{Bmatrix} e_1 \\ e_2 \\ \vdots \\ e_j \\ \vdots \\ e_N \end{Bmatrix} = \begin{bmatrix} x_1 & y_1 & x_1^2 & y_1^2 & x_1y_1 \\ x_2 & y_2 & x_2^2 & y_2^2 & x_2y_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_j & y_j & x_j^2 & y_j^2 & x_jy_j \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_N & y_N & x_N^2 & y_N^2 & x_Ny_N \end{bmatrix} \begin{Bmatrix} a \\ b \\ c \\ d \\ e \end{Bmatrix} - \begin{Bmatrix} u_1 - u_0 \\ u_2 - u_0 \\ \vdots \\ u_j - u_0 \\ \vdots \\ u_N - u_0 \end{Bmatrix}$$

In forma compatta si può scrivere come $\vec{e} = \mathbf{V}\vec{d} - \vec{u}$, in cui $\mathbf{V}$ è la matrice di Vandermonde e ha dimensioni $N \times 5$.

[1] Juan José Benito et al. "Application of the Generalized Finite Difference Method to improve the approximated solution of pdes". In: *Cmes-computer Modeling in Engineering & Sciences* 38 (2008), pp. 39–58.

Minimizzare il funzionale $J(u) = \sum w_j^2 e_j^2$ rispetto ai coefficienti a, b, c, d, e porta a scrivere un sistema di 5 equazioni in 5 incognite:

$$0 = \frac{\partial J}{\partial a} = 2\sum_{j=1}^{N} \left(ax_j + by_j + cx_j^2 + dy_j^2 + ex_jy_j - (u_j - u_0)\right) w_j^2 \cdot x_j$$

$$0 = \frac{\partial J}{\partial b} = 2\sum_{j=1}^{N} \left(ax_j + by_j + cx_j^2 + dy_j^2 + ex_jy_j - (u_j - u_0)\right) w_j^2 \cdot y_j$$

$$0 = \frac{\partial J}{\partial c} = 2\sum_{j=1}^{N} \left(ax_j + by_j + cx_j^2 + dy_j^2 + ex_jy_j - (u_j - u_0)\right) w_j^2 \cdot x_j^2$$

$$0 = \frac{\partial J}{\partial d} = 2\sum_{j=1}^{N} \left(ax_j + by_j + cx_j^2 + dy_j^2 + ex_jy_j - (u_j - u_0)\right) w_j^2 \cdot y_j^2$$

$$0 = \frac{\partial J}{\partial e} = 2\sum_{j=1}^{N} \left(ax_j + by_j + cx_j^2 + dy_j^2 + ex_jy_j - (u_j - u_0)\right) w_j^2 \cdot x_jy_j$$

In forma matriciale $\nabla J = 0 \rightarrow \mathbf{A}\vec{d} = \vec{b}$, dove:

$$\mathbf{A} = \begin{bmatrix} \sum w_j^2x_j^2 & \sum w_j^2x_jy_j & \sum w_j^2x_j^3 & \sum w_j^2x_jy_j^2 & \sum w_j^2x_j^2y_j \\ & \sum w_j^2y_j^2 & \sum w_j^2x_j^2y_j & \sum w_j^2y_j^3 & \sum w_j^2x_jy_j^2 \\ & & \sum w_j^2x_j^4 & \sum w_j^2x_j^2y_j^2 & \sum w_j^2x_j^3y_j \\ & & & \sum w_j^2y_j^4 & \sum w_j^2x_jy_j^3 \\ sym & & & & \sum w_j^2x_j^2y_j^2 \end{bmatrix} \qquad \vec{d} = \begin{Bmatrix} a \\ b \\ c \\ d \\ e \end{Bmatrix} \qquad \vec{b} = \begin{Bmatrix} \sum(u_j-u_0)w_j^2x_j \\ \sum(u_j-u_0)w_j^2y_j \\ \sum(u_j-u_0)w_j^2x_j^2 \\ \sum(u_j-u_0)w_j^2y_j^2 \\ \sum(u_j-u_0)w_j^2x_jy_j \end{Bmatrix}$$

L'espressione esplicita per i coefficienti a, b, c, d, e è possibile ma estremamente lunga; essa dipende dal numero nodi, dalla loro posizione, dalla funzione peso. Conviene quindi operare numericamente.

3.2 Metodo GFDM

L'idea alla base del Generalized Finite Difference Method (GFDM) è quella di risolvere l'equazione differenziale su ogni punto approssimandone le derivate usando le differenze finite generalizzate:

$$\vec{d} = \mathbf{A}^{-1}\vec{b} = \mathbf{A}^{-1}\mathbf{B}\vec{u} \qquad \rightarrow \qquad \mathbf{C} = \mathbf{A}^{-1}\mathbf{B}$$

Per cui è immediato ricavare la matrice dello stencil $\mathbf{C}$:

$$\begin{Bmatrix} a \\ b \\ c \\ d \\ e \end{Bmatrix} = \underbrace{\begin{bmatrix} \\ \mathbf{A}^{-1} \\ \\ \end{bmatrix} \begin{bmatrix} -\sum w_j^2x_j & w_1^2x_1 & \dots & w_j^2x_j & \dots & w_N^2x_N \\ -\sum w_j^2y_j & w_1^2y_1 & \dots & w_j^2y_j & \dots & w_N^2y_N \\ -\sum w_j^2x_j^2 & w_1^2x_1^2 & \dots & w_j^2x_j^2 & \dots & w_N^2x_N^2 \\ -\sum w_j^2y_j^2 & w_1^2y_1^2 & \dots & w_j^2y_j^2 & \dots & w_N^2y_N^2 \\ -\sum w_j^2x_jy_j & w_1^2x_1y_1 & \dots & w_j^2x_jy_j & \dots & w_N^2x_Ny_N \end{bmatrix}}_{\mathbf{C}} \begin{Bmatrix} u_0 \\ u_1 \\ \vdots \\ u_j \\ \vdots \\ u_N \end{Bmatrix}$$

Si evidenzia che la matrice $\mathbf{A}$ è simmetrica e ha dimensioni 5×5, mentre la matrice $\mathbf{B}$ è rettangolare e ha dimensioni $5 \times (N+1)$, perciò la matrice $\mathbf{C}$ ha anch'essa dimensioni $5 \times (N+1)$.

A questo punto è possibile approssimare le derivate nel nodo centrale:

$$\left.\frac{\partial u}{\partial x}\right|_0 = a = c_{1,1}u_0 + \sum_{j=1}^{N} c_{1,(j+1)}u_j$$

$$\left.\frac{\partial u}{\partial y}\right|_0 = b = c_{2,1}u_0 + \sum_{j=1}^{N} c_{2,(j+1)}u_j$$

$$\left.\frac{\partial^2 u}{\partial x^2}\right|_0 = 2c = 2c_{3,1}u_0 + 2\sum_{j=1}^{N} c_{3,(j+1)}u_j$$

$$\left.\frac{\partial^2 u}{\partial y^2}\right|_0 = 2d = 2c_{4,1}u_0 + 2\sum_{j=1}^{N} c_{4,(j+1)}u_j$$

$$\left.\frac{\partial^2 u}{\partial x \partial y}\right|_0 = e = c_{5,1}u_0 + \sum_{j=1}^{N} c_{5,(j+1)}u_j$$

Questo processo è ripetuto per ciascun nodo nel dominio. Sostituendo queste espressioni all'interno dell'equazione differenziale e delle condizioni al contorno, si ottiene un sistema di N equazioni algebriche (lineari oppure nonlineari a seconda della natura del problema).

Viene di seguito proposto un problema dimostrativo per illustrare il funzionamento del metodo: risolvere l'equazione di Laplace in un dominio 2D quadrato con condizioni al contorno di Dirichlet. Per semplicità si utilizza un approccio "fictious time integration":

$$\nabla^2 u = 0 \quad \rightarrow \quad \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0 \quad \rightarrow \quad \frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}$$

Le condizioni al contorno sono scelte in modo tale che la soluzione esatta sia la seguente:

$$u(x, y) = \frac{exp(8x)sin(8y)}{exp(8)}$$

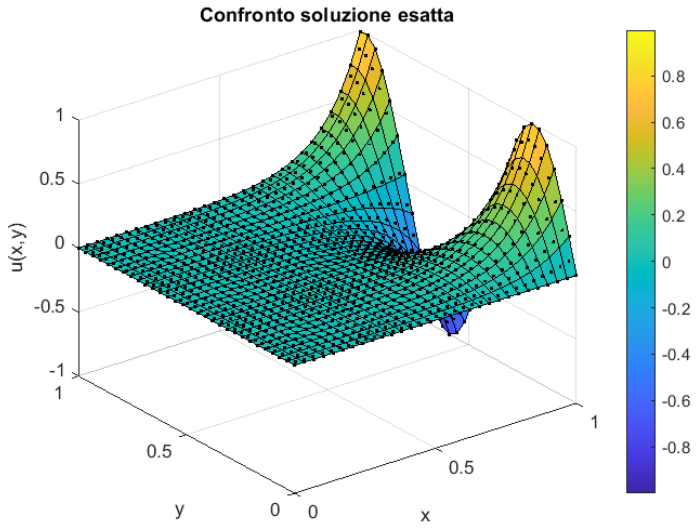


Figura 3.3: Soluzione $u(x, y)$ del problema dimostrativo

Applicando il metodo di Eulero Implicito:

$$\frac{u_0^{k+1} - u_0^k}{\Delta t} = \left(2c_{3,1}u_0^{k+1} + 2\sum_{j=1}^{N} c_{3,(j+1)}u_j^{k+1}\right) + \left(2c_{4,1}u_0^{k+1} + 2\sum_{j=1}^{N} c_{4,(j+1)}u_j^{k+1}\right)$$

Ad ogni step temporale quindi va risolto un sistema lineare le cui incognite sono i valori u associati a tutti i nodi interni. Dopo un breve transitorio temporale, il metodo converge alla soluzione stazionaria:

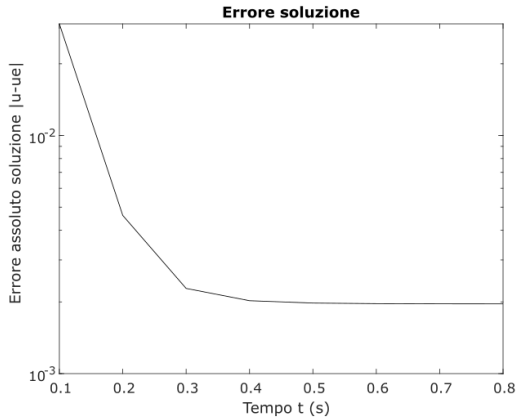


Figura 3.4: Andamento temporale errore

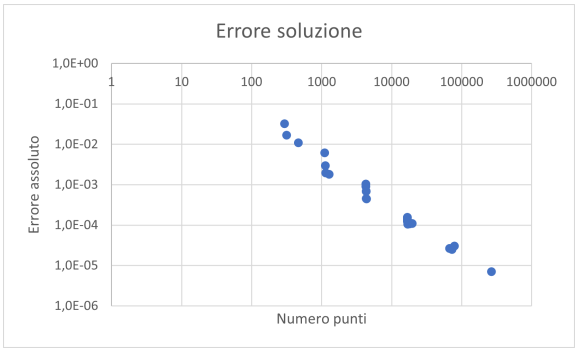


Figura 3.5: Convergenza del metodo GFDM all'infittire della griglia

Naturalmente raffinando la griglia la qualità della soluzione numerica migliora e si avvicina sempre più alla soluzione esatta.

3.3 M-matrici e proprietà

Nel metodo alle differenze finite tradizionale il sistema lineare da risolvere è simmetrico. Anche nel FEM e nel FVM vengono costruite matrici simmetriche, spesso aventi proprietà favorevoli come l'essere definite positive.
Questo non è più vero nel caso di metodi meshless. Dato che ciascun nodo tipicamente ha una smoothing length h diversa, è molto frequente avere un nodo j appartenente allo stencil del nodo i senza che quest'ultimo appartenga allo stencil del nodo j.

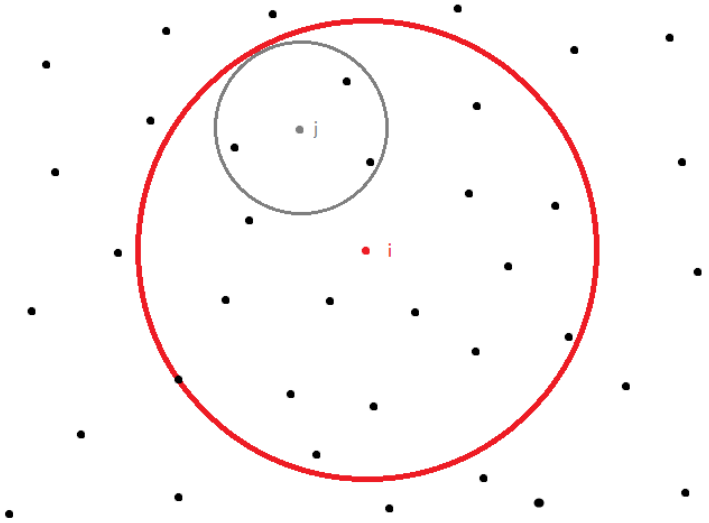


Figura 3.6: Esempio di mancanza di simmetria tra nodo i e nodo j

Nel problema dimostrativo analizzato in precedenza l'asimmetria della matrice del sistema è particolarmente evidente, anche perché i nodi al contorno non vengono risolti:

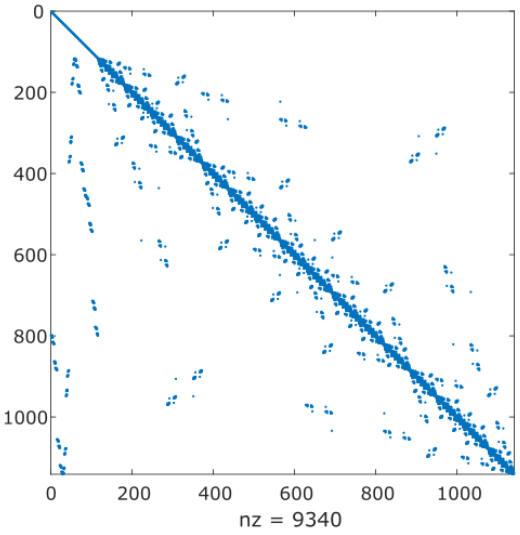


Figura 3.7: Struttura matrice sistema $\mathbf{M}$

Come fa notare Seibold[2] il funzionamento locale del metodo rende difficoltoso imporre una simmetria su tutte le coppie di nodi senza incorrere in pesanti controindicazioni sulla stabilità. Per questo è necessario lavorare con matrici non simmetriche ed è di interesse pratico classificarle, studiarle ed esaminarne le proprietà.

Definizione: Una matrice quadrata $\mathbf{M}$ è detta Z-matrice se gli elementi fuori dalla diagonale principale sono negativi o nulli $m_{ij} \leq 0$.
Definizione: Una matrice quadrata $\mathbf{M}$ è detta L-matrice se è una Z-matrice e gli elementi diagonali sono positivi $m_{ij} \leq 0,\ m_{ii} > 0$.
Definizione: Una matrice quadrata $\mathbf{M}$ è detta inversa positiva se tutti gli elementi (diagonali e non diagonali) della sua matrice inversa sono non negativi $\mathbf{M}^{-1} \geq 0$.
Definizione: Una matrice quadrata $\mathbf{M}$ è detta M-matrice se è una Z-matrice ed è inversa positiva.

$$m_{ij} \leq 0 \qquad \mathbf{M}^{-1} \geq 0$$

Ovviamente le matrici con cui si lavora sono molto grandi e anche per problemi semplici la matrice inversa $\mathbf{M}^{-1}$ non è calcolabile. Nella pratica perciò non sono disponibili dimostrazioni matematicamente rigorose, però può essere comunque utile esaminare alcune proprietà delle M-matrici perché frequenti in molte applicazioni e forniscono indicazioni preziose per la stabilità dei metodi.

Definizione: Una matrice quadrata $\mathbf{M}$ è detta strettamente diagonale dominante se ciascun elemento diagonale è maggiore della somma degli elementi non-diagonali sulla stessa riga.

$$|m_{ii}| > \sum_{j \neq i} |m_{ij}|$$

In modo del tutto analogo, $\mathbf{M}$ è detta debolmente diagonale dominante se $|m_{ii}| \geq \sum_{j \neq i} |m_{ij}|$. Facendo uso di tali definizioni, Seibold presenta un importante criterio per determinare se una matrice è una M-matrice, più facile da applicare rispetto alla definizione.

[2]Benjamin Seibold. "M-Matrices in meshless finite difference methods". In: (gen. 2006).

Teorema: Condizione sufficiente affinché una matrice quadrata $\mathbf{M}$ sia una M-matrice è che sia una L-matrice e sia strettamente diagonale dominante.

Ulteriori criteri sono disponibili; purtroppo però nella pratica le matrici spesso non sono nemmeno Z-matrici e tali teoremi non sono applicabili. Ovviamente trattandosi di condizioni sufficienti, ma non necessarie, non è possibile concludere nulla sulle proprietà del sistema.

Figura 3.8: Proprietà della matrice $\mathbf{M}$ di un semplice caso dimostrativo

Il motivo per il quale Seibold si sofferma sulle M-matrici sta nelle proprietà favorevoli che esse posseggono, sia da un punto di vista matematico sia per la convergenza dei solutori lineari.
In particolare, si può dimostrare che il metodo iterativo di Jacobi converge se la matrice $\mathbf{M}$ è strettamente diagonale dominante. Questo è vero a maggior ragione per le M-matrici, che costituiscono una classe più restrittiva.
Nota che questa proprietà è una condizione sufficiente e non necessaria, molto gradita anche perché se il metodo di Jacobi converge allora anche il metodo di Gauss-Seidel converge (e più velocemente grazie al teorema di Stein-Rosenberg).
Tali metodi da soli non sono particolarmente efficienti, ma possono dare luogo a buoni precondizionatori ed essere integrati all'interno di solutori più sofisticati come i metodi multigrid oppure i metodi di Krylov.

Schema iterativo	Precondizionatore $\mathbf{P}$
Richardson	$\mathbf{P} = \mathbf{I}$
Jacobi	$\mathbf{P} = \mathbf{D}$
Gauss-Seidel	$\mathbf{P} = \mathbf{D} + \mathbf{L}$
SOR	$\mathbf{P} = \frac{1}{\omega}\mathbf{D} + \mathbf{L}$

Si ricorda che $\mathbf{D}$, $\mathbf{L}$, $\mathbf{U}$ si ottengono scomponendo la matrice $\mathbf{M}$ rispettivamente nella sua diagonale, nella sua parte triangolare inferiore e nella sua parte triangolare superiore:

$$\mathbf{M} = \mathbf{D} + \mathbf{L} + \mathbf{U}$$

Ovviamente sono possibili diverse decomposizioni, ad esempio negli schemi ILU viene fatto uso della fattorizzazione $\mathbf{M} = \mathbf{LU}$ incompleta. Nel prosieguo di questo lavoro verranno fatti alcuni esperimenti numerici con i precondizionatori elencati sopra.

3.4 Proprietà numeriche

Come detto è difficile fare dimostrazioni matematiche rigorose su metodi le cui proprietà variano molto con la disposizione dei nodi e dal tipo di problema che si vuole affrontare. In generale però un metodo numerico per poter essere utilizzato in ambito ingegneristico deve soddisfare le seguenti proprietà:

- **convergenza**, la soluzione numerica discretizzata deve tendere alla soluzione esatta delle equazioni differenziali all'infittirsi della griglia;
- **consistenza**, l'errore di troncamento tra equazioni discretizzate ed equazioni differenziali esatte deve tendere a zero all'infittirsi della griglia;
- **stabilità**, il metodo non deve amplificare gli inevitabili errori di natura numerica;
- **conservazione**, il metodo deve conservare sia localmente sia globalmente le quantità relative al problema in oggetto (es: l'energia totale in un problema di dinamica elastica, oppure il bilancio dei flussi in un problema termico);

Ovviamente una soluzione numerica è "sempre sbagliata" e contiene al suo interno errori sistematici non eliminabili. È desiderabile che il metodo sia il più accurato possibile e che la soluzione non presenti picchi locali non fisici.

Per quanto riguarda convergenza e consistenza il metodo GFDM si presenta in modo analogo alle tradizionali differenze finite. Del resto i due metodi hanno in comune la formulazione mediante espansione in serie di Taylor, di conseguenza molte delle considerazioni sul FDM sono valide anche per il GFDM. Ordine del metodo ed errore di troncamento in particolare sono del tutto equivalenti.
Invece stabilità e conservazione non possono essere garantite in alcun modo, perlomeno nella formulazione base qui presentata. Ovviamente esistono varianti che affrontano e migliorano questi due aspetti, si cita ad esempio la variante Direct GFDM per la stabilità. Invece per la conservazione è possibile seguire il lavoro di Suchde e imporre un bilanciamento dei flussi approssimato costruendo una mesh locale. Date le complicazioni presenti in questo approccio e i risultati che non riflettono il costo computazionale significativamente superiore, in questo lavoro non verrà seguita questa strada.
Si possono fare considerazioni empiriche e congetture basate sulle proprietà delle M-matrici presentate in precedenza, ma nessun formalismo ad oggi è disponibile nel caso più generale.

3.5 Comparazione con altri metodi

Rispetto ai tradizionali metodi mesh-based quali il FEM e il FVM, il metodo GFDM presenta i seguenti vantaggi:

- ✓ Generazione pointcloud semplice, veloce e completamente automatizzabile anche su geometrie complesse;
- ✓ Parallelizzabile e scalabile quasi al 100%, può sfruttare al meglio le moderne architetture multicore e l'accelerazione grafica;
- ✓ Grandi deformazioni possibili senza remeshing dell'intero dominio;
- ✓ Raffinamento locale della griglia adattativo molto facile da implementare e poco costoso;
- ✓ Descrizione lagrangiana, molto vantaggiosa per un'ampia classe di problemi;

E i seguenti svantaggi:

- ✘ Più costoso da un punto di vista computazionale;
- ✘ Solo equazioni in forma differenziale;
- ✘ Problematiche legate alla stabilità numerica;
- ✘ Poca letteratura disponibile sull'argomento;

Rispetto ad altri metodi meshless basati sulle equazioni in forma differenziale (es: RBF, SPH), il metodo GFDM presenta i seguenti vantaggi:

- ✓ Condizioni al contorno imposte in modo molto semplice e chiaro, senza nodi virtuali/ghost;
- ✓ I nodi non trasportano massa, perciò possono essere inseriti o eliminati arbitrariamente;
- ✓ Formulazione compatibile sia con la descrizione euleriana sia con la descrizione lagrangiana;
- ✓ Immediata l'estensione ad ordini superiori;
- ✓ L'accuratezza non dipende in modo sensibile dalla scelta della funzione peso;

E i seguenti svantaggi:

- ✘ Problemi di malcondizionamento;
- ✘ Mancanza di conservazione;

Non vengono affrontate le differenze con i metodi basati sulle equazioni integrali.

3.6 Metodo GFDM classico

Si consideri ora un semplice problema termico in tre dimensioni: un'asta di sezione circolare soggetta ad un differenziale di temperatura costante ai suoi estremi:

$$\begin{cases} \frac{\partial u}{\partial t} = \frac{k}{\rho c_p}\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2}\right) + q_v & su\ \Omega \\ g_1 u + g_2 \frac{\partial u}{\partial n} = g_3 & su\ \partial\Omega \end{cases}$$

Dove $k, \rho, c_p = cost$ sono proprietà del materiale, mentre g_1, g_2, g_3 sono coefficienti utili per imporre le condizioni al contorno:

- $g_1 = 1, g_2 = 0, g_3 = T_{left}$ sulla faccia circolare sinistra;
- $g_1 = 1, g_2 = 0, g_3 = T_{right}$ sulla faccia circolare destra;
- $g_1 = 0, g_2 = 1, g_3 = 0$ sulla superficie esterna cilindrica;

Si ricorda che scomponendo le derivate direzionali vale $\frac{\partial u}{\partial n} = n_x \frac{\partial u}{\partial x} + n_y \frac{\partial u}{\partial y} + n_z \frac{\partial u}{\partial z}$, dove n_x, n_y, n_z sono le componenti del versore normale uscente.

Si utilizzi il Generalized Finite Difference Method con schema al II ordine per approssimare le derivate spaziali e il metodo di Eulero Implicito come integratore temporale. La nuvola di punti interna al dominio può essere generata con un semplice schema octree (si veda a tale proposito il paragrafo 4.5), mentre la selezione dei nodi vicini è affidata ad un semplice criterio di "circular neighborhood" con cell-classification (per maggiori informazioni si vada al paragrafo 4.6).
In tre dimensioni lo schema al II ordine prevede 9 coefficienti:

$$u_j = u_0 + ax_j + by_j + cz_j + dx_j^2 + ey_j^2 + fz_j^2 + gx_jy_j + hx_jz_j + gy_jz_j$$

Seguendo i passaggi visti in precedenza, si deve minimizzare il funzionale $J = \sum w_j^2 e_j^2$ rispetto a tali coefficienti per arrivare a scrivere un sistema di 9 equazioni in 9 incognite:

$$\mathbf{A} = \begin{bmatrix} \sum w_j^2 x_j^2 & \sum w_j^2 x_j y_j & \sum w_j^2 x_j z_j & \sum w_j^2 x_j^3 & \sum w_j^2 x_j y_j^2 & \sum w_j^2 x_j z_j^2 & \sum w_j^2 x_j^2 y_j & \sum w_j^2 x_j^2 z_j & \sum w_j^2 x_j y_j z_j \\ & \sum w_j^2 y_j^2 & \sum w_j^2 y_j z_j & \sum w_j^2 x_j^2 y_j & \sum w_j^2 y_j^3 & \sum w_j^2 y_j z_j^2 & \sum w_j^2 x_j y_j^2 & \sum w_j^2 x_j y_j z_j & \sum w_j^2 y_j^2 z_j \\ & & \sum w_j^2 z_j^2 & \sum w_j^2 x_j^2 z_j & \sum w_j^2 y_j^2 z_j & \sum w_j^2 z_j^3 & \sum w_j^2 x_j y_j z_j & \sum w_j^2 x_j z_j^2 & \sum w_j^2 y_j z_j^2 \\ & & & \sum w_j^2 x_j^4 & \sum w_j^2 x_j^2 y_j^2 & \sum w_j^2 x_j^2 z_j^2 & \sum w_j^2 x_j^3 y_j & \sum w_j^2 x_j^3 z_j & \sum w_j^2 x_j^2 y_j z_j \\ & & & & \sum w_j^2 y_j^4 & \sum w_j^2 y_j^2 z_j^2 & \sum w_j^2 x_j y_j^3 & \sum w_j^2 x_j y_j^2 z_j & \sum w_j^2 y_j^3 z_j \\ & & & & & \sum w_j^2 z_j^4 & \sum w_j^2 x_j y_j z_j^2 & \sum w_j^2 x_j z_j^3 & \sum w_j^2 y_j z_j^3 \\ & & & & & & \sum w_j^2 x_j^2 y_j^2 & \sum w_j^2 x_j^2 y_j z_j & \sum w_j^2 x_j y_j^2 z_j \\ & & & & & & & \sum w_j^2 x_j^2 z_j^2 & \sum w_j^2 x_j y_j z_j^2 \\ sym & & & & & & & & \sum w_j^2 y_j^2 z_j^2 \end{bmatrix}$$

$$\vec{d} = \begin{Bmatrix} a \\ b \\ c \\ d \\ e \\ f \\ g \\ h \\ i \end{Bmatrix} \qquad \vec{b} = \begin{Bmatrix} \sum(u_j - u_0) w_j^2 x_j \\ \sum(u_j - u_0) w_j^2 y_j \\ \sum(u_j - u_0) w_j^2 z_j \\ \sum(u_j - u_0) w_j^2 x_j^2 \\ \sum(u_j - u_0) w_j^2 y_j^2 \\ \sum(u_j - u_0) w_j^2 z_j^2 \\ \sum(u_j - u_0) w_j^2 x_j y_j \\ \sum(u_j - u_0) w_j^2 x_j z_j \\ \sum(u_j - u_0) w_j^2 y_j z_j \end{Bmatrix} \rightarrow \mathbf{B} = \begin{bmatrix} -\sum w_j^2 x_j & w_1^2 x_1 & \dots & w_j^2 x_j & \dots & w_N^2 x_N \\ -\sum w_j^2 y_j & w_1^2 y_1 & \dots & w_j^2 y_j & \dots & w_N^2 y_N \\ -\sum w_j^2 z_j & w_1^2 z_1 & \dots & w_j^2 z_j & \dots & w_N^2 z_N \\ -\sum w_j^2 x_j^2 & w_1^2 x_1^2 & \dots & w_j^2 x_j^2 & \dots & w_N^2 x_N^2 \\ -\sum w_j^2 y_j^2 & w_1^2 y_1^2 & \dots & w_j^2 y_j^2 & \dots & w_N^2 y_N^2 \\ -\sum w_j^2 z_j^2 & w_1^2 z_1^2 & \dots & w_j^2 z_j^2 & \dots & w_N^2 z_N^2 \\ -\sum w_j^2 x_j y_j & w_1^2 x_1 y_1 & \dots & w_j^2 x_j y_j & \dots & w_N^2 x_N y_N \\ -\sum w_j^2 x_j z_j & w_1^2 x_1 z_1 & \dots & w_j^2 x_j z_j & \dots & w_N^2 x_N z_N \\ -\sum w_j^2 y_j z_j & w_1^2 y_1 z_1 & \dots & w_j^2 y_j z_j & \dots & w_N^2 y_N z_N \end{bmatrix}$$

Da cui si ricava la matrice dello stencil:

$$\nabla J = 0 \;\rightarrow\; \mathbf{A}\vec{d} = \vec{b} \;\rightarrow\; \vec{d} = \mathbf{A}^{-1}\vec{b} = \mathbf{A}^{-1}\mathbf{B}\vec{u} \;\rightarrow\; \mathbf{C} = \mathbf{A}^{-1}\mathbf{B}$$

Nota che in 2D la matrice $\mathbf{C}$ aveva 5 righe, mentre in 3D le righe diventano 9. L'aumento del numero di colonne nella matrice di Vandermonde è motivo di preoccupazione perché il forte malcondizionamento limita l'estensione del metodo ad ordini superiori:

Ordine approssimazione	Numero colonne $\mathbf{A}$	Termini sviluppo Taylor
1	3	x, y, z
2	9	x, y, z, x^2, y^2, z^2, xy, xz, yz
3	18	x, y, z, x^2, y^2, z^2, xy, xz, yz, x^3, y^3, z^3, xyz, x^2y, x^2z, y^2x, y^2z, z^2x, z^2y
...	...	...

Seguendo i passaggi visti in precedenza si arriva a scrivere il seguente sistema sparso lineare:

$$u_0^{k+1}\left[\frac{1}{\Delta t} - 2\frac{k}{\rho c_p}\left(c_{4,1} + c_{5,1} + c_{6,1}\right)\right] - \sum_{j=1}^{N} u_j^{k+1}\left[2\frac{k}{\rho c_p}\left(c_{4,(j+1)} + c_{5,(j+1)} + c_{6,(j+1)}\right)\right] = q_v + \frac{1}{\Delta t}u_0^k$$

Per i nodi al contorno invece si ha:

$$u_0^{k+1}\left(g_1 + g_2 n_x c_{1,1} + g_2 n_y c_{2,1} + g_2 n_z c_{3,1}\right) + g_2 \sum_{j=1}^{N} u_j^{k+1}\left(n_x c_{1,(j+1)} + n_y c_{2,(j+1)} + n_z c_{3,(j+1)}\right) = g_3$$

Dove N è il numero dei nodi vicini al nodo 0 considerato. Queste equazioni vanno imposte su ciascun nodo del dominio e del contorno costruendo così il sistema riga per riga.

Per ricavare la matrice dello stencil $\mathbf{C}$ è necessario invertire la matrice di Vandermonde $\mathbf{A}$ per ciascun nodo. Si tratta di un processo molto oneroso da un punto di vista computazionale e che introduce problematiche di tipo numerico molto severe. Conviene invece usare la decomposizione di Cholesky:

$$\mathbf{A} = \mathbf{L}\mathbf{L}^\top \quad \rightarrow \quad \mathbf{A^{-1}} = [\mathbf{L}^\top]^{-1}[\mathbf{L}]^{-1} = [\mathbf{L}^{-1}]^\top[\mathbf{L}^{-1}]$$

Dove $\mathbf{L}$ è una matrice triangolare inferiore. Nota che la matrice $\mathbf{A}$ è il risultato di un problema ai minimi quadrati ed è simmetrica definita positiva, quindi è sempre possibile applicare la decomposizione di Cholesky.
Nota che per definizione $\mathbf{L} = \mathbf{VW}$; di questo aspetto verrà tratto vantaggio più avanti per semplificare l'implementazione e la derivazione del metodo.
In alternativa è possibile usare la decomposizione QR:

$$\mathbf{A} = \mathbf{QR} \quad \rightarrow \quad \mathbf{A^{-1}} = \mathbf{R}^{-1}\mathbf{Q}^\top$$

Dove $\mathbf{Q}$ è una matrice ortogonale (cioè $\mathbf{Q}^\top = \mathbf{Q}^{-1}$) e $\mathbf{R}$ è triangolare superiore.
Teoricamente la decomposizione QR è più lenta di quella di Cholesky ma per matrici così piccole la differenza è trascurabile. In compenso, sarà utile più avanti perché applicabile direttamente alla matrice di Vandermonde $\mathbf{V}$ (rettangolare e non simmetrica).

La soluzione di questo problema è banale e vede un andamento lineare della temperatura. Ovviamente il metodo GFDM con una formulazione al II ordine è in grado di trovare la soluzione esatta:

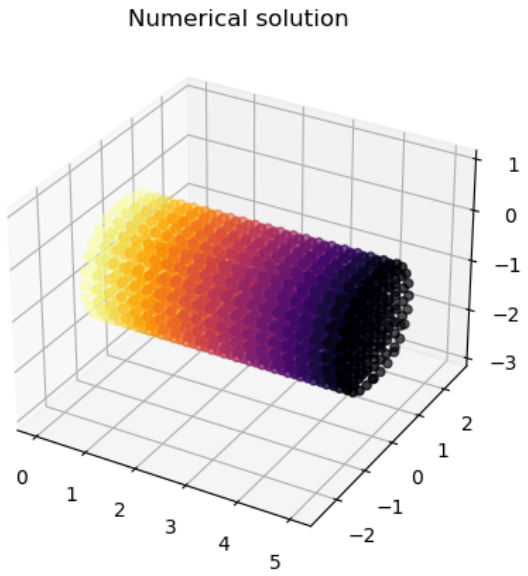


Figura 3.9: Soluzione problema termico asta 3D

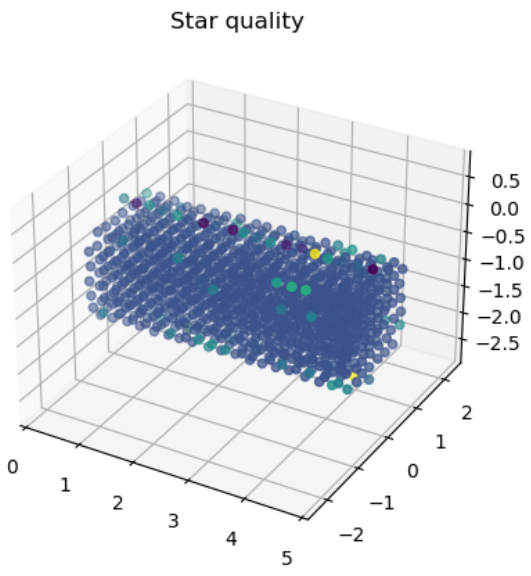


Figura 3.10: Star quality geometrico asta 3D

Come visto nel precedente caso 2D la formulazione implicita del metodo permette molta più libertà nella scelta del passo temporale ed evita l'insorgere di instabilità vicino ai bordi.
L'estensione in tre dimensioni però è accompagnata da molte difficoltà: numero condizionamento che esplode, assenza di dominanza diagonale, tempi di calcolo più lunghi e soprattutto da instabilità varie. In particolare, il metodo GFDM appena descritto risulta estremamente sensibile e arriva a convergenza solamente in pochi casi tarando i parametri meshSize e minNeighbors.
Tutto ciò fortunatamente è ben noto dalla letteratura. Si può osservare che la dominanza diagonale può diventare un problema in tutti quei casi in cui lo stencil è fortemente asimmetrico.
Come spiegato da Liszka,[3] infatti, è desiderabile che ciascun nodo sia connesso con tutti i vicini più prossimi, che i satelliti siano equamente distribuiti attorno al nodo centrale e che il "baricentro" dello stencil sia il più vicino possibile al nodo centrale.
Per quantificare tali aspetti Liszka definisce un parametro di "star quality" come segue:

$$\epsilon = \left| \frac{c_1 - \sum |c_{j+1}|}{c_1} \right|$$

[3]Tadeusz Liszka, Carlos Armando Duarte e W. W. Tworzydlo. "hp-Meshless cloud method". In: *Computer Methods in Applied Mechanics and Engineering* 139 (1996), pp. 263–288. DOI: 10.1016/S0045-7825(96)01086-9.

Dove c_1 è il coefficiente centrale dello stencil e c_{j+1} sono i coefficienti dei nodi satelliti. Per calcolarli è possibile fare riferimento all'operatore di Laplace $\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2}$:

$$c_1 = c_{4,1} + c_{5,1} + c_{6,1} \qquad c_{j+1} = c_{4,(j+1)} + c_{5,(j+1)} + c_{6,(j+1)}$$

Nota che uno stencil definito positivo ($c_1 < 0,\ c_{j+1} > 0$) è desiderabile ma non necessario alla convergenza. Tale parametro quindi fornisce solamente una indicazione e in presenza di instabilità si osserva essere sempre relativamente elevato.
Anche nel particolare problema termico 3D in oggetto si ha $\epsilon \gg 1$ nei nodi interni in prossimità dei bordi, come da aspettative.
Per risolvere la cosa si può agire sulla generazione dei nodi e/o sulla scelta dei vicini (vedere a tale proposito il paragrafo 4.6), oppure avvicinare il baricentro al nodo centrale estendendo il concetto di stencil:

- Nella variante Direct GFDM l'equazione di governo viene inglobata nello stencil (anche le condizioni al contorno per i relativi nodi);
- Nella variante MDLSM si cerca di ridurre l'ordine del metodo abbattendo così i problemi di condizionamento e la dimensione dello stencil. Inoltre, un approccio ai minimi quadrati risolve l'equazione di governo anche nel contorno;
- L'introduzione di gradi di libertà spettrali come incognite aggiuntive va a trattare e a risolvere specificatamente il problema dello stencil asimmetrico vicino ai contorni;

I prossimi paragrafi sono pertanto dedicati alla descrizione di questi approcci e alla loro formulazione matematica.

3.7 Variante Direct GFDM

La variante Direct GFDM, formulata da Suchde,[4] ha la peculiarità di inglobare nello stencil sia le equazioni di governo sia le condizioni al contorno. Per fare ciò si parte dall'approssimazione di una generica funzione $u(x, y)$ mediante differenze finite generalizzate:

$$u(x, y) = u_0 + ax + by + cx^2 + dy^2 + exy$$

Si consideri un semplice problema di conduzione termica 2D stazionario governato dalla nota equazione differenziale alle derivate parziali (PDE) ellittica:

$$\frac{k}{\rho c_p}\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) + q_v = 0$$

Sia per l'approssimazione polinomiale sia per l'equazione di governo è possibile definire un errore e come segue:

$$e_j = u_0 + ax_j + by_j + cx_j^2 + dy_j^2 + ex_jy_j - u_j$$

$$e_{PDE} = \frac{k}{\rho c_p}(2c + 2d) + q_v$$

Il primo è l'errore commesso dal polinomio approssimante rispetto alla funzione esatta $u(x, y)$, il secondo invece è l'errore di troncamento tra equazione discretizzata ed equazione differenziale.

[4] Pratik Suchde. "Conservation and Accuracy in Meshfree Generalized Finite Difference Methods". In: (gen. 2018).

Considerando un generico nodo con N nodi vicini e scrivendo l'errore e_j per ciascun satellite, in forma matriciale si arriva a:

$$\begin{Bmatrix} e_1 \\ e_2 \\ \vdots \\ e_j \\ \vdots \\ e_N \\ e_{PDE} \end{Bmatrix} = \begin{bmatrix} 1 & x_1 & y_1 & x_1^2 & y_1^2 & x_1y_1 \\ 1 & x_2 & y_2 & x_2^2 & y_2^2 & x_2y_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_j & y_j & x_j^2 & y_j^2 & x_jy_j \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_N & y_N & x_N^2 & y_N^2 & x_Ny_N \\ 0 & 0 & 0 & 2\frac{k}{\rho c_p} & 2\frac{k}{\rho c_p} & 0 \end{bmatrix} \begin{Bmatrix} u_0 \\ a \\ b \\ c \\ d \\ e \end{Bmatrix} - \begin{Bmatrix} u_1 \\ u_2 \\ \vdots \\ u_j \\ \vdots \\ u_N \\ q_v \end{Bmatrix}$$

In notazione compatta $\vec{e} = \mathbf{V}\vec{d} - \vec{u}$, dove $\mathbf{V}$ è una matrice di Vandermonde estesa.

I coefficienti incogniti $\vec{u}$ vengono calcolati attraverso una procedura di minimizzazione del seguente funzionale nel senso dei minimi quadrati:

$$minJ = \sum_{j=1}^{N} w_j^2 e_j^2 + \sum_{r=1}^{n_{PDE}} w_r^2 g_r^2$$

In notazione matriciale il problema si risolve con i seguenti passaggi formali:

$$minJ = \vec{e}^{\top}\mathbf{W}^2\vec{e} = \left(\mathbf{V}\vec{d} - \vec{u}\right)^{\top}\mathbf{W}^2\left(\mathbf{V}\vec{d} - \vec{u}\right) \quad \rightarrow \quad \vec{d} = \underbrace{\left[\left(\mathbf{V}\mathbf{W}^2\mathbf{V}\right)^{-1}\mathbf{V}^{\top}\mathbf{W}^2\right]}_{\mathbf{C}}\vec{u}$$

Dove $\mathbf{W}$ è una matrice diagonale dei pesi:

$$\mathbf{W} = \begin{bmatrix} w_1 & & & & & \\ & w_2 & & & & \\ & & \ddots & & & \\ & & & w_j & & \\ & & & & \ddots & \\ & & & & & w_N \\ & & & & & & w_{PDE} \end{bmatrix}$$

Questo calcolo ovviamente va eseguito per ogni punto del dominio. Gli stencil $\mathbf{C}$ così calcolati dipendono dalle proprietà geometriche dei nodi vicini, dalla funzione peso $w(x, y)$ scelta e dalle equazioni di governo discretizzate.

3.7.1 Condizioni al contorno

Per i nodi appartenenti al contorno, nel metodo GFDM è immediato imporre le relative condizioni aggiungendo una riga alla matrice di Vandermonde:

$$\begin{Bmatrix} e_1 \\ \vdots \\ e_N \\ e_{PDE} \\ e_{BC} \end{Bmatrix} = \begin{bmatrix} 1 & x_1 & y_1 & x_1^2 & y_1^2 & x_1y_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_N & y_N & x_N^2 & y_N^2 & x_Ny_N \\ 0 & 0 & 0 & 2\frac{k}{\rho c_p} & 2\frac{k}{\rho c_p} & 0 \\ g_1 & g_2n_x & g_2n_y & 0 & 0 & 0 \end{bmatrix} \begin{Bmatrix} u_0 \\ a \\ b \\ c \\ d \\ e \end{Bmatrix} - \begin{Bmatrix} u_1 \\ \vdots \\ u_N \\ q_v \\ g_3 \end{Bmatrix}$$

Così facendo la procedura di minimizzazione rimane inalterata ed è possibile imporre ulteriori equazioni ed espandere il metodo in modo estremamente semplice e senza alcuna difficoltà.
Chiaramente le condizioni al contorno sono rispettate solamente nel senso dei minimi quadrati, non in modo stretto. Si vedrà però che questo non costituirà un reale problema, anzi Suchde addirittura mostra come può aiutare ad alleviare alcune difficoltà presenti nella simulazione di fluidi incompressibili.

3.7.2 Sistema sparso finale

A questo punto è nota la matrice degli stencil $\mathbf{C} = \left[\left(\mathbf{V}\mathbf{W}^2\mathbf{V} \right)^{-1} \mathbf{V}^\top \mathbf{W}^2 \right]$ e si può passare alla scrittura del sistema sparso finale. Per ciascun nodo del dominio si hanno 6 equazioni:

$$\begin{Bmatrix} u_0 \\ a \\ b \\ c \\ d \\ e \end{Bmatrix} = \begin{bmatrix} c_{1,1} & c_{1,2} & \dots & c_{1,j} & \dots & c_{1,N} & c_{1,PDE} & c_{1,BC} \\ c_{2,1} & c_{2,2} & \dots & c_{2,j} & \dots & c_{2,N} & c_{2,PDE} & c_{2,BC} \\ c_{3,1} & c_{3,2} & \dots & c_{3,j} & \dots & c_{3,N} & c_{3,PDE} & c_{3,BC} \\ c_{4,1} & c_{4,2} & \dots & c_{4,j} & \dots & c_{4,N} & c_{4,PDE} & c_{4,BC} \\ c_{5,1} & c_{5,2} & \dots & c_{5,j} & \dots & c_{5,N} & c_{5,PDE} & c_{5,BC} \\ c_{6,1} & c_{6,2} & \dots & c_{6,j} & \dots & c_{6,N} & c_{6,PDE} & c_{6,BC} \end{bmatrix} \begin{Bmatrix} u_1 \\ u_2 \\ \vdots \\ u_j \\ \vdots \\ u_N \\ q_v \\ g3 \end{Bmatrix}$$

La prima riga fornisce l'equazione da mettere a sistema per ciascun nodo, sia esso interno oppure appartenente al contorno:

$$u_0 = c_{1,1}u_1 + c_{1,2}u_2 + \ldots + c_{1,j}u_j + \ldots + c_{1,N}u_N + c_{1,PDE}q_v + c_{1,BC}g_3$$

Il sistema sparso lineare da risolvere quindi assume la forma:

$$1\ u_i - \sum_{j=1}^{N} c_{1,j}u_j = c_{1,PDE}q_v + c_{1,BC}g_3$$

Come spiegato da Suchde, questa variante prende il nome di "Direct GFDM" perché nel sistema sparso finale compaiono soltanto gli stencil della funzione $u(x, y)$, senza fare uso degli stencil delle derivate.
Questi infatti possono essere usati per quantificare le varie derivate (ad esempio i flussi di calore $q_x = -k\frac{\partial u}{\partial x}$, $q_y = -k\frac{\partial u}{\partial y}$) in fase di post-processing, semplicemente esplicitando le corrispondenti righe della matrice $\mathbf{C}$:

$$\left.\frac{\partial u}{\partial x}\right|_0 = a = \sum_{j=1}^{N} c_{2,j}u_j + c_{2,PDE}q_v + c_{2,BC}g_3$$

$$\left.\frac{\partial u}{\partial y}\right|_0 = b = \sum_{j=1}^{N} c_{3,j}u_j + c_{3,PDE}q_v + c_{3,BC}g_3$$

$$\left.\frac{\partial^2 u}{\partial x^2}\right|_0 = 2c = 2\sum_{j=1}^{N} c_{4,j}u_j + 2c_{4,PDE}q_v + 2c_{4,BC}g_3$$

$$\left.\frac{\partial^2 u}{\partial y^2}\right|_0 = 2d = 2\sum_{j=1}^{N} c_{5,j}u_j + 2c_{5,PDE}q_v + 2c_{5,BC}g_3$$

$$\left.\frac{\partial^2 u}{\partial x \partial y}\right|_0 = e = \sum_{j=1}^{N} c_{6,j}u_j + c_{6,PDE}q_v + c_{6,BC}g_3$$

Ovvero rispetto al metodo classico sono presenti due termini ulteriori. Del resto l'idea alla base del metodo Direct GFDM è estendere l'espansione in serie di Taylor aggiungendo ai termini monomiali anche i membri destri (RHS) delle equazioni di governo.

Come fa notare Suchde un importante vantaggio del metodo Direct è quello di risolvere l'equazione di governo assieme alle condizioni al contorno nei nodi di frontiera. Questo conferisce

stabilità numerica al metodo senza ricorrere a termini artificiali e migliora l'accuratezza della soluzione. Come verrà esaminato al paragrafo 4.4 la migliore stabilità si traduce in un condizionamento dello stencil più basso e uniforme nel dominio.
Allo stesso modo il metodo può risolvere sistemi sovra-determinati con più equazioni che incognite, anche se in questo lavoro non ne verrà fatto uso.

3.8 Variante MDLSM

Abbiamo visto per sia il metodo GFDM classico sia la variante Direct GFDM impiegano uno stencil al II ordine per approssimare le incognite u_i su ciascun nodo del dominio. Una volta risolto il sistema sparso finale è facile calcolare le derivate con una procedura di post-processing. Nel Mixed Discrete Least Squares Method (MDLSM) invece le incognite e le derivate vengono determinate simultaneamente: le derivate del I ordine $\frac{\partial u}{\partial x}$, $\frac{\partial u}{\partial y}$ sono fatte entrare nel sistema sparso finale e sono risolte contemporaneamente alle incognite u.
Per fare ciò l'equazione di governo del II ordine viene scomposta in un sistema di equazioni del I ordine (formulazione mista):

$$\frac{k}{\rho c_p}\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}\right)+q_v=0 \;\rightarrow\; \begin{cases} \frac{\partial q_x}{\partial x}+\frac{\partial q_y}{\partial y}=\rho c_p q_v \\ q_x=-k\frac{\partial u}{\partial x} \\ q_y=-k\frac{\partial u}{\partial y} \end{cases}$$

Invece di risolvere una sola equazione di ordine 2 per una sola incognita (u) si possono risolvere tre equazioni accoppiate di ordine 1 per tre incognite (u, q_x, q_y). Abbassare l'ordine del metodo implica notevoli vantaggi dal punto di vista numerico:

- Incognite u_i e derivate q_x, q_y determinate simultaneamente, non è richiesto alcun post-processing;
- Le derivate di un ordine più basso possono essere approssimate con un polinomio di grado minore e con un numero di nodi vicini estremamente ridotto (anche meno di 10!);
- Le condizioni al contorno sui flussi di calore sono imposte come condizioni di Dirichlet. Per problemi termici non sono più presenti condizioni di Neumann o di Robin;

Le fasi di ricerca nodi vicini e di calcolo dello stencil quindi beneficiano enormemente di questo sviluppo. Anche l'assenza della fase di post-processing va ad abbassare l'onere computazionale, tuttavia il sistema sparso finale avrà il triplo delle incognite e la sua risoluzione brucia completamente i risparmi ottenuti nelle altre fasi. A parziale compensazione, la matrice $\mathbf{M}$ si vedrà essere simmetrica e definita positiva, quindi ha proprietà migliori per la convergenza e ammette l'uso di solutori lineari più efficienti.

Come mostrato da Faraji et al[5] il metodo lavora bene in vari casi 1D e 2D. Tuttavia, tenendo conto delle problematiche incontrate con il metodo classico, molti interrogativi nascono sulla stabilità e sull'estensione a casi 3D pratici.
Infatti nella sua formulazione base il metodo MDLSM manca di tutti quegli aspetti stabilizzanti visti nella variante Direct GFDM. A tale proposito il lavoro di Amani[6] cerca di introdurne qualcuno, come l'imposizione delle condizioni al contorno nel senso dei minimi quadrati e la scrittura delle equazioni di governo nei nodi di bordo.

[5] Saeb Faraji, M. H. Afshar e J. Amani. "Mixed Discrete Least Squares Meshless method for solution of quadratic partial differential equations". In: *Scientia Iranica* 21 (2014), pp. 492–504. DOI: 10.24200/sci.2017.4203.

[6] J. Amani, A. Saboor Bagherzadeh e Timon Rabczuk. "Error Estimate and Adaptive Refinement in Mixed Discrete Least Squares Meshless Method". In: *Mathematical Problems in Engineering* 2014 (2014), p. 721240. DOI: 10.1155/2014/721240.

Per popolare il sistema sparso finale si procede in modo analogo a quanto fatto con la variante Direct. In essa si scrivevano gli errori di approssimazione polinomiale sui satelliti di ciascun nodo e il metodo dei minimi quadrati veniva applicato per minimizzare tali errori locali. Si risolveva cioè un problema dei minimi quadrati per ciascun nodo del dominio.
Nel metodo MDLSM invece si scrivono solamente gli errori su equazioni di governo e su condizioni al contorno per ciascun nodo centrale. Vengono a mancare gli errori di approssimazione polinomiale sui satelliti:

$$e_{PDE1} = aq_x + bq_y - \rho c_p q_v$$

$$e_{PDE2} = q_x + kau$$

$$e_{PDE3} = q_y + kbu$$

$$e_{BC} = g_1 u + g_2 n_x q_x + g_2 n_y q_y - g_3$$

Per ciascun nodo quindi si hanno tre oppure quattro errori. A questo punto si usa il metodo dei minimi quadrati per minimizzare tali errori globalmente su tutto il dominio. In forma matriciale si scrive un "sistemone" del tipo:

$$\begin{Bmatrix} \vdots \\ e_{PDE1,i} \\ \vdots \\ e_{PDE2,i} \\ \vdots \\ e_{PDE3,i} \\ \vdots \\ e_{BC,i} \\ \vdots \end{Bmatrix} = \begin{bmatrix} & \vdots & & & \vdots & & & \vdots & \\ 0 \ \dots \ 0 \ \dots \ 0 & 2c_{2,1} & \dots & 2c_{2,j} & \dots & 2c_{2,N} & 2c_{3,1} & \dots & 2c_{3,j} \ \dots \ 2c_{3,N} \\ & \vdots & & & \vdots & & & \vdots & \\ kc_{2,1} \ \dots \ kc_{2,j} \ \dots \ kc_{2,N} & 0 & \dots & 1 & \dots & 0 & 0 & \dots & 0 \ \dots \ 0 \\ & \vdots & & & \vdots & & & \vdots & \\ kc_{3,1} \ \dots \ kc_{3,j} \ \dots \ kc_{3,N} & 0 & \dots & 0 & \dots & 0 & 0 & \dots & 1 \ \dots \ 0 \\ & \vdots & & & \vdots & & & \vdots & \\ 0 \ \dots \ g_1 \ \dots \ 0 & 0 & \dots & g_2 n_x & \dots & 0 & 0 & \dots & g_2 n_y \ \dots \ 0 \\ & \vdots & & & \vdots & & & \vdots & \end{bmatrix} \begin{Bmatrix} u_1 \\ \vdots \\ u_j \\ \vdots \\ u_N \\ q_{x,1} \\ \vdots \\ q_{x,j} \\ \vdots \\ q_{x,N} \\ q_{y,1} \\ \vdots \\ q_{y,j} \\ \vdots \\ q_{y,N} \end{Bmatrix} - \begin{Bmatrix} \vdots \\ \rho c_p q_v \\ \vdots \\ 0 \\ \vdots \\ 0 \\ \vdots \\ g_3 \\ \vdots \end{Bmatrix}$$

In notazione compatta $\vec{e} = \mathbf{V}\vec{u} - \vec{g}$, in cui $\vec{e}$ è il vettore colonna $4N \times 1$ degli errori, $\mathbf{V}$ è una matrice sparsa rettangolare di dimensioni $4N \times 3N$, $\vec{u}$ è il vettore colonna $3N \times 1$ delle incognite, $\vec{g}$ è il vettore colonna $4N \times 1$ dei termini noti.

Procedendo con il metodo dei minimi quadrati pesato (Weighted Least Squares, WLS), si definisce il funzionale $J = \sum_{i=1}^{N} w_{PDE}^2 e_{PDE,i}^2 + w_{BC}^2 e_{BC,i}^2$ da minimizzare come segue:

$$minJ = \vec{e}^{\top}\mathbf{W}^2\vec{e} = (\mathbf{V}\vec{u} - \vec{g})^{\top}\mathbf{W}^2(\mathbf{V}\vec{u} - \vec{g}) \qquad \rightarrow \qquad \underbrace{\mathbf{V}^{\top}\mathbf{W}^2\mathbf{V}}_{\mathbf{M}}\vec{u} = \underbrace{\mathbf{V}^{\top}\mathbf{W}^2\vec{g}}_{\vec{b}}$$

Che costituisce il sistema sparso finale $\mathbf{M}\vec{u} = \vec{b}$ da risolvere con un qualche solutore lineare. La soluzione $\vec{u}$ come detto contiene sia le incognite u_j sia le derivate $q_{x,j}$, $q_{y,j}$.
Ci si rende immediatamente conto che il sistema è quadrato, sparso, simmetrico e definito positivo. Come fatto notare da Amani il numero di incognite è triplicato rispetto al metodo GFDM classico aumentando l'onere computazionale di un fattore 3^{β}, dove β dipende dal solutore lineare scelto, ma le proprietà favorevoli della matrice $\mathbf{M}$ fanno sì che il sistema sia risolvibile mediante metodi più efficienti.

La simmetria in particolare è una proprietà che manca sia nel metodo classico sia nel metodo Direct, ma che è invece presente nel MDLSM. Tra le altre cose, essa permette di applicare l'algoritmo di ordinamento di Cuthill-McKee (inverso) per permutare la matrice $\mathbf{M}$ in modo tale da raccogliere il pattern di sparsità il più possibile vicino alla diagonale principale:

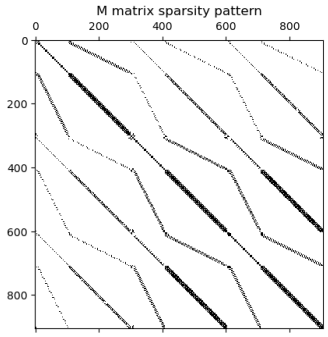


Figura 3.11: Pattern matrice $\mathbf{M}$ per un semplice caso 2D con nodi strutturati

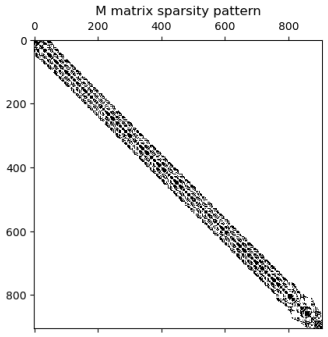


Figura 3.12: Pattern matrice $\mathbf{M}$ con ordinamento di Cuthill-McKee

In genere la nuvola di punti non ha una struttura ordinata. Specialmente in tre dimensioni l'ordine dei nodi è difficile che sia strutturato, se non in problemi molto semplici, e la matrice risultante mostra gli elementi non nulli distribuiti in modo abbastanza uniforme.

Ricapitolando, il metodo MDLSM ha i seguenti vantaggi:

- Le equazioni di governo non contengono più derivate seconde, permettendo così di impiegare uno schema di discretizzazione spaziale accurato solamente al I ordine. La fase di calcolo degli stencil quindi è enormemente più veloce;
- Lo stencil ridotto ha implicazioni benefiche anche sull'assemblaggio e sulla risoluzione del sistema finale, compensando in parte le dimensioni maggiori;
- Rispetto al metodo Direct GFDM il condizionamento dello stencil si mantiene estremamente basso ed è indipendente dalle equazioni di governo;
- Le derivate sono ottenute direttamente assieme alle incognite, senza dover svolgere alcuna attività di post-processing;
- Le condizioni al contorno possono essere formulate esclusivamente in termini di condizioni di Dirichlet, più semplici da implementare;

Purtroppo però l'implementazione in Julia fornisce risultati mediocri. Il metodo è generalmente instabile e la soluzione numerica risulta molte volte inaccurata, specialmente nei flussi q_x, q_y che mostrano schemi a scacchiera:

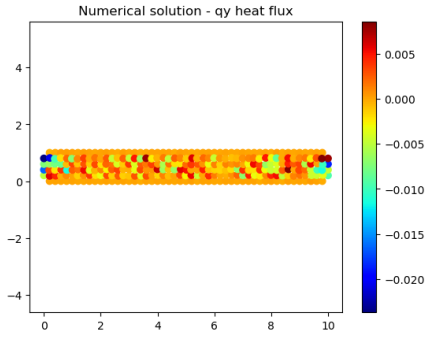


Figura 3.13: Illustrazione schema a scacchiera per il flusso q_x

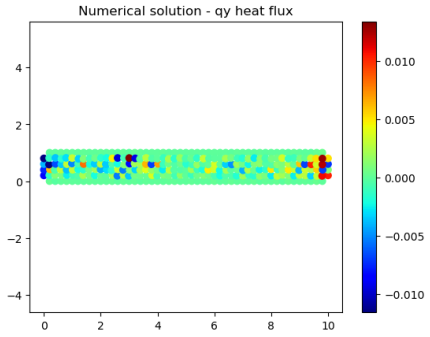


Figura 3.14: Illustrazione schema a scacchiera per il flusso q_y

Emerge pertanto la necessità di introdurre una qualche forma di stabilizzazione. Oñate[7] ad esempio propone una tecnica FIC (Finite Increment Calculus), che verrà presentata ed analizzata più in dettaglio al paragrafo 3.12.
Un'altra idea è quella di inglobare nel metodo MDLSM gli sviluppi visti nella variante Direct GFDM. Tale strada è certamente semplice ed agevole, ma va a compromettere molti dei vantaggi del MDLSM appena presentato. L'idea alla base del "Direct MDLSM" infatti è quella di aumentare lo stencil includendo le equazioni di governo (in formulazione mista) e le condizioni al contorno:

$$\begin{Bmatrix} \vdots \\ e_{uj} \\ \vdots \\ e_{qxj} \\ \vdots \\ e_{qyj} \\ \vdots \\ e_{PDE1} \\ e_{PDE2} \\ e_{PDE3} \\ e_{BC} \end{Bmatrix} = \begin{bmatrix} \vdots & & & \vdots & & & \vdots & & \\ 1 & x_j & y_j & 0 & 0 & 0 & 0 & 0 & 0 \\ \vdots & & & \vdots & & & \vdots & & \\ 0 & 0 & 0 & 1 & x_j & y_j & 0 & 0 & 0 \\ \vdots & & & \vdots & & & \vdots & & \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & x_j & y_j \\ \vdots & & & \vdots & & & \vdots & & \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & k & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & k & 0 & 0 & 0 & 1 & 0 & 0 \\ g_1 & 0 & 0 & g_2 n_x & 0 & 0 & g_2 n_y & 0 & 0 \end{bmatrix} \begin{Bmatrix} u_0 \\ a_u \\ b_u \\ q_{x0} \\ a_{qx} \\ b_{qx} \\ q_{y0} \\ a_{qy} \\ b_{qy} \end{Bmatrix} - \begin{Bmatrix} \vdots \\ u_j \\ \vdots \\ q_{xj} \\ \vdots \\ q_{yj} \\ \vdots \\ \rho c_p q_v \\ 0 \\ 0 \\ g_3 \end{Bmatrix}$$

Il sistema sparso finale si può scrivere seguendo i passasggi dello schema MDLSM in modo inalterato oppure proseguire con lo schema Direct GFDM.
La prima strada a prima vista sembra essere la più vantaggiosa visto che conduce ad un sistema simmetrico definito positivo. Tuttavia, come si vedrà meglio in seguito, l'approccio Direct conduce a sistemi con un condizionamento molto migliore ed è senza dubbio preferibile.
In questo modo viene dato luogo ad un metodo Direct GFDM che differisce da quello visto in precedenza solamente per le equazioni di governo in formulazione mista, e di conseguenza l'ordine di approssimazione ridotto al I.

Test in Julia mostrano come questa variante sia effettivamente stabile e correttamente funzionante, tuttavia l'onere di calcolo è sensibilmente maggiore e l'esecuzione risulta un ordine di grandezza più lenta se confrontata con il Direct GFDM descritto in precedenza. Del resto guardando alla matrice di Vandermonde il numero di colonne è sempre maggiore nel caso del nuovo "Direct MDLSM":

Problema	Numero colonne **V** Direct GFDM	Numero colonne **V** Direct MDLSM
Calore 2D	6	9
Calore 3D	10	12
Elasticità 2D	12	20
Elasticità 3D	30	36

Un altro svantaggio risiede proprio nell'ordine di approssimazione limitato al primo: per ottenere risultati accurati è necessario usare molti punti.
Il metodo Direct GFDM quindi viene scelto per il prosieguo di questo lavoro.

[7] E. Oñate, F. Perazzo e J. Miquel. "A finite point method for elasticity problems". In: *Computers & Structures* 79.22 (2001), pp. 2151–2163. ISSN: 0045-7949. DOI: https://doi.org/10.1016/S0045-7949(01)00067-0.

3.9 Gradi di libertà spettrali

Come detto in precedenza, il problema più grande per la stabilità del metodo GFDM classico sono gli stencil asimmetrici dei nodi interni prossimi ai bordi. In questi casi si incontrano spesso problemi di malcondizionamento e difficoltà nello scegliere i nodi vicini. La stessa cosa si osserva accadere in nodi interni a domini snelli, pur in modo più leggero, limitandosi ad una perdita di accuratezza nel calcolo delle derivate.
Per risolvere questi problemi Liszka[8] propone di inglobare nel vettore delle incognite $\vec{u}$ anche le derivate in direzione normale $\frac{\partial u}{\partial n}$ di ciascun nodo al contorno. Tali derivate prendono il nome di gradi di libertà spettrali (spectral DOFs).

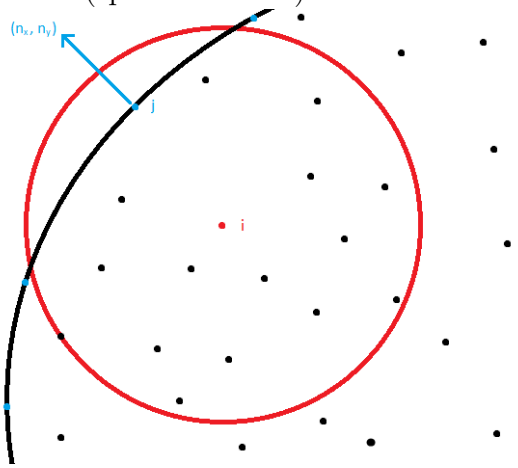


Figura 3.15: Esempio di stella con un grado di libertà spettrale

Vanno ovviamente aggiunte altrettante equazioni per mantenere il problema ben posto. Si consideri un generico nodo i con un satellite j appartenente al contorno del dominio e si scrivi lo sviluppo in serie di Taylor della derivata direzionale:

$$u_j = u_0 + ax_j + by_j + cx_j^2 + dy_j^2 + ex_jy_j \quad \rightarrow \quad \left.\frac{\partial u}{\partial x}\right|_j = a + 2cx_j + ey_j \ , \ \left.\frac{\partial u}{\partial y}\right|_j = b + 2dy_j + ex_j$$

$$\rightarrow \quad \left.\frac{\partial u}{\partial n}\right|_j = \left.\frac{\partial u}{\partial x}\right|_j n_x + \left.\frac{\partial u}{\partial y}\right|_j n_y = an_x + bn_y + 2cx_jn_x + 2dy_jn_y + e\left(n_xy_j + n_yx_j\right)$$

Questa equazione viene inglobata nello stencil del metodo classico semplicemente aggiungendo un numero di righe pari al numero di nodi vicini appartenenti al contorno ND:

$$\begin{bmatrix} 1 & x_1 & y_1 & x_1^2 & y_1^2 & x_1y_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_N & y_N & x_N^2 & y_N^2 & x_Ny_N \\ 0 & n_x & n_y & 2x_1n_x & 2y_1n_y & n_xy_1+n_yx_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & n_x & n_y & 2x_{ND}n_x & 2y_{ND}n_y & n_xy_{ND}+n_yx_{ND} \end{bmatrix} \begin{Bmatrix} u_0 \\ a \\ b \\ c \\ d \\ e \end{Bmatrix} = \begin{Bmatrix} u_1 \\ \vdots \\ u_N \\ \frac{\partial u}{\partial n}|_1 \\ \vdots \\ \frac{\partial u}{\partial n}|_{ND} \end{Bmatrix}$$

Nota che lo stencil continua a rimanere completamente disaccoppiato dalle equazioni di governo anche dopo l'introduzione dei gradi di libertà spettrali.

$$\mathbf{V}\vec{d} = \vec{u} \quad \rightarrow \quad \vec{d} = \mathbf{C}\vec{u}$$

[8] Liszka, Duarte e Tworzydlo, "hp-Meshless cloud method", cit.

Proseguendo con la costruzione del sistema sparso finale:

$$\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}=0 \;\rightarrow\; 2\left(\sum_{j=1}^{N} c_{4,j}u_j+\sum_{j=1}^{ND} c_{4,(N+j)}\left.\frac{\partial u}{\partial n}\right|_j\right)+2\left(\sum_{j=1}^{N} c_{5,j}u_j+\sum_{j=1}^{ND} c_{5,(N+j)}\left.\frac{\partial u}{\partial n}\right|_j\right)=0$$

$$g_1 u+g_2 n_x\left(\sum_{j=1}^{N} c_{2,j}u_j+\sum_{j=1}^{ND} c_{2,(N+j)}\left.\frac{\partial u}{\partial n}\right|_j\right)+g_2 n_y\left(\sum_{j=1}^{N} c_{3,j}u_j+\sum_{j=1}^{ND} c_{3,(N+j)}\left.\frac{\partial u}{\partial n}\right|_j\right)=g_3$$

Da cui si ricava un sistema sparso quadrato non simmetrico del tipo $\mathbf{M}\vec{u}=\vec{g}$.

Il metodo costruito in questo paragrafo può essere chiamato "Spectral GFDM" e consiste nell'aggiungere i gradi di libertà spettrali al metodo GFDM classico. Per un confronto tra tutte le varianti sviluppate finora si rimanda al paragrafo 4.4.
Ovviamente il concetto di GDL spettrali può essere applicato anche al metodo Direct GFDM e ai MDLSM, però per come sono costituiti si tratta di una complicazione non necessaria. La presenza di altre tecniche di stabilizzazione infatti renderebbe i spectral DOF superflui.

3.10 Discretizzazione problema termico

Si consideri il metodo Direct GFDM derivato al paragrafo 3.7 per un semplice caso didattico di conduzione termica 2D stazionaria. Adesso si vuole ripetere la derivazione nel caso più generale di conduzione termica 3D transitoria discretizzando le equazioni di bilancio termico ricavate al capitolo precedente:

$$\frac{\partial u}{\partial t}=\frac{k}{\rho c_p}\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}+\frac{\partial^2 u}{\partial z^2}\right)+q_v \qquad su\ \Omega$$

Con condizioni al contorno:

$$g_1 u+g_2\frac{\partial u}{\partial n}=g_3 \qquad su\ \partial\Omega$$

Per la continuazione del presente paragrafo si continui ad assumere $k,\rho,c_p=cost$ su tutto il dominio. Nota che è comunque possibile considerare materiali differenti su porzioni di dominio semplicemente separando le relative regioni e imponendo sul confine le opportune condizioni al contorno. È inoltre possibile tenere conto di nonlinearità del materiale, ad esempio $c_p=f(T)$, eseguendo la simulazione a piccoli step temporali e ricostruendo lo stencil ad ogni iterazione. In generale si tratta di un'attività molto onerosa, ma in molti casi pratici è possibile far uscire le nonlinearità dalla matrice $\mathbf{M}$ e portarle al termine noto:

$$c_p=f(T) \qquad \rightarrow \qquad \frac{k}{\rho}\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}+\frac{\partial^2 u}{\partial z^2}\right)=-c_p(T)\ q_v+c_p(T)\ \frac{\partial u}{\partial t}$$

In questo modo è possibile riciclare gli stencil del problema lineare durante la propagazione temporale e rendere l'onere computazionale sopportabile. Se si riesce anche a mantenere inalterata la matrice $\mathbf{M}$ si può ricorrere ad una qualche fattorizzazione (es: QR) da riutilizzare step dopo step e velocizzare sensibilmente la simulazione.
Purtroppo però questo sviluppo non è sempre fattibile, ad esempio in presenza di $k=f(T)$ e condizioni al contorno di convezione. In questi casi conviene abbandonare il metodo Direct GFDM in favore di varianti che non inglobino le equazioni all'interno dello stencil.

In ogni caso nella maggior parte degli impieghi a cui questo solutore termico è rivolto è ragionevole ipotizzare $k,\rho,c_p=cost$.

Ripetendo i passaggi visti al paragrafo 3.7, si consideri un generico nodo con N nodi vicini. Per costruire uno schema Direct GFDM accurato al II ordine si vuole localizzare un polinomio di secondo grado del tipo:

$$u(x,y,z) = u_0 + ax + by + cz + dx^2 + ey^2 + fz^2 + gxy + hxz + iyz$$

I coefficienti di tale polinomio vengono determinati minimizzando gli errori di approssimazione sui satelliti, gli errori sulle equazioni di governo e gli errori sulle condizioni al contorno:

$$e_j = u_0 + ax_j + by_j + cz_j + dx_j^2 + ey_j^2 + fz_j^2 + gx_jy_j + hx_jz_j + iy_jz_j - u_j$$

$$e_{PDE} = \frac{k}{\rho c_p}\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2}\right) + q_v - \frac{\partial u}{\partial t}$$

$$e_{BC} = g_1 u + g_2 n_x \frac{\partial u}{\partial x} + g_2 n_y \frac{\partial u}{\partial y} + g_2 n_z \frac{\partial u}{\partial z} - g_3$$

In forma matriciale quindi si può scrivere una relazione del tipo $\vec{e} = \mathbf{V}\vec{d} - \vec{u}$:

$$\begin{Bmatrix} e_1 \\ \vdots \\ e_j \\ \vdots \\ e_N \\ e_{PDE} \\ e_{BC} \end{Bmatrix} = \begin{bmatrix} 1 & x_1 & y_1 & z_1 & x_1^2 & y_1^2 & z_1^2 & x_1y_1 & x_1z_1 & y_1z_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_j & y_j & z_j & x_j^2 & y_j^2 & z_j^2 & x_jy_j & x_jz_j & y_jz_j \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_N & y_N & z_N & x_N^2 & y_N^2 & z_N^2 & x_Ny_N & x_Nz_N & y_Nz_N \\ 0 & 0 & 0 & 0 & 2\frac{k}{\rho c_p} & 2\frac{k}{\rho c_p} & 2\frac{k}{\rho c_p} & 0 & 0 & 0 \\ g_1 & g_2n_x & g_2n_y & g_2n_z & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{Bmatrix} u_0 \\ a \\ b \\ c \\ d \\ e \\ f \\ g \\ h \\ i \end{Bmatrix} - \begin{Bmatrix} u_1 \\ \vdots \\ u_j \\ \vdots \\ u_N \\ -q_v + \frac{\partial u}{\partial t} \\ g_3 \end{Bmatrix}$$

A questo punto si definisce il funzionale $J = \sum_{j=1}^{N} w_j^2 e_j^2 + w_{PDE}^2 e_{PDE}^2 + w_{BC}^2 e_{BC}^2$ da minimizzare con il metodo dei minimi quadrati pesato (WLS):

$$minJ = \|\mathbf{V}\vec{d} - \vec{u}\|^2 = \left(\mathbf{V}\vec{d} - \vec{u}\right)^\top \mathbf{W}^2 \left(\mathbf{V}\vec{d} - \vec{u}\right) = \vec{d}^\top \mathbf{V}^\top \mathbf{W}^2 \mathbf{V}\vec{d} - \vec{d}^\top \mathbf{V}^\top \mathbf{W}^2 \vec{u} - \vec{u}^\top \mathbf{W}^2 \mathbf{V}\vec{d}$$

$$0 = \frac{\partial J}{\partial \vec{d}} = 2\mathbf{V}^\top \mathbf{W}^2 \mathbf{V}\vec{d} - \mathbf{V}^\top \mathbf{W}^2 \vec{u} - \left(\vec{u}^\top \mathbf{W}^2 \mathbf{V}\right)^\top \quad \rightarrow \quad \mathbf{V}^\top \mathbf{W}^2 \mathbf{V}\vec{d} = \mathbf{V}^\top \mathbf{W}^2 \vec{u}$$

$$\rightarrow \quad \vec{d} = \underbrace{\left(\mathbf{V}^\top \mathbf{W}^2 \mathbf{V}\right)^{-1} \mathbf{V}^\top \mathbf{W}^2}_{\mathbf{C}} \vec{u}$$

Dove $\mathbf{W} = diag\,(w_1, \ldots, w_j, \ldots, w_N, w_{PDE}, w_{BC})$ è una matrice diagonale contenente i pesi di ciascun contributo. I vari w_j si possono calcolare con una distribuzione gaussiana utile a rendere più influenti i satelliti vicini al nodo centrale. Invece w_{PDE} e w_{BC} si possono ipotizzare costanti e pari a 2, come proposto da Suchde.[9] Più avanti verrà analizzata l'influenza dei vari pesi sulla soluzione numerica.

Noto lo stencil per ciascun nodo del dominio, è immediato popolare il sistema sparso finale a partire dalla prima riga dell'equazione $\vec{d} = \mathbf{C}\vec{u}$:

$$1\ u_i - \sum_{j=1}^{N} c_{1,j} u_j = c_{1,PDE}\left(q_v - \frac{\partial u}{\partial t}\right) + c_{1,BC} g_3$$

[9] Suchde, "Conservation and Accuracy in Meshfree Generalized Finite Difference Methods", cit.

Nel caso di problema stazionario il termine $\frac{\partial u}{\partial t}$ scompare, altrimenti per simulare un transitorio è possibile ricorrere al metodo di Eulero Implicito:

$$\frac{\partial u}{\partial t} \cong \frac{u_i^{k+1} - u_i^k}{\Delta t} \quad \rightarrow \quad \left(1 - \frac{c_{1,PDE}}{\Delta t}\right) u_i^{k+1} - \sum_{j=1}^{N} c_{1,j} u_j^{k+1} = c_{1,PDE} \left(-q_v - \frac{u_i^k}{\Delta t}\right) + c_{1,BC} g_3$$

Avendo denotato con l'apice k il passo temporale precedente e con $k+1$ il passo temporale successivo. I termini a sinistra entrano nella matrice sparsa $\mathbf{M}$, i termini a destra costituiscono il termine noto $\vec{b}$.

Si tratta quindi di uno schema accurato al II ordine nello spazio e al I ordine nel tempo. In alternativa, approssimare la derivata temporale con uno schema trapezoidale (metodo di Crank-Nicolson) darebbe luogo ad uno schema accurato al II ordine anche nel tempo, ma in caso di discontinuità nella soluzione potrebbero svilupparsi oscillazioni ad alta frequenza.
Per questo motivo ci si limita al metodo di Eulero Implicito del I ordine, che anzi in presenza di oscillazioni tende a smorzare numericamente il comportamento armonico.

3.11 Discretizzazione problema meccanico

I problemi strutturali elastici introducono un certo grado di complessità rispetto ai problemi termici. Per prima cosa le variabili dipendenti diventano due aumentando notevolmente il costo computazionale del metodo e peggiorando il condizionamento. Vi è poi la diversa natura delle equazioni di equilibrio: la derivata temporale è del II ordine e al RHS oltre al termine diffusivo c'è un termine aggiuntivo potenzialmente destabilizzante.

$$\rho \frac{\partial^2 \vec{u}}{\partial t^2} = \mu \nabla^2 \vec{u} + (\lambda + \mu) \nabla (\nabla \cdot \vec{u}) + \vec{F} \qquad su\ \Omega$$

Con condizioni al contorno:

$$g_1 \vec{u} + g_2 (\sigma \cdot \vec{n}) = \vec{g_3} \qquad su\ \partial\Omega$$

Come fatto notare nel capitolo precedente, questa scrittura consente di imporre spostamenti, trazioni e supporti elastici su un qualsiasi nodo.

La discretizzazione delle equazioni di Navier-Cauchy in tre dimensioni segue gli stessi passaggi visti al paragrafo precedente per il problema termico. Anche qui è necessario costruire uno schema accurato al II ordine per approssimare correttamente le derivate. Visto che nel metodo Direct GFDM le equazioni di governo entrano nello stencil, è d'obbligo definire tre diversi polinomi di secondo grado per le tre componenti di spostamento u, v, w:

$$u(x,y,z) = u_0 + a_u x + b_u y + c_u z + d_u x^2 + e_u y^2 + f_u z^2 + g_u xy + h_u xz + i_u yz$$

$$v(x,y,z) = v_0 + a_v x + b_v y + c_v z + d_v x^2 + e_v y^2 + f_v z^2 + g_v xy + h_v xz + i_v yz$$

$$w(x,y,z) = w_0 + a_w x + b_w y + c_w z + d_w x^2 + e_w y^2 + f_w z^2 + g_w xy + h_w xz + i_w yz$$

Ciascun polinomio quindi avrà coefficienti differenti. Il metodo dei minimi quadrati dovrà considerare un vettore delle derivate $\vec{d}$ triplicato nelle dimensioni rispetto al problema termico:

$$\vec{d} = \left\{ {}_{u_0,a_u,b_u,c_u,d_u,e_u,f_u,g_u,h_u,i_u},\ {}_{v_0,a_v,b_v,c_v,d_v,e_v,f_v,g_v,h_v,i_v},\ {}_{w_0,a_w,b_w,c_w,d_w,e_w,f_w,g_w,h_w,i_w} \right\}^\top$$

Per ricavare questi 30 coefficienti si deve procedere come fatto in precedenza, scrivendo gli errori e_j, e_{PDE}, e_{BC} commessi rispettivamente dall'approssimazione polinomiale, dalle equazioni di governo discretizzate e dalle condizioni al contorno discretizzate:

$$e_{ju} = u_0 + a_u x_j + b_u y_j + c_u z_j + d_u x_j^2 + e_u y_j^2 + f_u z_j^2 + g_u x_j y_j + h_u x_j z_j + i_u y_j z_j - u_j$$

$$e_{jv} = v_0 + a_v x_j + b_v y_j + c_v z_j + d_v x_j^2 + e_v y_j^2 + f_v z_j^2 + g_v x_j y_j + h_v x_j z_j + i_v y_j z_j - v_j$$

$$e_{jw} = w_0 + a_w x_j + b_w y_j + c_w z_j + d_w x_j^2 + e_w y_j^2 + f_w z_j^2 + g_w x_j y_j + h_w x_j z_j + i_w y_j z_j - w_j$$

$$e_{PDE1} = 2\,(2\mu + \lambda)\,d_u + 2\mu\; e_u + 2\mu\; f_u + (\lambda + \mu)\,g_v + (\lambda + \mu)\,h_w + F_x - \rho\frac{\partial^2 u}{\partial t^2}$$

$$e_{PDE2} = 2\mu\; d_v + 2\,(2\mu + \lambda)\,e_v + 2\mu\; f_v + (\lambda + \mu)\,g_u + (\lambda + \mu)\,i_w + F_y - \rho\frac{\partial^2 v}{\partial t^2}$$

$$e_{PDE3} = 2\mu\; d_w + 2\mu\; e_w + 2\,(2\mu + \lambda)\,f_w + (\lambda + \mu)\,h_u + (\lambda + \mu)\,i_v + F_z - \rho\frac{\partial^2 w}{\partial t^2}$$

$$e_{BC1} = g_{1u}u + g_{2u}\left[n_x\,(2\mu + \lambda)\,a_u + n_x\lambda\,(b_v + c_z) + n_y\mu\,(b_u + a_v) + n_z\mu\,(c_u + a_w)\right] - g_{3u}$$

$$e_{BC2} = g_{1v}v + g_{2v}\left[n_x\mu\,(b_u + a_v) + n_y\,(2\mu + \lambda)\,b_v + n_y\lambda\,(a_u + c_w) + n_z\mu\,(c_v + b_w)\right] - g_{3v}$$

$$e_{BC3} = g_{1w}w + g_{3w}\left[n_x\mu\,(c_u + a_w) + n_y\mu\,(c_v + b_w) + n_z\,(2\mu + \lambda)\,c_w + n_z\lambda\,(b_v + a_u)\right] - g_{3w}$$

In forma matriciale la nota relazione $\vec{e} = \mathbf{V}\vec{d} - \vec{u}$ assume una struttura a blocchi come quella illustrata nella seguente immagine:

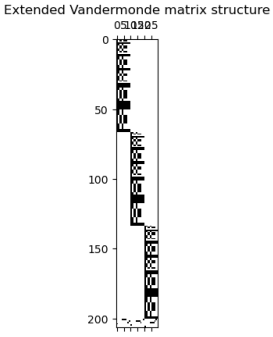


Figura 3.16: Struttura matrice Vandermonde estesa $\mathbf{V}$ per un nodo al contorno

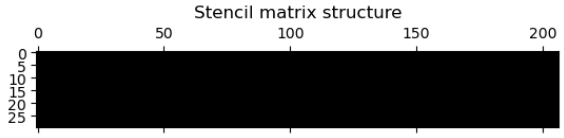


Figura 3.17: Struttura matrice stencil $\mathbf{C}$ per lo stesso nodo

La matrice dello stencil $\mathbf{C} = \left(\mathbf{V}^\top\mathbf{W}^2\mathbf{V}\right)^{-1}\mathbf{V}^\top\mathbf{W}^2$ invece è generalmente piena e ha dimensioni $30 \times (3N + 6)$, dove N è il numero di satelliti del nodo considerato.

Il sistema sparso finale si costruisce scrivendo tre equazioni per ciascun punto del dominio. Tali equazioni emergono direttamente dalla relazione $\vec{d} = \mathbf{C}\vec{u}$ estraendo le righe 1, 11, 21. Per lo spostamento lungo x si ha:

$$\begin{aligned} 1\,u_i - \sum_{j=1}^{N} c_{1,j}u_j - \sum_{j=1}^{N} c_{1,(N+j)}v_j - \sum_{j=1}^{N} c_{1,(2N+j)}w_j = \\ c_{1,PDE1}\left(-F_x + \rho\frac{\partial^2 u}{\partial t^2}\right) + c_{1,PDE2}\left(-F_y + \rho\frac{\partial^2 v}{\partial t^2}\right) \\ + c_{1,PDE3}\left(-F_z + \rho\frac{\partial^2 w}{\partial t^2}\right) + c_{1,BC1}g_{3u} + c_{1,BC2}g_{3v} + c_{1,BC3}g_{3w} \end{aligned}$$

E analogamente per gli spostamenti lungo y e lungo z.

Per un problema stazionario le derivate temporali $\frac{\partial^2 \vec{u}}{\partial t^2}$ si annullano e la soluzione in termini di spostamenti si ricava semplicemente dalla risoluzione del sistema lineare appena presentato e la matrice $\mathbf{M}$ è anche la matrice di rigidezza.

Nel caso di un problema transitorio invece è necessario approssimare in qualche modo le derivate seconde, oppure trasformare il sistema di equazioni del II ordine in un sistema di equazioni del I ordine nel tempo:

$$\rho \frac{\partial^2 \vec{u}}{\partial t^2} = \mu \nabla^2 \vec{u} + (\lambda + \mu) \nabla (\nabla \cdot \vec{u}) + \vec{F} \;\rightarrow\; \begin{cases} \rho \frac{\partial \vec{z}}{\partial t} = \mu \nabla^2 \vec{u} + (\lambda + \mu) \nabla (\nabla \cdot \vec{u}) + \vec{F} \\ \frac{\partial \vec{u}}{\partial t} = \vec{z} \end{cases}$$

E quindi procedere con la costruzione di un sistema lineare di dimensioni doppie.
Nota che per simulare il processo di stampa 3D è possibile trascurare i termini inerziali, dal momento che le accelerazioni in gioco sono praticamente nulle. In questo lavoro pertanto ci si limita a risolvere il problema elastico stazionario.

3.11.1 Stabilizzazione FIC

Sviluppata da Oñate,[10] la tecnica Finite Increment Calculus (FIC) permette di migliorare la stabilità e l'accuratezza dei metodi numerici in modo molto naturale. Essa infatti consiste nel modificare le equazioni di governo scrivendo i bilanci nel dominio discretizzato finito anziché in forma infinitesimale. Pertanto, è possibile applicarla ad un qualunque schema numerico (FEM, FVM, FDM, GFDM, ecc...).
Per le equazioni dell'elasticità statica lo schema si riconduce ad una stabilizzazione del tipo:

$$\mu \nabla^2 \vec{u} + (\lambda + \mu) \nabla (\nabla \cdot \vec{u}) + \vec{F} - \underbrace{\frac{1}{2} h \nabla \cdot \left[\mu \nabla^2 \vec{u} + (\lambda + \mu) \nabla (\nabla \cdot \vec{u}) + \vec{F} \right]}_{} = 0$$

Dove h è la smoothing length. Come fatto notare dall'autore, il metodo FIC introduce un termine diffusivo aggiuntivo che va a migliorare la stabilità e che scompare all'infittirsi della griglia.

Ovviamente anche le condizioni al contorno devono essere alterate:

$$g_1 \vec{u} + g_2 (\sigma \cdot \vec{n}) - g_3 - \underbrace{\frac{1}{2} h \vec{n} \left[\mu \nabla^2 \vec{u} + (\lambda + \mu) \nabla (\nabla \cdot \vec{u}) + \vec{F} \right]}_{} = 0$$

Il processo di discretizzazione chiaramente è del analogo a quello del problema meccanico non stabilizzato. Attenzione però che nelle nuove equazioni di governo compaiono termini del terzo ordine. Alzare l'ordine di approssimazione del polinomio interpolante può essere molto difficoltoso a causa di problemi di malcondizionamento. Di conseguenza Oñate propone per il metodo GFDM di introdurre le stabilizzazioni solamente nelle condizioni al contorno, lasciando le equazioni di governo inalterate.
Del resto si è visto che il metodo GFDM classico soffre proprio in prossimità dei bordi; il fatto che possa essere sufficiente una stabilizzazione più mirata in questa area va certamente a confermare le conclusioni del paragrafo 3.6.
Questo però implica anche che tale stabilizzazione non è necessaria per il metodo Direct GFDM, che va ad introdurre termini stabilizzanti in modo differente. Per questo motivo in questo lavoro non viene fatto alcun test sulla stabilizzazione FIC.

[10] Eugenio Oñate. "Finite increment calculus (FIC): a framework for deriving enhanced computational methods in mechanics". In: *Adv. Model. Simul. Eng. Sci.* 3 (2016), 14:1–14:18. DOI: 10.1186/s40323-016-0065-9.

3.12 Discretizzazione problema termo-meccanico

Nel capitolo precedente si è visto che le equazioni termo-meccaniche disaccoppiate differiscono da quelle elastiche solamente per la presenza di un carico volumetrico aggiuntivo e per le condizioni al contorno. Discretizzando secondo l'approccio Direct GFDM, tutte queste differenze si collocano al RHS, quindi la matrice di rigidezza $\mathbf{M}$ rimane del tutto identica a quella trovata al paragrafo precedente.
Adesso pertanto ci si occupa di esplicitate le modifiche da fare al termine noto $\vec{b}$ per risolvere problemi di natura termo-meccanica:

$$\begin{aligned}
&1\ u_i - \sum_{j=1}^{N} c_{1,j}u_j - \sum_{j=1}^{N} c_{1,(N+j)}v_j - \sum_{j=1}^{N} c_{1,(2N+j)}w_j \\
&= c_{1,PDE1}\left(-F_x + \frac{E\alpha}{1-2\nu}\frac{\partial T}{\partial x} + \rho\frac{\partial^2 u}{\partial t^2}\right) + c_{1,PDE2}\left(-F_y + \frac{E\alpha}{1-2\nu}\frac{\partial T}{\partial y} + \rho\frac{\partial^2 v}{\partial t^2}\right) \\
&\quad + c_{1,PDE3}\left(-F_z + \frac{E\alpha}{1-2\nu}\frac{\partial T}{\partial z} + \rho\frac{\partial^2 w}{\partial t^2}\right) + c_{1,BC1}\left(g_{3u} + g_{2u}n_x\frac{E\alpha\Delta T}{1-2\nu}\right) \\
&\quad + c_{1,BC2}\left(g_{3v} + g_{2v}n_y\frac{E\alpha\Delta T}{1-2\nu}\right) + c_{1,BC3}\left(g_{3w} + g_{2w}n_z\frac{E\alpha\Delta T}{1-2\nu}\right)
\end{aligned}$$

In due dimensioni con la formulazione plane-stress invece si ha:

$$\begin{aligned}
1\ u_i - \sum_{j=1}^{N} c_{1,j}u_j - \sum_{j=1}^{N} c_{1,(N+j)}v_j &= c_{1,PDE1}\left(-F_x + \frac{E\alpha}{1-\nu}\frac{\partial T}{\partial x} + \rho\frac{\partial^2 u}{\partial t^2}\right) \\
&\quad + c_{1,PDE2}\left(-F_y + \frac{E\alpha}{1-\nu}\frac{\partial T}{\partial y} + \rho\frac{\partial^2 v}{\partial t^2}\right) \\
&\quad + c_{1,BC1}\left(g_{3u} + g_{2u}n_x\frac{E\alpha\Delta T}{1-\nu}\right) + c_{1,BC2}\left(g_{3v} + g_{2v}n_y\frac{E\alpha\Delta T}{1-\nu}\right)
\end{aligned}$$

E analogamente per gli altri spostamenti. Il sistema sparso finale si costruisce concatenando le equazioni di tutti i nodi del dominio.
Si ricorda che l'approccio disaccoppiato considerato in questo lavoro consiste nel risolvere il problema termico e solo successivamente il problema termo-meccanico qui presentato.

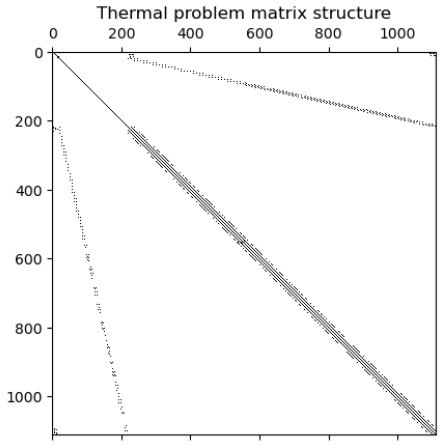


Figura 3.18: Struttura matrice $\mathbf{M}$ per un problema termico 2D (nodi strutturati)

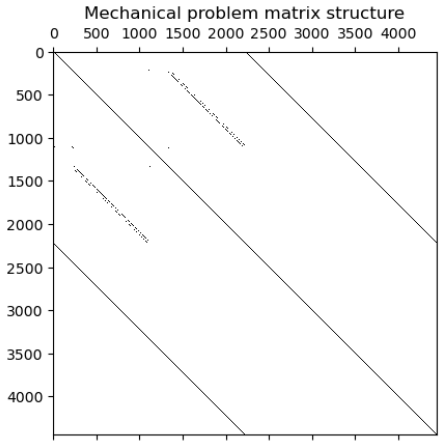


Figura 3.19: Struttura matrice $\mathbf{M}$ per un problema meccanico 2D (nodi strutturati)

3.13 Decomposizione QR e SVD

Facendo riferimento a quanto detto per il metodo Direct GFDM, la matrice dello stencil $\mathbf{C}$ si può calcolare usando la seguente definizione:

$$\mathbf{C} = \left(\mathbf{V}\mathbf{W}^2\mathbf{V}\right)^{-1}\mathbf{V}^\top\mathbf{W}^2$$

Ovviamente calcolarla in questo modo è fortemente sconsigliato ed estremamente controproducente sia in termini di malcondizionamento del problema sia per l'implementazione numerica. Una prima soluzione potrebbe consistere nell'applicare la decomposizione QR alla matrice normale $\mathbf{A} = \mathbf{V}^\top\mathbf{W}^2\mathbf{V}$:

$$\mathbf{A} = \mathbf{Q}\mathbf{R} \;\rightarrow\; \mathbf{A}^{-1} = \mathbf{R}^{-1}\mathbf{Q}^\top \;\rightarrow\; \mathbf{C} = \mathbf{R}^{-1}\mathbf{Q}^\top\mathbf{V}^\top\mathbf{W}^2$$

Dove $\mathbf{Q}$ è una matrice ortogonale e $\mathbf{R}$ è una matrice triangolare inferiore.
In questo modo si ottengono importanti benefici nella stabilità del metodo, ma è ancora richiesto il calcolo esplicito della matrice normale $\mathbf{V}^\top\mathbf{W}^2\mathbf{V}$. Questo si può dimostrare rendere il condizionamento del problema quadrato:

$$\kappa(\mathbf{V}^\top\mathbf{W}^2\mathbf{V}) = \left[\kappa(\mathbf{V}^\top\mathbf{W})\right]^2$$

Si ricorda che come regola spannometrica, l'ordine di grandezza di κ_2 indica il numero di cifre perse nell'accuratezza della soluzione. Un condizionamento al quadrato rende la soluzione numerica inutilizzabile molto facilmente.

Come è solito fare per i problemi ai minimi quadrati, conviene invece evitare il calcolo della matrice normale e decomporre invece la matrice di Vandermonde estesa. A tale proposito è comodo introdurre la seguente notazione: $\mathbf{A} = \mathbf{V}\mathbf{W}$, $\vec{x} = \vec{d}$, $\vec{b} = \mathbf{W}\vec{u}$.
Ricordando che la minimizzazione del funzionale J porta alla relazione $\mathbf{V}\mathbf{W}^2\mathbf{V}\vec{d} = \mathbf{V}^\top\mathbf{W}^2\vec{u}$, si procede come segue:

$$\mathbf{A}^\top\mathbf{A}\vec{x} = \mathbf{A}^\top\vec{b} \qquad \mathbf{A} = \mathbf{Q}\mathbf{R}$$

$$\left(\mathbf{Q}\mathbf{R}\right)^\top\left(\mathbf{Q}\mathbf{R}\right)\vec{x} = \left(\mathbf{Q}\mathbf{R}\right)^\top\vec{b} \;\rightarrow\; \mathbf{R}^\top\underbrace{\mathbf{Q}^\top\mathbf{Q}}_{\mathbf{I}}\mathbf{R}\vec{x} = \mathbf{R}^\top\mathbf{Q}^\top\vec{b} \;\rightarrow\; \mathbf{R}\vec{x})\mathbf{Q}^\top\vec{b} \;\rightarrow\; \vec{d} = \mathbf{R}^{-1}\mathbf{Q}\mathbf{W}\vec{u}$$

Da cui si estrae la matrice dello stencil:

$$\mathbf{C} = \mathbf{R}^{-1}\mathbf{Q}\mathbf{W}$$

Nota che calcolare l'inversa di $\mathbf{Q}$ è molto semplice ($\mathbf{Q}^{-1} = \mathbf{Q}^\top$), però calcolare l'inversa di una matrice triangolare $\mathbf{R}$ non è banale, specialmente nel caso di near-singular problems, ed è certamente il punto debole di tale approccio.

A tale proposito, vista anche la dimensione ridotta della matrice $\mathbf{A} = \mathbf{V}\mathbf{W}$, è possibile sostituire la decomposizione QR con l'approccio più potente ma costoso della decomposizione ai valori singolari (Singular Value Decomposition, SVD):

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top \;\rightarrow\; \vec{x} = \mathbf{V}\mathbf{\Sigma}^{-1}\mathbf{U}^\top\vec{b} \quad\rightarrow\quad \mathbf{C} = \mathbf{V}\mathbf{\Sigma}^{-1}\mathbf{U}^\top\mathbf{W}$$

Dove $\mathbf{\Sigma}$ è una matrice diagonale contenente i valori singolari di $\mathbf{A}$, mentre $\mathbf{U}$, $\mathbf{V}$ sono matrici ortogonali. Il grande vantaggio rispetto al QR è che non ci sono matrici da invertire (l'inversa di una matrice diagonale è a sua volta una matrice diagonale contenente il reciproco di ciascun elemento).
Se necessario, è anche possibile alterare leggermente i valori singolari contenuti in $\mathbf{\Sigma}$, ad esempio sommandoci un numero molto piccolo come 10^{-12} così da evitare singolarità. Ovviamente non conviene abusare di tale possibilità perché va a scapito dell'accuratezza.

3.14 Condizioni al contorno esatte

Una peculiarità del metodo Direct GFDM è quella di imporre le condzioni al contorno debolmente, nel senso dei minimi quadrati. Ciò si è visto permette di semplificare molto la formulazione e di migliorare la stabilità del metodo. In casi particolari però potrebbe essere necessario imporre delle condizioni esatte, possibilmente senza alterare il metodo numerico.

L'approccio proposto in questo lavoro consiste nello scegliere per ciascun nodo al contorno un polinomio approssimante che soddisfi in modo esatto la condizione al contorno. Ad esempio, se al nodo 42 si vuole imporre una parete adiabatica $\frac{\partial u}{\partial x} = 0$, sarà sufficiente scegliere per tale nodo un polinomio senza il termine lineare ax. A questo punto si può procedere con la medesima operazione di minimizzazione che si ha per i nodi interni.
Più in generale, per ciascun nodo al contorno si può costruire un sistema di riferimento locale $x^* - y^* - z^*$ in cui l'asse x^* coincide con la normale uscente, mentre gli altri assi y^*, z^* non sono rilevanti e si possono semplicemente ricavare da prodotto vettoriale.
In questo riferimento il polinomio che soddisfa automaticamente le condizioni al contorno si può determinare mettendo a sistema le seguenti equazioni:

$$\begin{cases} u(x^*, y^*, z^*) = u_0 + ax^* + by^* + cz^* + dx^{*2} + ey^{*2} + fz^{*2} + gx^*y^* + hx^*z^* + iy^*z^* \\ g_1 u + g_2 \frac{\partial u}{\partial x^*} = g_3 \quad \rightarrow \quad a = \frac{g_3 - g_1 u_0}{g_2} \end{cases}$$

A questo punto si prosegue come visto in precedenza scrivendo gli errori di approssimazione polinomiale e compilando la matrice di Vandermonde. Ad esempio, per una condizione di convezione si ha:

$$e_j = u_0 \left(1 - \frac{h}{k} x_j^*\right) + by_j^* + cz_j^* + dx_j^{*2} + ey_j^{*2} + fz_j^{*2} + gx_j^* y_j^* + hx_j^* z_j^* + iy_j^* z_j^* - \left(u_j - \frac{h}{k} u_{amb} x_j^*\right)$$

Nota che in questo caso non è più necessario aggiungere una riga alla matrice di Vandermonde per imporre le condizioni al contorno, essendo queste già soddisfatte implicitamente dall'intera classe dei polinomi così parametrizzati.
Si evidenzia inoltre che il vettore $\vec{d}$ ovviamente mancherà dell'elemento a e che la matrice di Vandermonde avrà una colonna in meno.

Capitolo 4

Implementazione

Nel capitolo precedente sono state ricavate le equazioni discretizzate secondo l'approccio GFDM. In questo capitolo è descritta l'implementazione all'interno del software OpenGFDM, partendo dal parsing del GCODE e arrivando a risolvere alcuni semplici problemi dimostrativi in 2D, passando per gli algoritmi di generazione pointcloud e di selezione dei nodi vicini.

4.1 Architettura generale del software

Come accennato in precedenza OpenGFDM è costituito da un frontend e da un backend indipendenti tra loro e che comunicano mediante un collegamento Websocket e messaggi testuali in formato JSON.

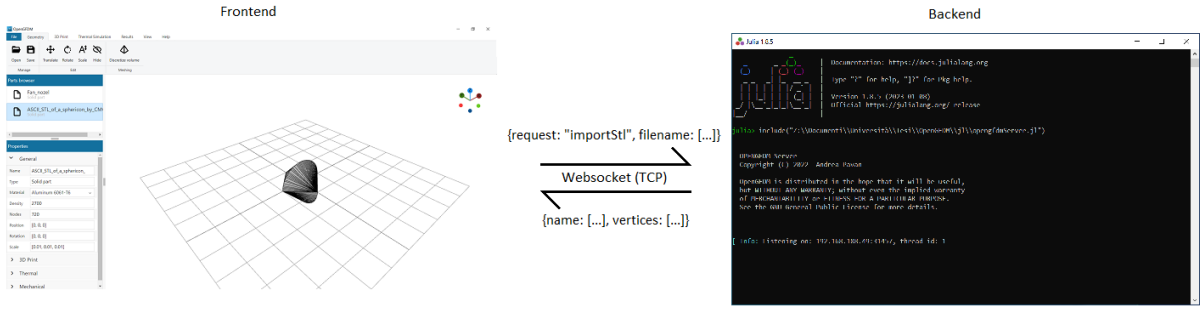


Figura 4.1: Illustrazione dell'architettura generale di OpenGFDM

Il **frontend** comprende l'interfaccia grafica, la gestione delle geometrie e il monitoring delle simulazioni. Costruito con tecnologie web (HTML, CSS, JavaScript) usando il framework di sviluppo open-source Electron,[1] esso si occupa di facilitare l'utente a definire le simulazioni, avviarle correttamente, mostrarne lo svolgimento in tempo reale e visualizzare i risultati.
La parte di Graphical User Interface (GUI) vera e propria fa uso del toolkit METROUI[2] per i controlli, il menù ribbon, le notifiche e le finestre popup.
La visualizzazione invece è affidata alla libreria three.js[3] e ad altri componenti minori (vedere il codice sorgente per ulteriori dettagli). Essa è generata da una camera prospettica con FOV di 45° e movibile dall'utente con mouse e/o touch screen. A seconda del livello di zoom anche la griglia x-y di aiuto viene più o meno raffinata.

[1] OpenJS Foundation. URL: `https://www.electronjs.org/`.
[2] Sergey Pimenov. URL: `https://korzh.com/metroui`.
[3] Three.js Project. URL: `https://threejs.org/`.

Due viste separate vengono renderizzate a seconda della fase in cui è la simulazione: durante il pre-processing viene visualizzata la vista principale con le geometrie STL e GCODE, mentre durante la simulazione e il post-processing la vista mostrata a schermo renderizza lo stato corrente.

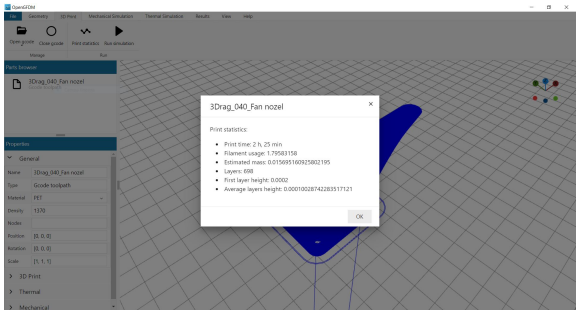


Figura 4.2: Screenshot vista principale

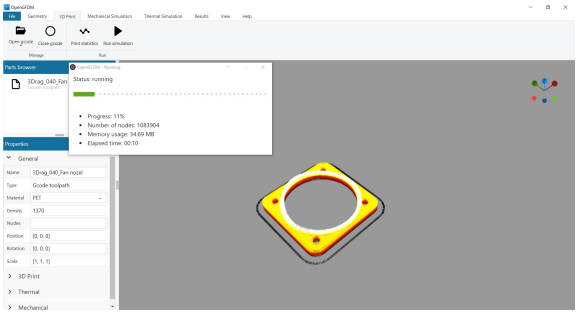


Figura 4.3: Screenshot vista simulazione

Il **backend** invece è strutturato in una serie di script e funzioni Julia che fanno capo ad un server (opengfdmServer.jl).
Julia[4] è un linguaggio di programmazione general-purpose rivolto al calcolo scientifico, progettato per essere veloce come il C pur essendo dinamico e avendo una notazione simile al Python. Tra le caratteristiche si annoverano:

- Elevate prestazioni, comparabili con quelle di linguaggi compilati staticamente come il C, anche senza vettorizzare il codice;
- Compilato dinamicamente durante l'esecuzione (Just In Time, JIT);
- Multiple-dispatch delle funzioni, in base al tipo degli argomenti viene generato automaticamente il codice specializzato;
- Progettato per il calcolo parallelo e distribuito;
- Macro, per generare codice Julia in modo programmatico;
- Open-source, il linguaggio di programmazione è interamente scritto in Julia stesso ed è liberamente consultabile;

Il fatto che la compilazione avvenga durante l'esecuzione del programma significa che la prima volta che viene lanciato si ha un considerevole ritardo (il cosiddetto "first time to plot"). Questo aspetto in particolare ha pesanti implicazioni nello sviluppo di OpenGFDM.

Il backend server implementato nello script opengfdmServer.jl non fa altro che lanciare un microservice Websocket implementato nel pacchetto HTTP.jl e stare in ascolto. In funzione del messaggio JSON ricevuto lo script si occupa di deserializzare i dati, eseguire il codice opportuno e rispondere sullo stesso canale.
L'approccio tradizionale dei software ingegneristici è quello di eseguire la simulazione e attendere il completamento per leggere un file di output e farne il parsing. Al contrario, mantenere un link IPC attivo offre importanti vantaggi:

- interoperabilità con un qualunque sistema in grado di leggere e scrivere stringhe JSON (non solo l'interfaccia grafica in Electron, ma anche eventuali webapp, servizi vari, linguaggi di programmazione);
- permette di controllare lo stato di esecuzione in real-time;
- si ricevono gradualmente e direttamente i risultati in un formato facilmente utilizzabile;
- workflow dell'utente integrato e semplificato;
- processo di esecuzione su macchina remota al 100% equivalente di quello in locale;

[4]Jeff Bezanson et al. "Julia: A fresh approach to numerical computing". In: *SIAM Review* 59.1 (2017), pp. 65–98. DOI: 10.1137/141000671. URL: https://epubs.siam.org/doi/10.1137/141000671.

Lo svantaggio principale sta nel dover trasmettere i dati da backend a frontend e viceversa; per simulazioni pesanti è un'attività onerosa e che rallenta in modo significativo l'esecuzione. A tale proposito OpenGFDM permette all'utente di cambiare la frequenza con cui i dati vengono trasmessi frontend e di disaccoppiare l'aggiornamento dello stato dalla preview dei risultati.

Sono stati provati diversi canali di Inter-Process-Communication (IPC): stdout, socket TCP, Websocket. Per confrontarne le prestazioni si misura più volte il tempo necessario ad ottenere e trasmettere le informazioni hardware all'interno della finestra "About OpenGFDM":

Canale	stdout	TCP socket	Websocket
Test #1	1571 ms	2169 ms	751 ms
Test #2	1143 ms	54 ms	24 ms
Test #3	1372 ms	17 ms	27 ms
Test #4	1568 ms	54 ms	91 ms
Test #5	1675 ms	25 ms	22 ms

La scelta è ricaduta sul protocollo Websocket per via delle sue performance e dell'assenza di problematiche di buffering. Sorprendentemente, il canale Websocket risulta a tratti più veloce del socket TCP su cui è basato, probabilmente a causa delle dimensioni del buffer.
Il canale stdout invece in questo test mostra un comportamento non competitivo, sia a causa del buffering sia dal fatto che Julia viene avviato e terminato ad ogni richiesta, esasperando così il problema del first-time-to-plot.

4.2 Parsing gcode e ricostruzione layer

Come accennato in precedenza, la sequenza di istruzioni che la stampante deve seguire per realizzare un pezzo è memorizzata in un file GCODE in formato testuale.

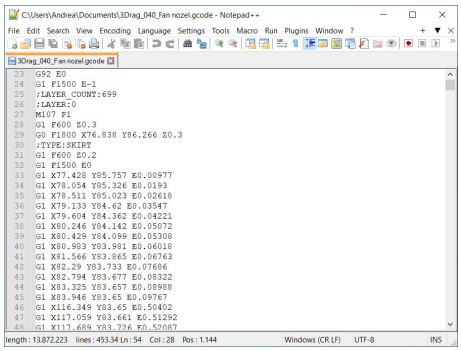


Figura 4.4: Esempio di file GCODE

Per poter eseguire la simulazione di processo è fondamentale tradurre queste istruzioni nell'andamento nel tempo di posizione, velocità, portata e temperatura materiale:

$$x(t), \qquad y(t), \qquad z(t), \qquad v(t), \qquad Q(t), \qquad T(t)$$

Facendo riferimento alla documentazione di Marlin Firmware,[5] ciascuna riga viene opportunamente separata ed analizzata estraendo comando e argomenti.
Data l'ampia varietà di comandi possibili e l'esistenza di varianti specifiche per singole macchine, solo un piccolo subset delle istruzioni più comuni è supportato da parseGcode.jl.

[5] *Marlin G-code Index*. URL: `https://marlinfw.org/meta/gcode/`.

Comando	Descrizione
G0	Movimento lineare rapido (senza estrusione)
G1	Movimento lineare
G28	Auto home
G90	Posizionamento assoluto
G91	Posizionamento relativo
G92	Specifica la posizione corrente in un nuovo sistema di riferimento definito dall'utente
G92.1	Resetta posizione corrente al sistema di riferimento macchina
M82	Estrusione assoluta
M83	Estrusione relativa
M104	Imposta temperatura hotend

Oltre a ciò l'utente può specificare dei fattori di conversione differenti per posizione, velocità, posizione filamento, temperatura. Si raccomanda da questo punto di vista di lavorare con le unità del Sistema Internazionale per evitare errori nelle successive attività di analisi. È utile pertanto impostare i fattori appropriati, ad esempio $scaleFactor = 0.001$ per convertire le posizioni da mm a m.
Nota che OpenGFDM lavora con numeri puri in modo simile a Nastran: se non si vuole impiegare il sistema MKS, è sufficiente aggiustare le proprietà del materiale.

Una volta ricavati gli andamenti di posizione, velocità, posizione filamento e temperatura può essere utile calcolare delle quantità di interesse: lunghezza di ciascun segmento lineare, tempo di stampa, velocità di estrusione, numero layer corrente, altezza layer corrente.
Per semplicità il tempo di stampa è calcolato per ciascun segmento a partire dalla velocità $v(t)$ senza tenere conto delle impostazioni di accelerazione/jerk presenti nel firmware:

$$l_i = \sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2 + (z_i - z_{i-1})^2}$$

$$t_i = t_{i-1} + \frac{l_i}{v_i}$$

$$v_i^{extr} = \frac{x_i^{extr} - x_{i-1}^{extr}}{t_i - t_{i-1}} \quad \rightarrow \quad Q_i = \rho A v_i^{extr}$$

Sono inoltre ipotizzati layer piani: altezza h costante all'interno di ciascun layer, ma è ammesso che possa variare da layer a layer. Attenzione che la simulazione potrebbe perdere in accuratezza se queste ipotesi di lavoro non vengono incontrate. Si tratta comunque di assunzioni ragionevoli e in linea con la pratica amatoriale.

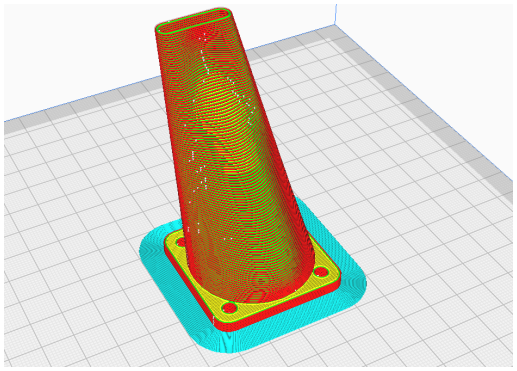


Figura 4.5: Esempio slicing geometria

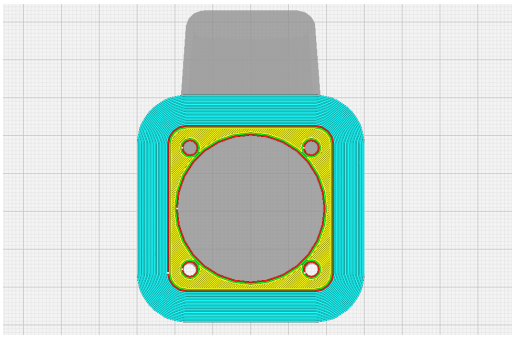


Figura 4.6: Illustrazione primo layer

Sono state provate diverse implementazioni per fare il parsing e i calcoli appena descritti:

- Python, lettura e parsing riga per riga;
- Julia (eachline), lettura e parsing riga per riga;
- Julia (hcat), lettura e parsing row-wise;
- Julia (allonce), lettura e parsing column-wise;

Le diverse opzioni provate in Julia fanno emergere le caratteristiche e le preferenze del linguaggio in termini di storage dei dati in memoria.

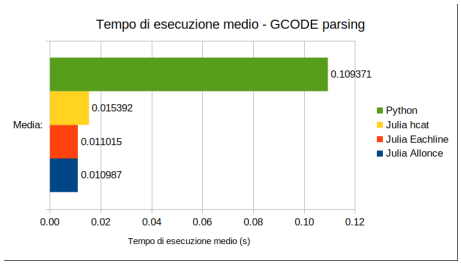


Figura 4.7: Comparazione tempi esecuzione parsing gcode

Si osserva che anche per un compito così semplice Python è 9.9 volte più lento di Julia, a conferma della scelta di scrivere il backend interamente in Julia.
Emerge anche che l'implementazione più vantaggiosa in termini di performance è quella che memorizza i dati column-wise, in accordo con quanto presente nella documentazione. La differenza con hcat in questo caso raggiunge addirittura il 40%.

4.2.1 Fast deposition solver

Prima di implementare il codice di simulazione vero e proprio conviene scrivere un solutore base che deposita solamente il materiale, senza fare alcuna analisi.
Seguendo il percorso di stampa calcolato come visto in precedenza, ad intervalli regolari specificati dall'utente viene depositata una sezione di materiale:

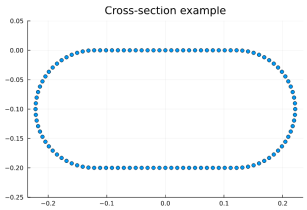


Figura 4.8: Illustrazione geometria sezione oblong

La sezione a forma di oblong è formata da un rettangolo arrotondato alle estremità da due semicerchi. Essa viene usata dal software di slicing Slic3r ed è considerata come metro di paragone da Raphaël et al.[6] Nonostante alcuni parametri non siano del tutto corrispondenti al vero, la sezione estrusa e depositata dall'hotend poco si discosta dall'oblong, a meno di non operare in condizioni estreme di sovra/sotto-estrusione.
In compenso, usare una sezione geometrica fissata senza necessità di interpolazioni porta notevoli vantaggi dal punto di vista computazionale. L'esecuzione di fastDepositionSolver.jl è estremamente veloce, solo 3.95 s per generare $> 6M$ nodi di una stampa di 3 ore circa.
Quando lanciato da interfaccia grafica però si incontra un rallentamento molto severo e il task richiede addirittura 95.51 s (135 s includendo anche il trasferimento dei dati). È chiaro che il sistema di comunicazione Websocket come implementato è molto più oneroso del previsto.

[6] Raphaël Comminal et al. "Numerical modeling of the strand deposition flow in extrusion-based additive manufacturing". In: *Additive manufacturing* 20 (2018), pp. 68–76. URL: `https://doi.org/10.1016/J.ADDMA.2017.12.013`.

4.3 Sviluppo backend

Viene adesso descritto brevemente il processo di sviluppo del backend attraverso degli esempi 2D dimostrativi di ciascuna classe di problemi.

4.3.1 Equazione Laplace

Dopo l'equazione di Laplace su dominio rettangolare con condizioni di Dirichlet risolta al capitolo precedente, si prosegue a piccoli passi analizzando alcuni casi con diverse condizioni al contorno e domini non rettangolari da validare in modo qualitativo:

- Campo elettrico all'interno di un condensatore piano (dominio rettangolare, condizioni Dirichlet su bordo e su nodi interni);[7]
- Test su dominio rettangolare forato con condizioni al contorno miste;[8]
- Corona circolare con condizioni di Dirichlet[9] e pointcloud generata con Flowmesher;

Di seguito ne vengono riportate le soluzioni:

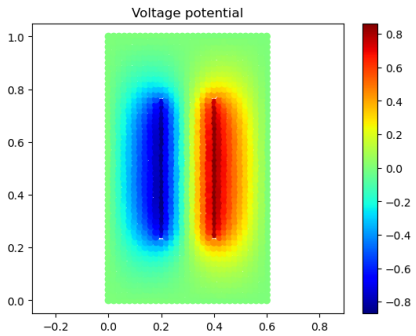


Figura 4.9: Soluzione numerica 1

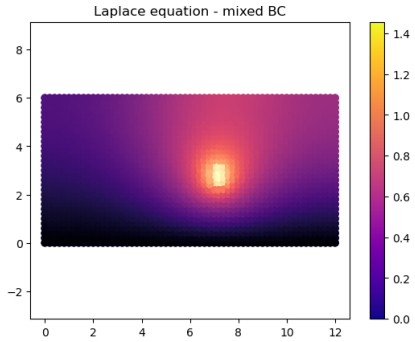


Figura 4.10: Soluzione numerica 2

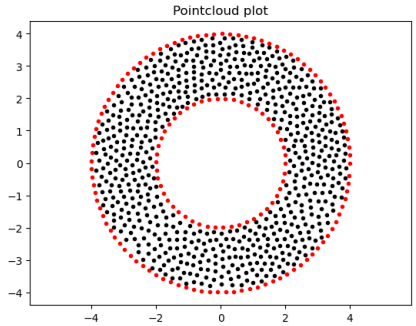


Figura 4.11: Pointcloud 3

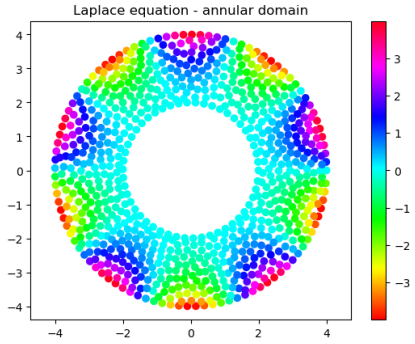


Figura 4.12: Soluzione numerica 3

Esse corrispondono alle soluzioni qualitative di riferimento; non è per ora svolta alcuna verifica quantitativa.

[7] James Richard Nagel. "Solving the Generalized Poisson Equation Using the Finite-Difference Method (FDM)". in: 2009.

[8] Mohammad Asif Zaman. "Numerical Solution of the Poisson Equation Using Finite Difference Matrix Operators". In: *Electronics* 11.15 (2022). DOI: 10.3390/electronics11152365. URL: https://www.mdpi.com/2079-9292/11/15/2365.

[9] Fourthirtytwo. URL: https://commons.wikimedia.org/wiki/File:Laplace%27s_equation_on_an_annulus.svg.

4.3.2 Equazione calore

Passando ai problemi di natura termica, l'esempio più semplice è quello di una barra ai cui estremi la temperatura è fissata:

- Temperatura lato sinistro $T_{left} = 400\ °C$;
- Temperatura lato destro $T_{right} = 300\ °C$;

Agli altri nodi del contorno è assegnata una condizione di parete adiabatica (condizione Neumann di flusso di calore nullo $q = 0$). Le proprietà del materiale ovviamente influenzano il solo transitorio e non sono rilevanti per la soluzione stazionaria.

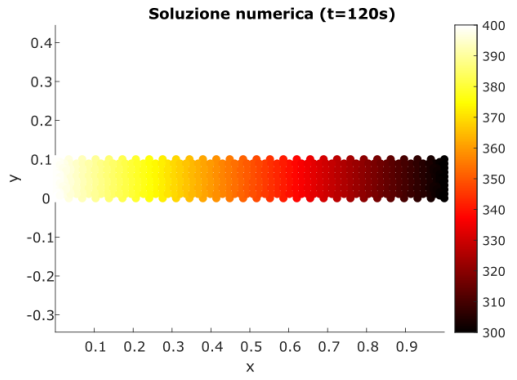


Figura 4.13: Soluzione numerica

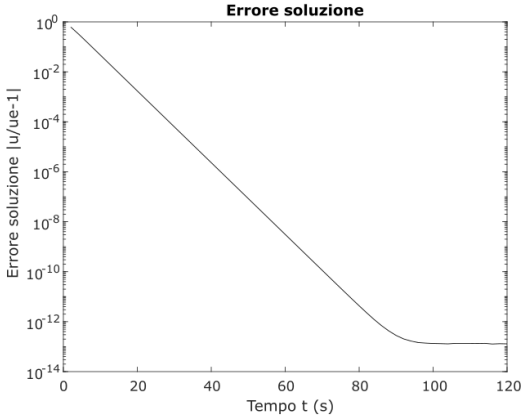


Figura 4.14: Andamento temporale errore

Come da aspettative la soluzione trovata è esatta entro l'errore di macchina (errore relativo $\sim 10^{-15}$); del resto il Direct GFDM implementa uno schema accurato al II ordine.

4.3.3 Equazioni elasticità

Il test più semplice per un problema meccanico è senza dubbio l'asta soggetta ad un carico assiale. La prova è condotta in controllo di spostamento, perciò le condizioni al contorno sono:

- Spostamento nullo al lato sinistro $u = 0, v = 0$;
- Spostamento imposto al lato destro $u = 1\ mm, v = 0$;
- Trazione nulla nel resto del contorno $t_x = 0, t_y = 0$;

Il materiale invece è una tipica lega di alluminio con $E = 68\ GPa$ e $\nu = 0.33$.

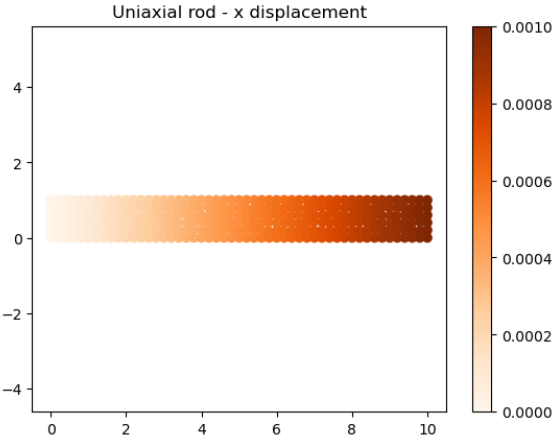


Figura 4.15: Soluzione numerica - spostamento

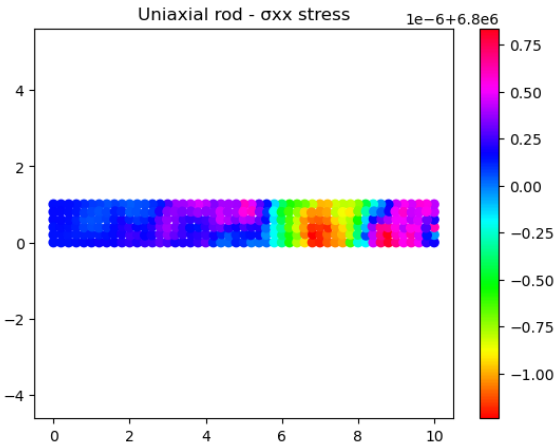


Figura 4.16: Soluzione numerica - stress

La soluzione numerica coincide perfettamente con il risultato fornito dalla teoria di Eulero-Bernoulli sia nello spostamento $u(x)$ sia nella tensione σ_{xx} entro l'errore di macchina. Anche qui infatti i campi sono costanti o lineari e quindi riproducibili esattamente dal metodo Direct GFDM stazionario.

Si decide allora di complicare leggermente il load case considerando una trave a sbalzo soggetta a flessione per effetto di un carico distribuito costante:

- Spostamento nullo al lato sinistro $u = 0$, $v = 0$;
- Carico distribuito verticale sul lato alto $t_x = 0$, $t_y = q = 1e5 N/m^2$;

Il materiale è la solita lega di alluminio 7075 con $E = 71.7\ GPa$ e $\nu = 0.33$:

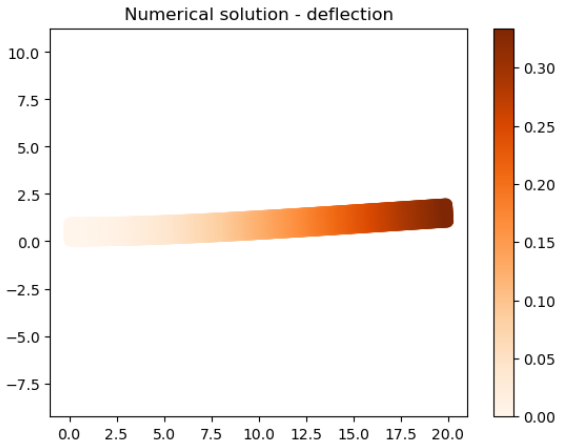


Figura 4.17: Soluzione trave deformata

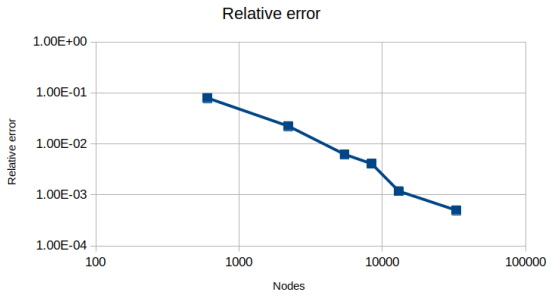


Figura 4.18: Errore sullo spostamento verticale massimo v_{tip} al variare del numero di nodi

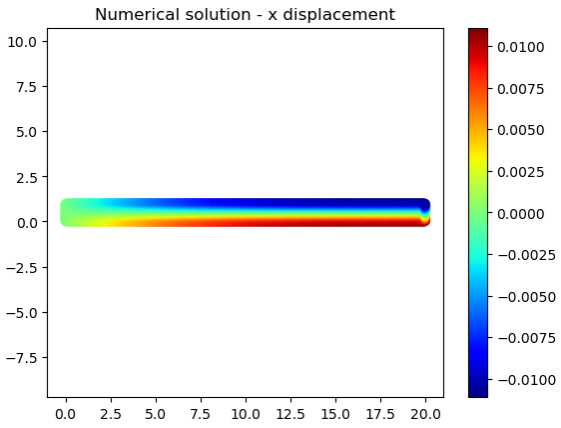


Figura 4.19: Soluzione numerica - spostamento in direzione x

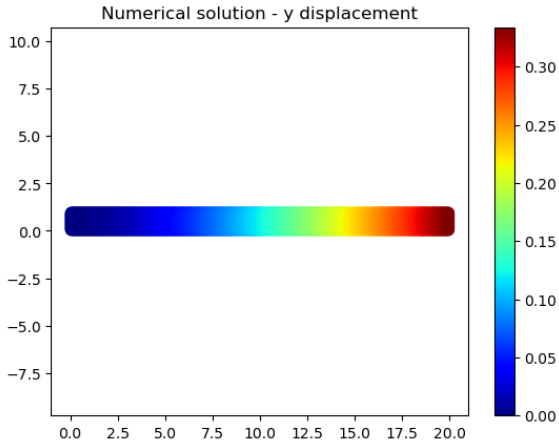


Figura 4.20: Soluzione numerica - spostamento in direzione y

Si vede chiaramente come la simulazione converga alla soluzione analitica data dalla teoria della trave di Eulero-Bernoulli:

$$v_{tip} = \frac{qL^4}{8EI} = 0.3347\ m$$

Dove $I = \frac{1}{12}h^4 = 0.0833\ m^4$ è il momento di inerzia di area della sezione quadrata.
Anche i campi di spostamento $u(x, y)$ e $v(x, y)$ corrispondono perfettamente a quelli ottenuti mediante FEM, vicino e lontano dal vincolo, ad ulteriore conferma della validità del metodo.

Si consideri adesso il classico test case del pannello infinito piano con foro centrale e soggetto ad un carico di trazione nel piano. Sfruttando le simmetrie del problema è possibile ridurre il dominio di calcolo e simulare solamente il quadrante in alto a destra.
Ovviamente non è possibile discretizzare un dominio infinito, e per limitarlo vengono fatti due tentativi con diversa estensione: 5×5 e 10×10.
Seguendo il lavoro di Oñate et al,[10] il materiale ha le seguenti proprietà: $E = 1000\ Pa$, $\nu = 0.3$. Invece le condizioni al contorno sono:

- Spostamento orizzontale nullo al bordo sinistro $u(0, y) = 0$;
- Spostamento verticale nullo al bordo inferiore $v(x, 0) = 0$;
- Carico orizzontale unitario al bordo destro $t_x = \sigma_0 = 1\ Pa$;
- Trazione nulla nel resto del contorno $t_x = 0$, $t_y = 0$;

La geometria è non banale e la nuvola di punti ha distribuzione irregolare, più raffinata in prossimità del foro e più diradata lontano dai bordi. Per evitare ambiguità nell'assegnazione delle normali, i nodi degli spingoli vengono rimossi.

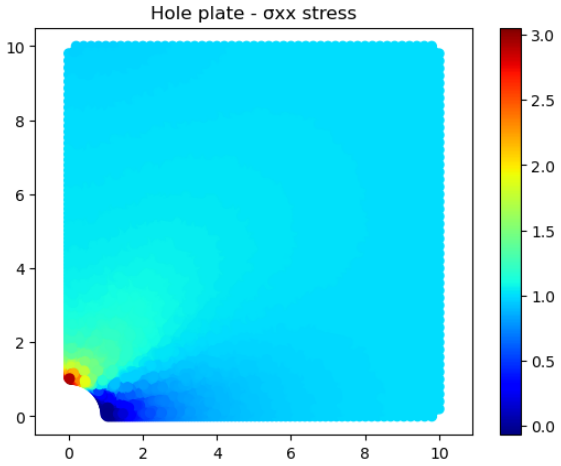


Figura 4.21: Soluzione numerica - stress σ_{xx} su dominio 10×10

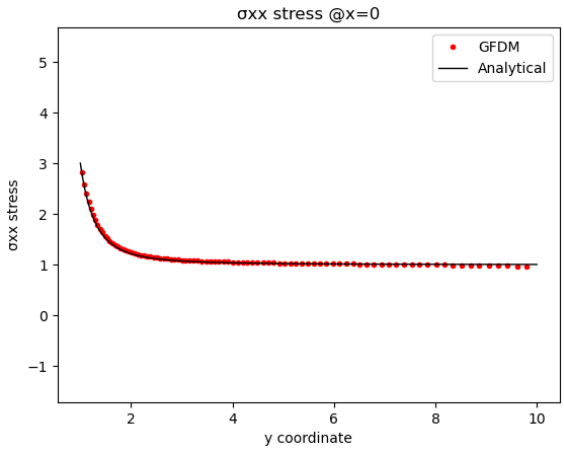


Figura 4.22: Confronto andamento $\sigma_{xx}(x = 0, y)$ numerico e analitico

Il problema ammette una soluzione analitica, utile per studiare l'errore commesso dal metodo:

$$\sigma_{xx}(r, \theta) = \sigma_0 \left[1 - \frac{a^2}{r^2} \left(\frac{3}{2} cos(2\theta) + cos(4\theta) \right) + \frac{3}{2} \frac{a^4}{r^4} cos(4\theta) \right]$$

La piastra di dimensioni 5×5 non produce risultati quantitativamente accettabili, pur riuscendo ad individuare in modo qualitativo l'andamento della soluzione. Anche nel caso 10×10 l'andamento non è perfetto, ma accettabile. In ogni caso all'infittire della griglia l'errore cala:

Nodi	$RMSE(\sigma_{xx})$	$\kappa_2(\mathbf{M})$
1158	5.19e-2	1.88e5
2262	2.35e-2	5.40e5
3198	1.65e-2	8.66e5

Ovviamente con un dominio finito l'errore non può tendere a zero. Per migliorare l'accuratezza sarebbe opportuno estendere ulteriormente le dimensioni della piastra, a prezzo però di un costo computazionale più pesante.
Analizzando invece il condizionamento del sistema sparso finale (matrice di rigidezza $\mathbf{M}$), si osserva un andamento crescente con il numero di nodi. Pur non costituendo un problema per

[10] Oñate, Perazzo e Miquel, "A finite point method for elasticity problems", cit.

questo test case, sarà bene monitorarlo attentamente durante le simulazioni più grandi nel prosieguo di questo lavoro.

Nota che in questo test case la validazione riguarda le tensioni, mentre il problema elastico lineare viene risolto con gli spostamenti come incognite. Come visto nel capitolo precedente è necessario fare un opportuno post-processing con gli stencil **C**.

4.3.4 Equazioni termo-meccaniche

Come visto in precedenza, i problemi termo-meccanici con accoppiamento debole analizzati in questo lavoro non sono altro che problemi lineari elastici con un carico volumetrico e condizioni al contorno leggermente modificate.
Procedendo per gradi, viene esaminato il problema banale di un'asta libera dilatata per effetto di una temperatura costante. Le dimensioni sono $L = 20\ m$, $H = 1\ m$ e il materiale è una lega di alluminio con $\alpha = 23.6\ \mu m/(m\ K)$. Le condizioni al contorno sono semplicemente:

- Incastro al lato sinistro $u = 0$, $v = 0$;
- Temperatura fissata al lato sinistro e al lato destro $T = 100\ °C$;

La temperatura di riferimento per lo zero dell'espansione termica invece è posto a $T_{ref} = 20\ °C$.

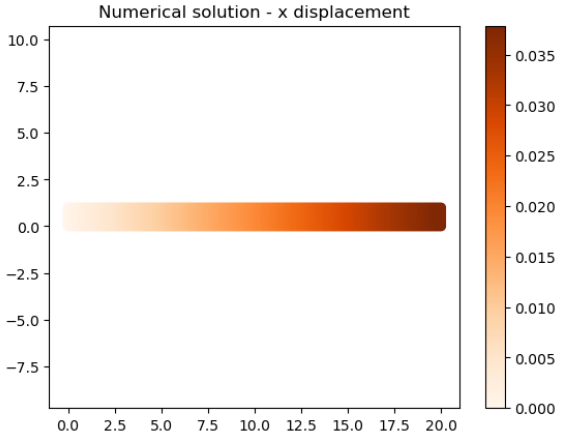


Figura 4.23: Soluzione numerica - spostamento in direzione x

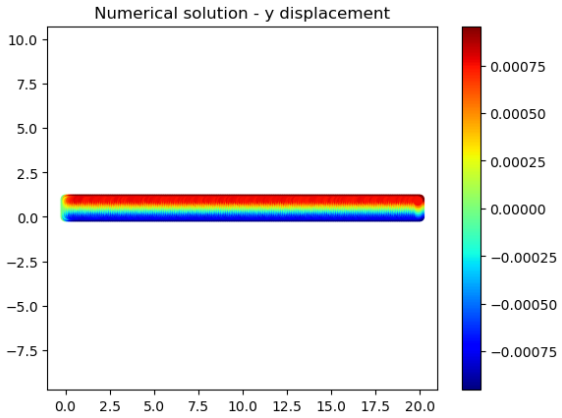


Figura 4.24: Soluzione numerica - spostamento in direzione y

La soluzione analitica è semplicemente $u_{tip} = L\alpha\Delta T = 0.03776\ m$, vale a dire $u(x)$ lineare. Il metodo Direct GFDM la prevede con ottima precisione, ma non in modo esatto per effetto del vincolo di incastro che fa nascere degli effetti locali. In ogni caso all'infittire della griglia e all'aumentare della snellezza della trave la soluzione arriva rapidamente a convergenza:

Altezza H	Nodi	u_{max}	v_{max}	Errore relativo
1.0	306	0.037884	0.001446	0.0033
1.0	1111	0.037859	0.000956	0.0026
1.0	5457	0.037851	0.000964	0.0024
1.0	8421	0.037851	0.000959	0.0024
0.1	4005	0.037776	0.001833	0.0004
0.1	6006	0.037772	0.000332	0.0003

Anche con una griglia estremamente grezza l'errore relativo si mantiene molto basso, attorno allo 0.3%, come da aspettative per un caso così semplice.

Si consideri ora un test case un poco più complesso preso dai tutorial di Wolfram:[11] una trave a sbalzo è scaldata gradualmente nel tempo e, per effetto dei gradienti di temperatura, sviluppa al suo interno degli sforzi termici deformandosi.

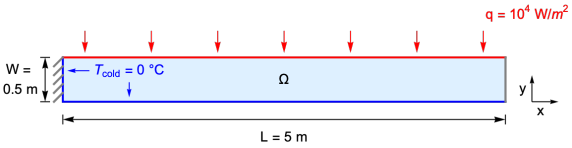


Figura 4.25: Illustrazione del problema (immagine presa dal riferimento)

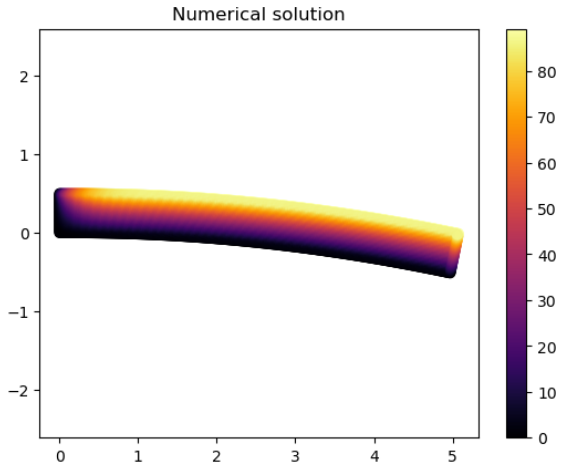


Figura 4.26: Soluzione numerica - campo di temperatura su geometria deformata

Rispetto al caso precedente qui sono presenti dei forti gradienti di temperatura e la simulazione è transitoria. L'implementazione quindi viene modificata e prevede l'esecuzione del modello termico e del modello meccanico ad ogni timestep, sempre in sequenza in modo disaccoppiato. Il materiale è una lega di acciaio avente le seguenti proprietà termomeccaniche:

- Modulo di Young $E = 210\ GPa$;
- Coefficiente di Poisson $\nu = 0.3$;
- Densità $\rho = 7500\ kg/m^3$;
- Coefficiente di espansione termica $\alpha = 12\ \mu m/(m\ K)$;
- Conducibilità termica $k = 44\ W/(m\ K)$;
- Calore specifico $c_p = 470\ J/(kg\ K)$;

Per le condizioni al contorno invece fare riferimento alla figura in alto a sinistra.
Come condizioni iniziali si prendono $T(x, y) = 0\ °C$, $u(x, y) = 0\ m$ e $v(x, y) = 0\ m$. Esse vengono fatte evolvere nel tempo con il metodo di Eulero Implicito a step costanti $\Delta t = 60\ s$ fino all'istante finale $t_f = 3\ h = 10800\ s$. La soluzione numerica così ricavata è confrontata con i risultati di riferimento forniti da Wolfram:

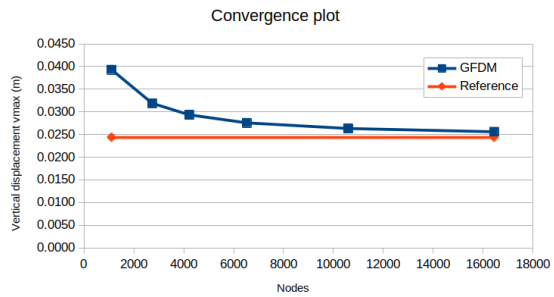


Figura 4.27: Grafico di convergenza all'infittire della griglia

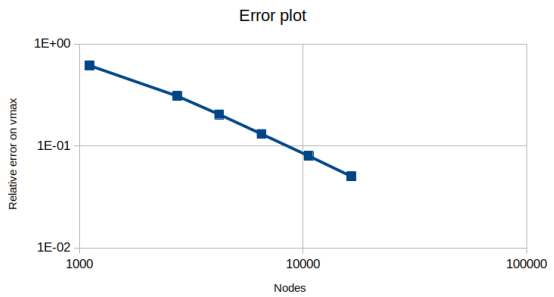


Figura 4.28: Andamento errore su v_{max} all'infittire della griglia

[11] Wolfram Documentation Center. URL: `https://reference.wolfram.com/language/PDEModels/tutorial/Multiphysics/ModelCollection/ThermalLoad.html`.

Come da aspettative il metodo Direct GFDM converge con successo alla soluzione di riferimento. A partire dagli errori e dalla dimensione ella griglia è immediato calcolare l'ordine di convergenza:

$$r = \frac{log(e^{k+1}/e^k)}{log(h^{k+1}/h^k)}$$

Ovviamente la formula è valida solo quando il parametro meshSize è infinitesimo; però all'infittire della griglia si vede chiaramente come l'ordine di convergenza tenda a 2:

Nodi	meshSize	T_{max}	u_{max}	v_{max}	Errore relativo	Ordine convergenza
1111	0.05	91.3301	0.0048	0.0393	0.6112	-
2737	0.03125	89.4717	0.0042	0.0319	0.3076	1.4608
4221	0.025	89.2410	0.0040	0.0293	0.2029	1.8640
6526	0.02	89.1827	0.0039	0.0276	0.1305	1.9773
10593	0.015625	89.1654	0.0038	0.0263	0.0796	2.0036
16441	0.0125	89.1631	0.0038	0.0256	0.0502	2.0683

Dalla tabella soprastante però si osserva che lo spostamento in direzione x differisce in modo sostanziale dalla soluzione di riferimento $u_{max} = 0.0049\ m$.

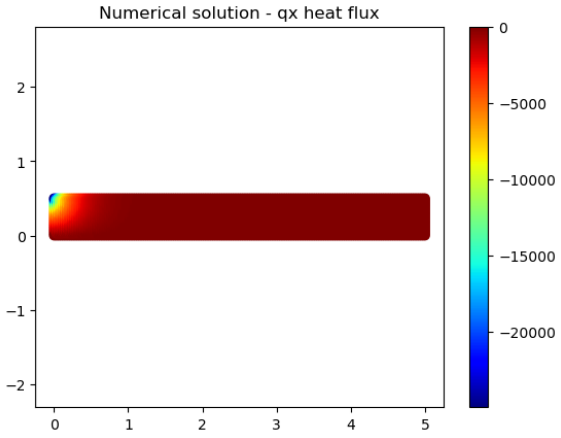


Figura 4.29: Soluzione numerica - flusso di calore q_x in direzione x

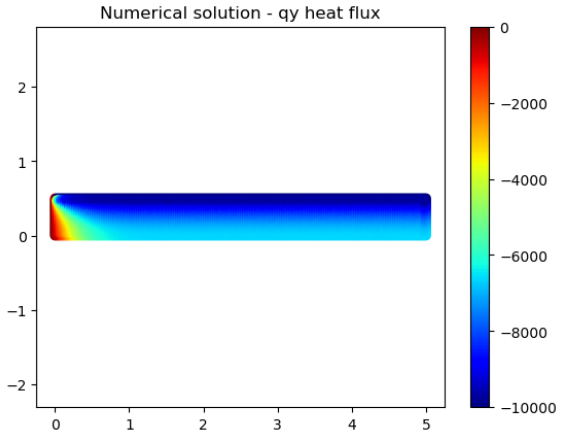


Figura 4.30: Soluzione numerica - flusso di calore q_y in direzione y

Analizzando più in dettaglio la distribuzione dei flussi di calore all'istante finale t_f, si vede come $q_y = -k\frac{\partial T}{\partial y}$ sia molto forte su tutta la trave, e infatti induce uno spostamento verticale considerevole. Invece il flusso in direzione x è nullo ovunque tranne che in prossimità dell'angolo superiore sinistro. Qui infatti è presente una singolarità tra la temperatura T_{cold} del bordo destro e il flusso q entrante dal bordo superiore. Mentre soluzione FEM è in grado di catturare bene la cosa, l'approccio ai minimi quadrati del Direct GFDM costituisce uno svantaggio da questo punto di vista.
Comunque è bene specificare che si tratta di un effetto secondario nella soluzione complessiva e che è risolvibile modificando il metodo opportunamente. Vista la poca rilevanza del fenomeno, si è deciso di non alterare l'implementazione e di proseguire con i metodi presentati.

4.4 Comparazione varianti GFDM

Nel capitolo precedente sono state introdotte diverse varianti al metodo GFDM classico: metodo Direct GFDM, metodo MDLSM, GDL spettrali. Ciascun approccio ha i suoi pro e i suoi contro, ma le proprietà di discretizzazione sono piuttosto simili; il metro di paragone deve invece guardare la stabilità e il condizionamento.
A tale proposito si considera il semplice caso di conduzione termica 2D stazionaria lineare su asta rettangolare:

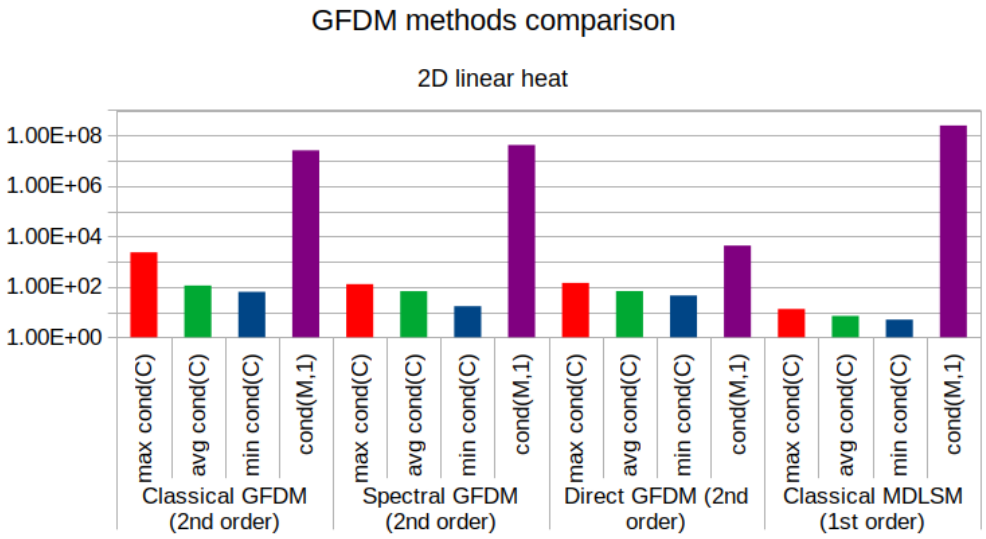


Figura 4.31: Comparazione condizionamento varianti GFDM

Il metodo classico è estremamente sensibile a piccole perturbazioni sulla nuvola di punti: $\kappa_1(\mathbf{M})$ oscilla tra due ordini di grandezza e $max(\kappa_2(\mathbf{C}))$ esplode rispetto al valore medio tra tutti i nodi. L'introduzione di gradi di libertà spettrali come da aspettative rende il metodo molto più stabile e il condizionamento dello stencil più basso. In particolare, il valore medio è ridotto alla metà e il valore massimo è abbattuto di un ordine di grandezza, un'enormità per un caso così semplice. Il sistema sparso finale però ha un condizionamento praticamente inalterato, anche se meno variabile con le perturbazioni.
La variante Direct ha un condizionamento degli stencil del tutto equivalente a quello dello Spectral GFDM. Ciò non sorprende dal momento che i contributi stabilizzanti sono molto simili tra i due metodi. La peculiarità e il grande vantaggio dell'approccio Direct sta nella costruzione e nel condizionamento del sistema sparso finale: qui $\kappa_1(\mathbf{M})$ è abbattuto di ben 4 ordini di grandezza rispetto ai metodi precedenti.

La variante MDLSM invece va analizzata a parte date le sue peculiarità. Il fatto di poter usare un'approssimazione al I ordine ha molte conseguenze, sia sulla stabilità sia sulla convergenza. Ovviamente il condizionamento dello stencil è il migliore possibile: con soli 5 nodi vicini $\kappa_2(\mathbf{C})$ è addirittura dell'ordine dell'unità, senza introduzione di GDL spettrali!
La cosa però va a scapito del sistema sparso finale, che ha un numero di incognite triplicato. Nonostante ciò $\kappa_1(\mathbf{M}) \sim 10^8$ è solamente un ordine di grandezza più grande rispetto al metodo classico, ma per casistiche più complesse ci si attende una differenza molto più marcata.
D'altro canto un ordine ridotto ha anche pesanti implicazioni sulla convergenza della soluzione numerica e in molti casi richiede una nuvola di punti molto fitta per ottenere risultati accettabili. In particolare, l'approssimazione dei flussi q_x e q_y è risultata molto poco accurata, specialmente su griglie grezze; in certi casi addirittura si vedono pattern a scacchiera, causati dallo stencil estremamente ridotto.

4.5 Generazione pointcloud

Come spiegato sin dall'inizio il vantaggio principale del metodo GFDM è la completa assenza di una mesh. La discretizzazione del dominio di calcolo avviene per opera di una nuvola di punti (pointcloud) senza alcuna connettività.
Proprio l'assenza di connettività tra nodi rende il metodo particolarmente flessibile e adatto a tecniche di raffinamento adattativo. In ogni caso, anche in presenza di raffinamento è necessario generare una distribuzione di nodi iniziale. Esaminando la letteratura presente sull'argomento, numerosi approcci sono possibili:

- Griglia cartesiana regolare;
- Griglia quadtree / octree;
- Distribuzione random;
- Metodo Flowmesher;
- Tecniche advancing front;

Molte di queste opzioni si ritovano all'interno degli algoritmi di generazione mesh, però l'assenza di connettività ne rende l'implementazione molto più semplice e veloce. Mancando la fase di triangolazione (o equivalenti), anche i requisiti di qualità sono più rilassati.

La **griglia cartesiana regolare** è certamente la tecnica di generazione della nuvola di punti più semplice. Pur avendo i pregi di essere estremamente veloce e parallelizzabile, purtroppo non è impiegabile con il metodo GFDM. Come visto in precedenza infatti il metodo soffre le situazioni con nodi allineati, perciò le griglie cartesiane sono di fatto limitate a casi 2D molto semplici e richiedono una piccola perturbazione random sulla posizione dei nodi.

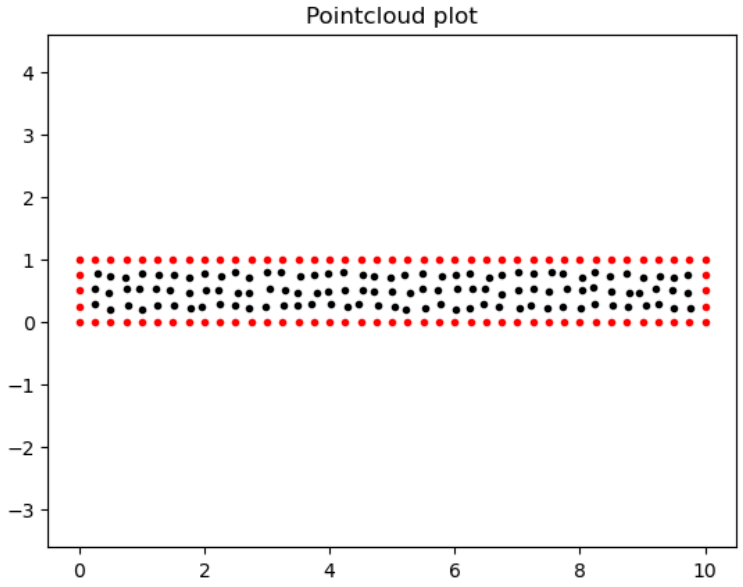


Figura 4.32: Esempio di pointcloud generata mediante griglia cartesiana

Una variante della griglia cartesiana regolare è la **tecnica quadtree** in 2D e octree in 3D. Essa prevede di raccogliere la geometria oggetto di analisi all'interno di un cubo di lato L e di suddividerlo in modo ricorsivo in 8 cubetti ognuno di lato $L/2$. La ricorsione può terminare quando all'interno di ciascun cubo non sono presenti più di uno o due punti appartenenti al contorno della geometria, oppure quando si raggiunge un livello massimo di profondità.
Un ulteriore criterio utile a migliorare la qualità della griglia è il bilanciamento 2:1. In questo lavoro una partizione quadtree/octree si dice bilanciata se ciascun blocco confina con blocchi di dimensioni 2x oppure 0.5x. Non ci devono essere blocchi vicini più grandi del doppio o più piccoli della metà.

Con un bilanciamento 2:1 la griglia si dirada in modo più regolare e non si hanno improvvise variazioni del fattore di espansione, come visibile nelle immagini seguenti:

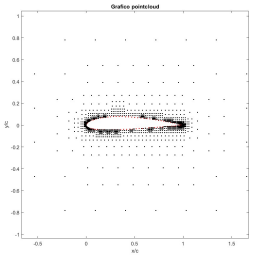


Figura 4.33: Esempio di pointcloud generata mediante tecnica quadtree bilanciata

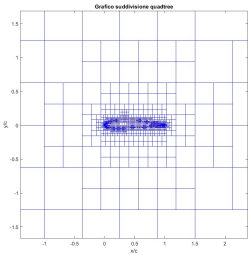


Figura 4.34: Illustrazione delle celle di una griglia quadtree bilanciata

Anche qui i nodi sono in molti casi allineati, ma la maggior flessibilità rispetto alla griglia cartesiana regolare fa sì che la tecnica quadtree/octree sia utilizzabile in uno spettro più ampio di casistiche 2D e anche in alcune geometrie 3D molto semplici.
Altri impieghi di una struttura dati di questo tipo si possono avere nella ricerca dei nodi vicini, come verrà accennato nel prossimo paragrafo, o nella determinazione di contatti/collisioni tra due oggetti. Per semplicità in questo lavoro tali aspetti non vengono approfonditi.

Un'alternativa molto semplice e comune per la generazione di una nuvola di punti è banalmente inserire punti generati casualmente all'interno del dominio in una **distribuzione random**. Purtroppo un approccio di questo tipo produce pointcloud dalla qualità molto scadente e difficilmente utilizzabili: i nodi sono troppo vicini o troppo lontani, la distribuzione è generalmente poco uniforme e piena di "buchi" (molto deleteri per il metodo GFDM).
Questi problemi in parte vengono risolti mediante approcci quasirandom. È possibile ad esempio costruire una struttura quadtree/octree e inserire punti random scegliendo la cella in modo casuale. Da semplici test condotti però si evince che anche per questa tecnica si ottengono risultati non ottimali, dalla qualità molto variabile e poco consistente.

Il **metodo Flowmesher**[12] invece è un algoritmo di generazione automatica in cui i punti vengono "iniettati" nel dominio e si respingono a vicenda fino a raggiungere una posizione di equilibrio. Le sorgenti (seed) che iniettano i nodi possono essere a loro volta punti posizionati casualmente, oppure è possibile andare a generare direttamente una nuvola di punti con distribuzione random.
Nell'implementazione in Julia viene scelta la seconda strada. Le particelle, con velocità iniziali nulle, vengono fatte propagare nel tempo mediante un semplice schema Leapfrog esplicito. Le forze considerate sono repellenti secondo una legge esponenziale con la distanza e sono calcolate considerando tutte le coppie di particelle vicine:

$$\mathbf{F}_{ij} = exp\left(-6.5\frac{\|\mathbf{P}_i - \mathbf{P}_j\|^2}{h_i^2}\right)\frac{\mathbf{P}_i - \mathbf{P}_j}{\|\mathbf{P}_i - \mathbf{P}_j\|}$$

I bordi del dominio sono hard-coded: se una particella esce viene immediatamente trasportata al suo interno con velocità normale invertita.
Per velocizzarne la convergenza, la velocità di ciascuna particella è dimezzata ad ogni iterazione; questo a tutti gli effetti agisce come un meccanismo di dissipazione.

[12] Zhujiang Wang et al. "FlowMesher: An automatic unstructured mesh generation algorithm with applications from finite element analysis to medical simulations". In: *CoRR* abs/2103.05640 (2021). URL: `https://arxiv.org/abs/2103.05640`.

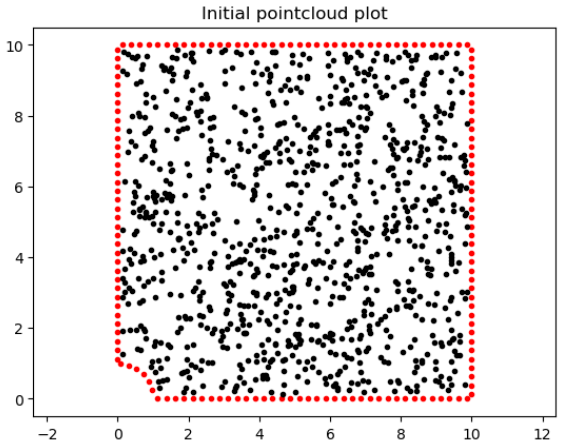


Figura 4.35: Pointcloud iniziale con distribuzione random

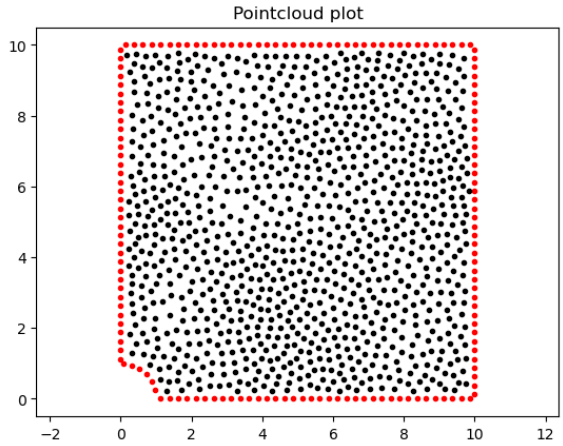


Figura 4.36: Pointcloud finale dopo 40 iterazioni Flowmesher

L'algoritmo Flowmesher ha un'implementazione molto semplice, ha una convergenza garantita, può gestire senza alcuna difficoltà nodi fissi, genera un pointcloud con il numero esatto di nodi assegnato dall'utente.
Purtroppo però la velocità di convergenza è molto lenta quando si hanno molte particelle. Un altro svantaggio è la presenza di alcune costanti che vanno tarate sulla base del problema considerato. Ne consegue che di fatto non è possibile impiegare l'algoritmo Flowmesher all'interno di simulazioni complesse ed è necessario cambiare tecnica.

La generazione della nuvola di punti mediante **Advancing Front** prevede l'inserimento successivo di nodi tenendo conto di una cloud density prescritta dall'utente. Un set di nodi presi dal contorno funge da sorgente per un layer interno, che a sua volta fa da sorgente per il layer successivo. La procedura continua in modo completamente automatico fino a quando l'intero dominio non è riempito. Una review di tali tecniche è fornita da Löhner et al,[13] ma per il presente lavoro si è iniziato rifacendosi ad un approccio più semplice e particolare (Biting method) presentato da Li et al:[14]

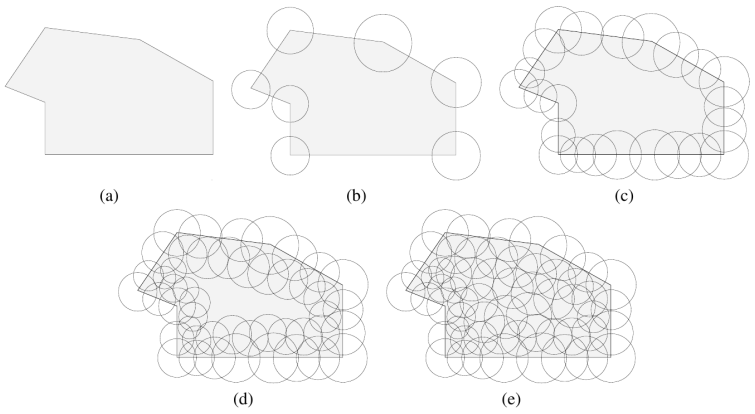


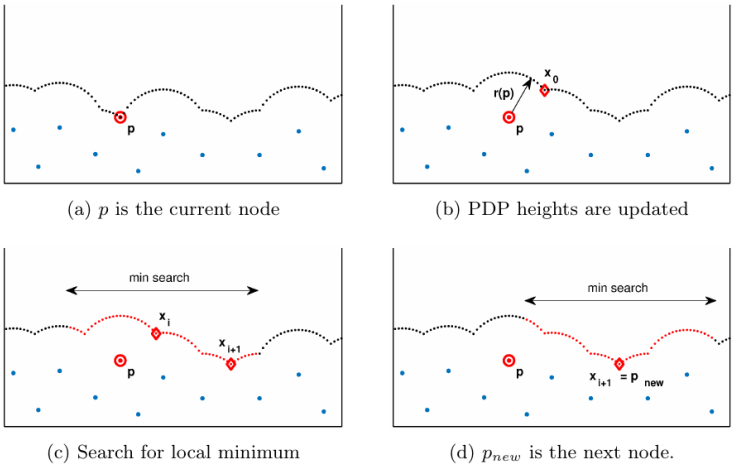
Figura 4.37: Illustrazione del funzionamento del Biting method. Immagine presa dal riferimento.

[13]Rainald Löhner e Eugenio Oñate. "Advancing front techniques for filling space with arbitrary separated objects". In: *Finite Elements in Analysis and Design* 46 (2010), pp. 140–151.

[14]Xiang-Yang Li, Shang-Hua Teng e Alper Üngör. "Point placement for meshless methods using Sphere packing and Advancing Front methods". In: 2001.

I nodi appartenenti al contorno del dominio fungono da fronte iniziale. Collocandoci delle sfere, il dominio viene "morsicato" diventando un set di segmenti ed archi. In corrispondenza delle intersezioni si collocano dei nodi, ove possibile. Essi vanno a costituire un nuovo fronte sul quale va ripetuta la procedura di biting fino ad esaurire il dominio.
Sulla carta il metodo è semplice e garantisce una buona qualità della nuvola di punti. L'implementazione però vede alcune criticità, in particolare sull'individuazione delle intersezioni e sull'estensione della formulazione in tre dimensioni.

In alternativa, il lavoro di Van Der Sande e Fornberg[15] propone un Advancing Front per riempire il bounding box del volume partendo dal basso e avanzando in altezza lungo l'asse z:



(a) p is the current node (b) PDP heights are updated

(c) Search for local minimum (d) p_{new} is the next node.

Figura 4.38: Illustrazione del funzionamento del metodo PDP. Immagine presa dal riferimento.

L'idea alla base dell'approccio è la stessa del Biting method. Qui però il fronte di avanzamento è definito da una griglia molto fitta di punti "virtuali", che partono dal bordo inferiore del dominio e procedono in altezza man mano che le sfere "mordono" il volume. Visto che la griglia è descritta da un vettore di altezze, $h(x)$ in 2D oppure $h(x, y)$ in 3D, il metodo prende il nome di "Potential Dot Placements" (PDP).
Tra i vantaggi del metodo PDP vi è la garanzia che i nodi siano correttamente spaziati, nonché la capacità di imporre zone più o meno raffinate semplicemente aggiustando il raggio delle sfere.
In questo modo vengono anche risolte le problematiche relative all'individuazione delle intersezioni, e l'estensione ai casi 3D è banale.

In letteratura sono ovviamente disponibili molte altre tecniche; si cita ad esempio la famiglia di algoritmi di particle packing che offre benefici importanti alle simulazioni SPH, come spiegato da Colagrossi et al.[16] In questo lavoro di tesi non viene però analizzata e ci si limita ai 5 approcci presentati.

[15]Kiera van der Sande e Bengt Fornberg. "Fast variable density 3-D node generation". In: *SIAM J. Sci. Comput.* 43 (2019), A242–A257. URL: `https://arxiv.org/abs/1906.00636`.

[16]Andrea Colagrossi et al. "Particle packing algorithm for SPH schemes". In: *Comput. Phys. Commun.* 183 (2012), pp. 1641–1653. URL: `https://www.openaccessrepository.it/record/22940`.

4.6 Selezione dei nodi vicini

I metodi meshless in genere lasciano molta libertà nella selezione dei nodi vicini. Come spiegato da Benito et al,[17] vari criteri sono possibili:

- Criterio della minima distanza, in cui vengono presi i primi N nodi più vicini. Si tratta di un approccio molto semplice e che può potenzialmente evitare l'approssimazione dei minimi quadrati; il problema è che spesso i nodi selezionati non sono ben distribuiti attorno al punto centrale e ciò porta a problemi di accuratezza se non addirittura di singolarità;
- Criterio dei quattro quadranti,[18] che si propone di trovare un buon compromesso tra distanza dei vicini e distribuzione a stella semplicemente selezionando i due nodi più vicini per ciascun quadrante. Esso però soffre quando la nuvola di punti è particolarmente irregolare, non garantisce di trovare un risultato soddisfacente, rende difficile cambiare il numero di vicini e ha un costo computazionale non indifferente per pointcloud dense. Spesso deve essere accoppiato con ulteriori criteri (es: minimo angolo tra due nodi vicini), ma allo stesso tempo è fondamentale evitare restrizioni eccessive;
- Criterio circolare, il più semplice da implementare. Tutti i nodi all'interno di un raggio R opportunamente definito vengono compresi nello stencil. Ovviamente per nuvole molto irregolari e vicino ai bordi il criterio potrebbe soffrire;
- Criterio del minimal positive stencil, proposto da Seibold,[19] prevede di sviluppare un calcolo di minimizzazione lineare in modo da trovare lo stencil di dimensioni minime tale da essere positivo. L'autore mediante argomentazioni matematiche dimostra che tale stencil esiste sempre tra quelli individuati da un criterio circolare, e che offre il miglior compromesso tra distanza e distribuzione. Il metodo GFDM beneficia enormemente da una procedura di questo tipo, ma il costo computazionale è importante visto che deve essere risolto un problema di Linear Programming per ciascun nodo del dominio con un metodo Simplex. L'autore da questo punto di vista fa notare che lo stencil minimo riduce il costo della successiva fase di risoluzione del sistema, bilanciando in parte l'onere maggiore;

In questo lavoro il criterio circolare è scelto per la sua semplicità e il suo basso onere computazionale. Le controindicazioni vengono affrontate come proposto da Suchde[20] nella variante Direct GFDM.

Ovviamente per trovare i nodi all'interno di una sfera di raggio R non è proponibile calcolare le distanze tra tutte le coppie di punti nel dominio. Per limitare il numero di calcoli richiesti viene costruita una griglia cartesiana di background sovrapposta al bounding box. Ciascuna cella ha lato $3 \times meshSize$ e solo i nodi all'interno della stessa cella (o di quelle vicine) sono analizzati.

In alternativa, un approccio quadtree/octree potrebbe essere implementato. Come trovato da Fernández-Fernández et al,[21] la maggiore efficienza può portare addirittura ad un dimezzamento dei tempi nella fase di ricerca.

[17] Benito et al., "Application of the Generalized Finite Difference Method to improve the approximated solution of pdes", cit.

[18] Augusto César Albuquerque Ferreira e Paulo Marcelo Vieira Ribeiro. "Reduced-order strategy for meshless solution of plate bending problems with the generalized finite difference method". In: *Latin American Journal of Solids and Structures* (2019). URL: https://doi.org/10.1590/1679-78255191.

[19] Seibold, "M-Matrices in meshless finite difference methods", cit.

[20] Suchde, "Conservation and Accuracy in Meshfree Generalized Finite Difference Methods", cit.

[21] José Antonio Fernández-Fernández et al. "Fast Octree Neighborhood Search for SPH Simulations". In: *ACM Transactions on Graphics (TOG)* 41 (2022), pp. 1 –13. URL: https://animation.rwth-aachen.de/publication/0579/.

Capitolo 5

Risultati

Dopo aver derivato i modelli, discusso il metodo numerico e descritto l'implementazione, è possibile procedere con la validazione e l'applicazione del codice alla simulazione di processo.

5.1 Test - sfera cava con generazione interna

Il primo test case è preso dal "validation manual" di Code_Aster e consiste in una semplice sfera cava soggetta ad una generazione volumetrica interna di calore.[1] Di seguito vengono riportati i parametri geometrici e fisici rilevanti:

- Raggio interno sfera $R_i = 1\ m$;
- Raggio esterno sfera $R_e = 2\ m$;
- Conducibilità termica $k = 1\ \frac{W}{mK}$;

Le condizioni al contorno prevedono solamente una temperatura imposta costante sulla superficie interna e sulla superficie esterna. Il carico volumetrico $q_v = 100\ W/m^3$ è anch'esso costante sull'intero volume:

- Temperatura imposta $T(r = R_i) = T(r = R_e) = 20\ °C$;

Grazie alle simmetrie presenti è possibile simulare soltanto una porzione di dominio di 30°. Sulle facce tangenziali vanno imposte condizioni di Neumann per impedire scambi termici. L'analisi svolta è stazionaria, non è richiesto specificare un campo di temperatura iniziale.

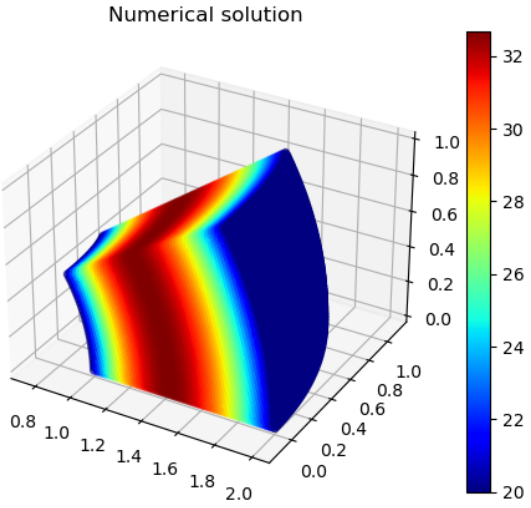


Figura 5.1: Soluzione numerica

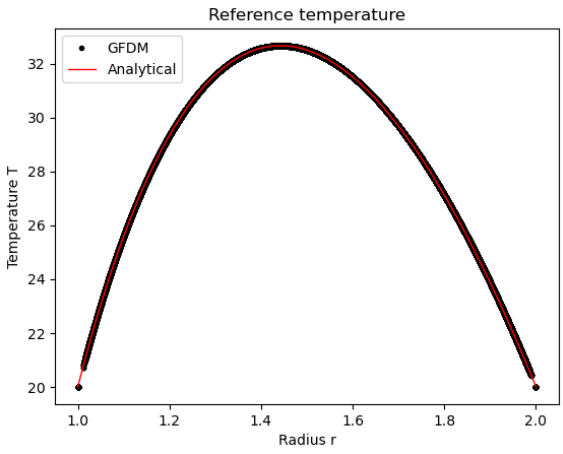


Figura 5.2: Profilo radiale di temperatura

[1] Haelewyn Jessica. *TPLV06 - Release of power in a hollow sphere*. URL: `https://code-aster.org/V2/doc/v14/en/man_v/v4/v4.04.006.pdf`.

È disponibile la soluzione analitica di questo problema:

$$T(r) = T_i + \frac{q_v}{6k}\left[\frac{(R_e^2 - R_i^2)\left(\frac{1}{R_i} - \frac{1}{r}\right)}{\frac{1}{R_i} - \frac{1}{R_e}} - (r^2 - R_i^2)\right]$$

Il profilo di temperatura quindi è una semplice parabola in direzione radiale. Pur essendo il metodo accurato al II ordine, esso non è in grado di risolvere in modo esatto questo test case dal momento che l'equazione del calore è scritta in coordinate cartesiane. In ogni caso con pointcloud di 6084, 23615, 70326 nodi il metodo Direct GFDM converge correttamente alla soluzione analitica in perfetto accordo con le proprietà teoriche.

5.2 Test - dissipatore di calore

Il test case successivo è ispirato al "PrePoMax Tutorial No.21". Si tratta una semplice analisi termica 3D stazionaria di un dissipatore di calore:

- La superficie piana inferiore è soggetta ad un flusso di calore $q_b = 18382.4\ W/m^2$;
- Le alette invece sono soggette a convezione con $h = 23\ W/(m^2\ K)$ e $T_{amb} = 20\ °C$;

Il materiale invece ha conducibilità termica $k = 237\ W/(m\ K)$. Non è richiesta la conoscenza di altre proprietà per svolgere l'analisi stazionaria con il metodo Direct GFDM:

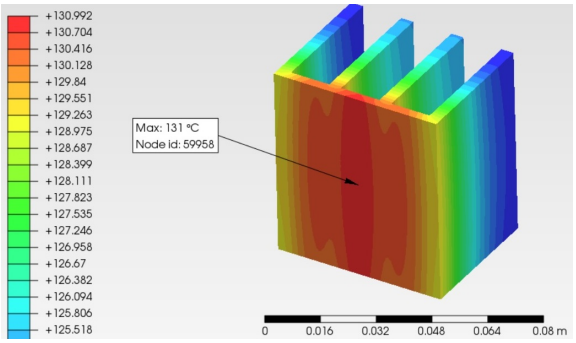


Figura 5.3: Soluzione FEM

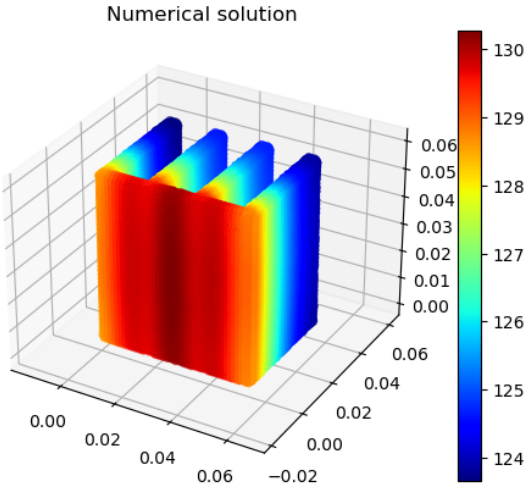


Figura 5.4: Soluzione numerica GFDM

Pur azzeccando il comportamento qualitativo della soluzione, l'implementazione ha riscontrato molti problemi di accuratezza. Parecchi tentativi e prove sono svolte per restringere la causa alla presenza di spigoli, dove il vettore normale alla superficie non è ben definito.
Rimuovendo gli spigoli dal pointcloud la soluzione diventa immediatamente coerente con i risultati predetti dal FEM. I nodi di bordo passano da 27382 a 26166 nel caso più grezzo, mentre i nodi interni rimangono 4700.
Attenzione che per certe combinazioni di parametri la convergenza può diventare molto lenta anche in assenza di spigoli; comunque l'errore non supera il 5% nello scenario peggiore.
Dal momento che questa problematica non compare in nessun altro test, la causa viene identificata nella geometria piuttosto snella in gran parte del dominio. I metodi a collocazione infatti tendono a soffrire stencil allungati, in particolare con nuvole di punti diradate. A conferma di ciò infittire la griglia va a migliorare la convergenza e a "sferificare" gli stencil.

Le attività di debugging non sono state vane e hanno permesso di studiare più in dettaglio le reali proprietà numeriche del metodo Direct GFDM:

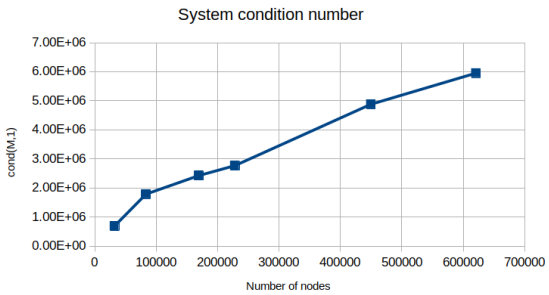


Figura 5.5: Numero condizionamento al variare del numero nodi

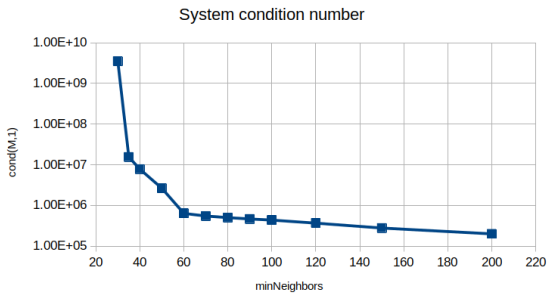


Figura 5.6: Numero condizionamento al variare di minNeighbors

Si osserva che il condizionamento del sistema finale cresce con il numero di nodi ma in modo relativamente modesto; per il prosieguo di questo lavoro quindi il numero di punti analizzato non costituisce un reale problema da questo punto di vista. Da verificare invece è l'effetto sullo stencil.
Aumentare il numero di vicini in generale è benefico per il condizionamento della matrice sparsa, ma a prezzo di un costo computazionale maggiore. Senza andare verso valori estremi, passare da $minNeighbors = 35$ a $minNeighbors = 60$ fa abbassare $\kappa_1(\mathbf{M})$ di più di 3 ordini di grandezza. Questo inoltre significa che uno stencil con pochi nodi tende facilmente a diventare singolare e che a differenza di quanto proposto da Seibold[2] per il metodo classico, nell'approccio Direct GFDM non è desiderabile trovare lo stencil minimo.
Per il prosieguo di questo lavoro quindi viene scelto il numero di vicini minimo pari a 10 in 2D e 60 in 3D.

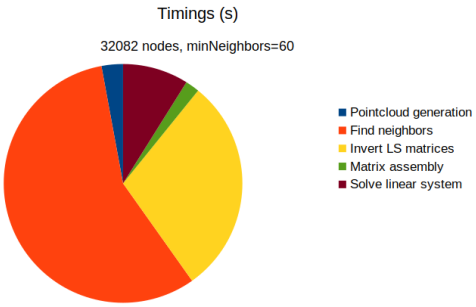


Figura 5.7: Tempo impiegato in ciascuna fase (32082 nodi)

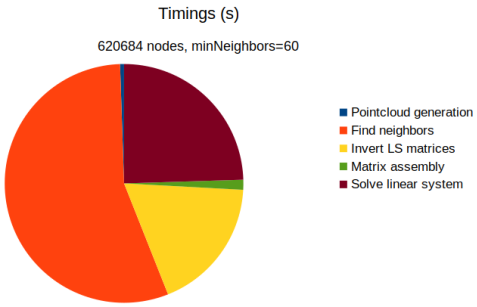


Figura 5.8: Tempo impiegato in ciascuna fase (620684 nodi)

Analizzando il costo computazionale di ciascuna fase del metodo, emerge chiaramente che la maggior parte del tempo è speso nella ricerca dei nodi vicini.
Anche l'inversione delle matrici dei minimi quadrati per calcolare gli stencil è estremamente impattante, soprattutto nei casi più grezzi. Per pointcloud fini il tempo di risoluzione del sistema sparso diventa sempre più importante, ma è anche quello in cui è più difficile agire.
Nota che questi grafici sono relativi ad uno script di debugging non ottimizzato in alcun modo: le variabili sono memorizzate globalmente, non viene usata alcuna procedura di parallelizzazione o vettorizzazione SIMD. L'esecuzione reale su OpenGFDM fa uso di un'altra implementazione e i risultati potrebbero essere molto differenti.

[2]Seibold, "M-Matrices in meshless finite difference methods", cit.

Passando ad uno schema del III ordine la convergenza è più rapida come da aspettative, ma il condizionamento è molto peggiore. Con $minNeighbors = 40$ si ha $\kappa_2(\mathbf{M}) \sim 10^{15}$ e addirittura non è possibile arrivare ad una soluzione, mentre con $minNeighbors = 60$ la situazione migliora ($\kappa_2(\mathbf{M}) \sim 10^8$) ma la griglia più grezza è molto inaccurata.

Imponendo le condizioni al contorno in modo esatto anziché nella procedura dei minimi quadrati la soluzione rimane invariata:

Nodi	Condizioni contorno	T_{max}	$\kappa_1(\mathbf{M})$
32082	Minimi quadrati	126.44 $°C$	6.91e5
32082	Esatte	126.44 $°C$	6.91e5

Questo risultato conferma la validità dell'approccio Direct GFDM, in cui le condizioni al contorno sono soddisfatte solamente in modo approssimato nel senso dei minimi quadrati. Infatti in questo caso particolare il residuo $e_{BC} = g_1 u + g_2 \vec{n} \cdot \nabla u - g_3$ è identicamente nullo su tutti i nodi del contorno. In altre casistiche in generale non è nullo ma quasi sempre trascurabile.

Altri test riguardano la funzione $w(r)$, che si vede influenzare il condizionamento del sistema sparso ma non il risultato, e l'effetto dei pesi w_{PDE} e w_{BC}, del tutto marginale (differenza su T_{max} inferiore a 0.1 $°C$).

5.3 Test - transitorio termico di un piatto perforato

Dopo due test statici si vanno a verificare le capacità transitorie del codice con un altro caso preso dal "validation manual" di Code_Aster.[3] La geometria in pianta del dominio è definita dall'intersezione di un triangolo equilatero con due circonferenze, come visibile in figura:

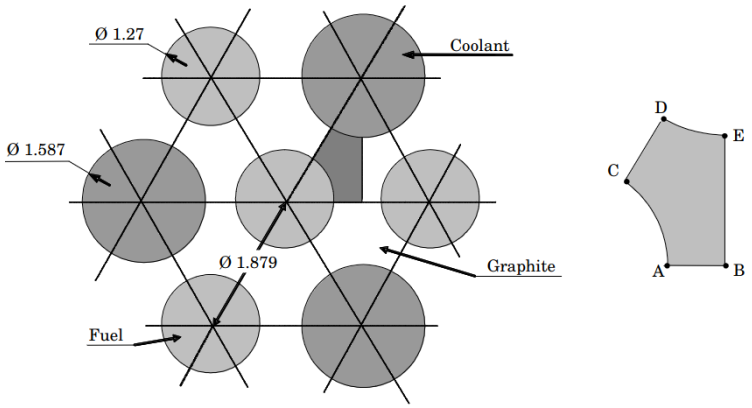


Figura 5.9: Forma in pianta del dominio (misure in cm). Immagine presa dal riferimento

Il materiale è caratterizzato dalle seguenti proprietà termiche:

- Conducibilità termica $k = 10\ W/(m\ K)$;
- Calore specifico volumetrico $\rho c_p = 1e6\ J/(m^3\ K)$;

Le condizioni al contorno invece prevedono:

- Flusso di calore entrante nel tratto AC $q_{AC} = 1e4\ W/m^2$;
- Convezione nel tratto DE con $h = 1e4\ W/(m^2\ K)$ e $T_{amb} = 0\ °C$;
- Scambio termico nullo su tutte le altre superfici $q = 0\ W/m^2$;

La condizione iniziale è $T(x, y) = 0\ °C$ all'istante $t = 0\ s$.

[3] Haelewyn Jessica. *TTLP301 - Transfer of heat in a perforated plate.* URL: `https://code-aster.org/V2/doc/default/en/man_v/v4/v4.23.301.pdf`.

La propagazione temporale vede l'impiego del metodo di Eulero Implicito con un timestep $\Delta t = 0.01\ s$ costante per un intervallo totale di $1.2\ s$, come specificato nel manuale di Code_Aster.

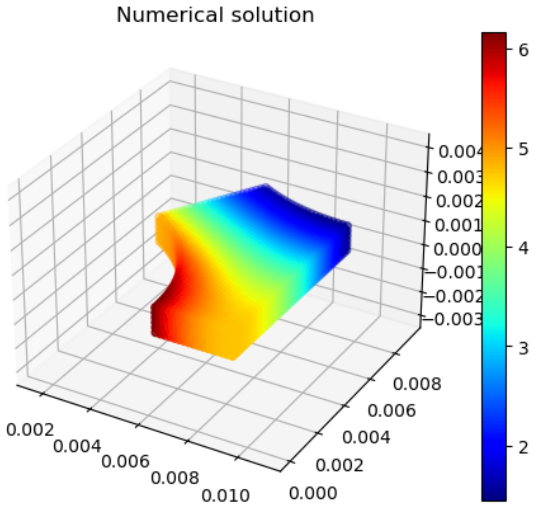


Figura 5.10: Soluzione numerica all'istante finale

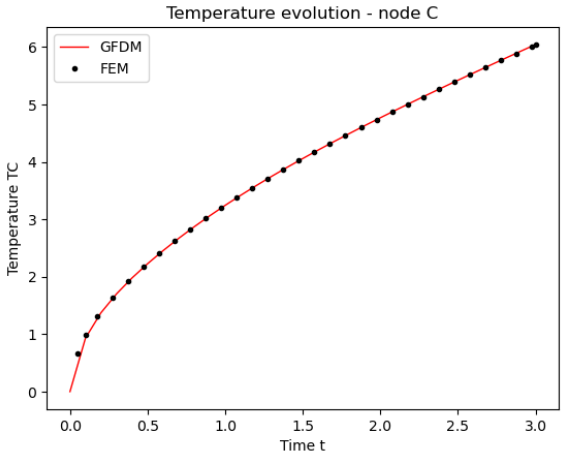


Figura 5.11: Andamento temperatura T_C nel tempo

Non è disponibile una soluzione analitica, il confronto va fatto con un'altra soluzione numerica ottenuta mediante FEM; il manuale propone di tracciare la temperatura del nodo C nel tempo e ne fornisce dei valori campionati ogni $0.1s$.
Come ben visibile dai grafici sopra, il metodo Direct GFDM è perfettamente consistente con i risultati di riferimento. Vengono eseguite diverse prove con pointcloud via via più fine (6238, 13140 e 38482 nodi), ma già con la griglia più grezza si ottengono risultati perfettamente sovrapponibili.

5.4 Test - raffreddamento oblong in plastica

Il test precedente è propedeutico alla simulazione termica del processo di stampa perché ne racchiude molti degli elementi tipici. Il suo superamento permette quindi di passare alla simulazione del raffreddamento di un tratto di plastica PET estrusa.
La geometria è un semplice oblong estruso di dimensioni $w = 0.55\ mm$, $h = 0.2\ mm$, $L = 1\ mm$ avente le seguenti proprietà termiche:

- Conducibilità termica $k = 0.24\ W/(m\ K)$;
- Densità $\rho = 1370\ kg/m^3$;
- Calore specifico $c_p = 1050\ J/(kg\ K)$;

Le condizioni al contorno vogliono simulare ciò che avviene durante la stampa:

- In basso il piatto riscaldato impone una temperatura $T = T_{bed} = 50\ °C$;
- In tutto il resto del contorno agisce un raffreddamento per convezione, con parametri stimati $h = 20\ W/(m^2\ K)$ e $T_{amb} = 20\ °C$;

Il materiale all'istante iniziale $t = 0\ s$ presenta una temperatura uniforme $T(x, y) = T_{extr} = 320\ °C$. L'obiettivo della simulazione è monitorare come il campo di temperatura evolve nel tempo. A tale proposito l'integrazione temporale è affidata al metodo di Eulero Implicito con timestep $\Delta t = 0.02\ s$ per un intervallo complessivo di $0.5s$.
I risultati naturalmente sono confrontati con quelli prodotti da una simulazione FEM, svolta con i software CalculiX e PrePoMax.

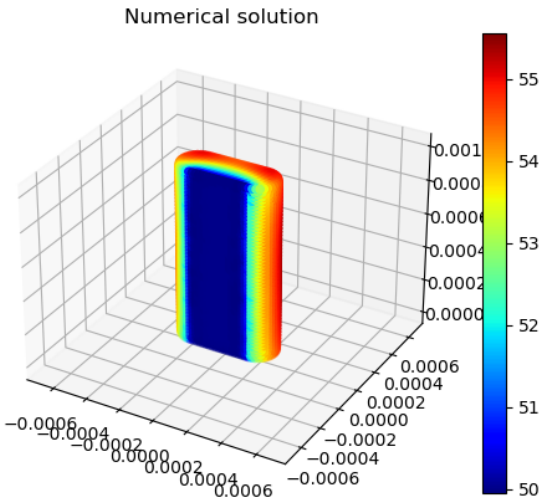


Figura 5.12: Soluzione numerica all'istante finale (14969 nodi)

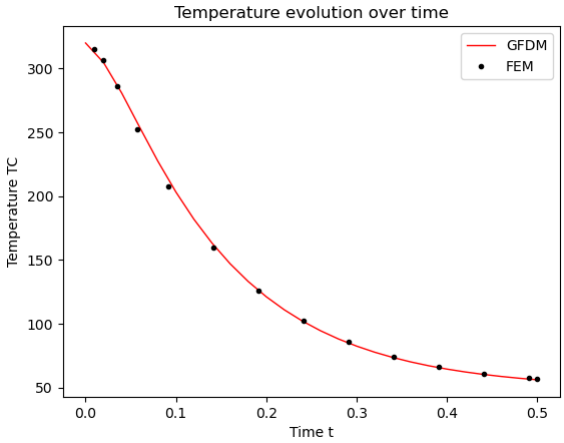


Figura 5.13: Andamento temperatura T_C nel tempo

Ancora una volta i risultati del codice Direct GFDM combaciano perfettamente con quelli prodotti dal FEM. Purtroppo però il metodo non riesce ad arrivare a convergenza se la nuvola di punti è troppo grezza; la decomposizione QR non è in grado di invertire la matrice R in quanto molto vicina alla singolarità. Questo problema in particolare si presenta con un pointcloud di 1613 nodi ed è assente con 14969 nodi.
Per un caso semplice come questo il maggior numero di punti non costituisce un problema, ma per una simulazione di processo reale avere un ordine di grandezza in più di nodi rende la trattazione impraticabile molto rapidamente.
Con la decomposizione SVD invece è possibile far girare anche il caso con 1613 nodi senza alcuna difficoltà, ovviamente accettando l'accuratezza un po' più bassa:

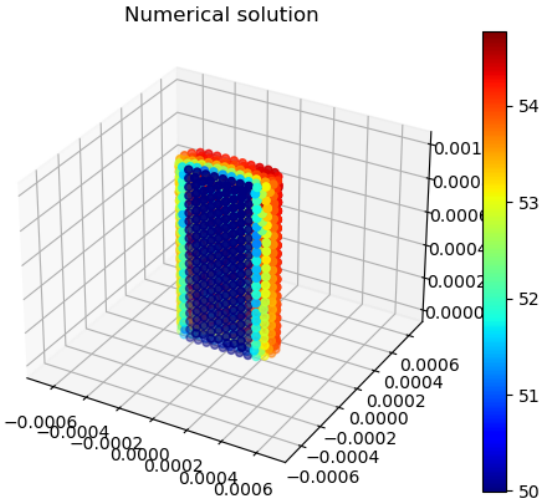


Figura 5.14: Soluzione numerica all'istante finale (1613 nodi)

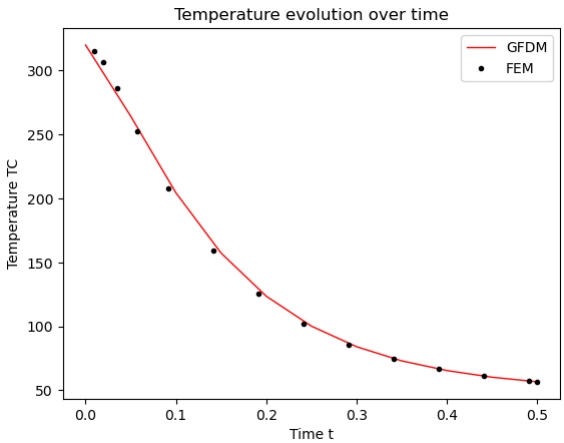


Figura 5.15: Andamento temperatura T_C nel tempo

Il vantaggio è il costo computazionale enormemente più basso; la fase di simulazione vera e propria infatti impiega solamente 3.4 s alzando il timestep a $\Delta t = 0.05$ s ed escludendo le fasi preparatorie, anch'esse dell'ordine di pochi secondi.
Per il prosieguo del lavoro quindi è d'obbligo adottare la decomposizione SVD.

5.5 Condizionamento stencil

Come detto sin dall'inizio il metodo GFDM in generale soffre di malcondizionamento. Si vuole adesso mostrare come questo evolve man mano che il problema da risolvere diventa sempre più complesso.
A tale scopo vengono estratti 4 casi dimostrativi con geometria del dominio molto simile:

- Calore 2D - 07_direct_2d_heat_neumann_steady.jl;
- Calore 3D - 17_3d_heat_filament.jl;
- Elasticità 2D - 18_2d_flexural_beam.jl;
- Elasticità 3D - 18b_3d_flexural_beam.jl;

Il primo ha una matrice di Vandermonde molto piccola, solo $(1 + N) \times 6$, mentre il secondo cresce a $(1 + N) \times 10$. Passando ai problemi di natura meccanica, il caso 2D ha una matrice grande $(2 + 2N) \times 12$ mentre il caso 3D esplode a ben $(3 + 3N) \times 30$ per i nodi interni.

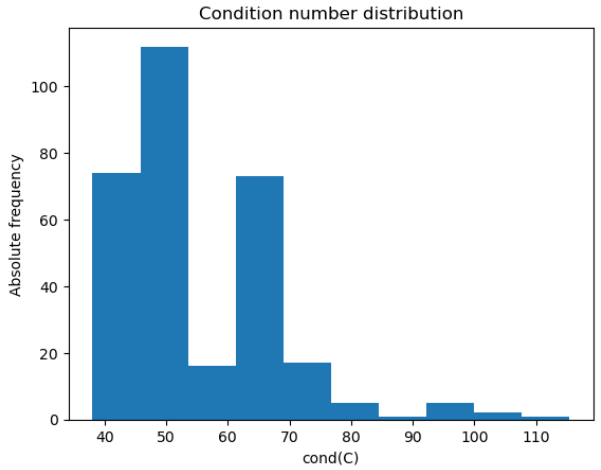


Figura 5.16: Distribuzione $\kappa_2(\mathbf{C})$ (metodo Direct GFDM, caso calore 2D)

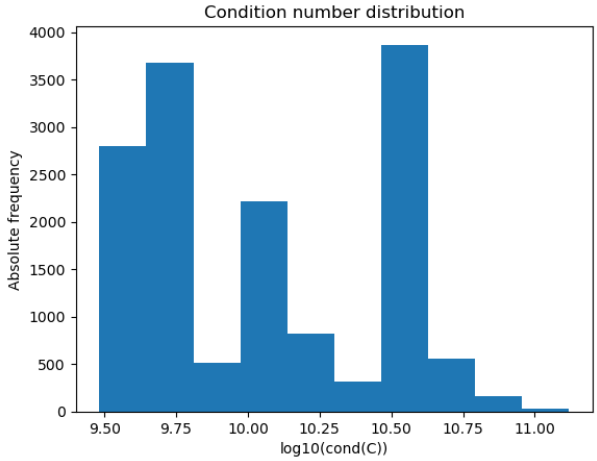


Figura 5.17: Distribuzione $\kappa_2(\mathbf{C})$ (metodo Direct GFDM, caso calore 3D)

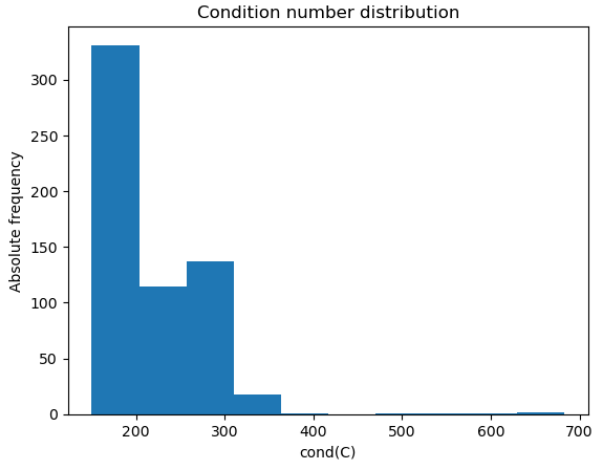


Figura 5.18: Distribuzione $\kappa_2(\mathbf{C})$ (metodo Direct GFDM, caso elasticità 2D)

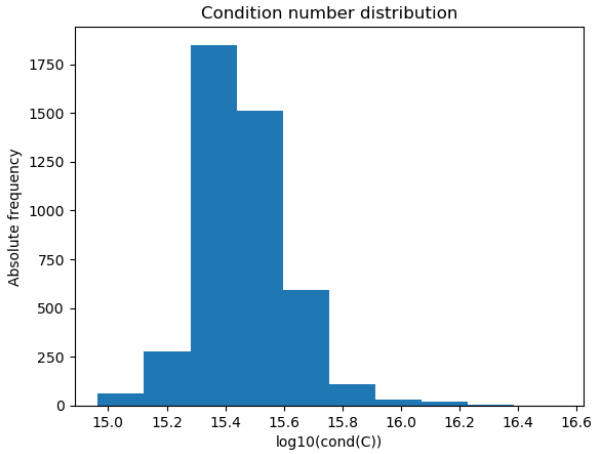


Figura 5.19: Distribuzione $\kappa_2(\mathbf{C})$ (metodo Direct GFDM, caso elasticità 3D)

Come da aspettative passando da due dimensioni a tre dimensioni il condizionamento dello stencil per il metodo Direct GFDM esplode di una decina di ordini di grandezza.

La sorpresa però è vedere il caso del problema elastico 3D rimanere relativamente contenuto se comparato con il problema termico nonostante il numero di righe e di colonne della matrice di Vandermonde triplicate.
Non si possono fare considerazioni sulla forma delle distribuzioni di $\kappa_2(\mathbf{C})$. Esse infatti variano fortemente con il raffinamento della griglia e soprattutto con il numero di nodi vicini. Per questi motivi le comparazioni sono svolte con parametri $minNeighbors$ e $meshSize$ costanti.

5.6 Test - trave a flessione

Si consideri lo stesso test case visto al paragrafo 4.3.3 di una trave elastica soggetta a carico flettente, ma esteso alle tre dimensioni. Le figure seguenti mostrano le componenti di spostamento di una semplice analisi stazionaria con 4459 nodi:

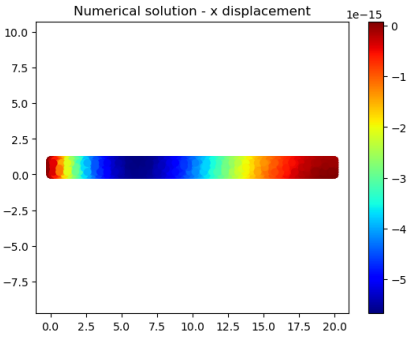


Figura 5.20: Soluzione numerica spostamento x

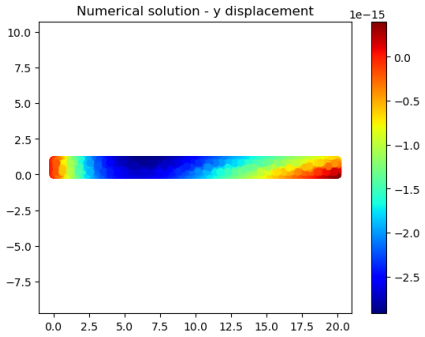


Figura 5.21: Soluzione numerica spostamento y

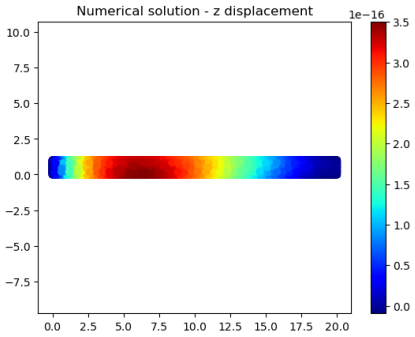


Figura 5.22: Soluzione numerica spostamento z

La soluzione è chiaramente errata e molto distante da quella esatta. Molti tentativi e reimplementazioni sono stati effettuati, purtroppo senza successo.
In tutti i casi si riscontrano problemi di convergenza simili a quelli visti nel test case dell'heatsink con metodo al III ordine con 40 nodi. Il principale indagato è certamente il malcondizionamento, che impedisce di ottenere una soluzione accettabile.

Date le dimensioni ridotte del problema in esame, si è scelto di applicare vari precondizionatori sinistri calcolando esplicitamente le matrici inverse $\mathbf{P}^{-1}$ e il numero condizionamento per capire più in dettaglio l'entità di tali difficoltà:

Precondizionatore	$\mathbf{P}$	$\kappa_1(\mathbf{M})$	v_{max}
Richardson	$\mathbf{P} = \mathbf{I}$	2.61E+19	1.79E-05
Jacobi	$\mathbf{P} = \mathbf{D}$	2.61E+19	1.05E-05
Gauss-Seidel	$\mathbf{P} = \mathbf{D} + \mathbf{L}$	1.40E+20	6.99E-05

Il precondizionamento quindi non sembra offrire alcun effetto tangibile, probabilmente a causa del condizionamento dello stencil $\mathbf{C}$ estremamente elevato che, popolando la matrice $\mathbf{M}$, a sua volta impedisce di costruire una matrice inversa $\mathbf{P}^{-1}$ in modo affidabile:

$$\mathbf{M}\vec{u} = \vec{b} \quad \rightarrow \quad \mathbf{P}^{-1}\mathbf{M}\vec{u} = \mathbf{P}^{-1}\vec{b}$$

Considerando invece un approccio più canonico all'applicazione dei precondizionatori, sono stati provati innumerevoli combinazioni tra solutori iterativi e strategie di decomposizione.
Usando i pacchetti IterativeSolvers.jl e Preconditioners.jl, infatti, Julia permette di integrare un ampio repertorio senza dover adattare il codice già scritto.

Tra i solutori iterativi provati si citano BiCGSTAB e GMRES, mentre per le decomposizioni si fa riferimento alle QR e LU implementate nel pacchetto LinearAlgebra.jl. I precondizionatori provati con tali solutori sono ILU e AMG, purtroppo anche qui senza successo. In tutti i casi non si riesce ad arrivare a convergenza.
L'unico modo per abbattere il condizionamento è trasformare il problema da stazionario a transitorio. Partendo dallo stato di quiete, la soluzione evolve nel tempo finché non raggiunge l'equilibrio statico. Il condizionamento del problema infatti migliora al diminuire del passo temporale Δt:

Passo temporale Δt	$\kappa_2\,(\mathbf{C}[1])$	$\kappa_1\,(\mathbf{M})$
0.1	1.58E+16	8.50E+18
0.01	1.58E+16	5.39E+17
1E-6	1.58E+16	6.14E+13

Purtroppo però il malcondizionamento è così severo da richiedere timestep estremamente bassi, del tutto impraticabili. Il metodo Direct GFDM quindi risulta inadatto a risolvere problemi elastici e termoelastici in tre dimensioni. Per il proseguimento di questo lavoro pertanto si è costretti a limitarsi a problemi termici 3d e problemi termomeccanici 2d.

5.7 Simulazione - stampa tratto rettilineo

I test case analizzati ai paragrafi precedenti sono preparatori alla simulazione del processo di stampa 3D, in particolare alla parte termica.
Per iniziare ci si limita ad un semplice tratto rettilineo lungo $L = 10\ mm$ estruso a step di $l_3 = 1\ mm$ da una testina che viaggia a $v = 40\ mm/s$. La geometria di ciascuno step e le proprietà del materiale sono le stesse viste al paragrafo 5.4:

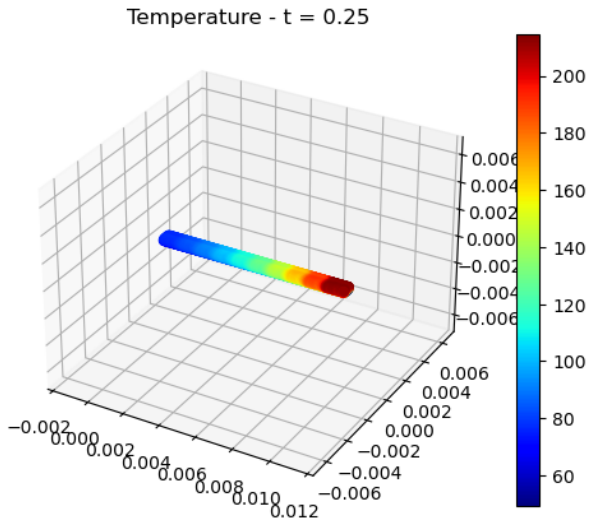


Figura 5.23: Soluzione numerica temperatura all'istante finale

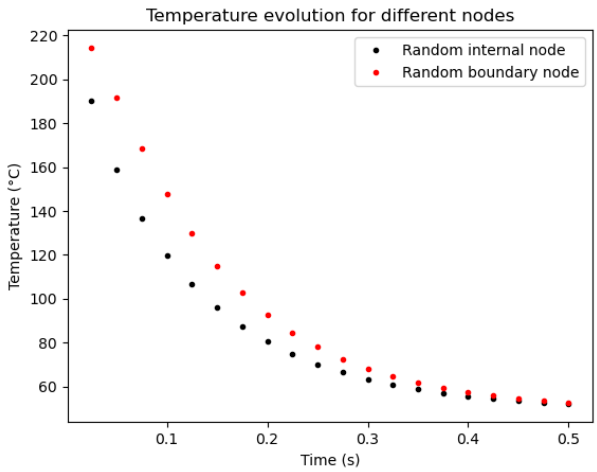


Figura 5.24: Andamento temperatura nel tempo di due nodi iniziali

Si osserva immediatamente che gli effetti diffusivi sono estremamente ridotti e che il driver principale che guida la simulazione è la temperatura del piatto fissata a 50 $°C$ per i nodi a contatto. Di fatto, ciascun tratto appare ad ogni timestep ad una temperatura di 230 $°C$ e si raffredda in modo indipendente dagli altri. Ciò emerge anche dal profilo di temperatura $T(t)$ di un generico nodo: l'andamento è del tutto identico a quello visto in precedenza per il

singolo tratto estruso, senza picchi dati dal nuovo materiale vicino. Inoltre un nodo interno si raffredda più velocemente di un nodo nel contorno superiore, in quanto più vicino al piatto. La convezione da questo punto di vista dovrebbe avere effetto contrario, ma risulta secondario.

Può essere interessante vedere cosa accade nel caso di deposizione multilayer: da un lato c'è del nuovo materiale caldo che va a scaldare nuovamente il layer sottostante, dall'altro vi è la cattiva conducibilità della plastica.

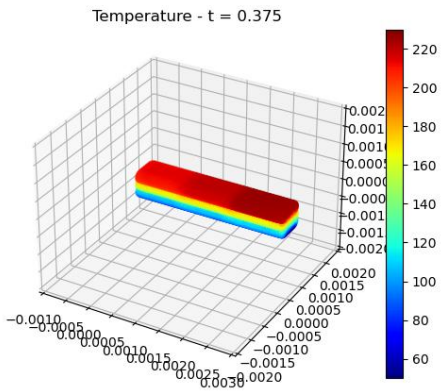


Figura 5.25: Soluzione numerica temperatura all'istante finale

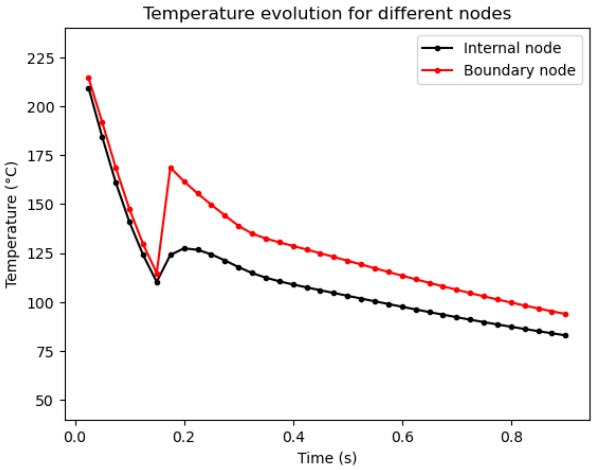


Figura 5.26: Andamento temperatura nel tempo di due nodi iniziali

In questo caso per semplicità si riduce la lunghezza del tratto stampato a $L = 3\ mm$. Come da aspettative il materiale appena estruso va a scaldare la plastica già depositata; l'effetto ovviamente è estremamente forte nei nodi appartenenti al contorno superiore e più debole per i nodi interni. Nel primo caso si ha un innalzamento di temperatura di addirittura 60 $°C$ in soli 0.025 s.

Dal profilo di temperatura si intravede anche il momento in cui il terzo layer viene depositato: sia il nodo al contorno sia il nodo interno infatti cominciano a raffreddarsi più lentamente, a causa della temperatura innalzata al secondo layer. Il fatto che la simulazione abbia catturato tale fenomenologia certamente va a confermare la correttezza del risultato.

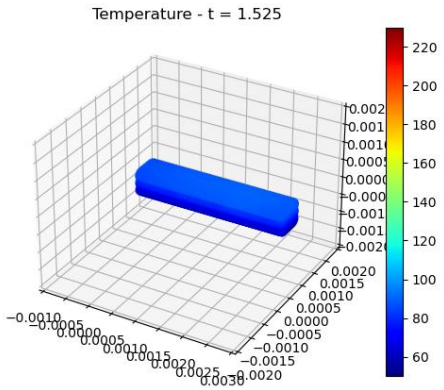


Figura 5.27: Soluzione numerica temperatura al termine del raffreddamento

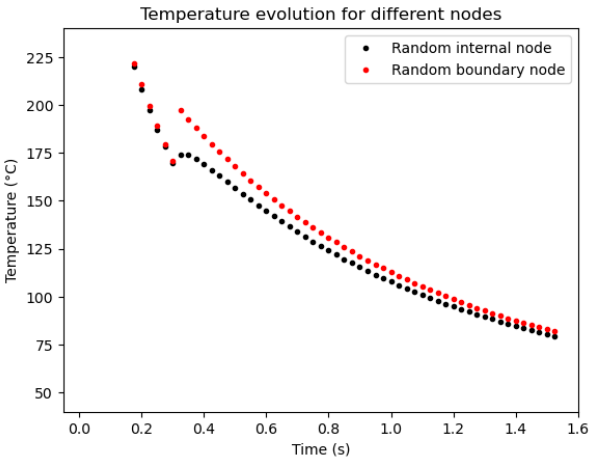


Figura 5.28: Andamento temperatura nel tempo di due nodi del secondo layer

A questo punto è naturale chiedersi cosa succede al variare dei parametri di stampa, ad esempio diminuendo la velocità di traslazione della testina di stampa.

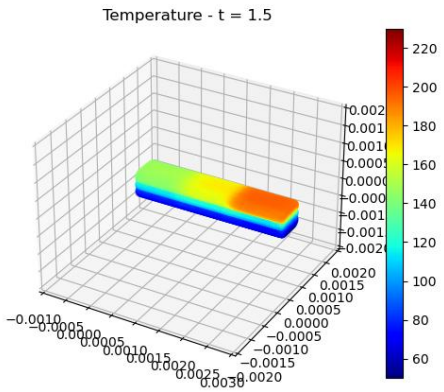


Figura 5.29: Soluzione numerica temperatura all'istante finale

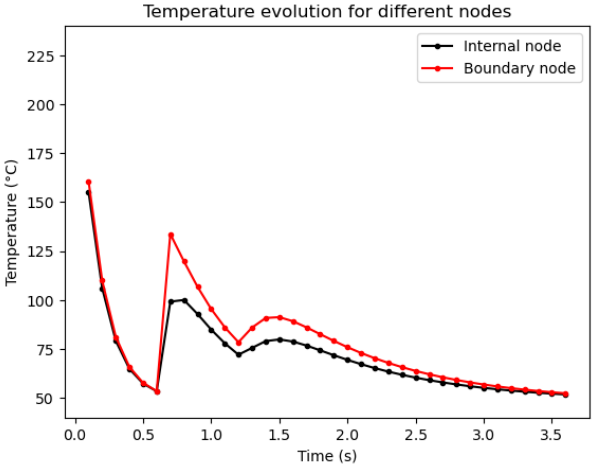


Figura 5.30: Andamento temperatura nel tempo di due nodi iniziali

Gli andamenti rimangono qualitativamente gli stessi ma le temperature si abbassano notevolmente per effetto dei tempi più lunghi. Gli sbalzi di temperatura risultano pertanto più intensi, come ben visibile dal profilo $T(t)$ di un generico nodo del I layer.
Per lo stesso motivo, dal grafico emerge chiaramente anche un riscaldamento del I layer dovuto alla deposizione del III layer più importante rispetto al caso precedente.
Si osserva anche come al termine della fase di raffreddamento la temperatura del nodo interno diventi più alta di quella del nodo del contorno superiore. A causa della convezione infatti i layer superiori tendono a cedere calore all'ambiente ($T_{amb} = 20\ °C$), quindi i nodi più lontani dal piatto raggiungono una temperatura di equilibrio più bassa.

Dal momento che il metodo Direct GFDM non consente la simulazione termomeccanica, non è possibile quantificare l'anisotropia delle proprietà meccaniche e l'adesione dei layer nei pezzi stampati. Ciononostante, facendo riferimento al lavoro di Khanafer et al,[4] è possibile definire una quantità detta "bonding potential" che descrive quanto i diversi layer sono saldi:

$$\varphi(x, y, z, t) = \int_0^t (T(x, y, z, t) - T_g)\, dt$$

Dove $T_g = 80\ °C$ è la temperatura di transizione vetrosa del PET.
Per ciascun nodo del dominio e per ciascun timestep si ha un valore di φ dato dall'area sottesa al grafico $T(t)$ e all'isoterma $T = T_g$. Il legame tra layer infatti si rafforza fintanto che la temperatura rimane superiore a quella di transizione vetrosa e si arresta quando scende al di sotto. Per effetto della deposizione dei layer successivi e l'incremento di temperatura per diffusione, si nota un significativo aumento del legame inter-layer nel tempo.

Una possibile applicazione di tale concetto potrebbe risiedere nella fase di post-processing, con la prospettiva di aggiustare la temperatura di stampa localmente e globalmente in funzione delle esigenze specifiche della parte.
Tornando al tratto rettilineo lungo $L = 10mm$, più rappresentativo delle stampe reali, si considerino tre diversi valori di velocità $v = 10\ mm/s, 20\ mm/s, 40mm/s$. Si vuole confrontare

[4]Khanafer et al., "Thermal analysis of fused deposition modeling process based finite element method: Simulation and parametric study", cit.

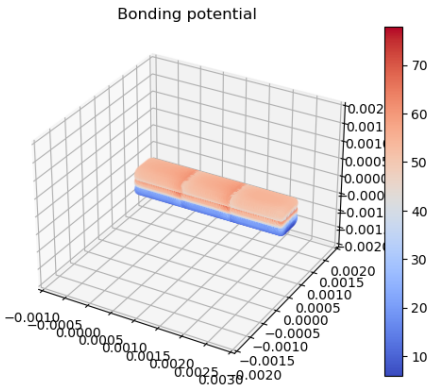


Figura 5.31: Bonding potential della soluzione all'istante finale

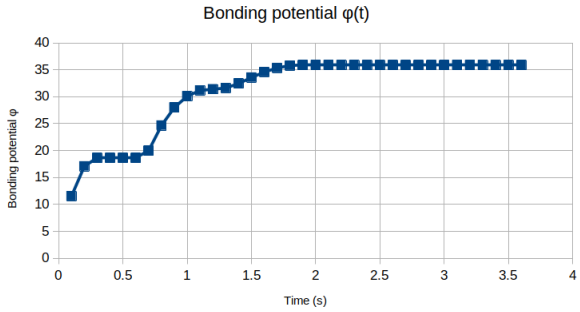


Figura 5.32: Andamento bonding potential nel tempo di un nodo di interfaccia

gli andamenti nel tempo della temperatura $T(t)$ e di bonding potential $\varphi(t)$ per ciascun caso con l'obiettivo di analizzare l'effetto della velocità di stampa sulla robustezza del pezzo finale. Ci si attende la presenza di due effetti contrastanti: un tempo maggiore rafforza il bonding ma le temperature più basse lo fanno arrestare prima. Il fattore dominante tra i due determina se è vantaggioso o meno aumentare la velocità di stampa.

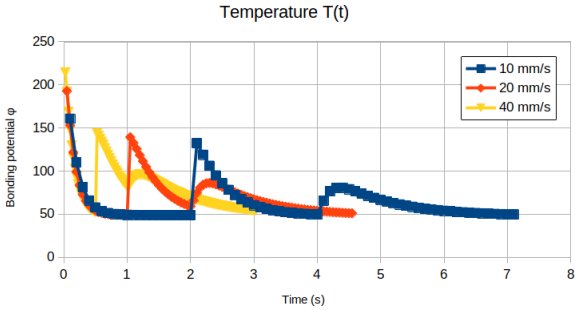


Figura 5.33: Comparazione andamenti temperatura $T(t)$ del nodo 1

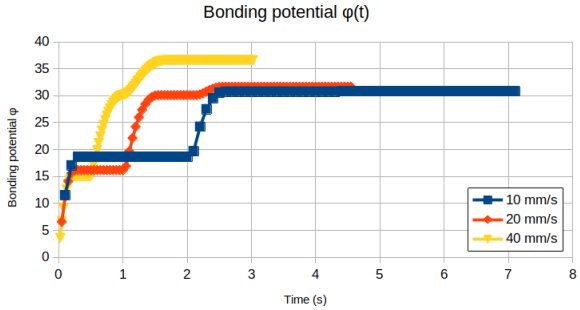


Figura 5.34: Comparazione andamenti bonding potential $\varphi(t)$ del nodo 1

Si osserva che in questo caso particolare prevale l'effetto delle temperature più basse e che conviene aumentare la velocità di stampa per ottenere un pezzo dalle proprietà più isotrope e con layer più saldi. Dal profilo di temperatura $T(t)$ si evince che questo è ancora più vero se il materiale già depositato non ha tempo di scendere sotto la temperatura di transizione vetrosa. Infatti, nel momento in cui $T < T_g$, parte dell'energia termica ricevuta per diffusione viene spesa nell'innalzamento di temperatura da T a T_g senza innalzare il bonding potential. Al contrario, se $T(t)$ si mantiene sempre al di sopra di T_g, tutto l'incremento di temperatura va a contribuire a rafforzare φ.
Si evidenzia comunque che questo è solamente un caso dimostrativo, molto lontano dalle geometrie realmente stampate. La lunghezza di 10 mm è estremamente contenuta e l'assenza di passate intra-layer vicine rende le interazioni termiche molto diverse dai casi pratici.
Si ricordano anche i limiti dettati dal modello fisico scelto, in particolare dell'ipotesi di materiale lineare le cui proprietà non variano con la temperatura.

5.8 Simulazione - stampa parete 2D

Si è visto che con il metodo Direct GFDM risulta impossibile eseguire simulazioni termo-meccaniche in tre dimensioni. Nel capitolo precedente ci si è limitati a risolvere il problema termico, adesso invece si vuole estendere il test-case della trave termo-meccanica in due dimensioni alla simulazione di processo.
Ovviamente tale approccio incontra forti limitazioni nell'applicabilità, però può essere considerato rappresentativo per certi casi particolari come pareti sottili. Inoltre la formulazione può essere opportunamente estesa a geometrie shell anche incurvate.

Si consideri una parete sottile di dimensioni $10 \times 0.8 \times 0.55$ mm stampata layer su layer seguendo delle istruzioni GCODE generate con il software Cura (4 layer da 0.2 mm ciascuno).

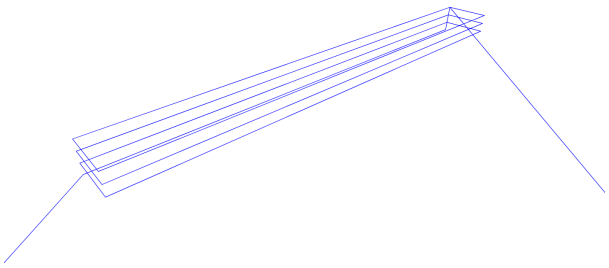


Figura 5.35: Screenshot di OpenGFDM con il percorso GCODE della simulazione

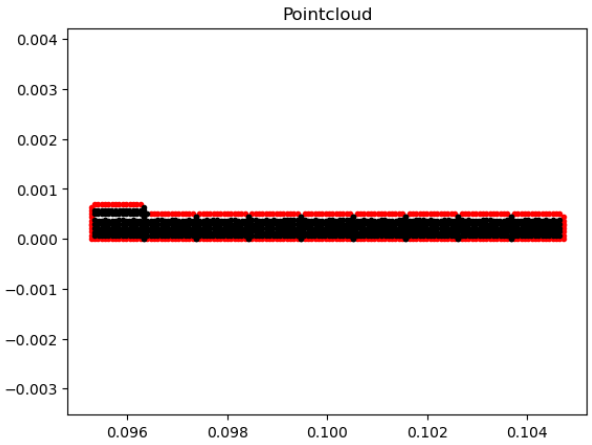


Figura 5.36: Pointcloud nel mezzo della simulazione del II layer

Come detto il simulatore segue le istruzioni depositando il materiale a blocchi. La lunghezza di tali blocchi, rettangolari in 2D, determina la risoluzione spaziale della simulazione e il passo temporale $\Delta t = \frac{\Delta x}{v}$.

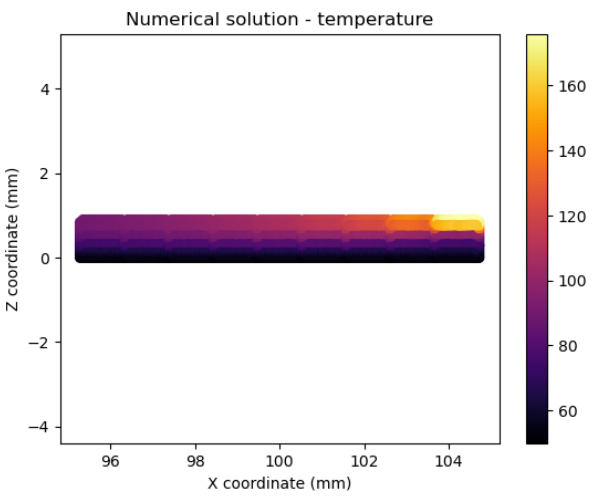


Figura 5.37: Soluzione numerica temperatura all'istante finale

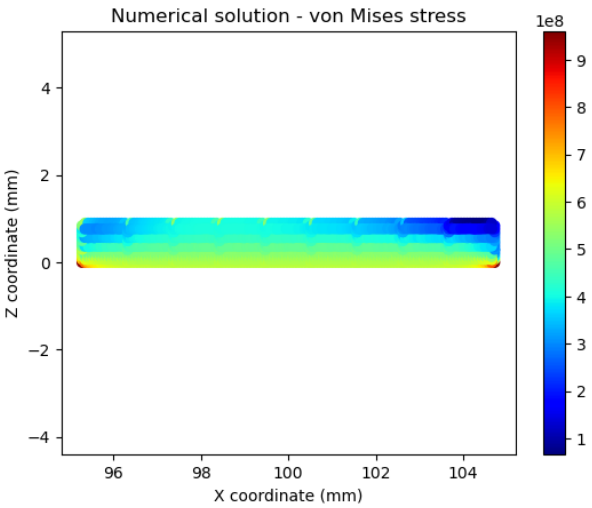


Figura 5.38: Soluzione numerica stress von Mises all'istante finale

Per ciascuno step quindi si genera un blocco e si fanno evolvere le soluzioni termica e meccanica. Come post-processing si possono calcolare le tensioni di von Mises, rappresentative dello stato tensionale del materiale istante per istante.

Nota che per ottenere un risultato corretto è necessario cambiare la temperatura di riferimento T_{ref} per l'espansione termica e farla coincidere con quella di estrusione $T_{extr} = 230\ °C$. In questo modo il materiale raffreddandosi si contrae e il vincolo sul piatto lo impedisce inducendo uno stato tensionale.
Infatti, osservando le tensioni di von Mises subito dopo la deposizione dell'ultimo tratto, si vede chiaramente che il materiale appena depositato è più caldo ed è quello meno sollecitato, mentre gli spigoli ancorati al piatto sviluppano delle importanti concentrazioni tensionali. Tali concentrazioni sono responsabili del famigerato fenomeno del warping che spesso affligge le stampe hobbistiche rovinandole in modo più o meno pesante.

Dal momento che la simulazione 2D è molto veloce (circa 10 s per questo caso semplificato), è possibile indagare l'effetto dei parametri di stampa sul warping.

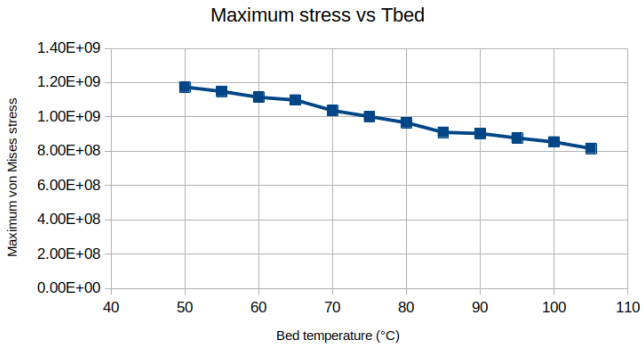


Figura 5.39: Effetto della temperatura del piatto sullo stress massimo

La soluzione classica al problema del warping è aumentare la temperatura del piatto di stampa. Questa indicazione è confermata numericamente dall'andamento decrescente dello stress massimo σ_v. Passando da 50 $°C$ a 70 $°C$ si ha una riduzione del 12% del picco tensionale, molto modesta probabilmente perché il pezzo è molto corto (solo 10 mm).
Si ricorda anche che queste sono solamente delle indicazioni qualitative: il modello non tiene conto della cristallizzazione del polimero e delle proprietà dipendenti dalla temperatura. Inoltre, il PET ha una temperatura di transizione vetrosa pari a $T_g = 80\ °C$ che va a limitare il range del piatto.

Capitolo 6

Conclusioni

Un metodo meshless alle differenze finite generalizzate (GFDM) è usato per risolvere numericamente problemi di natura termica, meccanica e termo-meccanica con lo scopo di simulare il processo di stampa 3D a deposizione di filamento. Limitandosi a risolvere modelli lineari, viene presentata l'implementazione all'interno di un nuovo software open-source rivolto all'utenza amatoriale hobbistica.
Per facilitarne l'uso, OpenGFDM include una interfaccia grafica semplice e moderna costruita con tecnologie web ed Electron. Il backend scritto in Julia può essere eseguito in locale oppure da remoto su un ampio numero di piattaforme.
Particolare cura è posta sull'accessibilità del software, che può essere usato da persone con disabilità visive senza modifiche ad-hoc nella maggior parte dei casi.
Il software si può definire accessibile anche nel senso che non pone alcuna barriera o restrizione all'utente: può essere utilizzato su qualunque sistema operativo (cross-platform) e il codice sorgente è liberamente ispezionabile e modificabile.

La presente tesi vuole descriverne l'attività di sviluppo articolata su più livelli:

- Interfaccia grafica;
- Generazione pointcloud e ricerca dei nodi vicini;
- Formulazione matematica del metodo e implementazione;
- Validazione su problemi termici, meccanici, termomeccanici semplificati;
- Presentazione risultati e applicazione alla simulazione di processo;

Nel primo capitolo sono descritte le esigenze che vuole soddisfare OpenGFDM e come queste si traducono nelle varie caratteristiche dell'applicativo. Il capitolo 2 è incentrato sulla presentazione processo di stampa 3D FDM e sulla simulazione di processo; qui si propone un modello termo-meccanico lineare disaccoppiato, in cui la parte termica è del tutto indipendente e calcolata separatamente dalla parte meccanica. La derivazione matematica mostra in particolare come gli stress termici entrino nel calcolo elastico come carichi volumetrici aggiuntivi ed alterino le condizioni al contorno.
Passando alla formulazione del metodo GFDM, nel capitolo 3 si mostra come il metodo classico soffra di severi problemi di stabilità e condizionamento. Le varianti Direct GFDM, MDLSM e Spectral GFDM propongono di risolverli ciascuno in modo leggermente diverso.
La scelta ricade sul metodo Direct GFDM, che quindi viene implementato e testato in dettaglio nel capitolo 4. Si eseguono test di complessità crescente, partendo dalla conduzione termica su dominio rettangolare 2D per arrivare a simulare il transitorio termico di un piatto perforato in 3D nel capitolo 5.

Una volta validato il metodo per problemi termici, meccanici e termo-meccanici, si procede con l'attività di simulazione di processo. A partire dalle istruzioni macchina GCODE il software

deposita tratti di plastica e ne segue l'evoluzione dei parametri termici e meccanici nel tempo. A differenza dei metodi numerici mesh-based, l'approccio meshless permette in questi casi di avere molta flessibilità e di gestire in modo molto naturale la presenza di vuoti e i contorni che mutano ad ogni passo temporale.
Le simulazioni termiche in particolare riescono a replicare molte delle fenomenologie presenti nel processo di stampa; si citano a titolo esemplificativo il riscaldamento del nuovo materiale sui layer già depositati e lo sviluppo di forti gradienti termici in corrispondenza delle interfacce.

Per quanto riguarda la parte termo-meccanica ed elastica in generale, purtroppo il malcondizionamento estremo impedisce di ottenere risultati accettabili in tre dimensioni. Molti sforzi sono stati spesi per risolvere la cosa, senza successo. La simulazione delle proprietà meccaniche pertanto è limitata alle due dimensioni, in cui comunque funziona molto bene.

Un'altra criticità emersa è il rallentamento molto severo dell'interfaccia grafica dovuto al sistema di comunicazione WebSocket, risultato molto più oneroso del previsto in presenza di un gran numero di nodi.
I miglioramenti possibili però sono innumerevoli e gli sviluppi futuri possono certamente mitigare se non risolvere queste problematiche. Dal lato metodo numerico è possibile esplorare l'introduzione di basi non monomiali, ad esempio con le Radial Basis Function (RBF), oppure introdurre una qualche forma di stabilizzazione artificiale, o ancora imporre la conservazione locale su quantità di interesse.
L'estensione verso problemi ed equazioni più complesse è teoricamente possibile e piuttosto semplice da implementare con il metodo Direct GFDM. In generale tuttavia si hanno fortissime limitazioni pratiche dettate dall'estremo malcondizionamento.

Bibliografia

OpenJS Foundation. URL: https://www.electronjs.org/.

Amani, J., A. Saboor Bagherzadeh e Timon Rabczuk. "Error Estimate and Adaptive Refinement in Mixed Discrete Least Squares Meshless Method". In: *Mathematical Problems in Engineering* 2014 (2014), p. 721240. DOI: 10.1155/2014/721240.

Artificial Intelligence Market Size, Share & Trends Analysis Report. Rapp. tecn. GVR-1-68038-955-5. Grand View Research, 2022.

Benito, Juan José et al. "Application of the Generalized Finite Difference Method to improve the approximated solution of pdes". In: *Cmes-computer Modeling in Engineering & Sciences* 38 (2008), pp. 39–58.

Bezanson, Jeff et al. "Julia: A fresh approach to numerical computing". In: *SIAM Review* 59.1 (2017), pp. 65–98. DOI: 10.1137/141000671. URL: https://epubs.siam.org/doi/10.1137/141000671.

Center, Wolfram Documentation. URL: https://reference.wolfram.com/language/PDEModels/tutorial/Multiphysics/ModelCollection/ThermalLoad.html.

Colagrossi, Andrea et al. "Particle packing algorithm for SPH schemes". In: *Comput. Phys. Commun.* 183 (2012), pp. 1641–1653. URL: https://www.openaccessrepository.it/record/22940.

Comminal, Raphaël et al. "Numerical modeling of the strand deposition flow in extrusion-based additive manufacturing". In: *Additive manufacturing* 20 (2018), pp. 68–76. URL: https://doi.org/10.1016/J.ADDMA.2017.12.013.

Comminal, Raphaël et al. "Numerical modeling of the strand deposition flow in extrusion-based additive manufacturing". In: *Additive Manufacturing* 20 (2018), pp. 68–76. URL: https://doi.org/10.1016/j.addma.2017.12.013.

Dávila, José Luis et al. "Hybrid manufacturing: a review of the synergy between directed energy deposition and subtractive processes". In: *The International Journal of Advanced Manufacturing Technology* 110 (2020), pp. 3377 –3390. URL: https://doi.org/10.1007/s00170-020-06062-7.

Faraji, Saeb, M. H. Afshar e J. Amani. "Mixed Discrete Least Squares Meshless method for solution of quadratic partial differential equations". In: *Scientia Iranica* 21 (2014), pp. 492–504. DOI: 10.24200/sci.2017.4203.

Fernández-Fernández, José Antonio et al. "Fast Octree Neighborhood Search for SPH Simulations". In: *ACM Transactions on Graphics (TOG)* 41 (2022), pp. 1 –13. URL: https://animation.rwth-aachen.de/publication/0579/.

Ferreira, Augusto César Albuquerque e Paulo Marcelo Vieira Ribeiro. "Reduced-order strategy for meshless solution of plate bending problems with the generalized finite difference method". In: *Latin American Journal of Solids and Structures* (2019). URL: https://doi.org/10.1590/1679-78255191.

Fourthirtytwo. URL: https://commons.wikimedia.org/wiki/File:Laplace%27s_equation_on_an_annulus.svg.

Industry-Proven Altair Radioss Finite Element Analysis Solver Now Available as Open-Source Solution. Altair. URL: https://www.altair.com/newsroom/news-releases/industry-proven-altair-radioss-finite-element-analysis-solver-now-available-as-open-source-solution.

Introduction to Material Jetting — MJ, NPJ, DOD. Dassault Systemes. URL: https://www.3ds.com/make/guide/process/material-jetting.

Jenkins, Meg. *The Past, Present, and Future of Cloud Computing and CFD*. Simscale. URL: https://www.simscale.com/blog/future-of-cloud-computing-cfd-engineers/.

Jessica, Haelewyn. *TPLV06 - Release of power in a hollow sphere*. URL: https://code-aster.org/V2/doc/v14/en/man_v/v4/v4.04.006.pdf.

— *TTLP301 - Transfer of heat in a perforated plate*. URL: https://code-aster.org/V2/doc/default/en/man_v/v4/v4.23.301.pdf.

Khanafer, Khalil et al. "Thermal analysis of fused deposition modeling process based finite element method: Simulation and parametric study". In: *Numerical Heat Transfer, Part A: Applications* 81 (2022), pp. 94 –118. DOI: 10.1080/10407782.2022.2038972.

Li, Xiang-Yang, Shang-Hua Teng e Alper Üngör. "Point placement for meshless methods using Sphere packing and Advancing Front methods". In: 2001.

Liszka, Tadeusz, Carlos Armando Duarte e W. W. Tworzydlo. "hp-Meshless cloud method". In: *Computer Methods in Applied Mechanics and Engineering* 139 (1996), pp. 263–288. DOI: 10.1016/S0045-7825(96)01086-9.

Löhner, Rainald e Eugenio Oñate. "Advancing front techniques for filling space with arbitrary separated objects". In: *Finite Elements in Analysis and Design* 46 (2010), pp. 140–151.

Marlin G-code Index. URL: https://marlinfw.org/meta/gcode/.

Nagel, James Richard. "Solving the Generalized Poisson Equation Using the Finite-Difference Method (FDM)". In: 2009.

Oñate, Eugenio. "Finite increment calculus (FIC): a framework for deriving enhanced computational methods in mechanics". In: *Adv. Model. Simul. Eng. Sci.* 3 (2016), 14:1–14:18. DOI: 10.1186/s40323-016-0065-9.

Oñate, E., F. Perazzo e J. Miquel. "A finite point method for elasticity problems". In: *Computers & Structures* 79.22 (2001), pp. 2151–2163. ISSN: 0045-7949. DOI: https://doi.org/10.1016/S0045-7949(01)00067-0.

Pimenov, Sergey. URL: https://korzh.com/metroui.

Project, Three.js. URL: https://threejs.org/.

Sande, Kiera van der e Bengt Fornberg. "Fast variable density 3-D node generation". In: *SIAM J. Sci. Comput.* 43 (2019), A242–A257. URL: https://arxiv.org/abs/1906.00636.

Scopigno, R. et al. "Digital Fabrication Techniques for Cultural Heritage: A Survey". In: *Computer Graphics Forum* 36.1 (2017), pp. 6–21. DOI: https://doi.org/10.1111/cgf.12781. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.12781.

Seibold, Benjamin. "M-Matrices in meshless finite difference methods". In: (gen. 2006).

Simulation Software Market Size, Share & Trends Analysis Report. Rapp. tecn. GVR-3-68038-751-3. Grand View Research, 2021.

Suchde, Pratik. "Conservation and Accuracy in Meshfree Generalized Finite Difference Methods". In: (gen. 2018).

Tan, Chaolin et al. "Selective laser melting of high-performance pure tungsten: parameter design, densification behavior and mechanical properties". In: *Science and Technology of Advanced Materials* 19.1 (2018), pp. 370–380. URL: https://doi.org/10.1080/14686996.2018.1455154.

Wang, Zhujiang et al. "FlowMesher: An automatic unstructured mesh generation algorithm with applications from finite element analysis to medical simulations". In: *CoRR* abs/2103.05640 (2021). URL: https://arxiv.org/abs/2103.05640.

Zaman, Mohammad Asif. “Numerical Solution of the Poisson Equation Using Finite Difference Matrix Operators”. In: *Electronics* 11.15 (2022). DOI: 10.3390/electronics11152365. URL: https://www.mdpi.com/2079-9292/11/15/2365.