



**Politecnico
di Torino**

Politecnico di Torino

Corso di Laurea Magistrale

A.A. 2021/2022

Ingegneria Dei Materiali Strutturali

Sessione di Laurea: Marzo 2022

**SVILUPPO DI UN ALGORITMO PER IL
CALCOLO DELLA SPECIFIC ENERGY
ABSORPTION DI LAMINATI
COMPOSITI IMPIEGATI IN
APPLICAZIONI CRASHWORTHINESS**

Relatore:

Paolino Davide Salvatore

Candidato:

Colonna Enio

Correlatore:

Vigna Lorenzo

Tutor aziendale: Galizia Giuseppe

Sommario

Sommario	2
1. Introduzione.....	6
1.1. Crashworthiness e materiali utilizzati.....	7
1.2. Curve di crash e setup sperimentale.....	8
2. Filtraggio delle curve di crash.....	18
3. Implementazione dell'algoritmo	25
3.1. Ricerca bibliografica.....	25
3.2. Approccio algoritmico per il calcolo della SEA.....	29
3.2.1. Algoritmo con spostamenti imposti dall'operatore.....	29
3.2.2. Algoritmo che considera lo spostamento finale variabile	33
3.2.3. Algoritmo con spostamenti variabili: prima versione	38
3.2.4. Algoritmo con spostamenti variabili: seconda versione	45
3.3. Approcci alternativi al calcolo della Specific Energy Absorption (SEA)	48
3.4. Algoritmo con spostamenti variabili: versione definitiva.....	52
3.5. Taratura dei parametri.....	66
4. Valutazione degli algoritmi.....	73
4.1. Setup sperimentale.....	73
4.2. Confronto degli algoritmi	81
5. Calcolo della SEA mediante algoritmo con spostamenti variabili: effetto della tipologia di misura dei provini.....	87
6. Conclusioni.....	90
7. Appendice: codici degli algoritmi.....	92
8. Bibliografia	124

Indice delle figure

Figura 1: Crash box utilizzata nelle autovetture di F1	8
Figura 2: Supporto Crashworthiness.....	9
Figura 3: a) Provino con sezione a C b) Provino con sezione angolare c) Provino piatto	10
Figura 4: Provino in materiale composito: Glass fiber	11
Figura 5: Instron Drop Tower 9450	12
Figura 6: Sistema di acquisizione della velocità di impatto	14
Figura 7: Curva tipica Forza-tempo di una prova crashworthiness.....	16
Figura 8: Curva tipica Forza-spostamento di una prova crashworthiness.....	16
Figura 9: Curva Forza-tempo di una prova crashworthiness	17
Figura 10: Curva Forza-spostamento di una prova crashworthiness	18
Figura 11: Curva Forza-spostamento non filtrata	19
Figura 12: Spettro in frequenza di una tipica curva di crash	19
Figura 13: Curva Forza-spostamento dopo la rimozione delle frequenze piu' rumorose.....	20
Figura 14: Curva Forza-spostamento dopo smoothing eccessivo.....	22
Figura 15: Curva Forza-spostamento dopo filtraggio con parametri ottimali	23
Figura 16: Curva Forza-tempo non filtrata e filtrata sovrapposte	23
Figura 17: Curve Forza-spostamento sperimentali e simulate ottenute su provini con geometria tubolare [4]	26
Figura 18: Curva media Forza-spostamento ottenuta su provini con geometria tubolare CFRP [8]	27
Figura 19: Diagramma di flusso che descrive il funzionamento dell'algoritmo che considera gli spostamenti imposti da un operatore.....	30
Figura 20: Tipica curva di crash Forza-Spostamento: algoritmo a spostamenti imposti.....	31
Figura 21: Posizioni spostamento iniziale e finale all'interno del vettore displacement	32
Figura 22: Diagramma di flusso relativo al codice che calcola la SEA imponendo lo spostamento iniziale e ricalcolando quello finale in maniera iterativa.....	35
Figura 23: Porzione di codice che illustra l'effetto del margine sul calcolo dello spostamento finale.....	36
Figura 24: Confronto sul calcolo dello spostamento finale utilizzando oppure non utilizzando un valore di margine.....	37
Figura 25: Curva di crash ottenuta per un provino G50(matrice in resina epossidica rinforzata con fibre di vetro, spessore 5 mm)	39
Figura 26: Range di spostamenti calcolato mediante l'algoritmo in Figura 23.....	39
Figura 27: Tratto di codice che regola il valore del margine sul calcolo dello spostamento iniziale	40
Figura 28: Diagramma di flusso relativo alla determinazione del margine da utilizzare per il calcolo dello spostamento iniziale	42
Figura 29: Curva di crash ottenuta su di un provino G30(matrice in resina epossidica rinforzata con fibre di vetro, spessore 3mm)	43
Figura 30: Risultati ottenuti mediante l'algoritmo che considera variabili entrambi gli spostamenti.....	44
Figura 31: a) porzione di codice raffigurante il processo iterativo per il calcolo dello spostamento iniziale b) porzione di codice raffigurante il processo iterativo per il calcolo dello spostamento finale.....	46
Figura 32: Curva di crash ottenuta su un provino di geometria tubolare e spessore 1 mm.....	47
Figura 33: Diagramma di flusso relativo al calcolo della SEA utilizzando gli spostamenti imposti in percentuale	49

Figura 34: Curva di crash ottenuta su di un provino C50(matrice in resina epossidica rinforzata con fibre di carbonio, spessore 5 mm)	50
Figura 35: diagramma di flusso per il calcolo della SEA considerando l'area sottesa a tutta la curva.....	51
Figura 36: Diagramma di flusso relativo alla metodologia di calcolo della forza media nella versione definitiva dell'algoritmo con spostamenti variabili	53
Figura 37: Matrice contenente sotto forma di vettori tutte le grandezze misurate dalla cella di carico prima dell'istante dell'impatto	55
Figura 38: Grandezze misurate dalla cella di carico nel momento esatto dell'impatto.....	56
Figura 39: Curva Forza-Tempo non filtrata Forza-Tempo filtrata Sovrapposizione delle due.....	57
Figura 40: Curva Forza-spostamento non filtrata Forza-Spostamento filtrata Sovrapposizione delle due.....	58
Figura 41: Calcolo della posizione corrispondente alla variabile "start_displacement"	59
Figura 42: Calcolo "final_position"	59
Figura 43: Curva ottenuta su di un provino G20(matrice in resina epossidica rinforzata con fibre di vetro, spessore 2 mm)	60
Figura 44: Curva di crash ottenuta su di un provino CFRP (matrice in resina epossidica rinforzata con fibre di carbonio)	62
Figura 45: Posizione relativa al limite di forza calcolato pe la curva in Figura 43.....	64
Figura 46: Calcolo della forza media eseguito considerando valori di forza: a) non filtrata b) filtrata.....	65
Figura 47: Curva di crash ottenuta su di un provino piatto realizzato in CFRP utilizzata come esempio per il processo di taratura dei parametri.....	67
Figura 48: Curva di crash utilizzata come esempio nel processo di taratura dei parametri: algoritmo con spostamenti variabili	69
Figura 49: Curva di crash utilizzata come esempio per il processo di taratura dei parametri: algoritmo che considera gli spostamenti imposti in percentuale	70
Figura 50: Curva di crash ottenuta per un provino piatto C50(matrice in resina epossidica rinforzata con fibre di carbonio, spessore 5mm)	71
Figura 51: Provino piatto G30 utilizzato per i test finalizzati alla valutazione degli algoritmi proposti	74
Figura 52: a) Provino 'glass-4' durante l'impatto b) provino 'glass-4' dopo l'impatto.....	80
Figura 53: Curva di crash ottenuta testando il provino 'glass-4'	81
Figura 54: Grafico a barre: deviazione standard calcolata per ciascuno degli algoritmi proposti	87

Abstract

Si definisce ‘crashworthiness’ la capacità di un veicolo di proteggere i passeggeri in caso di urto, obiettivo raggiunto tramite l’impiego di apposite strutture che si deformano in modo controllato e assorbono l’energia cinetica del veicolo evitando decelerazioni troppo elevate che possano recar danno ai passeggeri. In questo lavoro di tesi, realizzato in collaborazione con Instron, si propone di standardizzare i test di crashworthiness svolti sui materiali compositi. Questo è un obiettivo molto importante in quanto sino a questo momento, a causa della mancanza di uno standard, i risultati di tali test sono sempre stati analizzati con uno sguardo differente, e questo non permette un confronto oggettivo tra ricerche diverse. Il risultato principale di questa tipologia di test sono delle curve Forza-spostamento, le quali analizzate in maniera corretta permettono di determinare l’energia assorbita per unità di massa dal provino (SEA).

Ci sono numerosi aspetti che andrebbero definiti all’interno dello standard, questo lavoro tratta uno dei più discussi e più soggetto all’influenza di un qualsiasi operatore: l’individuazione del tratto utile di una curva di crash per il calcolo della Specific Energy Absorption (SEA). Senza uno strumento matematico che permetta di trovare per qualsiasi prova crashworthiness il tratto utile, il confronto tra due prove diverse non è indicativo della loro reale capacità di assorbire energia in quanto le curve, se analizzate da due diversi operatori, fornirebbero diversi valori di SEA.

Al fine di superare questa problematica, si rende necessario sviluppare un algoritmo che sia in grado di analizzare una qualsiasi curva di crash e di determinarne il tratto da considerare per il calcolo della SEA.

Per la scrittura dell’algoritmo si è partiti da quanto svolto in letteratura riguardo l’analisi delle curve sperimentali. Sono state sviluppate varie versioni di algoritmi in grado di determinare il tratto utile delle curve di crash, partendo da quella più semplice sino ad arrivare alla versione finale, la quale sarà la candidata ad essere proposta nello standard come metodologia di calcolo della SEA. L’ultima versione dell’algoritmo è in grado di escludere dall’analisi tutti quei tratti di curva che non risultano utili per l’analisi e di convergere verso il tratto in cui si ha una stabilizzazione del valore di forza (frattura progressiva del provino). È stata effettuata una validazione sperimentale dell’algoritmo per verificare che effettivamente sia in grado di fornire risultati accettabili e con la minore dispersione possibile.

Tra tutte le versioni, quella scelta come più opportuna per lo standard è quella che ha calcolato i vari valori di SEA garantendo la minima dispersione dei risultati su un certo numero di test svolti nelle stesse conduzioni e sullo stesso materiale, i quali teoricamente dovrebbero fornire lo stesso risultato in termini di SEA.

1. Introduzione

Nel mondo odierno, nel campo dell'ingegneria soprattutto, risulta sempre più di maggior peso una scelta adeguata dei materiali in una determinata applicazione al fine di ottenere la risposta desiderata dal sistema in analisi. Un campo di ricerca che viene esplorato da parecchi anni e che continua ad essere esplorato nel presente è quello degli impatti, in cui si sviluppano velocità di deformazione molto elevate e si porta il materiale a rottura.

In campo autoveicolistico, un impatto causa elevate decelerazioni sugli occupanti di una vettura, e di conseguenza è molto importante che i materiali utilizzati per la costruzione dello stesso abbiano determinate caratteristiche in termini di assorbimento di energia onde evitare di trasmettere l'energia cinetica del veicolo ai passeggeri tramite forze elevate che possono causare danni. Un parametro fondamentale che descrive l'efficacia di una struttura nell'assorbire l'energia di un urto deformandosi in modo controllato è la SEA (Energia assorbita per unità di massa) dei materiali utilizzati, la quale permette un confronto tra i vari materiali utilizzati in questo campo in termini di capacità di assorbimento energetico rapportato alla massa di materiale.

Storicamente, in applicazioni in cui è richiesto di assorbire l'energia di un urto sono sempre stati utilizzati materiali metallici, soprattutto alluminio ed acciaio, in quanto in grado di subire un'ampia deformazione plastica con conseguente riscaldamento, e questi meccanismi permettono di assorbire un'elevata quantità di energia durante l'impatto. Sta crescendo sempre più però un nuovo fronte di ricerca sull'utilizzo di laminati compositi in questa tipologia di applicazione in quanto sono materiali che permettono di ridurre notevolmente il peso di una crash box esibendo caratteristiche anche superiori in termini di SEA rispetto ad alluminio e acciaio [1]. I laminati compositi maggiormente impiegati in questo campo sono laminati con matrice polimerica in resina epossidica rinforzata con fibre di vetro o fibre di carbonio, ma stanno prendendo piede, negli ultimi anni, anche matrici termoplastiche (poliammidi, PEEK) e fibre di origine naturale (basalto, lino e altre fibre di origine vegetale).

Poter confrontare in maniera oggettiva i risultati dei test di impatto su materiali compositi permetterebbe un loro confronto equo e prettamente scientifico volto alla scelta del materiale corretto per ogni applicazione. Attualmente questo non è possibile in quanto non vi è uno standard che imponga la metodologia dei test di impatto su tali materiali, per questo motivo nasce questo lavoro di tesi, che si propone di risolvere o portare ad uno step successivo una delle tante problematiche relative al calcolo oggettivo della SEA.

La problematica che verrà affrontata nei prossimi capitoli è quella relativa all'individuazione del tratto utile delle curve dei test di crash su provini in maniera univoca, in modo da calcolare la SEA senza l'influenza di un operatore esterno e poter quindi confrontare le varie curve in maniera assolutamente oggettiva.

A questo proposito, l'obiettivo di questo lavoro è quello di sviluppare un algoritmo che in maniera automatica sia in grado di individuare il tratto utile di qualsiasi curva di crash e calcolarne la SEA. Verranno confrontate diverse varianti dell'algoritmo che sfruttano diverse modalità di ricerca del tratto utile, e sarà scelto il più efficace basandosi sui risultati di alcune prove sperimentali descritte in dettaglio all'interno del capitolo 3. Verrà inoltre proposto un metodo di filtraggio delle curve sperimentali al fine di consentire un più facile confronto visivo tra curve diverse e verrà infine proposto un confronto tra risultati di SEA ottenuti considerando le masse e dimensioni medie dei provini e la SEA calcolata tenendo conto di massa e dimensioni misurate per ogni singolo provino.

1.1. Crashworthiness e materiali utilizzati

La crashworthiness è definita come la capacità di una certa struttura, chiamata in gergo tecnico crash box, di assorbire un'elevata quantità di energia durante un urto e di fratturarsi in maniera progressiva con l'obiettivo di garantire una decelerazione uniforme e non impulsiva dei passeggeri all'interno di un determinato mezzo, evitando così ingenti danni fisici sugli stessi. Risulta notevolmente importante lo studio della crashworthiness in un mondo in cui i fenomeni di urto sono all'ordine del giorno. Questo spinge verso la necessità di progettare crash box sempre più prestanti e di conseguenza verso un'intensiva ricerca di materiali con SEA molto elevata. Applicazioni ingegneristiche in cui il peso non è un fattore trascurabile (crash box per autovetture da corsa, aeromobili, riduzione dei consumi dei mezzi di trasporto ecc...) appare necessario utilizzare materiali che presentino un valore di energia assorbita elevato e una densità ridotta. I candidati perfetti a questo appello sono i laminati compositi (soprattutto matrice polimerica in resina epossidica rinforzata con fibre di vetro o di carbonio), tutt'ora oggetto di studio in questo campo per le loro spiccate caratteristiche meccaniche unite alla loro densità inferiore rispetto ai rivali materiali metallici.

Qui di seguito, in **Figura 1**, è stata riportata a titolo di esempio un'immagine raffigurante la tipica crash box utilizzata nelle autovetture di F1:



Figura 1: Crash box utilizzata nelle autovetture di F1

Come si evince dalla **Figura 1** la geometria della crash box è realizzata in modo tale da innescare la frattura in maniera preferenziale in una certa direzione e garantire così un collasso il più progressivo e costante possibile della struttura, evitando fenomeni di buckling (rottura per instabilità elastica al carico di punta) delle pareti sottili che causerebbero un assorbimento energetico decisamente basso. Nel campo della crashworthiness la componente geometrica di innesco della frattura è chiamata trigger e svolge un ruolo molto importante in questa applicazione in quanto si rivela necessario per evitare fenomeni di rottura improvvisa con basso assorbimento di energia.

1.2. Curve di crash e setup sperimentale

Prima di parlare nel dettaglio di tutto il setup sperimentale e della strumentazione necessari all'ottenimento di una curva di crash, è importante risalire all'origine della necessità di test di caratterizzazione su materiali compositi. I compositi sono una classe di materiali molto interessante e ricercata nel mondo delle applicazioni strutturali in virtù delle loro spiccate caratteristiche meccaniche in sinergia con una densità estremamente bassa [1].

La problematica che è però intrinseca a questa classe di materiali è la loro struttura fortemente anisotropa, che comprende quindi svariati meccanismi di frattura con conseguente differenza nel comportamento della struttura in funzione del meccanismo implicato.

I principali meccanismi di frattura in condizioni di elevata velocità di deformazione individuati da [2] sono qui di seguito elencati:

- Tearing (meccanismo con maggiore assorbimento di energia)
- Transverse shearing
- Fragmentation
- Local buckling (meccanismo con minore assorbimento di energia)

La presenza di diversi meccanismi di rottura del provino porta ad un'elevata dispersione dei dati quando si effettuano dei test di caratterizzazione a causa del differente assorbimento di energia determinato da ciascun meccanismo, per tale motivo è estremamente importante che il setup sperimentale riduca drasticamente questa dispersione isolando una determinata modalità di frattura o riducendo al minimo la competizione tra le varie possibilità di danneggiamento. Quanto detto sino ad ora vale per tutti i test di caratterizzazione su materiali compositi, comprese le prove crashworthiness, oggetto di questo lavoro di tesi. Per la finalizzazione di tale obiettivo è fondamentale lo sviluppo di un supporto ottimizzato per i test di crash e di provini con geometria tale da sviluppare meccanismi di frattura tanto efficaci da renderli interessanti in questo campo applicativo. Verrà ora mostrato e descritto il supporto utilizzato per effettuare i test svolti in questo progetto.

Il supporto crashworthiness sviluppato, che svolge un ruolo da protagonista nell'ottenimento di curve di crash che siano utilizzabili per il calcolo della SEA, è raffigurato in **Figura 2**:

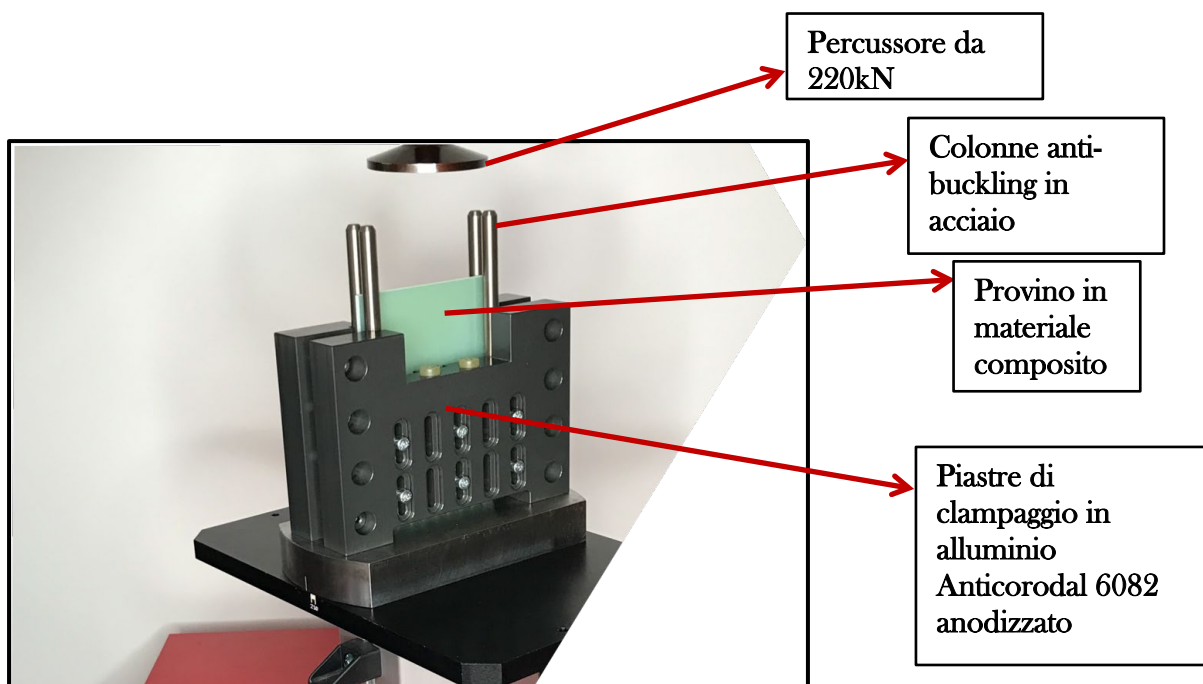


Figura 2: Supporto Crashworthiness

Gli elementi fondamentali che costituiscono il supporto sono le piastre di clampaggio, che hanno la funzione di tenere in posizione il provino durante il test, e le colonne anti-buckling che impediscono al provino di flettersi in direzione orizzontale. Impedendo questo fenomeno si garantisce l'innescare un meccanismo di frattura ad elevato assorbimento di energia e si riescono quindi ad ottenere dei risultati utilizzabili a livello sperimentale per il confronto tra diversi laminati compositi.

Oltre alle quattro colonne disposte agli angoli del supporto possono anche essere montate ulteriori colonne più corte in caso di test su provini più sottili, i quali sono soggetti a buckling per carichi di punta inferiori rispetto a provini spessi. Inoltre, può essere variata l'altezza delle colonne rispetto alla base del supporto, anche questo influisce sull'assorbimento di energia ma verrà spiegato più in dettaglio nei capitoli successivi.

Come si è potuto intuire la configurazione del supporto influenza i valori di SEA misurati, vi è però anche un contributo della geometria del provino che verrà ora discusso. Durante gli anni sono state testate numerose geometrie differenti, a titolo di esempio vengono riportati in **Figura 3** alcuni provini con differenti sezioni adoperati da [3]:

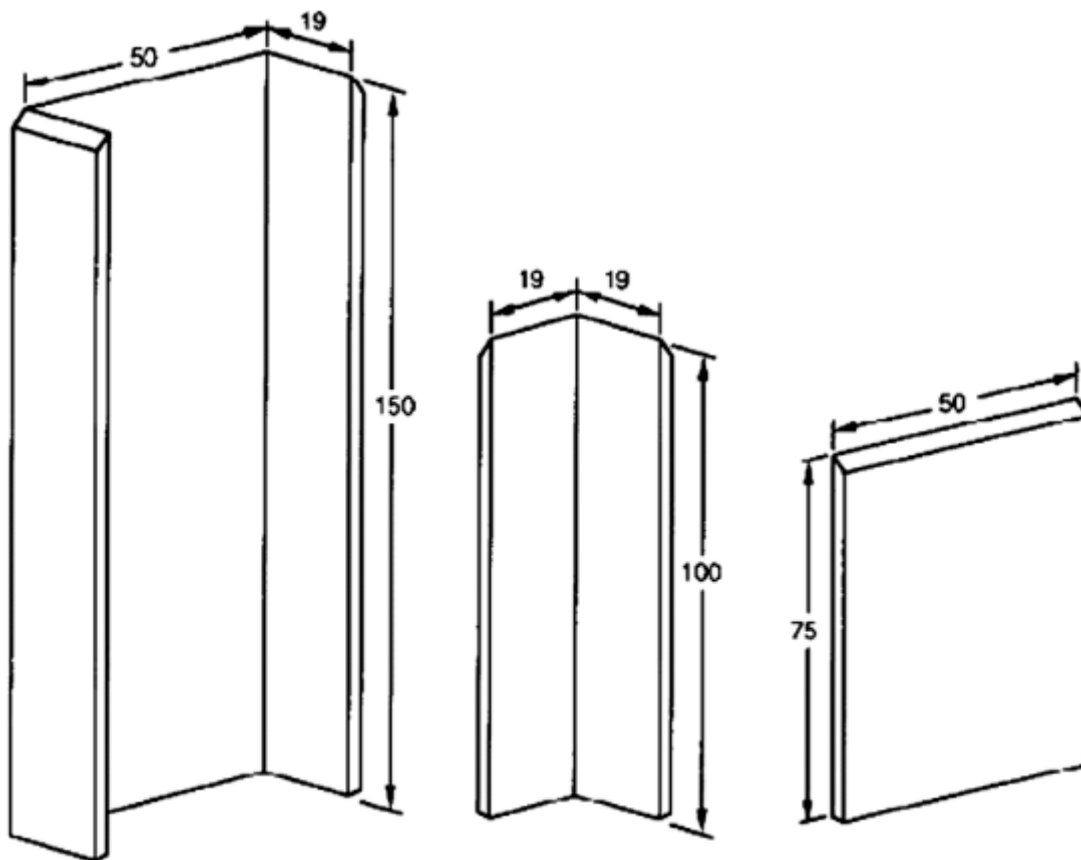


Figura 3: **a)** Provino con sezione a C **b)** Provino con sezione angolare **c)** Provino piatto

[3]

Il principale vantaggio di geometrie con sezione mostrata in **Figura 3** è l'auto supporto del provino durante il test, però nascono altre problematiche relative agli spigoli, in quanto possono indurre meccanismi di frattura non desiderati e che portano ad un basso assorbimento di energia. Per non essere soggetti alla geometria del provino ed ottenere provini più facili da realizzare, e quindi più uniformi a livello di proprietà meccaniche, si è pensato di utilizzare provini piatti. Durante i test svolti per questo lavoro di tesi sono stati utilizzati dei provini piatti con sezione rettangolare, i quali presentano le seguenti caratteristiche geometriche nominali:

- **Lunghezza:** 150 mm
- **Larghezza:** 100 mm
- **Spessore:** tra 2 mm e 6 mm
- **Altezza del trigger a dente di sega:** 5 mm

In **Figura 4** viene mostrato un tipico provino piatto con matrice in resina epossidica rinforzata con fibre di vetro, **NEMA FR4**, con spessore 3 mm:

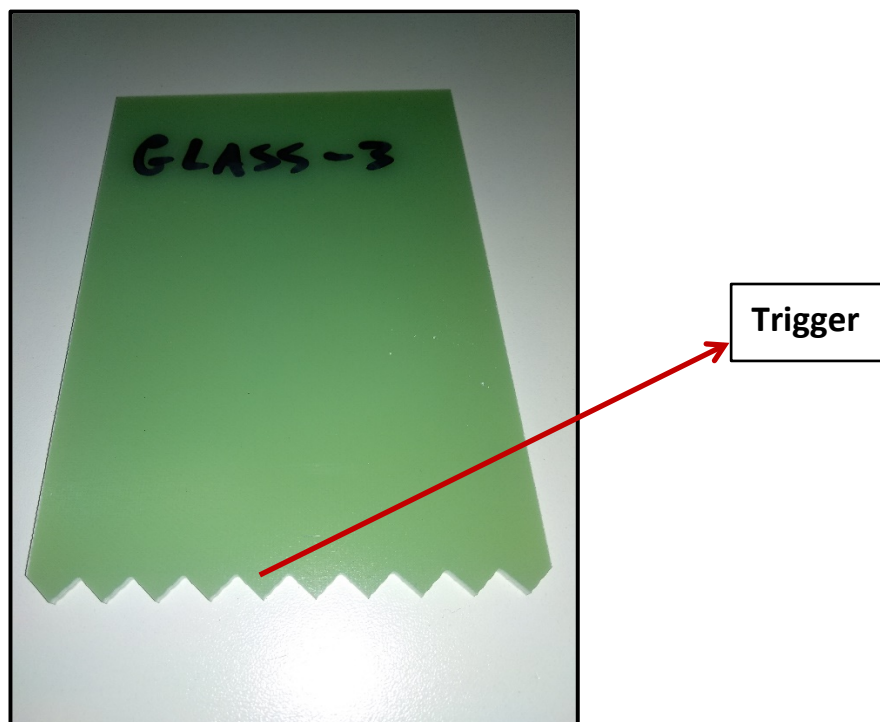


Figura 4: Provino in materiale composito: Glass fiber

Il trigger in questi provini costituisce l'elemento geometrico che garantisce un innescò di frattura preferenziale e di conseguenza facilita la rottura progressiva del provino.

Riassumendo quanto asserito in questa ultima parte, i materiali compositi presentano un'elevata dispersione dei risultati dei test di crash a causa dei molteplici meccanismi di danneggiamento che possono coesistere. Questo porta alla problematica della definizione di un test univoco che permetta di eliminare la soggettività dell'operatore nell'analisi dei dati unificando così il metro di confronto tra compositi differenti.

Al fine di arrivare alla definizione di una prova standard di crashworthiness per materiali compositi, il Politecnico di Torino ha avviato una collaborazione con Instron per definire una metodologia di prova che consenta di valutare le proprietà dei materiali compositi in prove di crash.

La parte primordiale dell'analisi sperimentale nel campo di studio dei compositi come materiali per la crashworthiness è quella relativa a tutto il setup sperimentale e all'apparecchiatura necessari all'ottenimento delle curve di crash, dalle quali successivamente si può arrivare a determinare la SEA e quindi confrontare diverse tipologie di laminati e stabilire quale sia più adeguata ad una certa applicazione.

In questo lavoro di tesi le curve sperimentali sono state ottenute utilizzando la drop tower di Instron modello 9450, la quale viene mostrata in **Figura 5**:

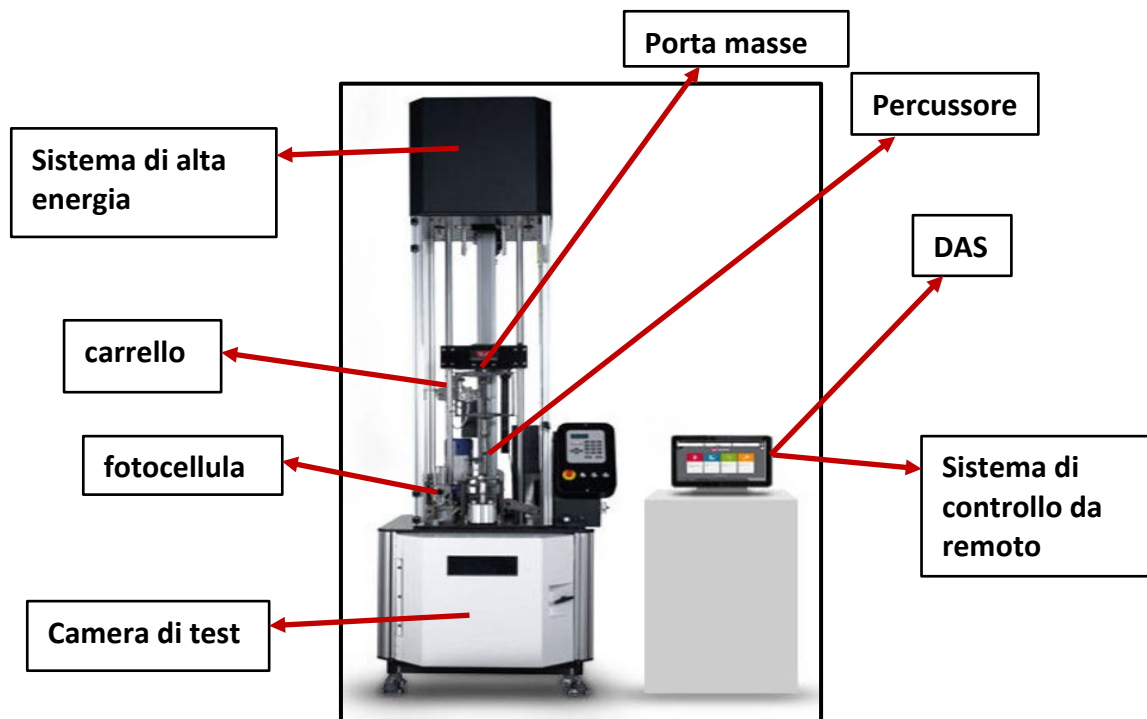


Figura 5: Instron Drop Tower 9450

Vengono ora analizzati più nel dettaglio gli strumenti costitutivi della torre:

- **Percussore**
- **Camera di test (può essere termostata)**
- **Fotocellula**
- **Carrello**
- **Porta masse**
- **Sistema di alta energia**
- **Sistema di controllo della macchina on board**
- **Sistema di controllo remoto da pc (mediante software Bluehill Impact)**
- **Sistema di acquisizione dei dati della cella di carico (DAS)**

L'apparecchiatura utilizzata per i test di crashworthiness, come si può intuire, svolge un ruolo importante nei test, non bisogna però trascurare l'importanza del setup sperimentale che parte dalla misura del provino e dal clampaggio dello stesso all'interno del supporto, sino all'impostazione corretta dei parametri di prova. Si analizza ora in dettaglio il setup che porta all'ottenimento di una curva di crash e successivamente verrà mostrato qualche esempio di curva registrata mediante il DAS di Instron.

Il primo step per eseguire un test di crash è la misura delle dimensioni del provino.

Una volta misurato il provino mediante l'utilizzo di un calibro, si procede con il suo clampaggio all'interno del supporto.

In funzione del tipo di prova che si intende svolgere si possono disporre le colonne anti-buckling ad altezze e posizioni differenti lasciando una diversa altezza non supportata nella parte inferiore del provino. Inoltre, per ridurre al minimo l'attrito ed eliminare così un parametro che potrebbe influire sul test si utilizzano solitamente colonne rivestite con un sottile strato di lubrificante solido al PTFE (Teflon, ovvero polimero fluorurato con un bassissimo coefficiente di attrito). Per il clampaggio all'interno del supporto si procede disponendo il provino con il trigger rivolto verso il basso e avvitando le quattro viti di fissaggio fino a battuta.

A questo punto si procede stringendo le viti con chiave dinamometrica impostata ad un certo valore di coppia (solitamente 1 Nm, corrispondenti a una forza di afferraggio di 4 kN) in modo da effettuare dei test con la stessa forza di clampaggio ed evitare un attrito eccessivo che porterebbe a maggiori valori di energia. Fissate le colonne ed il provino si monta il supporto sulla base della torre all'interno della camera termostatica e la si chiude.

Il prossimo step è quello di creare un nuovo metodo con il software Bluehill Impact della torre e impostare i parametri di test in funzione della tipologia di prova che si vuole svolgere.

Sono stati eseguiti test di impatto variando diversi parametri quali:

- Tipologia di materiale e spessore
- Velocità di impatto
- Massa di impatto
- Energia di impatto
- Forza di clampaggio
- Altezza del provino non supportata

Alcuni test prevedono velocità di impatto superiori a 5 m/s, in questo caso interviene il sistema di alta energia che è dotato di molle che forniscono ulteriore energia al carro oltre quella potenziale gravitazionale.

La scelta del parametro da variare dipende da vari fattori quali possono ad esempio essere lo spessore del provino e la tipologia di materiale analizzato.

Un accorgimento estremamente importante per la misura corretta della velocità di impatto è il posizionamento della fotocellula, la quale deve essere posizionata in modo che il coltellino superiore del carro sia ad un'altezza leggermente inferiore a quella del laser della fotocellula in modo che la misura della velocità avvenga esattamente nell'istante dell'impatto e non prima. Prima di proseguire con la trattazione, viene spiegato brevemente il funzionamento della fotocellula e viene mostrata in **Figura 6** una didascalia raffigurante tale sistema di misura:

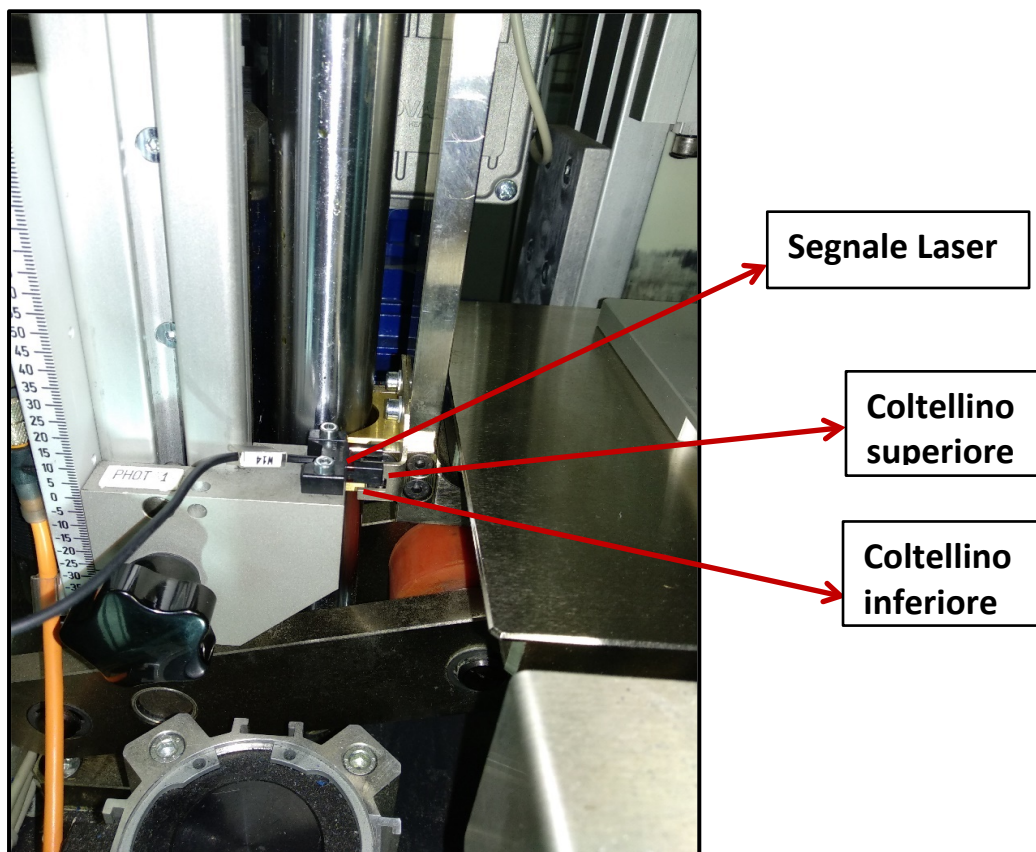


Figura 6: Sistema di acquisizione della velocità di impatto

Essa è dotata di due coltellini posti ad una distanza nota e di un laser la cui interruzione provocata dal passaggio dei coltellini solidali al carro determina l'inizio della misura di velocità del carro. Finché il percussore non raggiunge la posizione tale per cui il segnale laser viene interrotto dal passaggio del carro la fotocellula non misura, appena la parte superiore del carro interrompe il segnale parte la misura e il sistema valuta il tempo che intercorre tra il passaggio dei due coltellini solidali con il carro (tempo di interruzione del laser) e calcola la velocità di impatto come il rapporto tra la distanza nota tra i due coltelli e il tempo impiegato tra il passaggio del primo e quello del secondo.

Posizionata correttamente la fotocellula, in modo da conoscere l'esatta velocità di impatto, si può procedere con il test vero e proprio.

Il test può essere fatto partire attraverso il sistema di controllo on board oppure mediante il controllo da remoto, in questo caso le prove sono state eseguite con controllo da remoto, che prevede l'utilizzo di un software esterno per il controllo del test e l'acquisizione dei dati. È molto importante ricordare che la cella di carico integrata all'interno del percussore è in grado di misurare dei valori di forza, di conseguenza le curve Forza-Tempo possono essere ottenute direttamente dalla misura della forza associata ad un certo istante temporale, mentre le curve Forza-spostamento sono ottenute mediante integrazione numerica, come mostrato in eq. (1)

$$\delta(t) = \delta_0 + V_0 * t + \frac{1}{2} * g * t^2 - \int_0^t (\int_0^t \frac{F(t)}{m} dt) dt \quad (1)$$

Per capire a fondo questa equazione si considerino i termini presenti al suo interno. δ_0 è lo spostamento iniziale, all'istante 0, che fisicamente rappresenta l'istante in cui avviene l'impatto tra il percussore ed il provino. V_0 è la velocità al tempo 0, quindi la velocità di impatto del percussore con il provino. Il terzo termine dell'equazione descrive semplicemente la caduta del percussore soggetto al potenziale gravitazionale terrestre. Il termine più interessante di questa equazione è sicuramente l'ultimo, il quale rappresenta la decelerazione subita dal percussore a causa dell'impatto con il provino. Tale decelerazione viene misurata dalla cella di carico in termini di forza, tra le due grandezze vi è di mezzo solamente la massa del percussore. Partendo dai valori di forza, il software bluehill Impact li divide per la massa del percussore, e successivamente esegue una doppia integrazione numerica nel tempo in modo da calcolare il terzo termine dell'eq. (1), e di conseguenza la posizione del percussore ad un certo istante, coincidente con lo spostamento del provino. Fino ad ora è stato descritto in maniera dettagliata il setup sperimentale completo necessario all'ottenimento di una curva di crash.

Ultimo, ma non meno importante è assicurarsi che sia acceso e funzionante il DAS (Data Acquisition System), che per questo lavoro acquisisce i dati alla frequenza di 1MHz, ovvero ogni mille punti temporali registrati scorre 1 ms.

Si può ora procedere con l'analisi di una curva di crash mediante qualche esempio.

In **Figura 7** e **Figura 8** vengono mostrate le curve rispettivamente nel dominio del tempo e degli spostamenti ottenute per un provino con matrice in resina epossidica rinforzata con fibre di vetro:

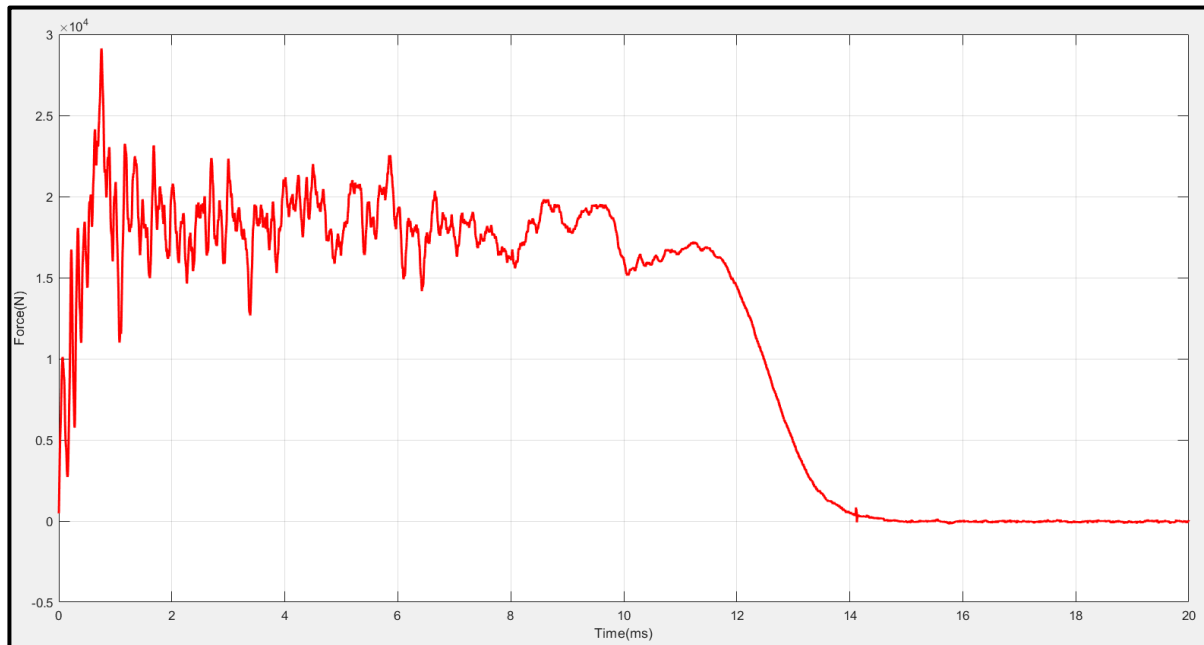


Figura 7: Curva tipica Forza-tempo di una prova crashworthiness

Come si può osservare in **Figura 7** la curva nel dominio del tempo è molto schiacciata, questo è indice del fatto che il test è stato eseguito ad una velocità di impatto abbastanza elevata e quindi la durata del test è molto piccola. In questo caso il test è stato eseguito a circa 5 m/s su di un provino con spessore 3 mm.

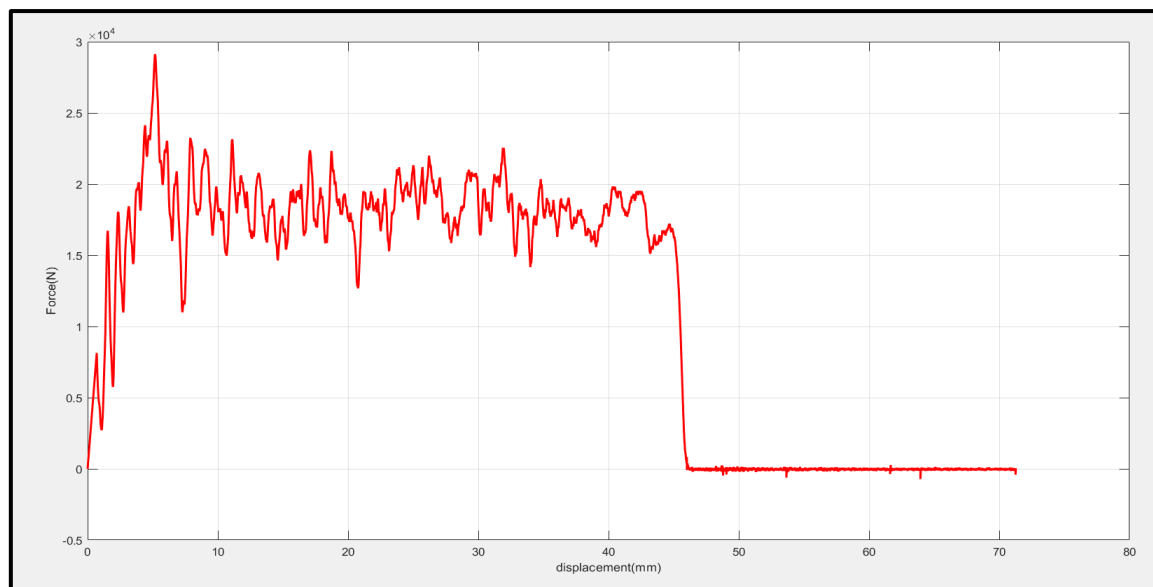


Figura 8: Curva tipica Forza-spostamento di una prova crashworthiness

Osservando invece la curva nel dominio degli spostamenti, **Figura 8**, si può notare un picco di forza nella zona iniziale, questo è dovuto alla presenza del trigger, il quale ha la funzione di favorire un certo meccanismo di frattura. La geometria acuta del trigger causa picchi di forza che possono anche arrivare a 100kN.

Dopo il picco di forza si ha una discesa della forza sino ad una stabilizzazione attorno ad un valore medio. Il tratto in cui le oscillazioni sono ridotte e la variazione del valore di forza è minima viene definito tratto utile, importante per il calcolo della SEA. Successivamente al tratto utile si colloca il tratto di discesa della forza, dovuta al progressivo arresto del percussore sino ad arrivare a valori di forza praticamente nulli, indice che il percussore ha terminato la sua corsa e dissipato tutta l'energia iniziale. Un aspetto che verrà spiegato in dettaglio nel successivo capitolo è quello legato alle oscillazioni della forza, ben visibili sulle curve appena mostrate. Per questo momento ci si limita ad affermare che tali vibrazioni sono relative alla dinamica del sistema durante il test di crash.

Quanto detto sino ad ora è un esempio che dimostra come osservando le curve di crash si possano intuire determinati comportamenti del provino e valutare quali zone della curva devono essere escluse dall'analisi per il calcolo della SEA. Verrà ora mostrato un altro esempio di curve sperimentali, ottenute per un provino in fibra di carbonio con spessore **2.5 mm**, **Figura 9** e **Figura 10**, mediante la drop tower 9450:

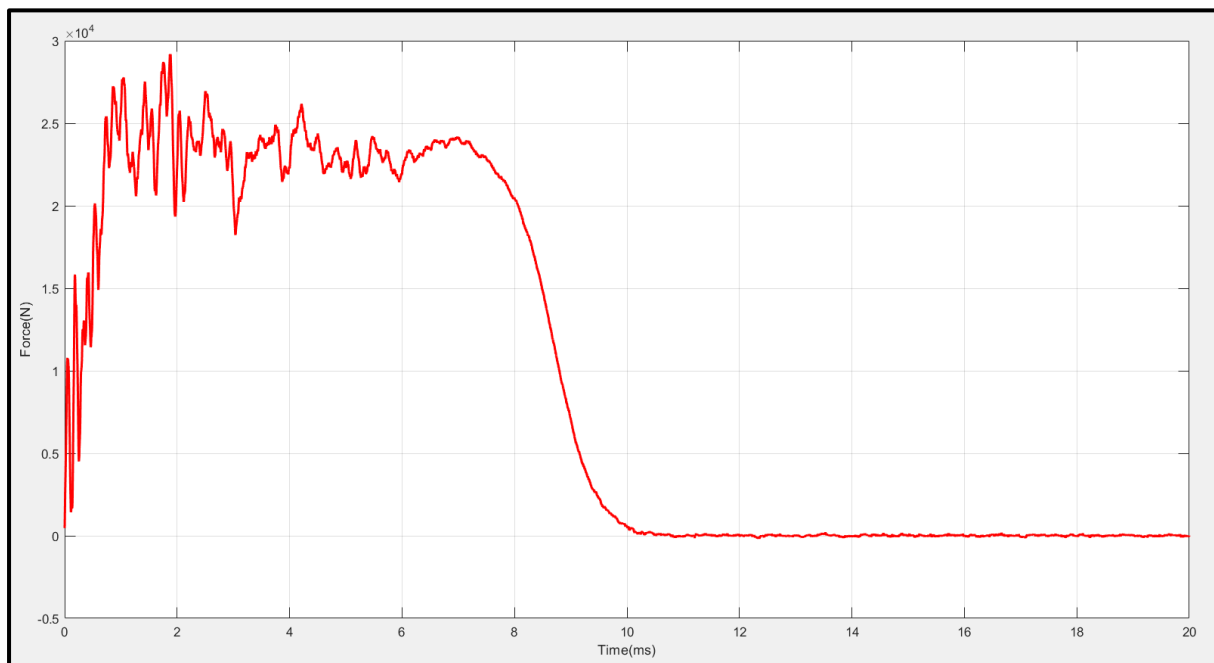


Figura 9: Curva Forza-tempo di una prova crashworthiness

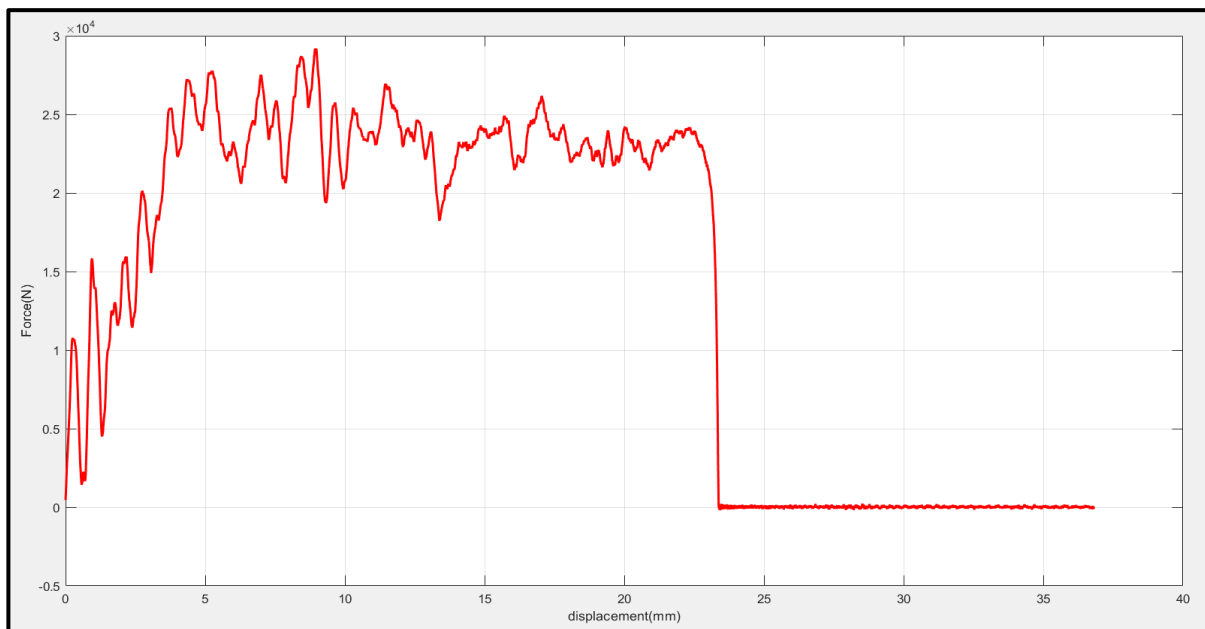


Figura 10: Curva Forza-spostamento di una prova crashworthiness

Analizzando le ultime due curve mostrate si nota una similitudine per la curva forza-tempo con la curva precedente, mentre per quanto riguarda la curva forza-spostamento ci sono sicuramente delle differenze. Prima di tutto si osservi come la forza media risulta più alta in **Figura 10**, e in secondo luogo si noti come il tratto di frattura progressiva sia nettamente più corto nella stessa didascalia. Tutto questo indica che il provino ha uno spessore maggiore rispetto al caso precedente, **Figura 8**.

Quanto accennato sino ad ora è solo un preludio di quello che è il cuore di questo lavoro di tesi, il quale verrà descritto in maniera dettagliata nei capitoli successivi.

2. Filtraggio delle curve di crash

Prima di discutere nel dettaglio la parte relativa all'individuazione del tratto utile delle curve di crash per il calcolo della SEA è assolutamente necessario trattare la problematica della rumorosità di tali curve in quanto essa potrebbe influire in maniera negativa sul funzionamento dell'algoritmo.

L'elevato rumore del segnale di forza nei test crashworthiness è dovuto alla dinamica del sistema; infatti, appena il percussore collide con il provino nascono una serie di vibrazioni nel percussore che vengono registrate dalla cella di carico integrata al suo interno e vengono poi visualizzate sulla curva finale.

Si rende quindi necessario l'utilizzo di strumenti matematici per il filtraggio delle curve in modo che risultino più leggibili dall'operatore incaricato di svolgere i test e che non vadano ad interferire con l'algoritmo nel calcolo della SEA.

Questo ultimo aspetto verrà approfondito in seguito quando si parlerà in dettaglio del funzionamento dello script.

Terminato questo breve preambolo sul problema della rumorosità, si può procedere con la descrizione delle operazioni eseguite e degli strumenti matematici utilizzati per filtrare il segnale ed ottenere così delle curve non influenzate dalla dinamica del sistema.

Quando si ha a che fare con un segnale, l'elemento di partenza da analizzare è sicuramente lo spettro in frequenza, in quanto fornisce informazioni estremamente importanti riguardo le frequenze che lo costituiscono e mette in mostra quali sono quelle che costituiscono il rumore e non sono quindi parte dell'informazione utile della curva.

Per tale scopo è stato utilizzato il software per calcolo scientifico MATLAB, ed in particolare l'applicazione di signal processing integrata nel software.

Qui di seguito vengono riportati rispettivamente in **Figura 11** e **Figura 12** un esempio di curva di crash prima di essere filtrata e del relativo spettro in frequenza di Fourier:

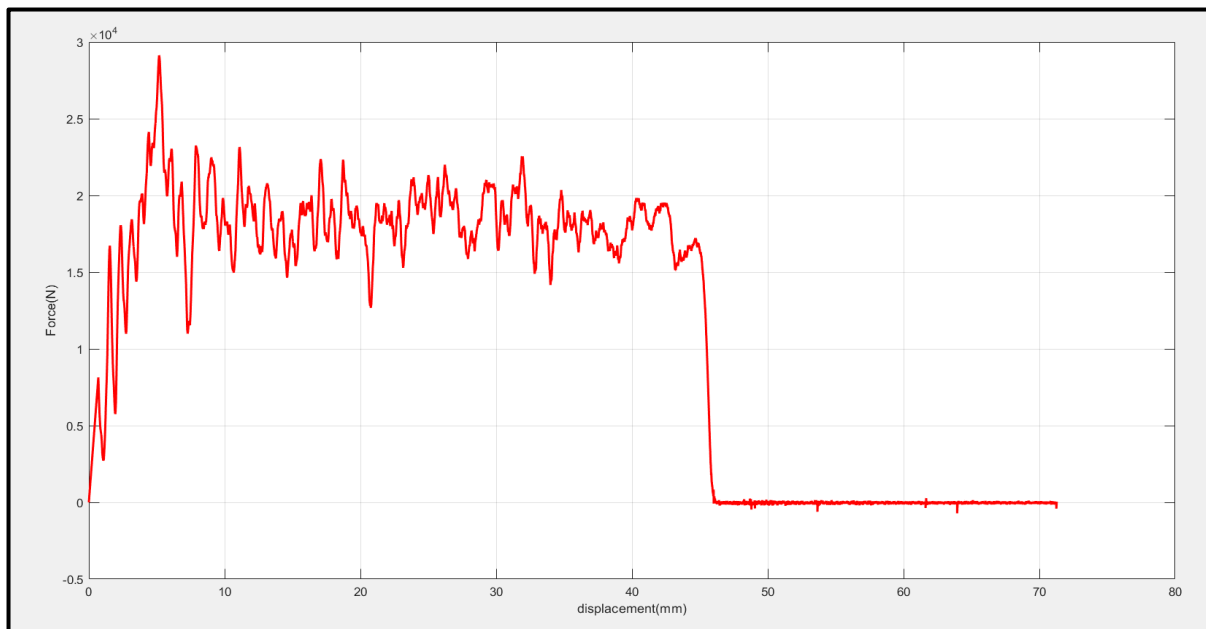


Figura 11: Curva Forza-spostamento non filtrata

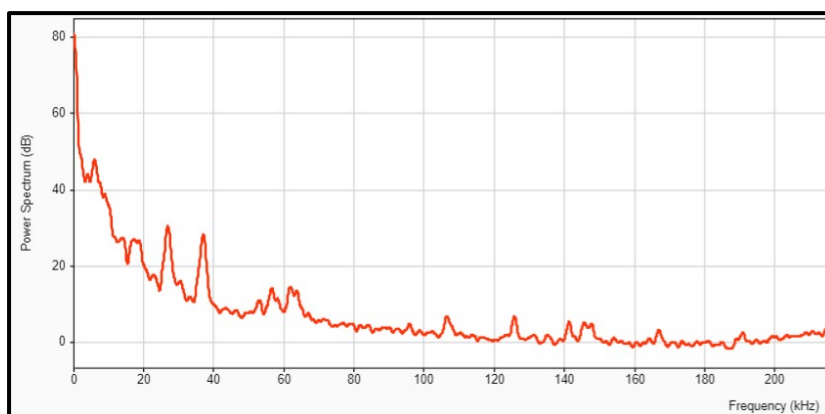


Figura 12: Spettro in frequenza di una tipica curva di crash

Come si osserva in **Figura 11** la curva non filtrata presenta delle oscillazioni della forza molto ampie per i motivi citati precedentemente ad inizio del capitolo, questo rende molto complessa la lettura della curva e l'individuazione del tratto utile per il calcolo della SEA.

Per semplicità di rappresentazione è stata riportata la curva Forza-spostamento e non Forza-tempo in quanto quest'ultima risulta molto più schiacciata sull'asse delle ascisse vista la piccolissima durata dell'impatto e non risultano ben visibili le oscillazioni del segnale.

Questa scelta non influenza in nessun modo le considerazioni che possono essere fatte anche sulla curva forza-tempo in quanto l'operazione di filtraggio agisce solamente sui valori di forza, indipendentemente dal fatto che essa venga rappresentata nel dominio del tempo o nel dominio degli spostamenti.

Osservando invece la **Figura 12** si evince come le frequenze più rumorose e che quindi disturbano maggiormente il segnale sono situate nell'intervallo tra uno e quaranta kHz. Considerando questo punto di partenza è stato applicato un filtro bandstop nella finestra di frequenze tra uno e quaranta kHz.

In **Figura 13** viene mostrato il risultato di questa prima operazione di filtraggio sulla curva Forza-spostamento.

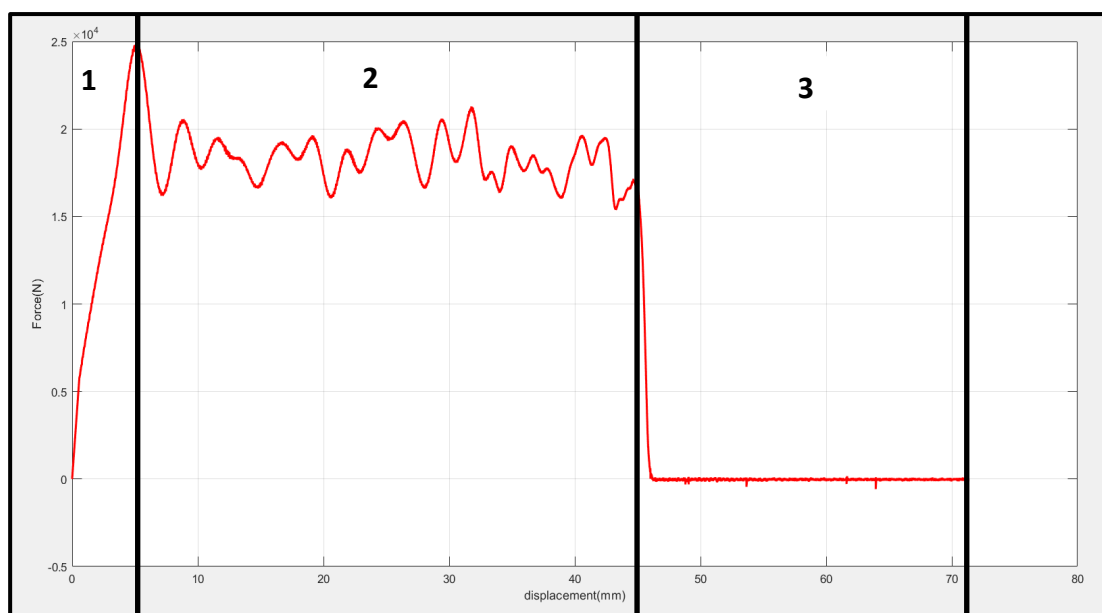


Figura 13: Curva Forza-spostamento dopo la rimozione delle frequenze più rumorose

Dopo aver eliminato la finestra di frequenze più rumorose risalta subito come le ampie oscillazioni ben visibili in **Figura 11** si sono ridotte al minimo e la curva risulta sicuramente più leggibile ed interpretabile.

L'operazione di filtraggio ha permesso di individuare i tratti caratteristici di una curva di crash, i quali sono principalmente tre:

- **Tratto 1:** porzione di curva che va dallo zero degli spostamenti fino a circa lo spostamento coincidente con la dimensione del trigger (nel caso specifico di questa curva il trigger è un triangolo isoscele con altezza 5 mm).
- **Tratto 2:** parte centrale della curva, in cui la forza ha un andamento circa costante.
- Questo tratto è rappresentativo della frattura progressiva del provino ed è proprio la zona di interesse per il calcolo della SEA.
- **Tratto 3:** comprende tutta la porzione di curva che va dalla fine del tratto utile sino all'ultimo degli spostamenti calcolati. Solitamente questo tratto è ben visibile in quanto si ha un rapido decremento della forza dovuto al progressivo arresto del percussore sino a quando il valore di forza tende a zero. A questo punto il percussore è completamente fermo e quindi la cella di carico non misura più alcuna accelerazione.

Ricapitolando quanto fatto fino a questo momento, si è applicato un bandstop nella finestra tra uno e quaranta kHz per ridurre il rumore e le oscillazioni della curva.

La qualità del segnale può essere ulteriormente migliorata in modo da rendere meno acuti i picchi, eliminare il rumore non rimosso dall'operazione di bandstop ed evidenziare così al meglio il comportamento del provino durante il test. Al fine di raggiungere tale obiettivo è stato adoperato uno strumento di smoothing. Il setup dei parametri in questa operazione è molto delicato in quanto anche piccole variazioni dell'ampiezza della finestra di smoothing può portare ad un'eccessiva perdita di informazioni da parte della curva. La finestra di smoothing è definita come il numero di punti appartenenti ad una certa finestra che si muove lungo il campo di esistenza della curva. Per chiarire le idee si consideri l'esempio riportato in **Figura 14**:

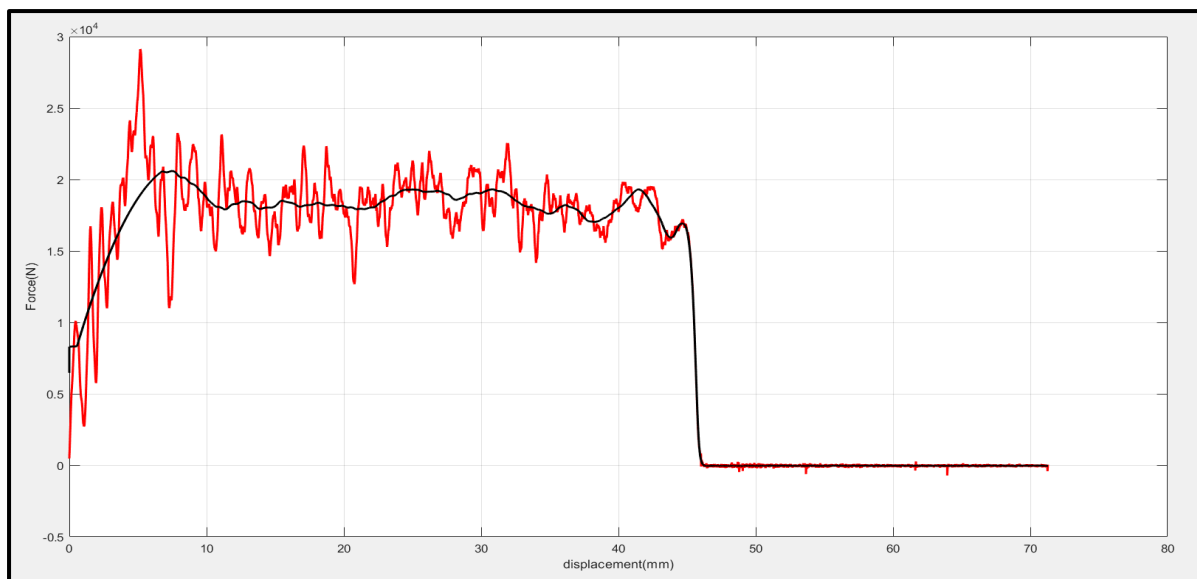


Figura 14: Curva Forza-spostamento dopo smoothing eccessivo

Nella precedente didascalia è stato effettuato uno smoothing del segnale utilizzando l'algoritmo di Savitzky-Golay e una finestra di duemila punti, ciò significa che sono stati presi in considerazione dall'algoritmo di filtraggio duemila punti alla volta fino alla fine della curva. Osservando la **Figura 14** si può dedurre che l'algoritmo utilizzando da MATLAB per lo smoothing riesce molto bene ad attenuare i picchi e a renderli meno acuti, però la finestra di punti utilizzata non è ottimale in quanto si ha un'eccessiva perdita di informazioni.

Partendo dal primo tratto della curva si nota come il picco iniziale non risulta praticamente distinguibile dai picchi presenti nel tratto centrale, questo è un fenomeno assolutamente non desiderato in quanto non permette di individuare agevolmente il tratto utile della curva e il tratto di influenza del trigger.

Sulla base delle osservazioni fatte sino ad ora si è ricercata una finestra di smoothing ottimale che permettesse di conservare le informazioni fondamentali della curva di crash eliminando solamente il rumore dell'ambiente circostante. Dopo essere state testate numerose finestre e analizzate molteplici curve è stata trovata come finestra ottimale una finestra di 700 punti, sempre utilizzando lo stesso algoritmo di filtraggio. Viene ora mostrato in **Figura 15** la stessa curva di crash filtrata con i parametri ottimali:

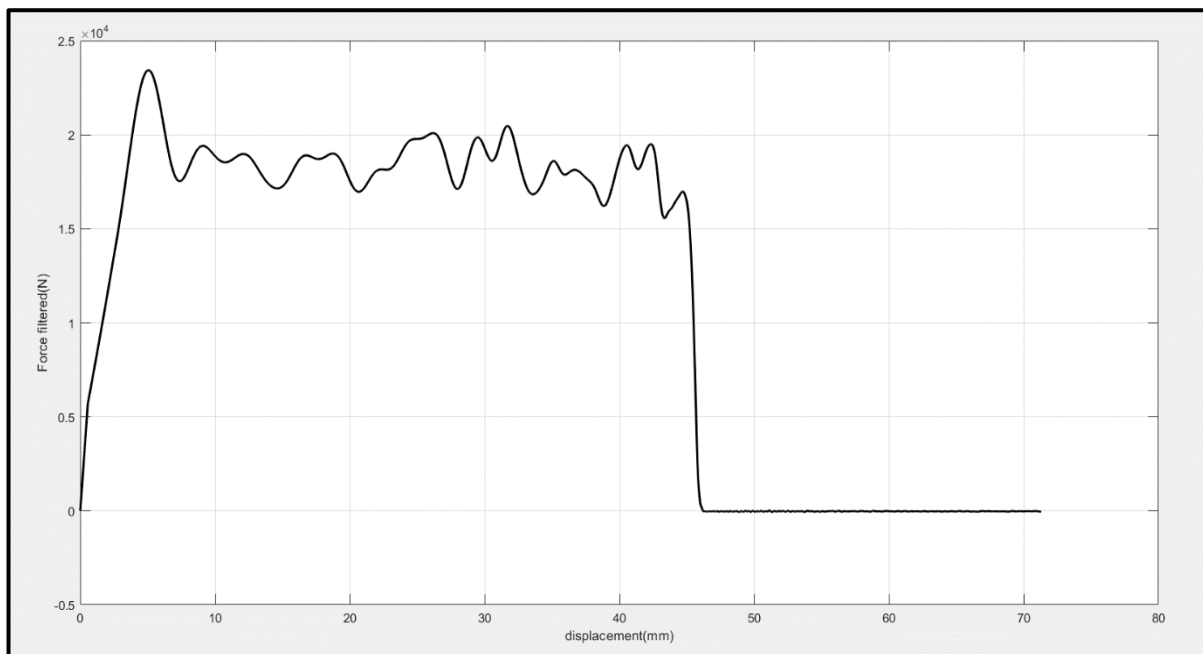


Figura 15: Curva Forza-spostamento dopo filtraggio con parametri ottimali

Confrontando la **Figura 14** con la **Figura 15** si può osservare come nella prima non si riconoscono visibilmente i tratti caratteristici della curva e ci sia una perdita di informazioni riguardanti il comportamento del provino durante il test a differenza di quanto avviene nella seconda.

Come ultima parte di questo capitolo riguardante il filtraggio delle curve di crash verrà mostrata una didascalia in **Figura 16** in cui sono sovrapposte sullo stesso grafico una curva di crash non filtrata e la stessa curva dopo un filtraggio completo, bandstop e smoothing, eseguito con parametri ottimizzati.

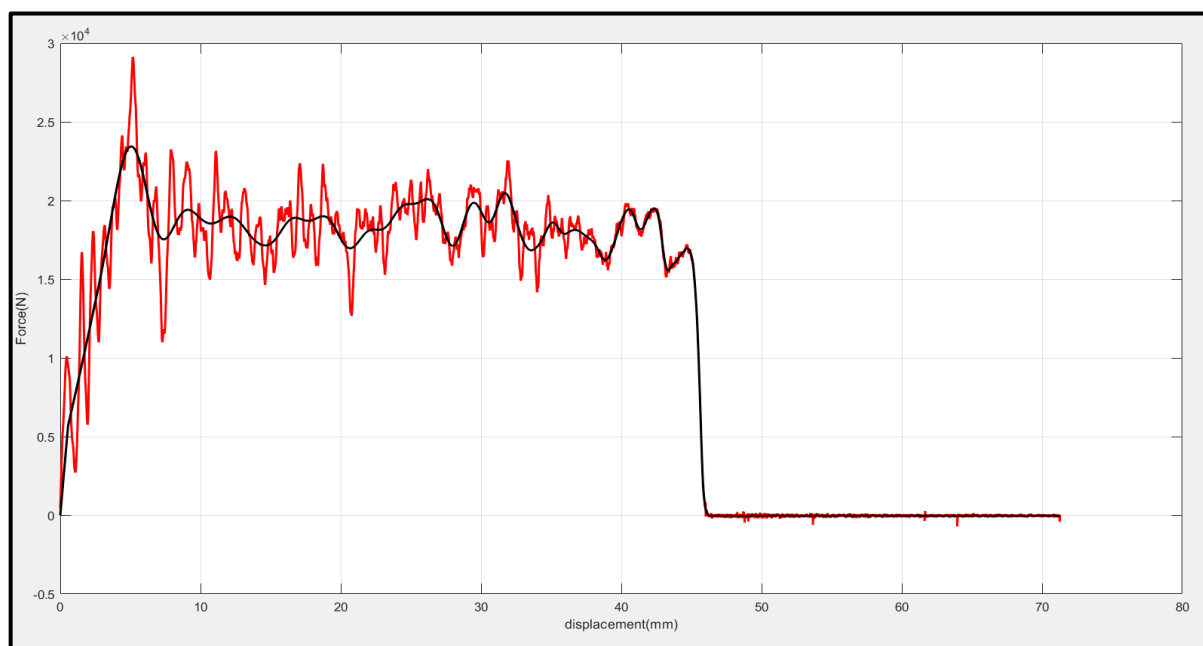


Figura 16: Curva Forza-tempo non filtrata e filtrata sovrapposte

Da questo ultimo confronto viene messo in risalto la differenza tra una curva di crash non filtrata ed una curva di crash dopo il filtraggio. L'operazione di filtraggio del segnale di forza permette di ridurre al minimo le oscillazioni e questo è un primo grosso vantaggio in quanto garantisce all'algoritmo oggetto della tesi di lavorare in condizioni ottimali, annullando il rischio di calcolo di spostamenti errati per il calcolo della SEA. Inoltre, si può osservare come l'applicazione del filtro conserva perfettamente l'andamento originale della forza e questo è un aspetto estremamente importante in quanto un andamento differente da quello reale porterebbe ad un calcolo errato degli spostamenti da parte dell'algoritmo e quindi ad un valore errato della SEA.

Prima di concludere in maniera definitiva questo capitolo relativo al filtraggio, merita spazio un approfondimento riguardo gli strumenti di bandstop e di smoothing utilizzati per filtrare le curve di crash. Lo strumento di bandstop, come già accennato in precedenza, permette di eliminare un range di frequenze rumoroso, che disturba il segnale di forza. Lo strumento di smoothing invece, ha la funzione di "arrotondare" i picchi e renderli meno acuti. Questo garantisce un funzionamento ottimale dell'algoritmo e limita variazioni brusche del segnale di forza, le quali possono portare ad anomalie nel calcolo della SEA. La funzione e l'importanza dei vari parametri che compaiono nelle successive equazioni sono state già discusse precedentemente all'interno del capitolo.

I relativi comandi Matlab di bandstop e smoothing sono riportati qui di seguito, bandstop (2) e smoothing (3):

(2)

$y = \text{bandstop}(x, f_{min}, f_{max}, fs)$

(3)

$y = \text{smoothdata}(x, \text{algorithm}, w)$

Legenda:

- y : vettore che restituisce in output la funzione bandstop o smoothing
- x : vettore che la funzione bandstop o smoothing prende in input
- f_{min} : limite inferiore del range di frequenze da tagliare
- f_{max} : limite superiore del range di frequenze da tagliare
- fs : frequenza di campionamento
- algorithm : algoritmo di smoothing utilizzato dalla funzione

Concludendo questo capitolo si evince come la parte relativa al filtraggio è notevolmente influente sul resto del lavoro di tesi; quindi, averla sviluppata nella maniera ottimale ha permesso di proseguire con la successiva parte del lavoro, che riguarda il calcolo vero e proprio della SEA e lo sviluppo dell'algoritmo in questione, senza alcun problema e rallentamento.

3. Implementazione dell'algoritmo

3.1. Ricerca bibliografica

Questo capitolo costituisce il cuore di questo lavoro di tesi in quanto descrive in maniera approfondita tutti i passaggi concettuali che hanno permesso di raggiungere e sviluppare la versione finale dell'algoritmo utilizzato per individuare il tratto utile delle curve di crash e per il calcolo della SEA.

Come è stato accennato in precedenza, fino a questo momento non è mai stato definito uno standard riguardo il tratto di curva da prendere in considerazione, questo perché ogni curva presenta un andamento leggermente differente in funzione dei parametri di prova e dei meccanismi di frattura coinvolti nel test. Per questo motivo il confronto tra diversi materiali compositi nel campo della crashworthiness ha sempre subito l'influenza dell'operatore che osserva ed analizza le curve di crash scegliendo in base alla propria esperienza quali parti della curva forza-spostamento considerare e quali invece scartare. Differenti operatori possono analizzare una stessa curva di crash in maniera simile ma mai identica, ciò non permette un confronto oggettivo nemmeno tra due provini dello stesso materiale oltre che tra materiali differenti. Questa lacuna nel campo della crashworthiness ha spinto la solida collaborazione tra il Politecnico di Torino e Instron, volta alla definizione di un test standard e del calcolo della SEA mediante un algoritmo che sia in grado di individuare il tratto di plateau della forza di una curva di crash in maniera autonoma ed univoca, senza l'ausilio di un operatore esterno. Il raggiungimento di tale obiettivo elimina una delle grosse problematiche relative alla standardizzazione dei test crashworthiness, anche se ci sono in gioco molti altri parametri in fase di elaborazione.

La fase di implementazione dell'algoritmo ha inizio con lo studio e l'analisi delle metodologie di calcolo della SEA sino ad ora adottate, mediante un'approfondita ricerca bibliografica. Analizzando il lavoro pregresso svolto nel campo della crashworthiness dei materiali compositi si è constatato che nella maggior parte dei paper pubblicati in letteratura non è specificato il criterio di scelta di un certo tratto di curva e del calcolo della SEA, ma vengono semplicemente riportati i risultati ottenuti con diversi materiali compositi e posti a confronto. A titolo di esempio vengono ora riportati alcuni esempi di analisi delle curve ritrovati in bibliografia. Il primo esempio citato riguarda lo studio condotto da [4] su provini realizzati in CFRP (matrice polimerica rinforzata con fibre di carbonio) con geometria tubolare. In **Figura 17** vengono raffigurati alcuni dei risultati ottenuti da [4] mediante prove sperimentali e simulazioni agli elementi finiti (FEM):

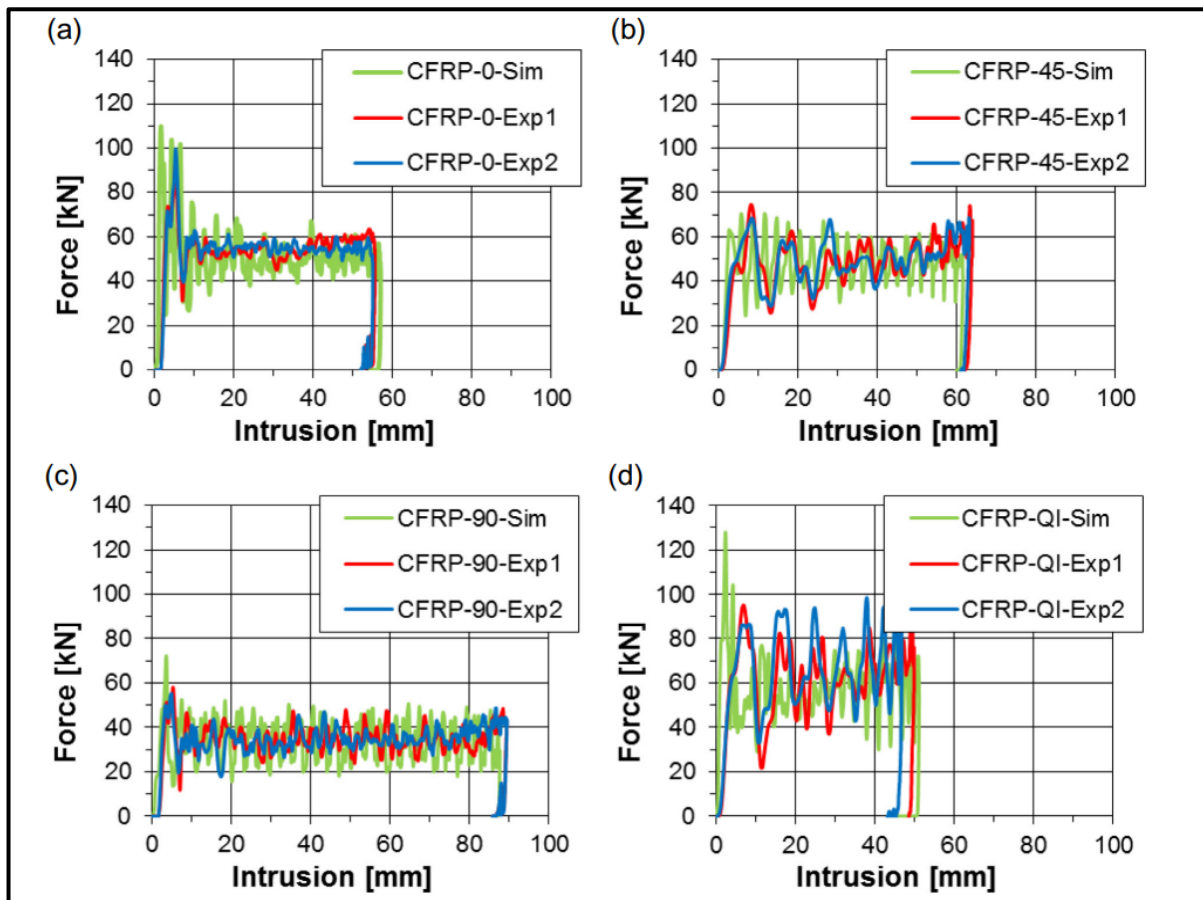


Figura 17: Curve Forza-spostamento sperimentali e simulate ottenute su provini con geometria tubolare [4]

Osservando la **Figura 17** si può asserire che le curve ottenute sono utilizzabili per il calcolo della SEA in quanto non presentano andamenti anomali. Inoltre, le curve simulate si sovrappongono abbastanza bene alle curve sperimentali. In questo studio però non viene specificato il tratto di curva preso in considerazione per il calcolo ma vengono solamente riportati i risultati finali; quindi, formalmente non è stato utilizzato alcun criterio univoco ed oggettivo[4].

Un altro esempio che viene riportato è lo studio approfondito sulla standardizzazione dei test crashworthiness svolto da [5]. In questo caso viene indagata la possibilità di utilizzare provini con geometria differente da quella tubolare, ad esempio con sezioni auto supportanti come già citato nel capitolo 2, in quanto riducono i meccanismi di frattura indesiderati e risultano più semplici da produrre sottoforma di pre-impregnati. Riguardo il calcolo della SEA, anche in questo caso vengono confrontati diversi provini in termini di assorbimento di energia ma non viene specificato il criterio di scelta del tratto di curva di crash da considerare.

Considerazioni analoghe possono essere fatte su molti altri articoli ritrovati in letteratura [6]-[7].

Un punto di partenza per lo sviluppo e implementazione dell'algoritmo può essere considerato lo studio condotto da [8], in cui vengono testati provini con geometria tubolare a varie energie di impatto. In questo articolo, per tutti i test svolti è stato individuato un tratto di frattura progressiva del provino nel range di spostamenti tra **10 mm** e **40 mm**. Viene di seguito raffigurata in **Figura 18** la curva media ottenuta da [8] dopo aver effettuato una serie di test su provini dello stesso materiale, dalla quale è stata poi calcolata la SEA:

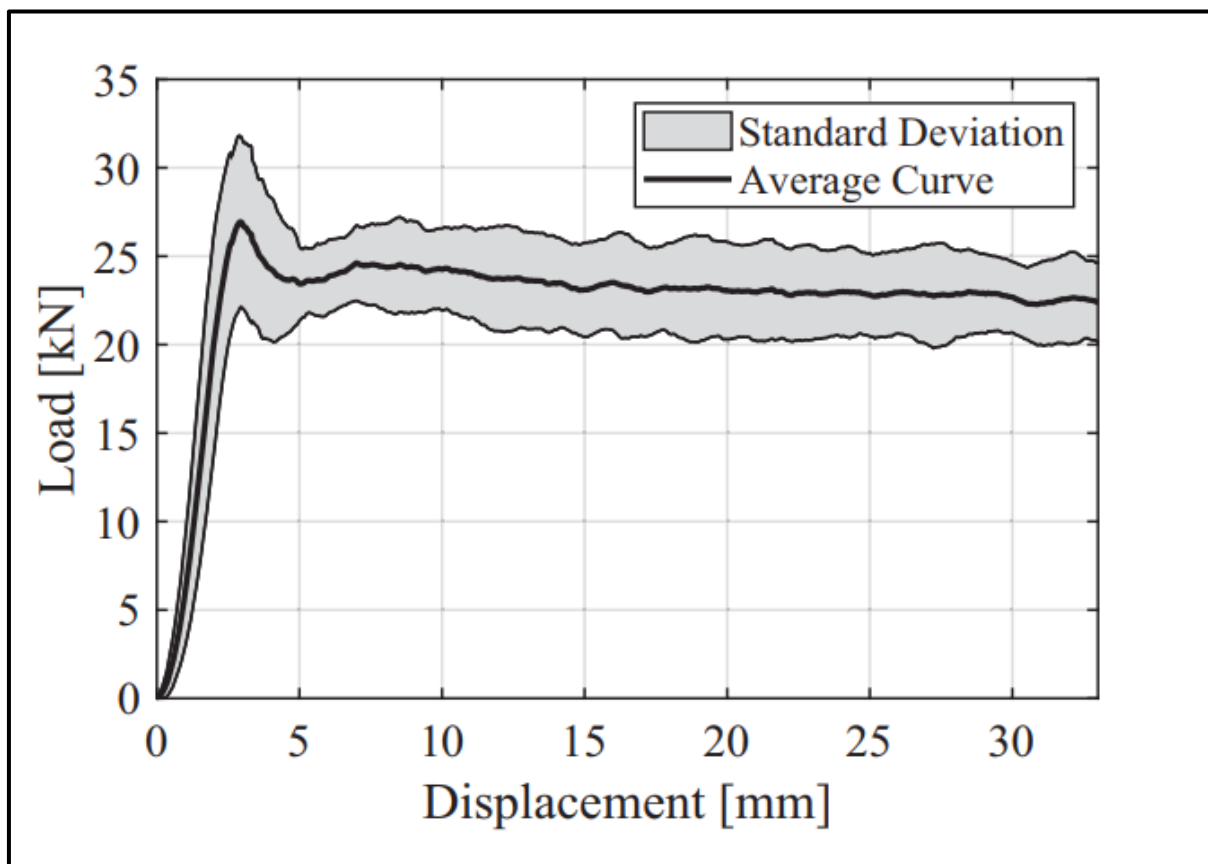


Figura 18: Curva media Forza-spostamento ottenuta su provini con geometria tubolare CFRP [8]

I vari test effettuati da [8] sono stati condotti in condizioni quasi statiche, ovvero a bassa velocità di deformazione, in modo da individuare più facilmente la zona di frattura progressiva del provino. Dalla curva in **Figura 18** vengono eliminate, ai fini del calcolo della SEA, le parti con lunghezza inferiore a **10 mm** e superiore a **40 mm**, per mettere in evidenza solamente la parte in cui si ha una frattura stabile e progressiva. La SEA è stata calcolata utilizzando L'eq. (3), scritta in due forme differenti in modo da evidenziare la definizione basilare di SEA, ovvero il rapporto tra l'energia assorbita dal provino e la massa di provino fratturato:

(4)

$$SEA = \frac{F_{mean} * \delta}{\rho * A * \delta} \qquad SEA = \frac{Energy}{m_{crush}}$$

- **F_{mean}**: Valore di forza media calcolato sul tratto di rottura progressiva del provino
- **δ**: Delta di spostamenti calcolato come differenza tra spostamento finale ed iniziale del tratto utile della curva
- **ρ**: Densità del provino
- **A**: Sezione resistente del provino
- **Energy**: Energia assorbita dal provino durante l'impatto
- **m_{crush}**: Massa del provino fratturato

Per questo caso specifico il delta di spostamenti vale **30 mm**. Come accennato in precedenza è stato considerato questo come punto di inizio per l'implementazione dell'algoritmo. La prima versione dello script prende quindi come limite inferiore e superiore di spostamenti dei valori inseriti dall'operatore e calcola la SEA come mostrato nell'eq. (4).

Quello esposto in [8] è in realtà solo uno dei criteri che possono essere utilizzati ai fini del calcolo della SEA in una prova di crashworthiness. È infatti immediato comprendere come possa essere non adatto ad effettuare un confronto tra due curve di lunghezza molto diversa quali possono essere ottenute ad esempio da prove effettuate su due materiali con spessore molto diverso a parità di energia di impatto. Nei seguenti sotto capitoli verranno infatti descritte ed analizzate in dettaglio tutte le versioni dei vari algoritmi sviluppati che prevedono criteri diversi per l'identificazione del tratto utile della curva, mettendo in luce le problematiche che hanno spinto al miglioramento progressivo della prima versione per arrivare all'ultima.

3.2. Approccio algoritmico per il calcolo della SEA

3.2.1. Algoritmo con spostamenti imposti dall'operatore

L'approccio algoritmico al calcolo della SEA risulta essere lo strumento migliore per un'analisi completamente oggettiva delle curve di crash. In questo capitolo, come già anticipato, verranno illustrati nel dettaglio i passaggi concettuali che hanno permesso l'evoluzione dell'algoritmo che identifica in modo automatico il tratto migliore della curva di crash ai fini del calcolo della SEA. Partendo dalla metodologia più semplice, che richiede che il tratto utile della curva sia identificato dall'operatore, si arriva all'ultima versione, in cui il codice in maniera autonoma analizza una qualsiasi curva di crash, ne individua il tratto utile ed è in grado di calcolarne la SEA. Gli unici dati richiesti in input per il funzionamento dell'algoritmo sono la densità, la sezione e la dimensione del trigger del provino, che possono variare in funzione della campagna sperimentale condotta. Tutti gli algoritmi sviluppati all'interno di questo lavoro sono basati su script sviluppati tramite il software Matlab.

Per prima cosa l'algoritmo a spostamenti imposti importa dalla directory di Matlab tutte le curve che si vogliono analizzare e crea una tabella in cui vengono memorizzati i valori di SEA calcolati per ogni curva.

Terminato l'import delle curve vengono richiesti all'utente i dati del provino necessari per il calcolo finale della SEA. I dati richiesti dal codice sono:

- **Densità del provino**
- **Sezione resistente del provino**
- **Altezza del trigger**
- **Spostamento iniziale**
- **Spostamento finale**

Si noti che i dati di input richiesti all'utente sono quelli medi misurati sul set di provini oggetto di analisi. In alternativa, è possibile analizzare un provino alla volta inserendo di volta in volta i dati relativi al singolo provino.

Si fa inoltre presente che le curve che costituiscono l'input del codice derivano dall'export di Bluehill Impact. Tale export contiene tutti i dati numerici relativi al test sottoforma di file con estensione 'csv' e un file 'pdf' riassuntivo di tutte le prove e tutti i parametri utilizzati per una determinata prova di crash.

Una volta che sono stati inseriti e memorizzati i dati precedentemente citati, all'interno dell'algoritmo è implementato un ciclo 'while' che per ogni curva adopera le seguenti operazioni:

- Import dei vettori: 'Time', 'Force', 'Displacement'
- Filtraggio della curva (descritto in maniera dettagliata nel capitolo 2)
- Individuazione del tratto utile
- Calcolo della SEA
- Memorizzazione del valore calcolato all'interno della tabella creata all'inizio dello script

La parte relativa al filtraggio è stata ampiamente descritta nel capitolo 2; quindi, in questo capitolo viene messa in risalto la componente algoritmica relativa al calcolo degli spostamenti iniziale e finale e di conseguenza della SEA. Si raffigura in **Figura 19** uno schema relativo al calcolo dell'energia assorbita dal provino mediante l'algoritmo che considera gli spostamenti imposti da un operatore, successivamente viene spiegato in dettaglio ogni step del diagramma di flusso:



Figura 19: Diagramma di flusso che descrive il funzionamento dell'algoritmo che considera gli spostamenti imposti da un operatore

Il primo comando eseguito è quello riportato al primo step in **Figura 19**, in cui vengono importate all'interno del codice le curve presenti nella directory di Matlab. Successivamente vengono acquisiti i dati in input dall'utente. Nella fase di calcolo vera e propria l'algoritmo individua la posizione relativa allo spostamento iniziale e finale inseriti dall'operatore, calcola la forza media e di conseguenza la SEA. Per far sì che il concetto sia perfettamente trasparente viene qui di seguito riportato un esempio di calcolo delle posizioni degli spostamenti iniziale e finale per la curva in **Figura 20**:

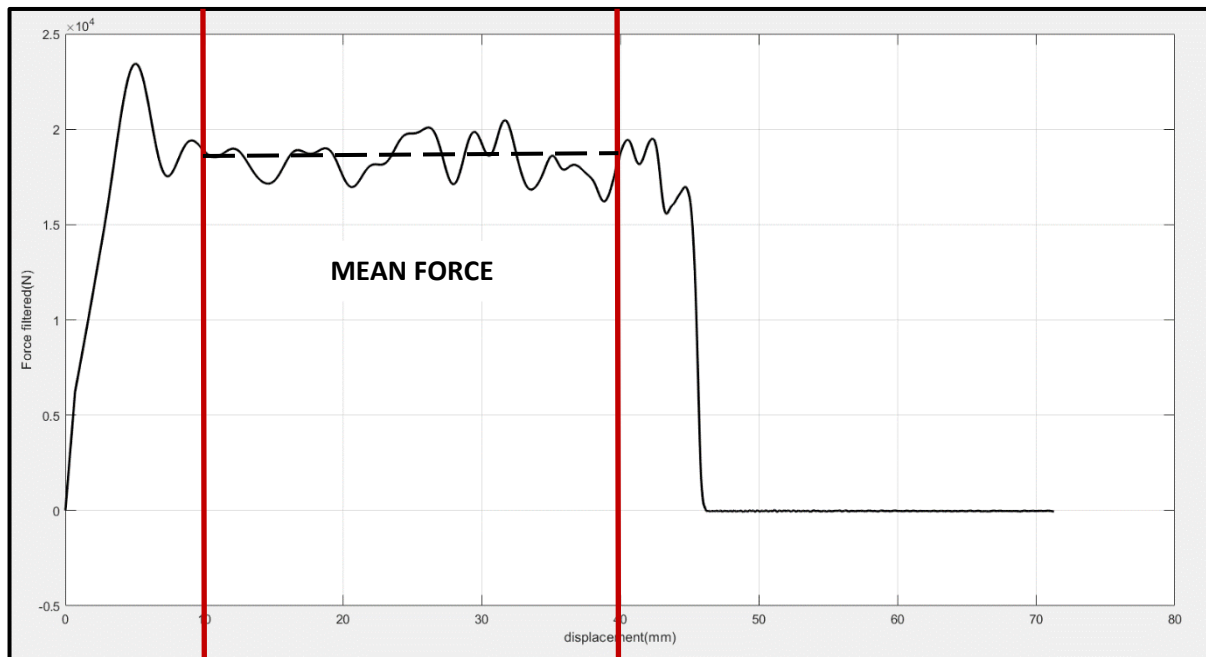


Figura 20: Tipica curva di crash Forza-Spostamento: algoritmo a spostamenti imposti

Per semplicità di rappresentazione è stata riportata la curva filtrata. Si supponga che un potenziale utente, osservando la precedente curva, decida di considerare il tratto di frattura progressiva del provino quello compreso tra i seguenti spostamenti:

- Spostamento iniziale: 10 mm
- Spostamento finale: 40 mm

A questo punto l'algoritmo mediante l'ausilio di un contatore individua in quale posizione del vettore degli spostamenti è collocato lo spostamento iniziale e lo spostamento finale. Dal calcolo risulta che lo spostamento iniziale si colloca alla posizione **1503**, mentre lo spostamento finale si colloca nella posizione **8453**, come si può osservare in **Figura 21**:

displacement	
62667x1 double	
	1
1503	9.9990
1504	10.0050
1505	10.0110
1506	10.0170
1507	10.0230
1508	10.0300
1509	10.0360
1510	10.0420
1511	10.0480
1512	10.0550
1513	10.0610
1514	10.0670
1515	10.0730
1516	10.0800
1517	10.0860

displacement	
62667x1 double	
	1
8453	39.9980
8454	40.0010
8455	40.0030
8456	40.0060
8457	40.0080
8458	40.0110
8459	40.0130
8460	40.0160
8461	40.0180
8462	40.0200
8463	40.0230
8464	40.0250
8465	40.0280
8466	40.0300
8467	40.0330

Figura 21: Posizioni spostamento iniziale e finale all'interno del vettore displacement

Osservando la **Figura 21** si noti che gli spostamenti individuati non sono esattamente **10 mm** e **40 mm** ma **9.99 mm** e **39.99 mm**. Questo risiede nel fatto che la funzione utilizzata per individuare la posizione di un certo spostamento si ferma nel caso limite allo spostamento ricercato, altrimenti individua quello precedente e non successivo, ma questo non influenza in alcun modo il calcolo della SEA.

Ora che sono state individuate le posizioni relative agli spostamenti inseriti in input l'algoritmo procede scansionando il vettore della forza non filtrata, inserendo in un altro vettore vuoto tutti i valori di forza all'interno dell'intervallo di posizioni determinato allo step precedente. Ci sono quindi tutti gli elementi per poter determinare la SEA. Lo step successivo prevede il calcolo della forza media mediante l'apposita funzione di Matlab e il calcolo vero e proprio della SEA secondo l'eq. (4). Quanto fatto sino ad ora rappresenta un approccio molto basilare all'analisi di una qualsiasi curva di crash, che entra in conflitto con l'obiettivo proposto da questo lavoro di tesi, ovvero la definizione di un metodo standard ed oggettivo per il calcolo della SEA.

Secondo l'approccio appena descritto un potenziale operatore inserirebbe manualmente in input lo spostamento iniziale e finale analizzando visivamente la curva e tagliando i tratti che secondo il suo parere non descrivono la frattura progressiva del provino.

Come si può ben intuire questa tipologia di analisi non è in grado di fornire dei risultati oggettivi, in quanto differenti operatori potrebbero individuare spostamenti diversi anche se simili per una stessa curva, in funzione dell'esperienza pregressa e della capacità di analisi di curve crashworthiness. Questo filone di pensiero non rispecchia le caratteristiche di uno standard, volto invece ad un'indagine metodica e una ripetibilità dei calcoli previsti. Nel mondo scientifico sono proprio le controversie e le problematiche il motore che spinge verso il cambiamento, funzionale al miglioramento di determinati criteri di analisi di un qualsiasi problema. Nel caso specifico di questo progetto, l'inefficacia del primo approccio algoritmico studiato è stato un campanello di allarme che ha portato alla ricerca di una nuova metodologia di calcolo degli spostamenti e della SEA data una qualsiasi curva di crash. Come in qualunque ambito di ricerca, lo sviluppo di nuove metodologie di calcolo prevede di procedere per gradi onde evitare di trascurare approcci validi, e di varare ogni possibilità al fine di non tralasciare fattori determinanti per l'analisi condotta.

3.2.2. Algoritmo che considera lo spostamento finale variabile

La prima variazione presa in considerazione applicabile all'algoritmo è stata imporre costante solamente uno dei due spostamenti e calcolare l'altro in maniera iterativa. Si è optato per mantenere costante lo spostamento iniziale, il quale sembra essere il meno influenzato dall'andamento della curva, e di partire da uno spostamento finale imposto ma di ricalcolarlo in maniera iterativa sino a convergere verso il tratto utile della curva.

Tutto il procedimento di calcolo verrà ora illustrato in maniera dettagliata, partendo dal diagramma di flusso in **Figura 22**. La sezione di codice relativa al filtraggio (capitolo 2) e al calcolo della SEA (**Figura 19**) sono le medesime del precedente algoritmo.

Come nel caso dell'algoritmo con spostamenti imposti, lo spostamento iniziale viene definito in input dall'utente e la posizione relativa ad esso viene calcolata in maniera analoga a quanto scritto al primo step della **Figura 19**. Lo spostamento finale, invece, viene calcolato in maniera iterativa partendo dal valore massimo del vettore displacement. Una volta definito lo spostamento finale l'algoritmo procede in maniera iterativa, calcolando la forza media nel range di spostamenti tra quello iniziale imposto e quello finale. Alla prima iterazione lo spostamento finale è quello massimo del vettore 'displacement'.

In ogni iterazione il codice confronta la forza media calcolata con un valore limite, che per questo momento è stato imposto come **10 kN**. Tale valore è stato settato considerando l'ordine di grandezza tipico delle forze in gioco sul tratto utile di una curva di crash. Durante il confronto dei due valori di forza si aprono due possibilità:

- **Forza media < Forza limite**
- **Forza media > Forza limite**

Finché si verifica la prima condizione l'algoritmo continua ad iterare riducendo lo spostamento finale di un valore definito in input sino al raggiungimento della seconda condizione, la quale implica che si è trovato uno spostamento finale abbastanza vicino al tratto utile tale per cui la forza media calcolata rispecchia l'ordine di grandezza tipico per una curva di crash. Terminata questa fase iterativa, il codice procede con il calcolo della SEA come descritto dall'eq. (4). Per cercare di comprendere meglio i passaggi logici dell'algoritmo appena descritto è utile riassumere quanto detto con un diagramma di flusso, che viene qui di seguito mostrato in **Figura 22**:

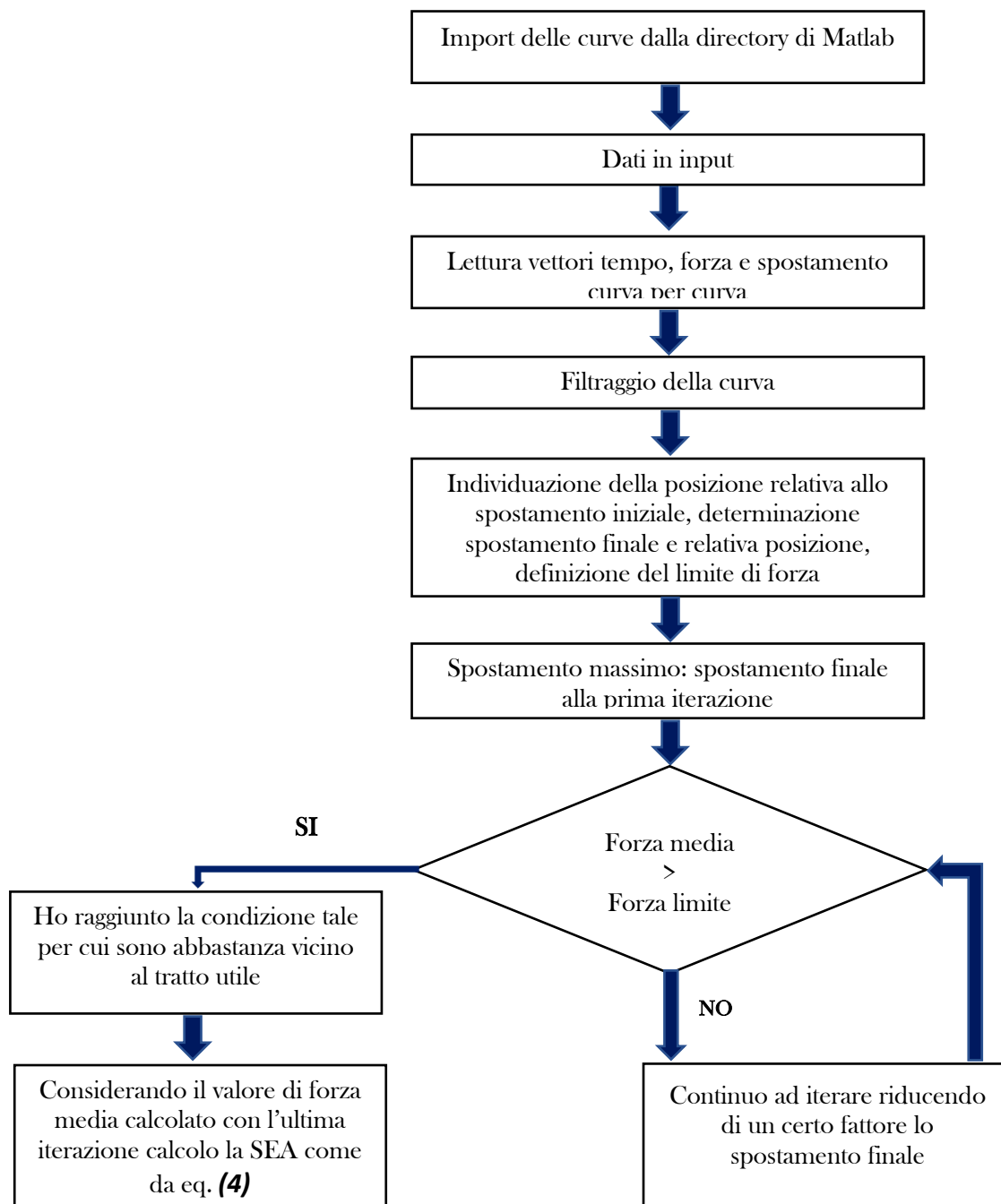


Figura 22: Diagramma di flusso relativo al codice che calcola la SEA imponendo lo spostamento iniziale e ricalcolando quello finale in maniera iterativa

L'algoritmo appena descritto è stato utilizzato per analizzare una serie di curve sperimentali in modo da verificarne il corretto funzionamento. Ciò che è emerso da tale analisi è uno spostamento finale calcolato alla fine delle iterazioni che non è situato perfettamente sul tratto di plateau della forza ma sul tratto di discesa della stessa. Questo è dovuto ad un decremento della forza che non è necessariamente immediato ma può essere graduale, e porta ad una sottostima del valore di SEA in quanto nel calcolo della forza media vengono considerati anche valori di forza non appartenenti al tratto utile.

Al fine di superare questo intoppo si è pensato di stabilire un ulteriore margine di riduzione dello spostamento finale una volta raggiunta la condizione limite. In

Figura 23 viene mostrato la porzione di codice che include questa scelta:

```

135 - force_final=zeros(1,(final_position-start_position)+1); %azzerò il v
136 - subtractive_factor= start_position-1;% definisco un fattore sot
137 - % inserisco all'interno del vettore force_final tutti i valori
138 - % forza nell'intervallo di interesse
139 - for k=start_position:final_position
140 -     force_final(k-subtractive_factor)=force(k);
141 - end
142 - new_calculated_force_average=mean(force_final);
143 - %valuto se la forza media calcolata alla h-esima iterazione e'
144 - %superiore o inferiore al limite imposto
145 - if new_calculated_force_average>=force_limit
146 -     final_displacement=final_displacement-decrease_1;

```

Figura 23: Porzione di codice che illustra l'effetto del margine sul calcolo dello spostamento finale

Dal confronto con il diagramma precedente, **Figura 19**, si nota immediatamente la differenza che riguarda il calcolo dello spostamento finale. Si osservino in particolare le righe 145-146 della **Figura 23**, quando viene raggiunta la condizione limite l'ultimo spostamento calcolato viene ulteriormente ridotto di un certo margine, che in questo caso è stato definito come “decrease_1”. Tale margine, a questo livello di sviluppo dell'algoritmo, è stato settato ad un valore di prova, richiede quindi un successivo processo di taratura. Una volta ricalcolato lo spostamento finale il calcolo della forza media e della SEA viene eseguito in maniera identica a quanto fatto in **Figura 19**. Sino a questo momento il funzionamento step-by-step dei due algoritmi in questione è stato spiegato esclusivamente in maniera letterale, con la conseguenza di una potenziale confusione del lettore che potrebbe non aver trattato tali argomenti in precedenza.

Per rendere più chiari i concetti si consideri un esempio applicativo di calcolo sulla curva raffigurata in **Figura 24**:

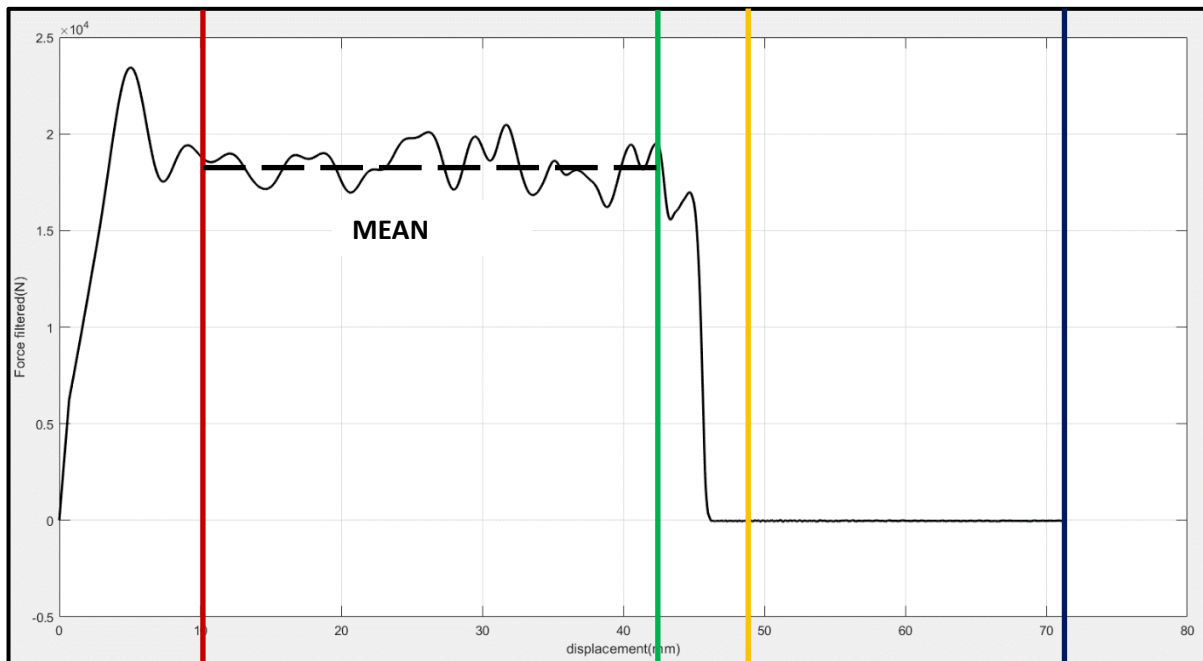


Figura 24: Confronto sul calcolo dello spostamento finale utilizzando oppure non utilizzando un valore di margine

Legenda:

- **Linea blu:** Spostamento massimo (71.29 mm)
- **Linea rossa:** Spostamento iniziale imposto dall'utente (10 mm)
- **Linea gialla:** Spostamento finale calcolato senza margine (47.29 mm)
- **Linea verde:** Spostamento finale calcolato considerando un margine di 5 mm (42.29 mm)

Una prima considerazione che si può fare osservando lo spostamento finale calcolato nei due casi è che nel primo caso, in cui non si applica alcun margine, non si riesce a convergere in maniera ottimale verso il tratto utile della curva. Appena si applica il margine si riesce invece a convergere abbastanza bene verso il plateau della forza. Il valore del margine utilizzato in questo calcolo è stato settato a **5 mm**, ma non è un valore definitivo in quanto per giungere alla versione finale dell'algoritmo sarà necessaria una taratura di tale parametro, come verrà descritto nel dettaglio in seguito.

Prima di proseguire con la trattazione si riassume quanto detto sino ad ora in modo da compattare tutti i concetti e comprendere a pieno il seguito. L'approccio algoritmico al calcolo della SEA ha avuto come punto di origine quanto osservato in letteratura, ovvero data una certa curva di crash, un qualsiasi operatore analizzando visivamente la curva, impone gli spostamenti iniziale e finale in input e calcola l'energia assorbita dal provino considerando tale range. Questa versione dell'algoritmo è stata considerata in contrapposizione con l'obiettivo dello standard, rendere oggettivo e ripetibile il calcolo della SEA, portando all'applicazione di alcune variazioni all'interno del codice. La prima modifica apportata è quella relativa al calcolo dello spostamento finale, che nella versione successiva alla prima viene calcolato in maniera iterativa partendo dallo spostamento massimo del vettore. Dall'analisi su diverse curve si è constatato che nonostante l'approccio iterativo lo spostamento finale calcolato non ricade sempre sul tratto utile, per questo motivo si è optato per la definizione di un margine di riduzione ulteriore che garantisca con maggiore probabilità la convergenza verso il plateau della forza.

3.2.3. Algoritmo con spostamenti variabili: prima versione

Lo step successivo verso il raggiungimento della versione finale dell'algoritmo è stato valutare di adottare il medesimo ragionamento iterativo anche sullo spostamento iniziale. Questa idea è divenuta concreta nel momento in cui si è constatato che su alcune curve di crash imporre lo spostamento iniziale a **10 mm** comporta una sottostima ulteriore della SEA in quanto si taglia una porzione eccessiva di tratto utile nella parte iniziale della curva. Al fine di visualizzare meglio questo concetto si consideri la curva in **Figura 25**:

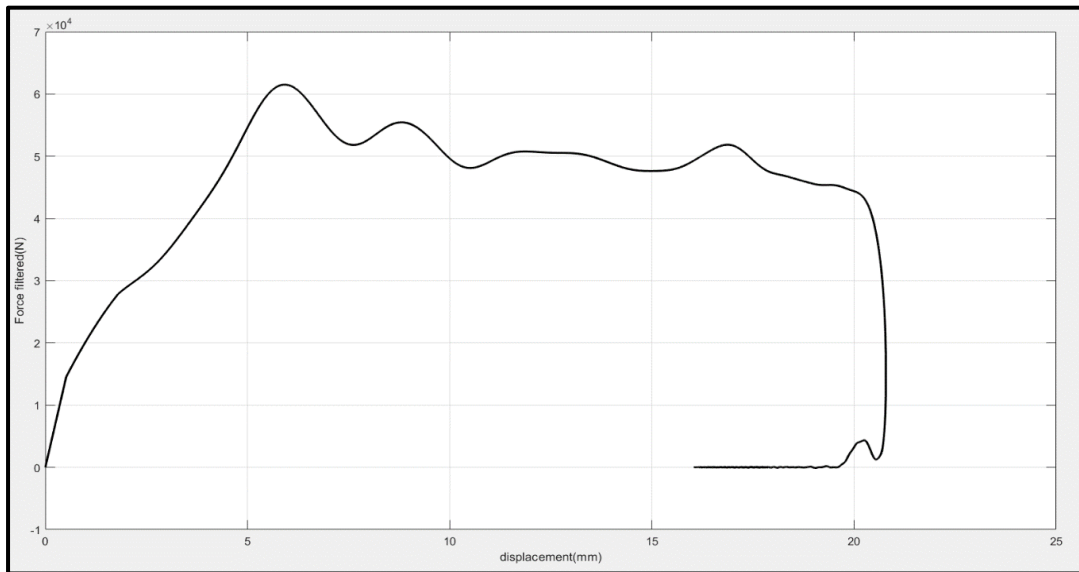


Figura 25: Curva di crash ottenuta per un provino G50(matrice in resina epossidica rinforzata con fibre di vetro, spessore 5 mm)

Osservando la curva appena mostrata il primo particolare che risalta è la lunghezza ridotta. Ciò è dovuto allo spessore elevato del provino, il quale porta allo sviluppo di una forza media maggiore nel tratto utile rispetto a provini più sottili, a spese di una corsa del provino nettamente inferiore. Per questa tipologia di curve è impensabile tagliare i primi **10 mm** di curva a priori, in quanto si trascurerebbe un ampio tratto di curva utilizzabile per il calcolo della SEA. Supponiamo di effettuare il calcolo del range di spostamenti utilizzando l'algoritmo descritto in **Figura 22** nel paragrafo precedente, il risultato è riportato qui di seguito in **Figura 26**:



Figura 26: Range di spostamenti calcolato mediante l'algoritmo in Figura 23

Legenda:

- **Linea rossa:** Spostamento iniziale inserito in input dall'utente (10 mm)
- **Linea verde:** Spostamento finale calcolato mediante iterazione (15.79 mm)

Dalla **Figura 26** risulta chiaro come si stia trascurando un'importante porzione di curva per il calcolo della SEA, anche perché all'errore sullo spostamento iniziale bisogna sommare quello derivante dal calcolo non preciso dello spostamento finale derivante da una mancata taratura dei parametri del codice, la quale è stata eseguita successivamente in linea temporale.

Per arrivare alla versione finale dell'algoritmo sarà ovviamente necessaria una taratura dei margini utilizzati, ma si procede con ordine seguendo quanto è stato fatto concettualmente anche negli step intermedi. In questa versione intermedia dell'algoritmo il metro di confronto con il limite di forza, sempre settato a **10 kN** come nell'algoritmo descritto al paragrafo precedente, risulta ancora essere la forza media. Di conseguenza il processo iterativo che porta al calcolo dello spostamento iniziale e finale non può essere effettuato in maniera separata vista la necessità di un range di spostamenti per il calcolo della forza media stessa.

Bisogna inoltre tenere in considerazione che l'andamento della forza nella parte iniziale e finale della curva è differente; quindi, non possono essere adottati gli stessi valori di margine, la cui funzione è stata definita nel precedente paragrafo.

Qui di seguito, in **Figura 27**, viene riportato il tratto di codice che definisce quale sia il margine più adatto da utilizzare per il calcolo dello spostamento iniziale:

```
70      %% CALCOLO FORZA MEDIA
71      %ragiono sullo spostamento iniziale imposto. Se e' maggiore di 5 aumento
72      %piu' lentamente in modo da perdere meno tratto dicurva possibile mentre se
73      %e' minore di 5 tengo un margine piu' alto in modo da non ritrovarmi prima
74      %del plateau della forza
75 -    increase_start=0;
76 -    increase_start_1=0;
77 -    if start_displacement>5
78 -        increase_start=0.2;
79 -        increase_start_1=0.2;
80 -    elseif start_displacement<=5
81 -        increase_start=0.02;
82 -        increase_start_1=5;
83 -    end
```

Figura 27: Tratto di codice che regola il valore del margine sul calcolo dello spostamento iniziale

Per comprendere al meglio il valore concettuale del tratto di codice riportato in **Figura 27** risulta necessario spiegare passo per passo i comandi eseguiti. Si parte con le prime due righe, **75** e **76**, nelle quali le due variabili 'increase_start' ed 'increase_start_1' vengono azzerate in modo da creare un'allocazione di memoria all'interno di Matlab. Prima di proseguire bisogna fare luce sulla funzione delle due variabili appena menzionate. La variabile 'increase_start' è l'incremento attuato ad ogni ciclo iterativo della variabile 'start_displacement', il cui valore di partenza è definito in input dall'utente. La variabile 'increase_start_1' invece è il margine utilizzato per il calcolo dello spostamento iniziale. La funzione del margine non viene nuovamente illustrata in quanto è stata ampiamente descritta nel paragrafo precedente.

Proseguendo con la spiegazione delle righe di codice, si considerino le righe **77-79**. Se lo spostamento inserito in input dall'utente è superiore a **5 mm**, allora i valori di 'increase_start' e 'increase_start_1' adottati sono quelli mostrati alle righe **78-79**. Al contrario, nel caso in cui lo spostamento iniziale inserito in input dall'utente sia inferiore o uguale a **5mm** i valori di tali variabili viene settato ai valori mostrati alle righe **81-82**. Il valore di riferimento nelle due condizioni, righe **77** e **80**, non è preso casualmente ma è stato scelto considerando che proprio i primi **5 mm** di curva Forza-spostamento sono quelli influenzati dalla presenza del trigger nel caso di provini piatti. Il valore delle variabili 'increase_start' ed 'increase_start_1' è stato definito in maniera spannometrica per testare il codice, e non è andato incontro al processo di taratura in quanto questo approccio è stato eliminato nella versione finale dell'algoritmo con spostamenti variabili. Quello che rimane da spiegare al meglio è il perché sia necessario adottare due margini differenti per il calcolo dello spostamento iniziale. A questo scopo, si consideri il seguente esempio pratico: si supponga che un operatore inserisca in input uno spostamento iniziale (start_displacement) superiore ai **5 mm**. In questo caso si può ben comprendere che il valore di partenza dell'iterazione sia oltre il tratto di influenza del trigger, di conseguenza molto vicini al tratto utile; quindi, non si può pensare di utilizzare un valore di margine elevato.

Nel caso in cui l'operatore inserisse un valore inferiore a **5 mm** ci si ritroverebbe nel tratto di influenza del trigger, in cui la forza è in fase di crescita e non si è stabilizzata. Per questo motivo, in questa casistica, si è optato per un valore di margine superiore, in grado di tagliare il tratto di curva in cui è presente il picco di forza causato dalla presenza del trigger nel provino.

Per chiarire al meglio delle possibilità quanto detto sino ad ora viene riportato in **Figura 28** il diagramma di flusso relativo al funzionamento del tratto di codice in **Figura 27**:

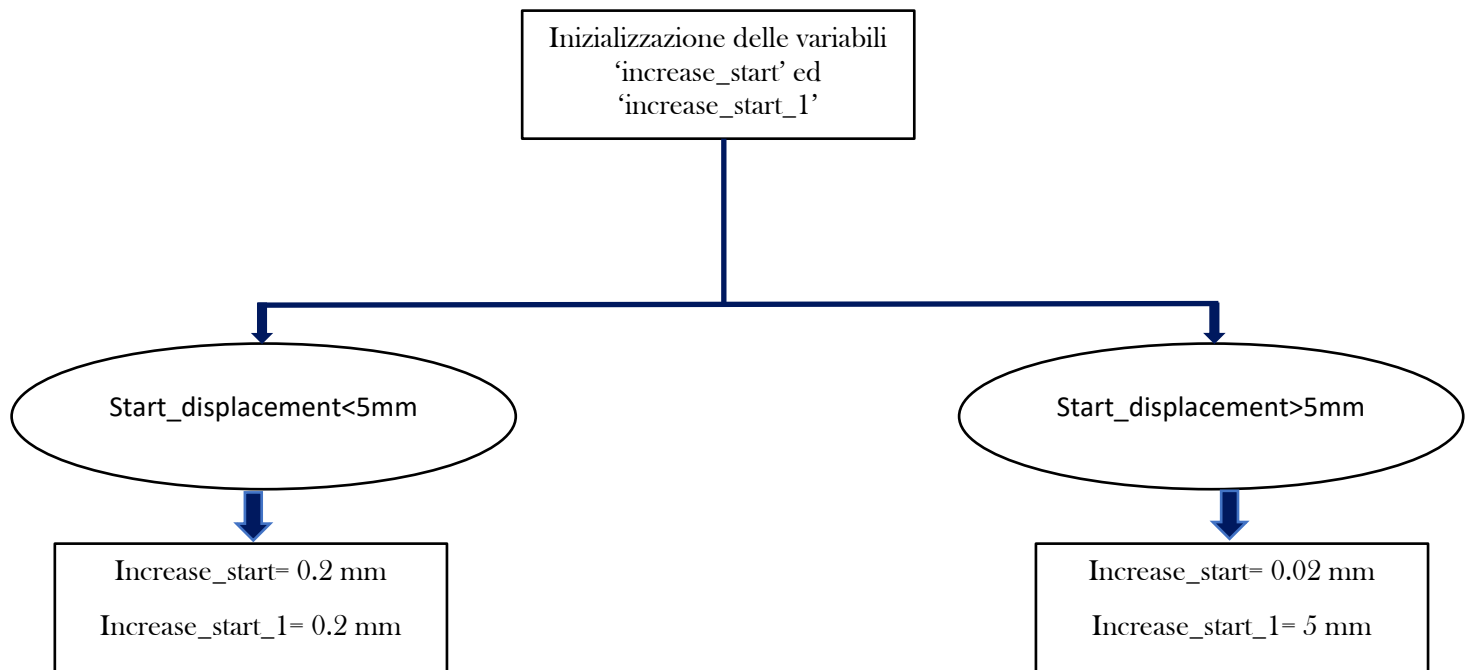


Figura 28: Diagramma di flusso relativo alla determinazione del margine da utilizzare per il calcolo dello spostamento iniziale

Quanto descritto sino ad ora è la porzione di algoritmo che varia rispetto a quello descritto precedentemente, la parte restante lavora esattamente come già descritto nello scorso paragrafo.

Riassumendo si può affermare che l'unica variazione apportata rispetto alla versione precedente è quella relativa al calcolo in maniera iterativa sia dello spostamento finale sia di quello iniziale, anche se con margini differenti per i motivi già citati precedentemente.

Pur utilizzando questo nuovo approccio, si rischia in ogni caso di calcolare valori di SEA non adeguati in quanto si fa riferimento alla forza media e non alla forza puntuale, chiariamo ora meglio questo concetto. Entrambe le ultime due versioni dell'algoritmo (paragrafo 3.2.2 e 3.2.3), che calcolano in maniera iterativa, ad ogni step confrontano la forza media con la forza limite (definita nel paragrafo 3.2.1) e questo può portare a considerare valori di spostamento iniziale e finale che non si trovano sul tratto utile. Un errore di questo tipo è dovuto al confronto di una media e non dei valori puntuali; infatti, la forza media calcolata può anche essere dell'ordine di grandezza richiesto per il confronto pur considerando punti che si trovano al di là del tratto piatto, coprendo in qualche modo l'inefficacia dell'iterazione. Si consideri ad esempio la curva raffigurata in **Figura 29**:

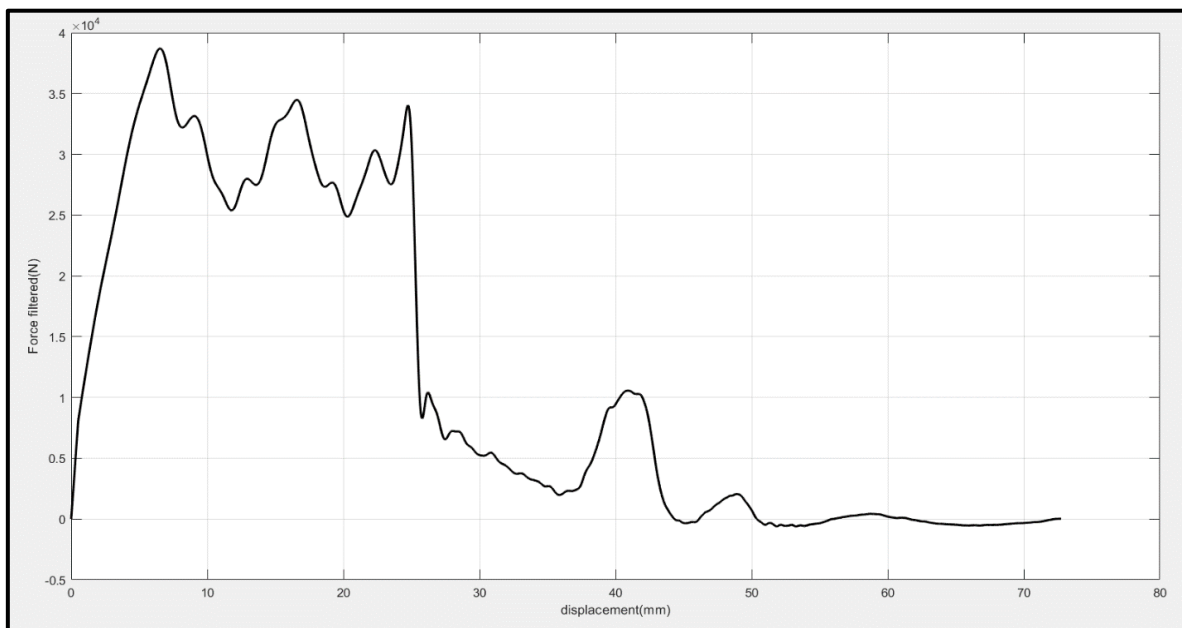


Figura 29: Curva di crash ottenuta su di un provino G30(matrice in resina epossidica rinforzata con fibre di vetro, spessore 3mm)

Tale curva si è provato ad analizzarla mediante l'algoritmo che considera variabili entrambi gli spostamenti, i risultati sono riportati qui di seguito in **Figura 30**:

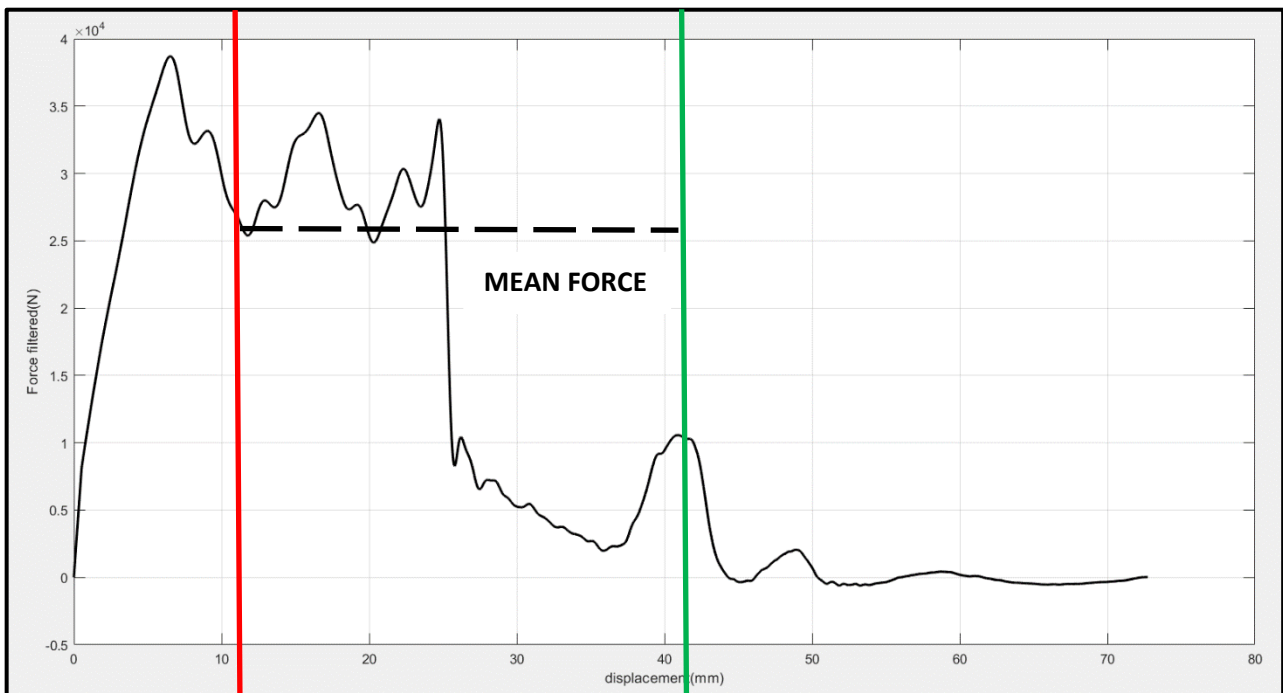


Figura 30: Risultati ottenuti mediante l'algoritmo che considera variabili entrambi gli spostamenti

Legenda:

- **Linea rossa:** Spostamento iniziale calcolato a fine iterazione (10.52 mm)
- **Linea verde:** Spostamento finale calcolato a fine iterazione (41.74 mm)

Osservando gli spostamenti calcolati si può chiaramente vedere che la curva non è stata tagliata in maniera ottimale per il calcolo della SEA. Lo spostamento iniziale taglia una porzione di tratto utile che andrebbe considerata (da circa 8 mm), ma la situazione più critica si riscontra sullo spostamento finale, il quale risulta completamente al di là del plateau della forza. Questa aberrazione nel calcolo, come anticipato, è dovuta all'utilizzo della forza media come metro di confronto, che tampona il considerare anche valori di forza molto bassi (sicuramente al di fuori del tratto utile). In questo caso specifico tutti i valori di forza oltre l'ultimo picco sul tratto utile non andrebbero considerati nel calcolo della media, però anche considerandoli la forza media calcolata continua ad essere dell'ordine della decina di kilo Newton e di conseguenza la condizione limite si verifica anche tagliando in maniera errata la curva. Tutto questo porta a calcolare valori di spostamento sbagliati e a sottostimare il valore reale di SEA. Nota: il picco nella zona finale della curva è dovuto ad una frattura del provino nella parte superiore oltre che in quella inferiore. Questo potrebbe causare problemi nel calcolo dello spostamento finale, ma questo problema è stato affrontato nel paragrafo 3.3 relativo al funzionamento della versione definitiva dell'algoritmo con spostamenti variabili.

3.2.4. Algoritmo con spostamenti variabili: seconda versione

Al fine di abbandonare il concetto di forza media e abbracciare quello di forza puntuale viene proposta un'ulteriore versione dell'algoritmo. La successiva versione del codice effettua sempre un confronto tra due valori di forza per individuare la condizione limite, però in questo caso il metro di confronto non è più la forza media ma bensì i singoli valori di forza. Così facendo si riesce a valutare in maniera sicuramente più realistica per quale valore di spostamento la condizione di confronto si verifica. Inizialmente si era pensato di partire a scansionare la curva dal primo elemento del vettore della forza (si ragiona sulla forza filtrata in quanto presenta un andamento più regolare e senza ampie oscillazioni rispetto alla forza non filtrata) sino ad arrivare alla posizione in cui il valore di forza scende sotto un certo limite scelto, solamente basandosi sull'osservazione di curve precedenti. Successivamente sono state apportate delle modifiche in questo senso, infatti si è scelto di imporre i seguenti estremi della curva entro i quali lavora il ciclo iterativo dell'algoritmo:

- **Posizione iniziale:** posizione corrispondente ad un valore di spostamento pari al trigger
- **Posizione finale:** posizione oltre la quale i valori di forza divengono negativi

La prima scelta è stata fatta considerando la funzione del trigger, ovvero inizializzare la frattura del provino. In conseguenza di questo il primo tratto di curva, che va dallo spostamento **0 mm** sino allo spostamento corrispondente all'altezza del trigger, può essere trascurato nell'analisi della SEA in quanto lungo la dimensione del trigger la frattura non è sicuramente progressiva ma bensì si ha una crescita repentina della forza sino a raggiungere un picco. Solamente superato il picco iniziale la forza tende a scendere e a stabilizzarsi sul tratto utile. La seconda scelta deriva dalla necessità di definire un valore di spostamento finale da cui l'algoritmo inizia ad iterare. Si è deciso di prendere lo spostamento tale per cui la forza diventa negativa. Questa scelta è stata adottata considerando che quando la forza assume valori negativi il test è già terminato, anche se la cella di carico continua a misurare per qualche altro milli secondo.

A titolo di esempio vengono mostrate, in **Figura 31**, le righe di codice relative ai processi iterativi sullo spostamento iniziale e sullo spostamento finale:

```

171 - %Iterazione sullo spostamento iniziale
172 - for i=start_position:final_position
173 -     if force_filtered(i)>=force_limit
174 -         start_displacement=displacement(i)+increase_start;
175 -         break
176 -     elseif force_filtered(i)<=force_limit
177 -         i=i+1;
178 -     end
179 - end
192 - %Iterazione sullo spostamento finale
193 - for j=final_position:-1:start_position
194 -     if force_filtered(j)>=force_limit
195 -         final_displacement=displacement(j)-decrease_final;
196 -         break
197 -     elseif force_filtered(i)<=force_limit
198 -         j=j-1;
199 -     end
200 - end

```

a

b

Figura 31: **a)** porzione di codice raffigurante il processo iterativo per il calcolo dello spostamento iniziale **b)** porzione di codice raffigurante il processo iterativo per il calcolo dello spostamento finale

Nell'intervallo di righe 172-179 in **Figura 31** viene eseguito dal codice il processo iterativo che porta alla determinazione dello spostamento iniziale da utilizzare per il calcolo della SEA. Il ciclo 'for', come si può osservare, ha come estremi le variabili chiamate all'interno del codice "start_position" e "final_position", le quali rappresentano i due limiti che sono stati precedentemente descritti. L'algoritmo confronta ogni valore di forza compreso in questo intervallo con il limite di forza, per il momento fermo a **10 kN**. Fino a che non trova un valore di forza superiore o uguale al limite continua ad iterare incrementando l'indice del ciclo, appena viene individuato un valore di forza che rispetta la condizione alla riga 173 in **Figura 31** l'iterazione viene interrotta e si considera come spostamento iniziale il valore collocato alla posizione i-esima nel vettore displacement. In realtà bisogna precisare che a tale spostamento viene aggiunto un certo margine, etichettato nel codice come "increase_start", avente la medesima funzione del margine utilizzato nelle due precedenti versioni dello script. Lo stesso identico ragionamento è stato implementato per il calcolo dello spostamento finale, con l'unica differenza che in questo caso l'iterazione parte dal fondo dell'intervallo e ragiona a ritroso. Sino ad ora l'algoritmo il ragionamento dell'algoritmo è stato spiegato a sprazzi, la descrizione dettagliata di ogni passaggio del codice verrà effettuata per la versione definitiva nel capitolo successivo.

Lasciando da parte il concetto di forza media ed utilizzando come indice di controllo la forza puntuale si è riusciti a rendere più accurato il calcolo del range di spostamenti per il calcolo della SEA. L'ultimo intoppo all'implementazione della versione definitiva dell'algoritmo si è presentato durante test crashworthiness su provini avente geometria tubolare. Questi provini sono realizzati sempre con matrice in resina epossidica rinforzata con fibre di vetro, però la resina e le fibre utilizzate possono essere differenti da quelle impiegate per i provini piatti. La versione dell'algoritmo realizzata sino a questo momento ha riscontrato delle difficoltà nel calcolo degli spostamenti iniziale e finale per alcune curve ottenute dai provini di geometria tubolare con spessore **1 mm**. Per comprendere al meglio il motivo dell'inefficacia del codice si consideri la curva raffigurata in **Figura 32**:

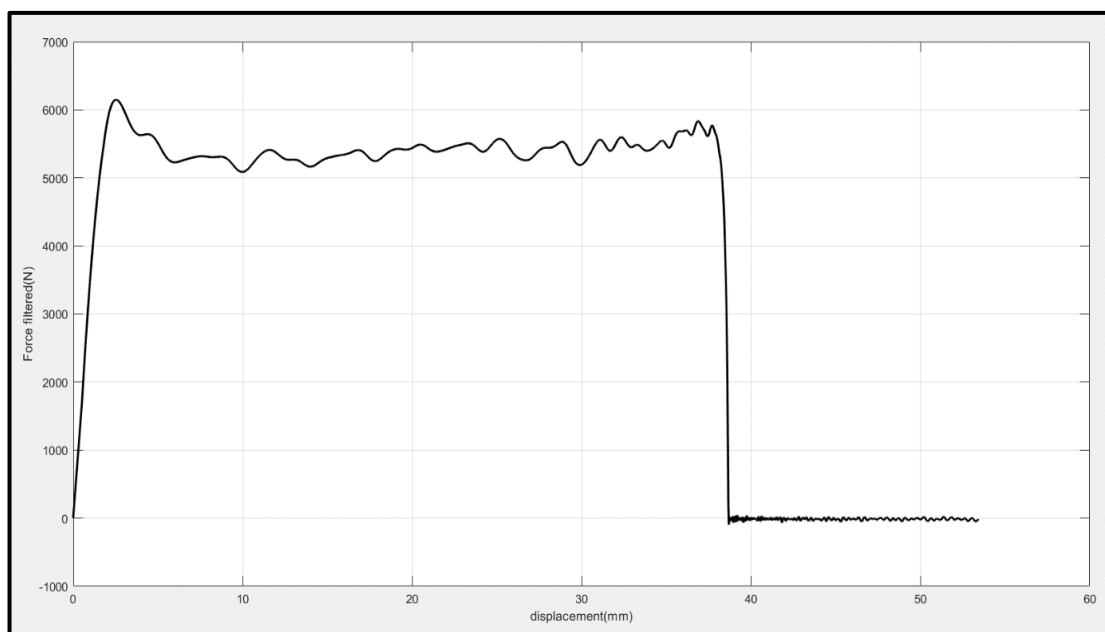


Figura 32: Curva di crash ottenuta su un provino di geometria tubolare e spessore 1 mm

Osservando la curva si noti come le forze in gioco non superano i **7 kN**, mentre il limite di forza utilizzato come confronto all'interno del codice è **10 kN**. Questa condizione è assolutamente critica per il funzionamento dell'algoritmo perché, come si può ben immaginare, la condizione di confronto definita in **Figura 22** non viene mai raggiunta. Di conseguenza le iterazioni relative allo spostamento iniziale e finale non arrivano in nessun modo a convergenza data la permanente condizione di forza *i*-esima inferiore alla forza limite e il calcolo fallisce.

Si è iniziato a pensare ad una soluzione alternativa per superare questo punto morto ed analizzando tutte le curve di crash ottenute sino ad ora si è giunti alla conclusione che il limite di forza non può essere definito in maniera assoluta, ma deve essere calcolato sulla singola curva oggetto di analisi.

L'ultimo controllo necessario al corretto funzionamento dell'algoritmo riguarda l'analisi del tratto di curva compreso tra il limite di forza e la fine della curva.

In particolare, è fondamentale che in questo tratto non si presentino valori di forza che siano uguali o superiori al limite definito all'interno codice. Nell'ultima parte di questo capitolo sono state accennate delle problematiche, la cui risoluzione porta alla luce la versione definitiva del codice. Essa non viene mostrata qui in dettaglio per non appesantire la trattazione. Tutto quello che riguarda il funzionamento dell'ultima versione dell'algoritmo sarà spiegato in maniera dettagliata e con accezione logica nel paragrafo 3.4.

3.3. Approcci alternativi al calcolo della Specific Energy Absorption (SEA)

Prima di entrare nel dettaglio del funzionamento dell'ultima versione dell'algoritmo con spostamenti variabili risulta utile illustrare due approcci alternativi per il calcolo della SEA, i quali verranno poi messi a confronto con la versione finale dell'algoritmo appena citata al fine di valutare quale delle tre metodologie di calcolo sia più efficace, in termini di minimizzazione della dispersione dei risultati, nel calcolo dell'energia assorbita dal provino. I due approcci alternativi sono elencati qui di seguito:

- Calcolo della SEA utilizzando spostamenti imposti in percentuale
- Calcolo della SEA considerando l'area sottesa a tutta la curva di crash

Il primo dei due approcci proposti è molto simile a quello in cui vengono imposti gli spostamenti; quindi, è legato in qualche modo al lavoro svolto da [8].

Il secondo approccio, adottato da [9], è stato preso in considerazione e confrontato con gli altri in modo da varare tutte le possibilità di calcolo della SEA e avere alle spalle della definizione dello standard una campagna sperimentale solida. In questo sotto capitolo ci si limita a descrivere la metodologia di calcolo secondo i due approcci, il confronto tra i vari algoritmi verrà effettuato e descritto successivamente.

Si parte con l'analisi della prima metodologia alternativa proposta, fissando come punto di inizio lo schema mostrato in **Figura 33**:

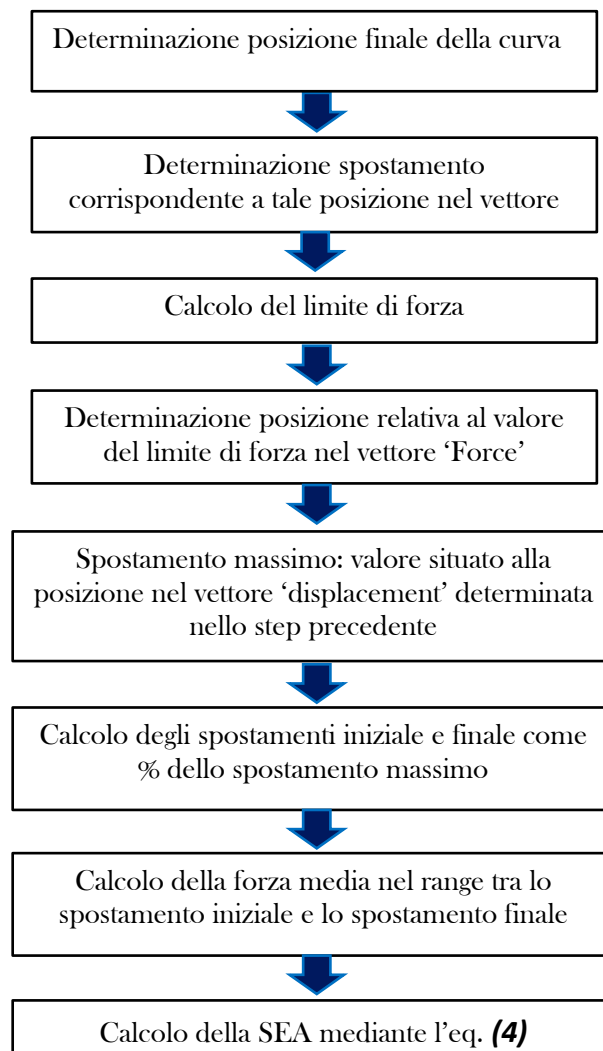


Figura 33: Diagramma di flusso relativo al calcolo della SEA utilizzando gli spostamenti imposti in percentuale

Viene riportato solamente il processo relativo al calcolo della forza media in quanto le altre parti (import delle curve, lettura dei dati, filtraggio delle curve) dell'algoritmo sono le stesse viste precedentemente.

Il primo step è dedicato all'individuazione della fine della curva, più precisamente come termine della curva viene preso lo spostamento oltre il quale la forza assume valori negativi. Una volta determinato questo punto, l'algoritmo prende il valore di spostamento associato ad esso, e lo utilizza per calcolare il limite di forza, con la medesima modalità della versione finale dell'algoritmo con spostamenti variabili, la quale viene spiegata nel dettaglio nel paragrafo successivo per rispettare l'ordine cronologico di sviluppo delle varie versioni (per il calcolo sono stati utilizzati i parametri derivanti da processo di taratura). A questo punto individua la posizione alla quale è collocato il limite di forza all'interno del vettore 'Force' e considera lo spostamento associato a tale posizione nel vettore 'displacement' come spostamento massimo. Nello step successivo calcola gli spostamenti iniziale e finale come % dello spostamento massimo. I valori ottimali delle % sono stato ottenuti mediante il processo di taratura, il quale verrà illustrato in seguito. Determinati i valori spostamento sul tratto utile all'algoritmo non rimane che calcolare la forza media nel range di tali valori e conseguentemente la SEA, eq. (4).

I valori delle percentuali da considerare derivano anch'essi dal processo di taratura come già detto. Si consideri un esempio di calcolo sulla curva in **Figura 34**:

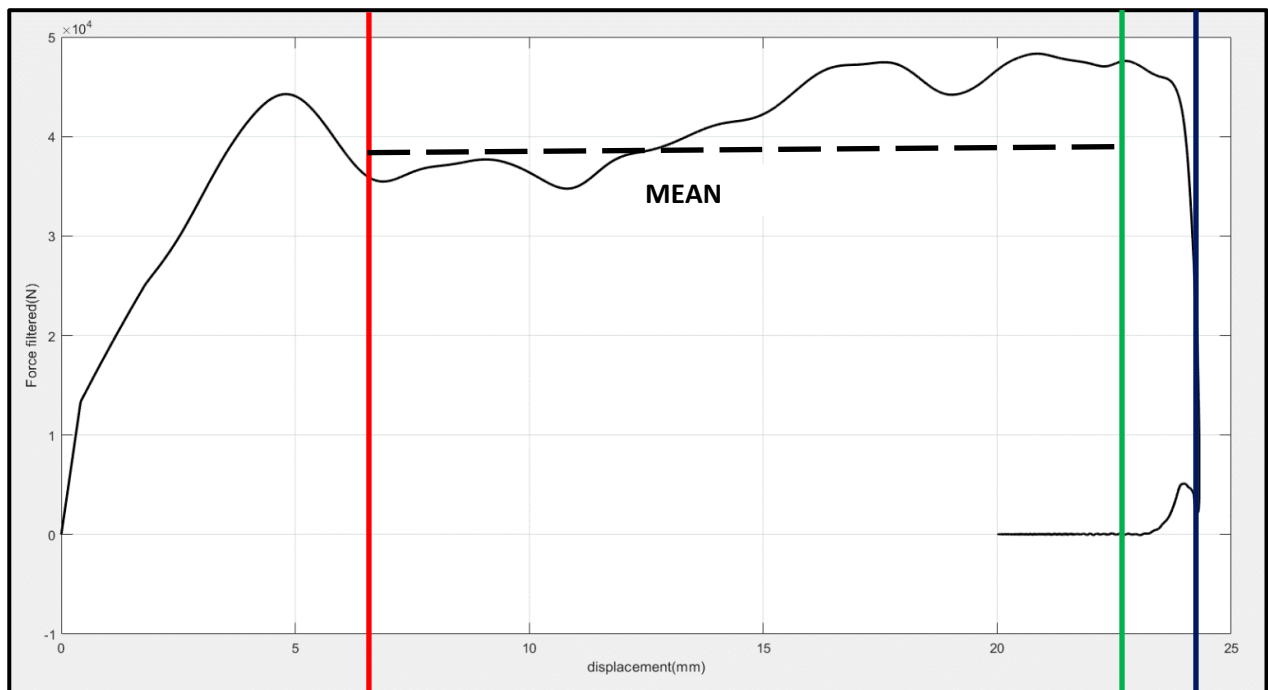


Figura 34: Curva di crash ottenuta su di un provino C50 (matrice in resina epossidica rinforzata con fibre di carbonio, spessore 5 mm)

Legenda:

- **Linea rossa:** spostamento iniziale (6.06 mm)
- **Linea verde:** spostamento finale (23.06 mm)
- **Linea blu:** spostamento massimo (24.27 mm)
- % utilizzata per il calcolo dello spostamento iniziale: 25 %
- % utilizzata per il calcolo dello spostamento finale: 95 %

Determinati gli spostamenti iniziale e finale la SEA viene calcolata utilizzando l'eq. (4).

Si passa ora a descrivere il secondo approccio proposto, secondo il quale la SEA viene calcolata considerando l'area sottesa a tutta la curva di crash. Questa metodologia di calcolo concettualmente risulta anche più semplice della precedente; infatti, non è necessaria alcuna iterazione per il calcolo degli spostamenti.

Prendendo come esempio la curva in **Figura 34**, la SEA viene calcolata mediante il diagramma di flusso in **Figura 35**:

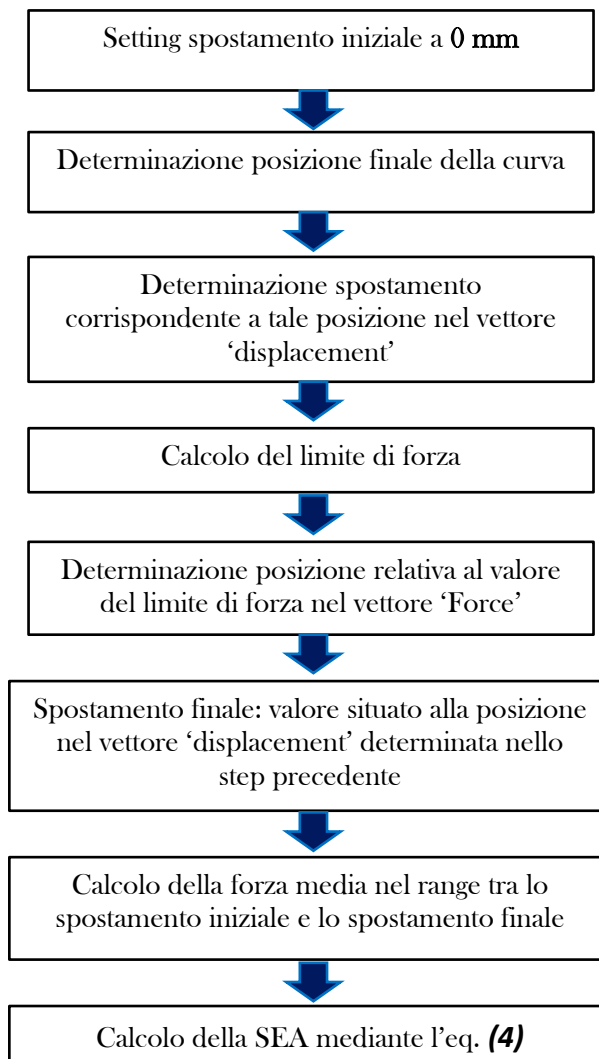


Figura 35: diagramma di flusso per il calcolo della SEA considerando l'area sottesa a tutta la curva

Come si può osservare lo spostamento iniziale è settato a zero, in quanto questo approccio prevede di considerare l'area sottesa a tutta la curva per il calcolo della SEA; quindi, non viene tagliato il tratto di influenza del trigger.

Successivamente vengono determinati la posizione finale della curva ed il limite di forza, in maniera analoga a quanto visto per l'algoritmo con spostamenti imposti in percentuale (i parametri utilizzati sono quelli derivanti dal processo di taratura).

A questo punto l'algoritmo individua la posizione alla quale è collocato il limite di forza nel vettore 'Force' ed assume come spostamento finale il valore di spostamento corrispondente a tale posizione nel vettore 'displacement'. Negli step finali il codice calcola la forza media nel range tra lo spostamento iniziale e quello finale, ed infine la SEA mediante l'eq. (4).

Con questo approccio si concludono le metodologie di calcolo utilizzate per calcolare la SEA, le quali verranno messe a confronto nei capitoli dedicati.

3.4. Algoritmo con spostamenti variabili: versione definitiva

In questo sotto capitolo viene spiegato in maniera dettagliata il funzionamento della versione definitiva dell'algoritmo per il calcolo della SEA, la quale verrà implementata all'interno del software Bluehill Impact installato sulle macchine di test Instron. Prima di addentrarsi nel cuore del funzionamento di questa ultima versione dell'algoritmo è utile fare un riassunto dei passaggi concettuali che hanno condotto al suo sviluppo ed effettuare una panoramica generale sul suo funzionamento. Il punto di partenza è stato quello di considerare gli spostamenti imposti dall'operatore che analizza la curva di crash. Si è subito compreso che questo approccio non soddisfa i requisiti di uno standard; quindi, si è pensato di apportare delle modifiche importanti. Prima di tutto si è provato a tenere lo spostamento iniziale e calcolare quello finale in maniera iterativa senza l'utilizzo di alcun margine, ma anche questo approccio è risultato inefficace. Per migliorare ulteriormente l'algoritmo si è optato per calcolare anche lo spostamento iniziale in maniera iterativa e adottare dei valori di margine che consentissero di convergere il più possibile verso il tratto utile della curva. Gli ultimi due step concettuali che hanno condotto alla versione definitiva del codice sono stati quelli di ragionare sulla forza puntuale e non più sulla forza media, e calcolare il limite di forza per ogni curva.

Per quanto riguarda la parte di import delle curve, input dei dati e filtraggio delle curve la versione definitiva dell'algoritmo ragiona esattamente come le versioni precedenti, la differenza principale sta nella metodologia di calcolo della forza media. In particolare, si può dire che le iterazioni che permettono il calcolo degli spostamenti iniziale e finale sono quelle mostrate in **Figura 31**. Quindi, ciò che distingue sostanzialmente le ultime due versioni del codice è il calcolo del limite di forza, che nella precedente versione veniva definito in maniera assoluta, mentre in questo caso viene calcolato sulla singola curva. Questo accorgimento apporta un notevole miglioramento nel calcolo degli spostamenti e di conseguenza della forza media, in quanto utilizzando come limite di forza un valore tarato sulla curva in analisi si riesce a convergere in maniera molto più precisa verso il tratto utile. All'interno del codice sono poi implementati altri controlli volti a tagliare tratti indesiderati di curva, ma questi ultimi verranno spiegati in dettaglio successivamente.

A questo punto si consideri il diagramma di flusso in **Figura 36**, che descrive con accezione algoritmica tutti gli step coinvolti nel calcolo della forza media, garantendo così una visione generale del suo funzionamento prima che il lettore si imbatte nella descrizione minuziosa del codice.

Per comodità non vengono riportati gli step relativi all'import delle curve, lettura dei dati e filtraggio della curva in quanto sono analoghi a quanto visto nei precedenti paragrafi.

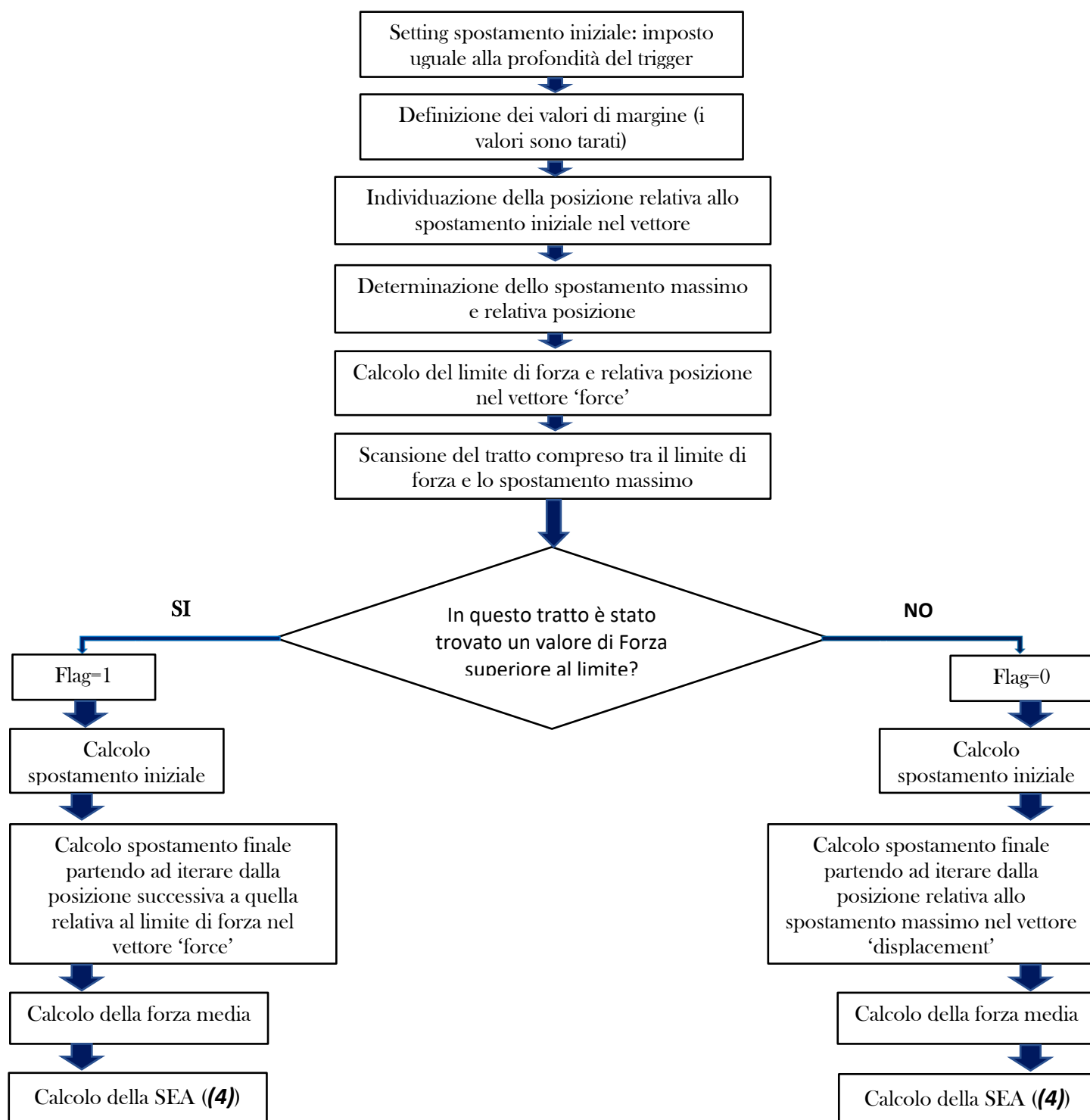


Figura 36: Diagramma di flusso relativo alla metodologia di calcolo della forza media nella versione definitiva dell'algorithm con spostamenti variabili

A questo punto si può procedere con la spiegazione del funzionamento dell'algoritmo. Partendo dalla testa codice, il primo step è dedicato all'import di tutte le curve presenti nella cartella aperta nella directory di Matlab. Affinché l'import vada buon fine è fondamentale che la cartella contenente le curve da analizzare e lo stesso algoritmo siano state aggiunte alla directory precedentemente.

La sezione "DATI IN INPUT" è invece dedicata all'input dei dati necessari per il calcolo della SEA, quali:

- Densità del provino in kg/m^3
- Sezione del provino in m^2
- Altezza del trigger in mm

Una volta inseriti e memorizzati i dati è implementato un ciclo iterativo che comprende praticamente tutto il codice ragionando come segue:

- Prende in input la prima curva importata precedentemente dalla directory di Matlab
- Legge dal file con estensione '.csv' i vettori tempo, forza e spostamento, denominati all'interno del codice come time, force e 'displacement'
- Filtra la curva
- Esegue il calcolo della forza media
- Calcola la SEA
- Memorizza in una tabella tutti i risultati utili

Le operazioni sopra citate vengono svolte ciclicamente analizzando le curve presenti nella cartella designata in maniera sequenziale, sino a che non vengono scansionate tutte.

Ora verranno illustrate nel dettaglio tutte le operazioni svolte durante le iterazioni.

L'input vero e proprio fa sì che venga memorizzata una matrice in Matlab, le cui colonne sono dei vettori e rappresentano le misure eseguite dalla cella di carico del percussore. All'interno del codice questa matrice è stata denominata 'Input', e si presenta come mostrato in **Figura 37**:

	1	2	3	4	5	6	7
1	1	-3.4730	-5.4460	NaN	NaN	NaN	NaN
2	2	-3.4720	-5.4460	NaN	NaN	NaN	NaN
3	3	-3.4710	-62.7750	NaN	NaN	NaN	NaN
4	4	-3.4700	8.8860	NaN	NaN	NaN	NaN
5	5	-3.4690	-34.1100	NaN	NaN	NaN	NaN
6	6	-3.4680	23.2180	NaN	NaN	NaN	NaN
7	7	-3.4670	-34.1100	NaN	NaN	NaN	NaN
8	8	-3.4660	-62.7750	NaN	NaN	NaN	NaN
9	9	-3.4650	23.2180	NaN	NaN	NaN	NaN
10	10	-3.4640	-19.7780	NaN	NaN	NaN	NaN
11	11	-3.4630	23.2180	NaN	NaN	NaN	NaN
12	12	-3.4620	-34.1100	NaN	NaN	NaN	NaN
13	13	-3.4610	8.8860	NaN	NaN	NaN	NaN
14	14	-3.4600	-19.7780	NaN	NaN	NaN	NaN
15	15	-3.4590	37.5500	NaN	NaN	NaN	NaN
16	16	-3.4580	23.2180	NaN	NaN	NaN	NaN
17	17	-3.4570	-5.4460	NaN	NaN	NaN	NaN
18	18	-3.4560	37.5500	NaN	NaN	NaN	NaN
19	19	-3.4550	23.2180	NaN	NaN	NaN	NaN
20	20	-3.4540	51.8820	NaN	NaN	NaN	NaN
21	21	-3.4530	51.8820	NaN	NaN	NaN	NaN
22	22	-3.4520	23.2180	NaN	NaN	NaN	NaN
23	23	-3.4510	66.2140	NaN	NaN	NaN	NaN
24	24	-3.4500	51.8820	NaN	NaN	NaN	NaN

Figura 37: Matrice contenente sotto forma di vettori tutte le grandezze misurate dalla cella di carico prima dell'istante dell'impatto

Legenda:

- Colonna 1: Sample
- Colonna 2: Time
- Colonna 3: Force
- Colonna 4: Velocity
- Colonna 5: Energy
- Colonna 6: Displacement
- Colonna 7: Impulse

Il primo aspetto che risalta osservando la didascalia precedente è senza dubbio il fatto che compaia nelle ultime colonne (4-7) la scritta 'NaN'. Questo a primo impatto potrebbe sembrare un errore di import dei dati ma non è assolutamente corretto in quanto, in realtà, è intrinseco alle misure della cella di carico. Di fatto, tutti gli istanti precedenti all'impatto del percussore con il provino sono riportati come negativi e in questo lasso di tempo non viene misurata alcuna grandezza. La misura ha inizio solamente nell'istante dell'impatto, considerato istante zero, come si può osservare in **Figura 38**:

3474	3474	0	782.8190	5.1900	0	0	0
3475	3475	1.0000e-03	969.1360	5.1900	0.0050	0.0050	1.0000e-03
3476	3476	0.0020	1.1698e+03	5.1900	0.0100	0.0100	0.0020
3477	3477	0.0030	1.2988e+03	5.1900	0.0170	0.0160	0.0030
3478	3478	0.0040	1.5568e+03	5.1900	0.0240	0.0210	0.0050
3479	3479	0.0050	1.7001e+03	5.1900	0.0320	0.0260	0.0060
3480	3480	0.0060	1.9437e+03	5.1900	0.0420	0.0310	0.0080
3481	3481	0.0070	2.1300e+03	5.1900	0.0520	0.0360	0.0100
3482	3482	0.0080	2.2877e+03	5.1900	0.0640	0.0420	0.0120
3483	3483	0.0090	2.5027e+03	5.1900	0.0760	0.0470	0.0150
3484	3484	0.0100	2.7177e+03	5.1900	0.0900	0.0520	0.0170
3485	3485	0.0110	2.8610e+03	5.1900	0.1040	0.0570	0.0200
3486	3486	0.0120	3.0330e+03	5.1900	0.1200	0.0620	0.0230
3487	3487	0.0130	3.2479e+03	5.1900	0.1360	0.0670	0.0260
3488	3488	0.0140	3.3626e+03	5.1900	0.1530	0.0730	0.0290
3489	3489	0.0150	3.5346e+03	5.1900	0.1710	0.0780	0.0330
3490	3490	0.0160	3.6492e+03	5.1900	0.1900	0.0830	0.0370
3491	3491	0.0170	3.7782e+03	5.1900	0.2090	0.0880	0.0400
3492	3492	0.0180	3.9359e+03	5.1890	0.2290	0.0930	0.0440
3493	3493	0.0190	4.1079e+03	5.1890	0.2500	0.0990	0.0480
3494	3494	0.0200	4.2082e+03	5.1890	0.2710	0.1040	0.0520
3495	3495	0.0210	4.4518e+03	5.1890	0.2940	0.1090	0.0570

Figura 38: Grandezze misurate dalla cella di carico nel momento esatto dell'impatto

Per lo scopo dell'analisi sulla SEA tutto ciò che precede l'impatto del percussore con il provino non è di interesse, per tale motivo l'algoritmo è in grado di importare solamente le colonne di interesse della matrice, per questo lavoro di tesi 2 3 e 6, e di leggere i dati a partire dall'istante temporale zero. Questo particolare valore temporale viene identificato dal software Bluehill Impact come l'istante tale per cui viene registrato un incremento del valore di forza, indice dell'inizio della decelerazione del percussore.

Successivamente all'import dei vettori tempo, forza e spostamento viene memorizzato nella tabella creata all'origine del codice il valore massimo del vettore 'Force', il quale può tornare utile per un confronto tra le varie curve. La sezione relativa al filtraggio delle curve è stata ampiamente spiegata e descritta all'interno del capitolo 2, quindi non verrà ripercorsa in questo punto per non rendere pesante e ripetitiva la trattazione. La parte di codice dedicata al plot delle curve è in grado di mostrare i seguenti risultati:

- Forza-Tempo non filtrata
- Forza-Tempo filtrata
- Entrambe sovrapposte sullo stesso grafico
- Forza-Spostamento non filtrata
- Forza-Spostamento filtrata
- Entrambe sovrapposte sullo stesso grafico

In **Figura 39** e **Figura 40** viene mostrato un esempio di output:

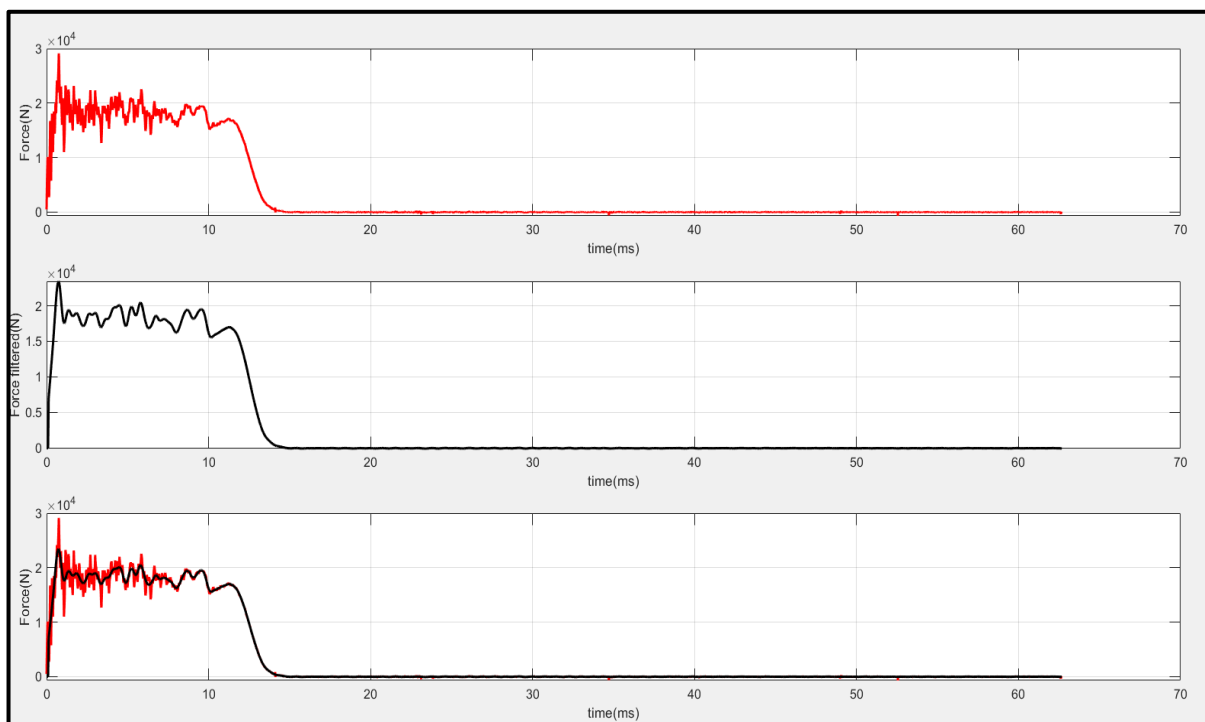


Figura 39: Curva Forza-Tempo non filtrata || Forza-Tempo filtrata || Sovrapposizione delle due

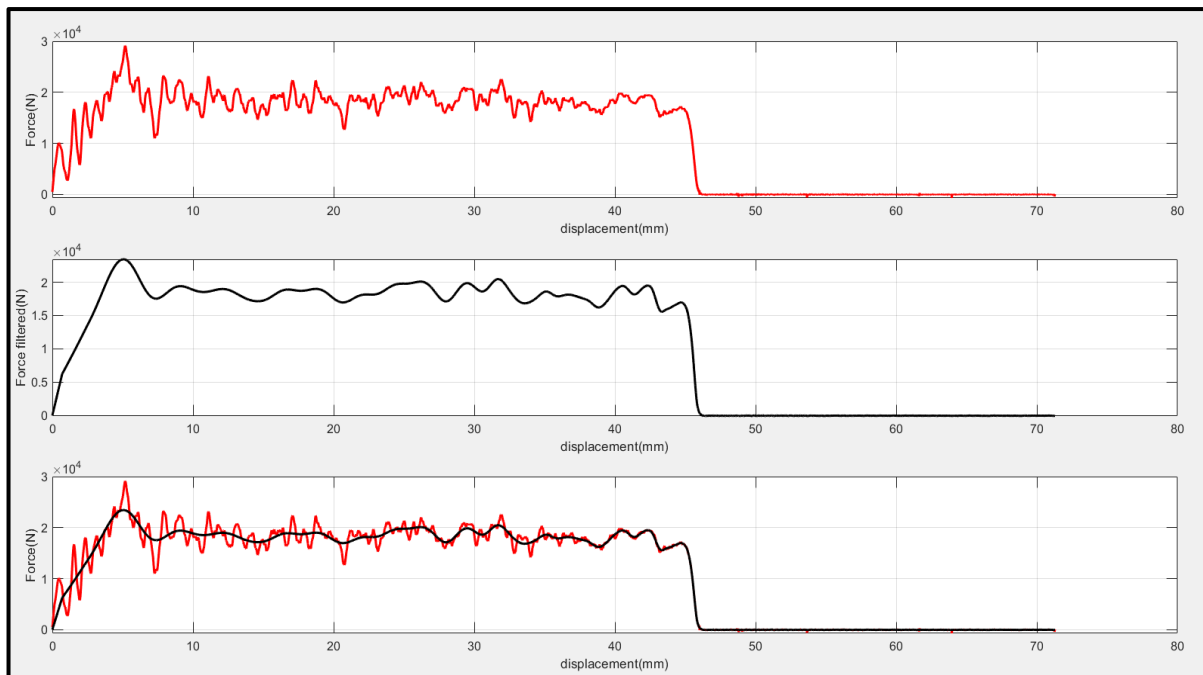



Figura 40: Curva Forza-spostamento non filtrata || Forza-Spostamento filtrata || Sovrapposizione delle due

Terminata la parte relativa al filtraggio subentra il cuore dell'algoritmo, la sezione di codice dedicata al calcolo degli spostamenti e della forza media. Il calcolo ha inizio con il primo step in **Figura 36**, in cui la variabile 'start_displacement' viene assunta pari all'altezza del trigger. Questa assunzione è importante per la logica dell'algoritmo in quanto sancisce il punto di inizio dell'iterazione sul calcolo dello spostamento iniziale. Nello step successivo vengono definiti i margini, la cui funzione è stata ampiamente discussa precedentemente. Il loro valore è stato scelto in seguito ad un meticoloso processo di taratura dei parametri del codice, che verrà esplicitato successivamente, per ora si assumano come parametri tarati e funzionali al calcolo della SEA.

Una volta imposti i margini l'algoritmo procede determinando la posizione corrispondente a 'start_displacement' nel vettore degli spostamenti. Prendendo come esempio la curva in **Figura 40** 'start_displacement' vale **5 mm** vista la geometria del trigger nei provini piatti. L'algoritmo ha calcolato che tale valore di spostamento è situato alla posizione **728**, definita come 'start_position'.

Per verificare la correttezza del calcolo si può osservare il vettore ‘displacement’ alla posizione designata, come mostrato in **Figura 41**:

724	4.9720
725	4.9790
726	4.9860
727	4.9920
728	4.9990
729	5.0060
730	5.0120
731	5.0190
732	5.0260



START POSITION


Figura 41: Calcolo della posizione corrispondente alla variabile "start_displacement"

Come spiegato precedentemente, la funzione dedicata alla determinazione della posizione di un certo valore all'interno di un qualsiasi vettore trova un valore minore o uguale a quello ricercato, per tale motivo in questo caso è stato individuato lo spostamento **4.99 mm** e non **5 mm**.

Ora che è stata determinata la posizione da cui partirà l'iterazione risulta necessario definire l'estremo superiore entro il quale l'algoritmo lavora.

Questo viene svolto nello step successivo, in cui la posizione finale, denominata nel codice 'final_position', corrisponde al valore di spostamento oltre la quale i valori di forza divengono negativi. Considerando sempre la curva in **Figura 40**, la 'final_position' trovata è la **14821**, come indicato in **Figura 42**:

14819	0.4649
14820	0.2281
14821	-0.2572
14822	-0.4701
14823	-0.8113



FINAL POSITION

Figura 42: Calcolo "final_position"

Riassumendo quanto fatto sino ad ora, l'algoritmo ha semplicemente determinato il range di posizioni entro il quale si svolgeranno le iterazioni che portano al calcolo dello spostamento iniziale e finale impiegati per il calcolo della SEA.

A questo punto si può procedere con la metodologia di calcolo del limite di forza da utilizzare nel confronto tra lo stesso e i valori di forza del vettore 'Force'.

Basandosi sul 'max_displacement' calcolato, l'algoritmo prende in considerazione un valore leggermente inferiore e ne determina la metà. Sostanzialmente individua un valore di spostamento collocato sul tratto utile. Successivamente prende il valore di forza associato ad esso e calcola il limite di forza come il 50% di tale valore. La % è stata definita nel processo di taratura dei parametri, del quale si parlerà successivamente.

Il codice considera un valore leggermente inferiore al 'max_displacement' per assicurare che lo spostamento rispetto al quale si calcola il limite di forza sia situato sul tratto utile, soprattutto nel caso di curve la cui la fine della curva si individui per spostamenti maggiori rispetto ad altre. La metodologia di calcolo del limite di forza proposta permette di distaccarsi da un valore assoluto e di poterlo determinare in funzione della singola curva. Questo è un grosso vantaggio vista l'importanza di questo parametro per la convergenza dell'algoritmo, e di conseguenza per il calcolo della SEA.

Il passo successivo è individuare la posizione del limite di forza nel vettore denominato 'force_filtered' per poter effettuare un ulteriore controllo necessario al funzionamento ottimale dello script. È importante verificare se nell'intervallo di posizioni che va dal limite di forza sino alla fine della curva siano presenti valori di forza superiori al limite, perché se ciò accade si può ben comprendere come l'algoritmo arresterebbe l'iterazione prima del punto designato. Si consideri come esempio la curva in **Figura 43**:

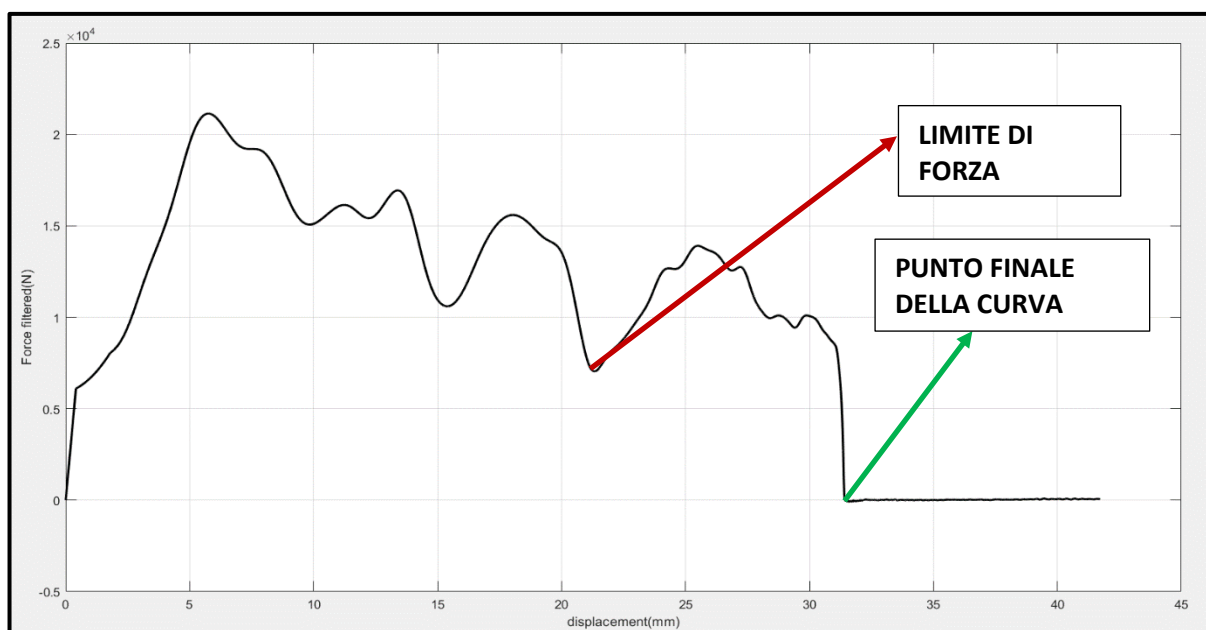


Figura 43: Curva ottenuta su di un provino G20(matrice in resina epossidica rinforzata con fibre di vetro, spessore 2 mm)

Per questa curva il limite di forza calcolato dall'algoritmo risulta essere **7.084 kN**, come indicato dalla freccia rossa in **Figura 43**, mentre la fine della curva è stata individuata allo spostamento **31.45 mm** indicato dalla freccia verde nella medesima didascalia.

Come è ben visibile, nell'intervallo tra il limite di forza e il termine della curva vi sono valori di forza superiori al limite, di conseguenza l'iterazione (che parte dal punto finale della curva e procede a ritroso) si arresta non appena viene individuato un valore superiore a **7.084 kN**, e questo si verifica prima del punto designato (freccia rossa). Prima di procedere con la soluzione a questa problematica si consideri un'osservazione: la ricerca della posizione corrispondente al limite di forza nel vettore 'force_filtered' fallisce solo nel caso in cui nel mezzo del tratto utile vi siano oscillazioni della forza filtrata superiori al **50%** rispetto alla media. Ciò vorrebbe dire che le oscillazioni sulla forza non filtrata potrebbero essere anche del **100%** rispetto al valore medio, ma in questo caso il test probabilmente risulterebbe non valido; quindi, di fatti questa problematica non sussiste.

Tornando al discorso principale, tratti di curva situati tra la freccia rossa e la freccia verde in **Figura 43**, che presentano valori di forza superiori al limite, devono essere tagliati in quanto portano l'algoritmo ad un calcolo errato dello spostamento finale, in particolare si sovrastima tale valore e si sottostima la SEA. A tale scopo si è deciso di adottare una soluzione che prevede due opzioni di calcolo degli spostamenti in funzione dell'andamento della curva nel tratto appena citato. Prima di illustrare le due diverse soluzioni risulta necessario per l'algoritmo comprendere se oltre il limite di forza esistono effettivamente valori superiori ad esso; quindi, è implementato un ciclo di controllo, il quale utilizza un flag come indicatore. Il valore del flag viene inizializzato a zero, successivamente viene scansionato il tratto di curva nell'intervallo di posizioni tra il limite di forza e il 'max_displacement' confrontando ogni valore del vettore 'Force' con lo stesso limite: se non viene trovato alcun valore di forza superiore ad esso allora il flag rimane zero, altrimenti il flag assume valore uno.

A questo punto il codice può procedere con il calcolo in funzione del valore della variabile flag:

- **Flag=0**: l'algoritmo esegue il calcolo senza alcuna variazione partendo ad iterare da 'final_position' sino a che non incrocia il limite di forza
- **Flag=1**: indica che è presente nel tratto incriminato un valore di forza superiore al limite, di conseguenza l'algoritmo parte ad iterare dalla posizione successiva al limite di forza in modo da tagliare il tratto di curva anomalo

Per capire a fondo il funzionamento dell'algoritmo è necessario esplorare a fondo e non in maniera superficiale le due metodologie di calcolo sopra citate, per questo motivo verranno ora illustrati due esempi che rispecchiano le due casistiche presentate.

Si consideri la curva in **Figura 44**:

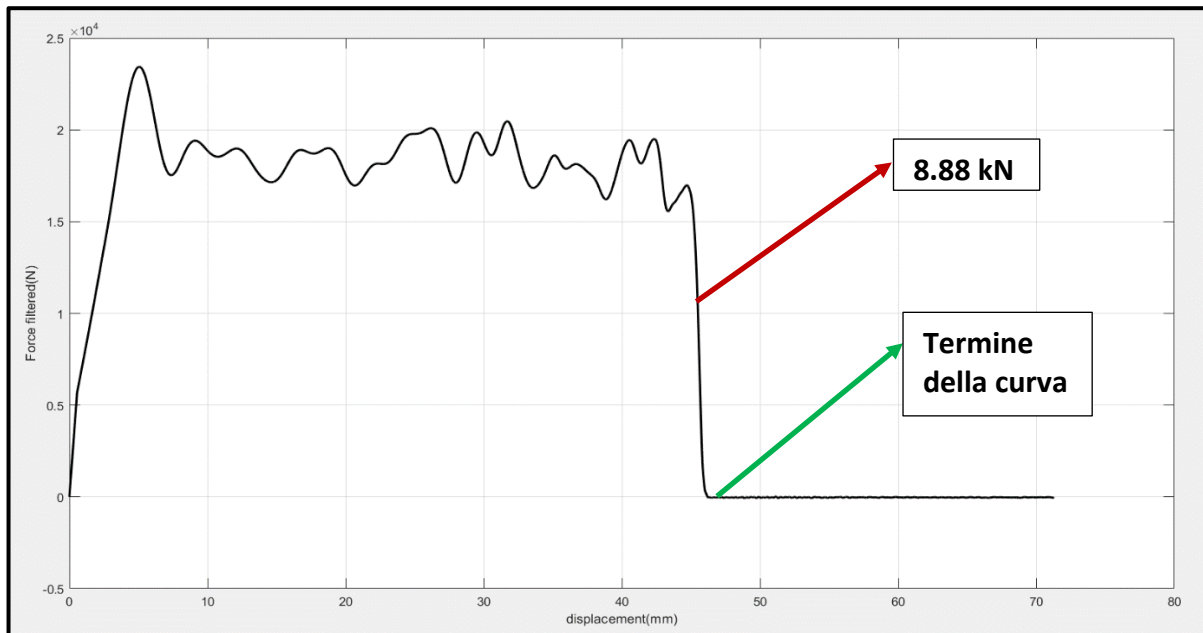


Figura 44: Curva di crash ottenuta su di un provino CFRP (matrice in resina epossidica rinforzata con fibre di carbonio)

Il limite di forza calcolato dall'algoritmo per la curva appena mostrata risulta essere **8.88 kN**, come indicato dalla freccia rossa in **Figura 44**, mentre la il limite superiore della curva è indicato dalla freccia di colore verde nella medesima didascalia. Si può osservare chiaramente che nell'intervallo di spostamenti compreso tra il limite di forza e l'estremo superiore della curva non vi sono valori di forza eccedenti il limite previo determinato; quindi, si ricade nella prima casistica di calcolo (**flag=0**). In questo caso il codice prima di tutto esegue un'iterazione sullo spostamento iniziale. L'iterazione parte da 'start_position' ed avanza scansionando il vettore della forza filtrata sino a che non viene incrociato un valore di forza superiore a quello limite. A questo punto l'algoritmo si arresta, prende il valore di spostamento corrispondente a tale valore di forza e ne aggiunge ancora il margine. Il valore di spostamento calcolato nel codice viene chiamato 'start_displacement' e viene memorizzato nella tabella creata all'inizio dello script.

Ora che è stato determinato lo spostamento iniziale l'algoritmo individua la posizione corrispondente a tale valore ed essa sarà il punto di partenza per calcolare la forza media sul tratto utile.

Un ragionamento del tutto speculare viene eseguito per quanto riguarda il calcolo dello spostamento finale, chiamato nel codice 'final_displacement', con la differenza che in questo caso l'iterazione avviene a ritroso.

Bisogna sottolineare però che sullo spostamento finale viene eseguito un ulteriore controllo necessario visti i limiti fisici del supporto crashworthiness: una volta calcolato il 'final_displacement' l'algoritmo verifica che esso sia inferiore a **45 mm**, in modo da capire se sono stati coinvolti i deceleratori nell'impatto. Per capire meglio questo concetto si consideri il supporto crashworthiness in **Figura 2** con il relativo provino montato. I provini piatti sono lunghi nominalmente **150 mm**, e il tratto di provino che può fratturarsi prima di toccare i deceleratori è proprio di **45 mm** per come è realizzato il sistema; quindi, se il percussore non dovesse arrestarsi entro la corsa utile andrebbe a urtare i deceleratori. Tale fenomeno nella curva si visualizza con un picco di forza oltre i **45 mm** di spostamento misurato, e tutta la parte di curva oltre questo limite deve essere tagliato in quanto non indicativo della frattura progressiva del provino.

Ritornando quindi al controllo eseguito dall'algoritmo, se lo spostamento finale calcolato è superiore a **45 mm** significa che sono stati coinvolti i deceleratori, di conseguenza all'interno del codice è implementata una soluzione che fa in modo di considerare come 'final_displacement' proprio tale valore. Mediante tale accorgimento si è sicuri di non considerare un tratto di curva che andrebbe a sovrastimare il valore di SEA visti i tipici picchi di forza che nascono a causa dei deceleratori, i quali per contro sono necessari per evitare danni al supporto quando si verifica una tale condizione.

Eseguito il calcolo dello spostamento iniziale, finale e delle relative posizioni l'algoritmo procede la determinazione della forza media sul tratto individuato. Al fine di effettuare questo calcolo è stata implementata un'iterazione che scansiona il vettore della forza nell'intervallo di posizioni precedentemente determinato, e memorizza tutti i valori in questo range in un ulteriore vettore, denominato "force_final". A questo punto non resta che utilizzare la funzione di Matlab per il calcolo della media.

Contemporaneamente al calcolo della media vengono anche determinate altri parametri utili per l'analisi della curva, i quali sono memorizzati nella tabella generata ad inizio codice('tabella_SEA').

Sino a questo momento è stata analizzata la prima metodologia di calcolo degli spostamenti e della forza media, esploriamo ora la seconda casistica partendo dalla curva in **Figura 43**.

Come già detto in precedenza, per questa curva l'algoritmo calcola un limite di forza di **7.084 kN** indicato dalla freccia rossa, e il termine della curva indicato dalla curva di colore verde. Rispetto alla curva in **Figura 44** quest'ultima presenta valori di forza superiori al limite nell'intervallo di spostamenti compreso tra la freccia rossa e la freccia verde. Ci si trova quindi di fronte alla seconda casistica, in cui la variabile flag ha valore unitario. La logica iterativa che porta al calcolo dello spostamento iniziale e finale rimane invariata, l'unica differenza risiede nel punto di inizio dell'iterazione per il calcolo dello spostamento finale. Se si suppone far partire l'iterazione dal termine della curva si può immaginare che andando a ritroso si incrocia un valore superiore al limite prima del punto designato (freccia rossa), di conseguenza il 'final_displacement' calcolato risulta errato. Al fine di tagliare la coda della curva che presenta l'andamento anomalo, l'algoritmo parte ad iterare dalla posizione successiva a quella corrispondente al limite di forza nel vettore 'Force', in questo modo il calcolo converge verso il tratto piatto anche per curve con andamento simile a quella in **Figura 43**. Per chiarire meglio questo approccio si consideri la **Figura 45** :

5479	7.0939e+03
5480	7.0909e+03
5481	7.0880e+03
5482	7.0852e+03
5483	7.0825e+03
5484	7.0798e+03
5485	7.0772e+03

Figura 45: Posizione relativa al limite di forza calcolato pe la curva in Figura 43







La posizione individuata dall'algoritmo è la **5482** nel vettore "force_filtered". Non essendo presente nel vettore esattamente il valore del limite di forza calcolato dal codice l'algoritmo è implementato in modo da individuare il valore subito successivo in ordine crescente. Se invece fosse stato presente esattamente lo stesso valore calcolato dal codice sarebbe stata determinata la posizione relativa a tale valore.

Una volta trovata la posizione corrispondente al limite di forza l'algoritmo procede ad iterare partendo dalla successiva, in questo caso la **5483**, eseguendo i successivi calcoli come già implementato per il caso precedente.

A questo punto il calcolo della forza media e del range di spostamenti è avvenuto con successo qualsiasi sia la curva importata dal codice Matlab. L'ultima parte dell'algoritmo è dedicata al calcolo vero e proprio della SEA, il quale avviene esattamente con la stessa modalità descritta dall'eq. (4).

Si conclude questo sotto capitolo con una nota: tutti i ragionamenti svolti dall'algoritmo sui valori di forza sono effettuati sul vettore della forza filtrata in modo da evitare problematiche relative alle ampie oscillazioni del segnale come accennato nel capitolo 2, mentre la forza media viene calcolata su valori di forza non filtrata in modo da essere più fedele possibile alla realtà del fenomeno fisico in atto. Questo è possibile in quanto la forza media calcolata considerando la forza filtrata e non filtrata risulta praticamente identica come si può osservare in

Figura 46:

	flag	0	a)
	force	62667x1 double	
	force_average	1.8198e+04	
	flag	0	b)
	force	62667x1 double	
	force_average	1.8207e+04	

*Figura 46: Calcolo della forza media eseguito considerando valori di forza: **a)** non filtrata || **b)** filtrata*

3.5. Taratura dei parametri

Nei precedenti sotto capitoli è capitato sovente di prendere per ottimale un certo parametro senza porsi domande riguardo il processo di taratura che ha dato frutto ad un determinato valore implementato nell'algoritmo. Il processo di taratura dei vari parametri in gioco non è stato descritto in precedenza in modo da non appesantire la trattazione e focalizzarsi sul puro funzionamento logico dei vari codici analizzati, ma verrà spiegato in maniera dettagliata in questo sotto capitolo in modo che il lettore abbia il quadro completo riguardo l'approccio algoritmico al calcolo della SEA.

Entrando più nel dettaglio, gli approcci al calcolo della SEA che necessitano di una taratura dei parametri sono:

- **Algoritmo che calcola la SEA considerando gli spostamenti variabili**
- **Algoritmo che calcola la SEA considerando gli spostamenti imposti in percentuale**
- **Algoritmo che calcola la SEA considerando l'area sottesa a tutta la curva**

I parametri che necessitano taratura nel primo algoritmo elencato sono i seguenti:

- **Margine sullo spostamento iniziale**
- **Margine sullo spostamento finale**
- **Percentuale per il calcolo del limite di forza**

Quelli che invece devono essere tarati nel secondo algoritmo sono:

- **Percentuale rispetto allo spostamento massimo per il calcolo dello spostamento iniziale**
- **Percentuale rispetto allo spostamento massimi per il calcolo dello spostamento finale**

Il parametro che invece necessita di una taratura in entrambi gli ultimi due algoritmi elencati è il seguente:

- **Termine della curva: valore di spostamento preso come riferimento per il calcolo degli spostamenti iniziale e finale**
- **Percentuale per il calcolo del limite di forza**

Il processo di calibrazione di tutti i parametri elencati verrà spiegato in riferimento ad un esempio specifico al fine di visualizzare più chiaramente il ragionamento adottato.

Per un qualsiasi procedimento di calibrazione è estremamente importante avere dei valori di riferimento rispetto ai quali valutare i parametri in gioco.

Osservando tutti i parametri elencati precedentemente si conviene facilmente che essi sono tutti funzionali al calcolo dello spostamento iniziale e finale, il cui ruolo è stato ampiamente illustrato nel paragrafo 3.2, relativo all'approccio algoritmico al calcolo della SEA.

Per questo motivo i valori di riferimento ottimali per questa tipologia di calibrazione sono risultati essere quelli di spostamento iniziale e finale individuati da un operatore esperto in maniera visiva per una qualsiasi curva di crash. Al fine di comprendere a fondo questa ultima affermazione consideriamo un esempio di taratura dei margini rispettivamente sullo spostamento iniziale e finale per la curva in **Figura 47**:

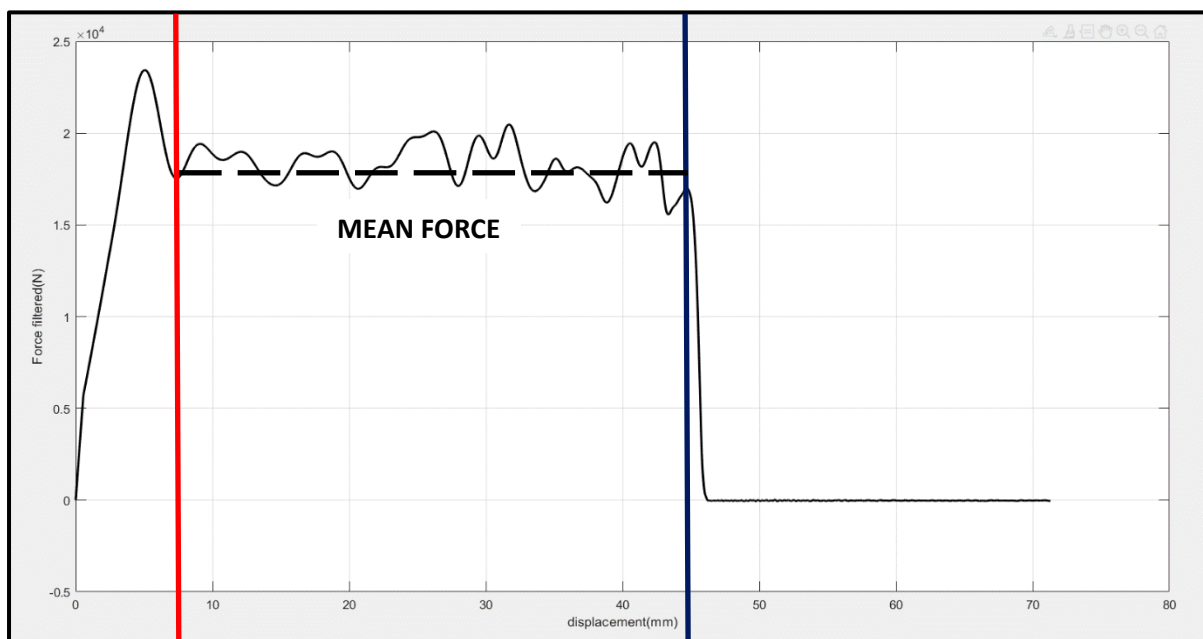


Figura 47: Curva di crash ottenuta su di un provino piatto realizzato in CFRP utilizzata come esempio per il processo di taratura dei parametri

Legenda:

- **Linea rossa:** Spostamento iniziale che un potenziale operatore inserirebbe manualmente analizzando visivamente la curva: preso come valore vero per la taratura (7.33 mm)
- **Linea blu:** Spostamento finale che un potenziale operatore inserirebbe manualmente analizzando visivamente la curva: preso come valore vero per la taratura (44.7 mm)

A questo punto si hanno tutti gli elementi di base per poter descrivere il processo di calibrazione in maniera dettagliata. Considerando l'esempio proposto, i valori di spostamento individuati come riferimento sono quelli indicati in **Figura 47**.

Una volta che si hanno a disposizione gli spostamenti di riferimento può essere attuata la calibrazione vera e propria.

Il processo di taratura è un processo prettamente empirico, infatti si parte da valori di prova dei margini e si valuta l'errore relativo degli spostamenti calcolati dall'algoritmo utilizzando tali margini, rispetto ai valori di riferimento (valori di esempio riportati in **Figura 47**). Questo procedimento viene ripetuto per un certo numero di curve, nel caso specifico di questo lavoro di tesi ne sono state analizzate **20**, ed una volta terminata l'analisi su di esse si valuta per quante risulta verificata la seguente condizione, che per comodità chiameremo condizione limite: errore relativo inferiore o al limite uguale al $\pm 5\%$ rispetto ai valori presi come riferimento. Se per almeno il **90 %** delle curve utilizzate come campione la condizione appena citata risulta verificata si può concludere che i margini scelti sono abbastanza affidabili per il calcolo degli spostamenti, in caso contrario si prendono le curve tali per cui non risulta verificata la condizione limite e si testano altri due valori dei margini in funzione degli spostamenti calcolati su tali curve utilizzando i margini di prova iniziali. Quanto detto sino ad ora può sembrare confusionario, ma si cerca ora di chiarire meglio il concetto. Si supponga di aver preso **20** curve come campione per l'analisi, che per **15** di queste risulti verificata la condizione limite mentre per le restanti **5** l'errore sia superiore al **5 %**. Si supponga inoltre che i margini utilizzati per l'analisi primaria siano rispettivamente per lo spostamento iniziale e finale **2 mm** e **2 mm**, e che per le **5** curve in cui non si è verificata la condizione limite gli spostamenti calcolati siano maggiori di più del **5 %** rispetto agli spostamenti di riferimento. Risulta chiaro quindi che affinché l'errore calcolato sia accettabile e rientri nel range, il margine sullo spostamento iniziale deve essere ridotto (viene sommato) mentre quello sullo spostamento finale deve essere incrementato (viene sottratto). Nel caso in cui si fosse verificato il caso opposto, ovvero che gli spostamenti calcolati fossero inferiori di più del **5 %** rispetto a quelli di riferimento, allora si sarebbe dovuto incrementare il margine sullo spostamento iniziale e ridurre quello sullo spostamento finale. Tale ragionamento è stato applicato in maniera iterativa sino ad ottenere un margine di errore nel range $\pm 5\%$ rispetto agli spostamenti di riferimento su almeno il **90 %** delle **20** curve considerate. Nel caso specifico preso come esempio ciò implica che delle **5** curve tali per cui non era rispettato il margine di errore, almeno per **3** di esse devono essere trovati dei margini che garantiscano il verificarsi della condizione limite precedentemente citata facendo attenzione che rimanga tale anche nel caso di tutte le altre **15** curve. Il ragionamento descritto sino ad ora è stato applicato in maniera identica su tutti i parametri da tarare per entrambi gli algoritmi in esame.

Prima di mostrare i risultati ottenuti, è importante aprire una piccola parentesi sul valore di errore relativo scelto come limite.

Si è deciso di adottare un valore del **5 %** come riferimento in quanto in numerose applicazioni nel campo ingegneristico risulta essere il valore al di sotto del quale viene garantita con buona probabilità la solidità dei calcoli effettuati per lo sviluppo di un determinato progetto.

Vengono qui di seguito riportati i parametri risultanti dalla calibrazione e un esempio di calcolo degli spostamenti mediante i parametri tarati per entrambi gli algoritmi:

Algoritmo che considera variabili gli spostamenti

- Margine sullo spostamento iniziale: 1.5 mm
- Margine sullo spostamento finale: 1.5 mm
- Percentuale per il calcolo del limite di forza: 50 %

Algoritmo che considera gli spostamenti imposti in percentuale

- Percentuale rispetto allo spostamento massimo per il calcolo dello spostamento iniziale: 25 %
- Percentuale rispetto allo spostamento massimo per il calcolo dello spostamento finale: 95 %

Algoritmo che considera gli spostamenti imposti in percentuale e algoritmo che considera l'area sottesa a tutta la curva

- Termine della curva: valore di spostamento tale per cui la forza risulta < 0 kN
- Percentuale per il calcolo del limite di forza: 30 %

Come accennato precedentemente vengono ora riportati due esempi di calcolo utilizzando i parametri ottimali per entrambi gli algoritmi considerati.

Si parte con un esempio relativo all'algoritmo che considera gli spostamenti variabili per la curva raffigurata in **Figura 48**, la quale viene riportata nuovamente per comodità di visualizzazione:

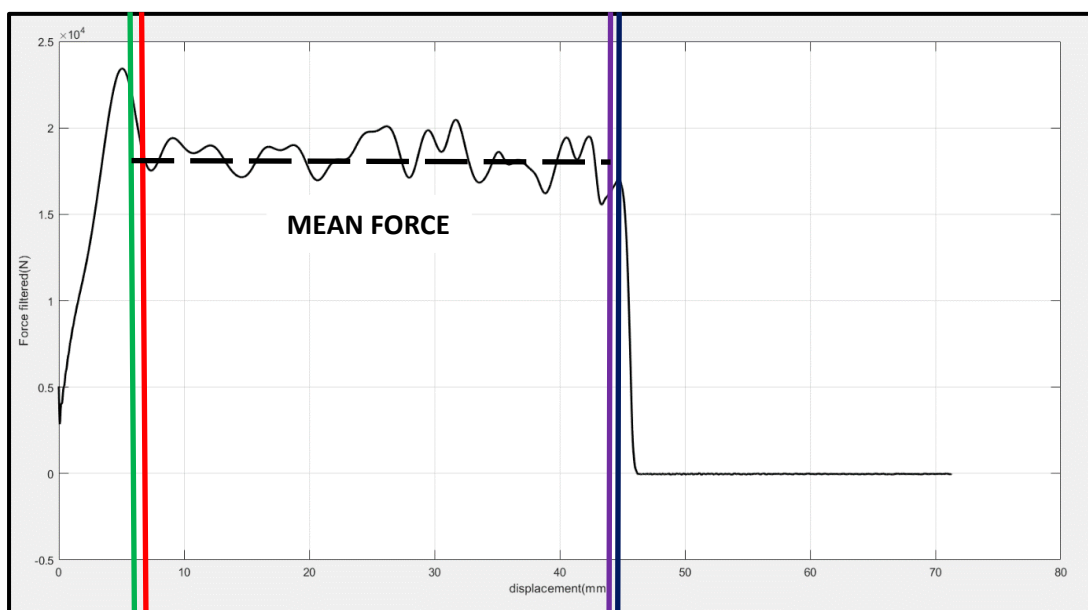


Figura 48: Curva di crash utilizzata come esempio nel processo di taratura dei parametri: algoritmo con spostamenti variabili

Per la curva raffigurata nella precedente didascalia gli spostamenti presi come valori di riferimento per la taratura sono indicati dalla seguente legenda:

- **Linea rossa:** spostamento iniziale preso come riferimento (6.8 mm)
- **Linea blu:** spostamento finale preso come riferimento (44.71 mm)

Vengono ora riportati gli spostamenti calcolati dall'algoritmo con i margini calibrati:

- **Linea verde:** spostamento iniziale calcolato dall'algoritmo (6.49 mm)
- **Linea viola:** spostamento finale calcolato dall'algoritmo (44.05 mm)

Come si può osservare gli spostamenti calcolati dall'algoritmo rientrano pienamente nel $\pm 5\%$ rispetto ai valori di riferimento, si può quindi affermare che il processo di taratura ha permesso di raggiungere risultati positivi.

Come parte conclusiva di questo sotto capitolo si riporta un ultimo esempio di calcolo degli spostamenti per la stessa curva precedente mediante l'algoritmo che considera gli spostamenti imposti in percentuale. Si consideri la **Figura 49**:

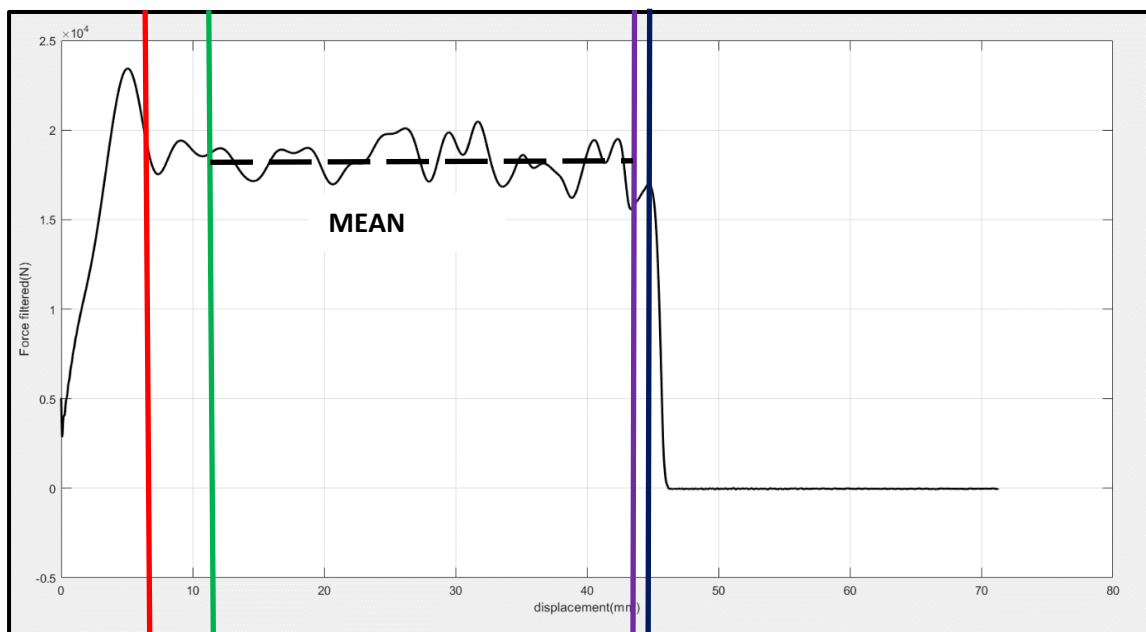


Figura 49: Curva di crash utilizzata come esempio per il processo di taratura dei parametri: algoritmo che considera gli spostamenti imposti in percentuale

Gli spostamenti presi come riferimento sono i medesimi del caso precedente e sono indicati in **Figura 49** facendo capo alla stessa legenda.

Gli spostamenti calcolati dall'algoritmo sono i seguenti:

- **Linea verde:** spostamento iniziale (11.42 mm)
- **Linea viola:** spostamento finale (43.41 mm)

Dai risultati ottenuti si evince che lo spostamento finale calcolato rientra nel margine del **5 %** rispetto al valore di riferimento, mentre lo spostamento iniziale calcolato risulta essere al di fuori di tale range. Questo esempio è stato riportato appositamente per sottolineare che il processo di taratura non garantisce in maniera assoluta un errore inferiore al **5 %**, ma fa sì che mediamente i parametri implementati nel codice permettano di ottenere degli spostamenti utilizzabili per il calcolo della SEA, trascurando il minor tratto utile possibile in tale operazione. Quanto detto sino ad ora non implica che il processo di taratura è risultato inefficace, ma solamente che ci sono dei casi in cui si possono riscontrare errori superiori al **5 %** sul calcolo degli spostamenti. Come controesempio viene riportato un caso in cui il calcolo degli spostamenti, svolto mediante l'algoritmo che considera gli spostamenti imposti in percentuale, fornisca risultati perfettamente nel range. Si osservi la curva in **Figura 50**:

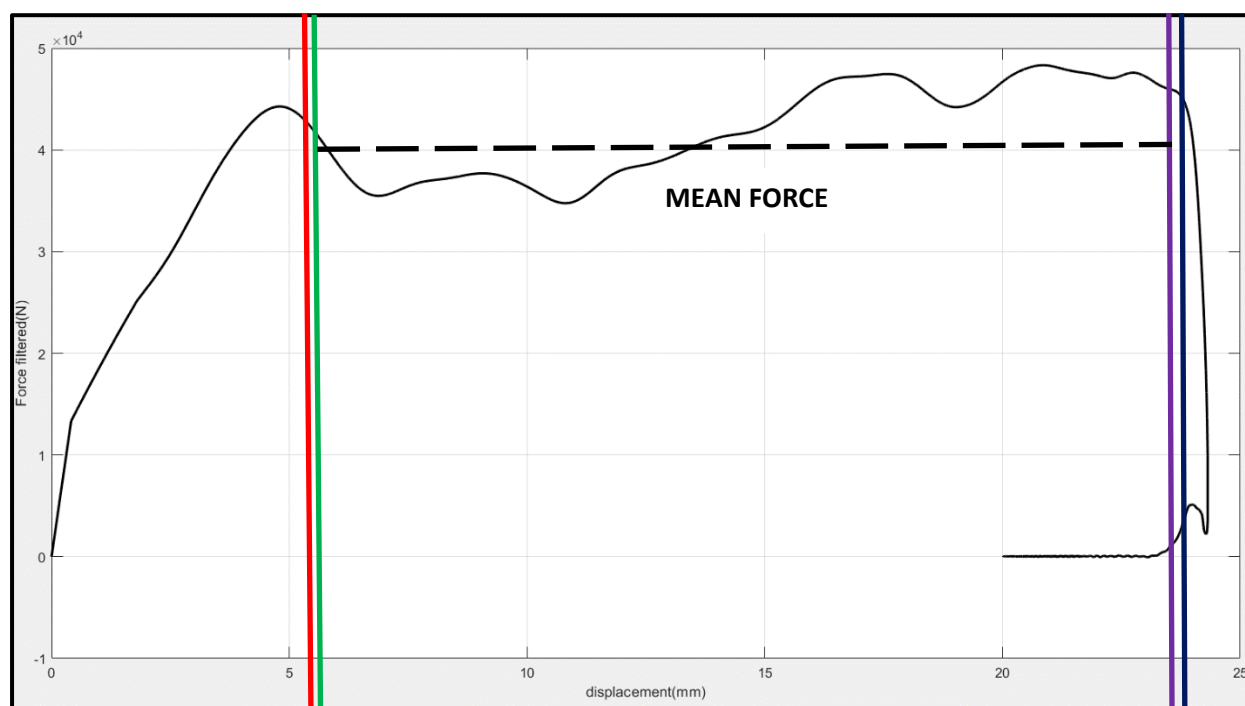


Figura 50: Curva di crash ottenuta per un provino piatto C50(matrice in resina epossidica rinforzata con fibre di carbonio, spessore 5mm)

Legenda:

- **Linea rossa:** Spostamento iniziale preso come riferimento (6 mm)
- **Linea blu:** spostamento finale preso come riferimento (23.67 mm)

I valori di spostamento calcolato dall'algoritmo sono i seguenti:

- **Linea verde:** spostamento iniziale (6.08 mm)
- **Linea viola:** spostamento finale (23.1 mm)

Si vede chiaramente come in questo caso gli spostamenti calcolati rientrano nel range del **5 %** rispetto a quelli di riferimento.

Questo conferma quanto ribadito precedentemente, ovvero che il processo di calibrazione è oneroso e non garantisce un calcolo degli spostamenti nel range di errore prefissato per il **100 %** dei casi, ma assolutamente necessario al fine di ridurre al minimo il margine di errore medio sul calcolo della SEA, e questo è un risultato importante che si potrà maggiormente apprezzare nel capitolo dedicato al confronto tra i vari algoritmi.

4. Valutazione degli algoritmi

4.1. Setup sperimentale

Nel capitolo precedente sono stati esplorati e descritti in maniera dettagliata diversi approcci al calcolo della SEA per laminati compositi, di conseguenza sono stati sviluppati tre differenti algoritmi che vengono elencati qui di seguito:

- **Algoritmo che considera variabili gli spostamenti**
- **Algoritmo che considera gli spostamenti imposti in percentuale**
- **Algoritmo che considera l'area sottesa a tutta la curva**

A questo punto è necessario effettuare una scrematura finale in quanto all'interno del software Bluehill Impact verrà implementato solamente uno degli algoritmi per il calcolo della SEA.

Al fine di compiere una scelta adeguata bisogna scegliere un fattore discriminante che permetta un confronto oggettivo tra gli algoritmi e costituisca un metro di confronto valido a livello sperimentale. Nel caso specifico di questo progetto di tesi si è deciso di seguire il seguente percorso logico per la valutazione degli algoritmi:

- **Eseguire un determinato numero di test di impatto nelle stesse condizioni in termini di setup del supporto e di provini utilizzati**
- **Eseguire il calcolo della SEA per tutte le curve ottenute utilizzando ciascun algoritmo**
- **Valutare quale dei tre algoritmi proposti fornisce la minor deviazione standard e garantisce la minor dispersione possibile dei dati**

Il setup sperimentale che conduce alla scelta di uno dei tre algoritmi valutati come migliore per il calcolo delle SEA, ha inizio con la misura dei vari provini impiegati nella campagna sperimentale. A titolo di esempio viene illustrato in **Figura 51** uno dei provini utilizzati per i test:

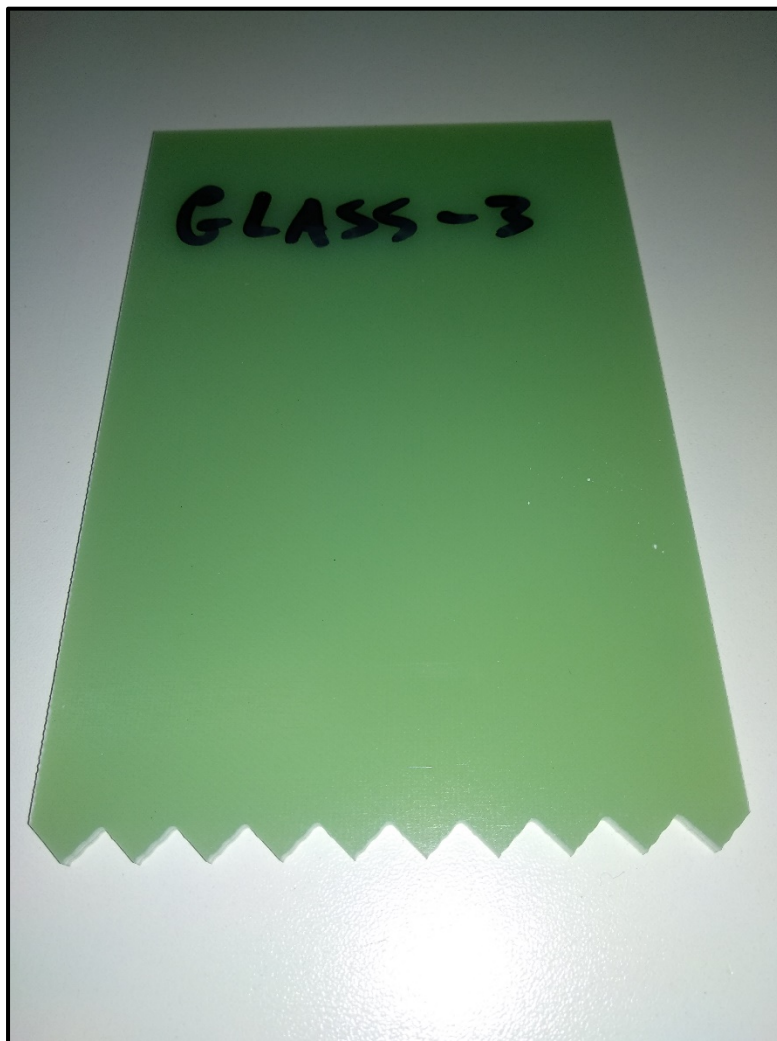


Figura 51: Provino piatto G30 utilizzato per i test finalizzati alla valutazione degli algoritmi proposti

Il provino raffigurato nella precedente didascalia è un provino **G30**, quindi avente matrice in resina epossidica rinforzata con fibre di vetro ed uno spessore di **3 mm**.

Le dimensioni nominali del provino sono riportate qui di seguito:

- **Larghezza: 100 mm**
- **Lunghezza: 150 mm**
- **Spessore: 3 mm**
- **Altezza Trigger a dente di sega: 5 mm**

La misura dei provini è necessaria in quanto è ben noto che nella realtà le dimensioni di un manufatto possono essere leggermente differenti da quelle nominali, questo è un aspetto intrinseco al processo produttivo.

Per la misura dei provini è stato utilizzato un calibro elettronico che ha presenta una sensibilità del centesimo di millimetro. In **Tabella 1** vengono riportate tutte le misure relative ai provini utilizzati per la valutazione degli algoritmi:

Tabella 1: Tabella contenente le misure relative ai provini utilizzati per la campagna sperimentale

Colonn a1	Colonn a2	Colonn a3	Colonn a4	Colonn a5	Colonn a6	Colonn a7	Colonn a8	Colonn a9	Colonna 10
Nome (Provin o)	Massa (g)	Larg 1 (mm)	Larg 2 (mm)	Lung 1 (mm)	Lung 2 (mm)	Spess 1 (mm)	Spess 2 (mm)	Spess 3 (mm)	Spess 4 (mm)
glass_1	92.98	99.95	99.95	149.97	150.00	3.12	3.13	3.07	3.03
glass_2	94.14	100.36	100.42	149.68	149.77	3.15	3.16	3.13	3.12
glass_3	93.87	100.53	100.31	149.90	149.88	3.14	3.11	3.05	3.03
glass_4	93.99	100.31	100.63	150.01	150.00	3.02	3.02	3.13	3.15
glass_5	94.85	100.40	100.36	149.93	149.95	3.15	3.16	3.18	3.15
glass_6	95.05	100.41	100.31	149.96	149.95	3.17	3.14	3.15	3.17
glass_7	95.11	100.42	100.39	149.88	150.04	3.12	3.16	3.17	3.15
glass_8	95.09	100.32	100.28	150.11	150.02	3.16	3.13	3.18	3.19
glass_9	94.86	100.31	100.35	149.97	149.94	3.15	3.17	3.15	3.14
glass_1 0	94.64	100.60	100.45	149.97	149.97	3.17	3.14	3.11	3.14
glass_1 1	92.99	100.04	100.08	149.79	149.79	2.95	2.95	3.11	3.11

Partendo con ordine, nella colonna 1 in **Tabella 1** sono stati inseriti i nomi dei vari provini utilizzati, mentre nelle colonne 2-10 sono state riportate le varie misure eseguite sugli stessi.

Il nome dei provini è stato scelto in modo da risalire facilmente alla tipologia di provino; infatti, sono stati denominati con il prefisso “glass” per ricordare che sono realizzati con matrice in resina epossidica rinforzata da fibre di vetro. Il suffisso numerico indica semplicemente il numero del provino, per questo lavoro sono stati utilizzati 11 provini nominalmente identici. I provini utilizzati per questa campagna sperimentale sono del tipo **NEMA FR4**, realizzati mediante taglio di una piastra in pre-impregnato (tessuto plain in fibra di vetro e matrice in resina epossidica) prodotta come semilavorato commerciale dall’omonima azienda. Vengono inoltre riportate in **Tabella 2** le caratteristiche meccaniche nominali di questa tipologia di laminato:

Tabella 2: Caratteristiche fisiche e meccaniche nominali provini FR4

E _{LONG} (Gpa)	Densita' (g/cm ³)	Resistenza flessionale (Mpa)	Resistenza a trazione (Mpa)	Resistenza a compressione (Mpa)
24	1.9	340	260	330

Viene ora riportato uno schema riassuntivo delle misure effettuate sui provini:

- **Massa: 1 misurazione**
- **Larghezza: 2 misurazioni**
- **Lunghezza: 2 misurazioni**
- **Spessore: 4 misurazioni**

Per la massa è stata effettuata un'unica misurazione in quanto misure ripetute su uno stesso provino non hanno evidenziato variazioni del valore misurato. È stato invece riscontrato che, le dimensioni presentano una certa variabilità derivante dal processo produttivo e quindi su di esse è stata presa più di una misura.

Più precisamente per la larghezza e la lunghezza sono state eseguite due misurazioni, una per ogni lato del provino, mentre per lo spessore sono state effettuate due misurazioni per lato in quanto esso presenta una variabilità maggiore rispetto alle altre grandezze se rapportata alla dimensione nominale, contribuendo in maniera decisiva all'incertezza di misura. Una volta terminata la fase di misura di tutti i provini da testare per ciascun provino vengono determinate le grandezze medie, riportate in **Tabella 3**.

Tabella 3: Tabella contenente i valori medi delle misure per ogni provino della campagna sperimentale

Colonna1	Colonna2	Colonna3	Colonna4	Colonna5	Colonna6	Colonna7
Massa (g)	Larg (mm)	Lung (mm)	Spessore (mm)	Sezione (mm ²)	Volume (mm ³)	Densità (g/cm ³)
92.98	99.95	149.99	3.09	308.60	45512.84	2.04
94.14	100.39	149.73	3.14	315.22	46412.00	2.03
93.87	100.42	149.89	3.08	309.54	45627.02	2.06
93.99	100.47	150.01	3.08	309.45	45648.69	2.06
94.85	100.38	149.94	3.16	317.20	46771.09	2.03
95.05	100.36	149.96	3.16	316.89	46729.37	2.03
95.11	100.41	149.96	3.15	316.28	46641.21	2.04
95.09	100.30	150.07	3.17	317.45	46846.81	2.03
94.86	100.33	149.96	3.15	316.29	46641.19	2.03
94.64	100.53	149.97	3.14	315.65	46552.81	2.03
92.99	100.06	149.79	3.03	303.18	44656.10	2.08

A questo punto sono state calcolate le grandezze medie e relativa deviazione standard considerando tutto il set di provini, riportate in **Tabella 4**, determinate come la media di tutte le misure riportate in ciascuna delle colonne **1-7** in **Tabella 3**.

Tabella 4: Misure medie calcolate considerando tutto il set di provini

Colonna1	Colonna2	Colonna3	Colonna4	Colonna5	Colonna6	Colonna7	Colonna8
Media	94.33	100.35	149.95	3.12	314.26	46338.30	2.04
Dev st	0.70	0.16	0.09	0.03	3.56	526.69	0.01
Dev st/media	0.74%	0.16%	0.06%	1.10%	1.13%	1.14%	0.56%

Analizzando quanto detto sino ad ora si può facilmente comprendere che il calcolo della SEA, a parità di algoritmo, può essere eseguito con due differenti modalità:

- Utilizzando le misure medie di tutto il set di provini (**Tabella 4**)
- Utilizzando le misure medie del singolo provino (**Tabella 3**)

Nel caso specifico di questo lavoro di tesi il calcolo è stato eseguito seguendo la prima modalità elencata, ciò però non toglie che la seconda è stata comunque valutata. Ovviamente seguire l'una o l'altra modalità porta a valori di SEA leggermente differenti, questo effetto verrà descritto più nel dettaglio nel capitolo successivo, ora è importante concentrarsi sul setup sperimentale che ha permesso di ottenere le varie curve di crash utilizzate per l'analisi dei dati.

Conclusa la parte relativa alla misura dei provini e al calcolo delle misure medie il prossimo step prevede il montaggio del provino all'interno del supporto crashworthiness. Prima di tutto le colonnine anti-buckling del supporto sono state rivestite con un sottile strato di Teflon in modo da ridurre al minimo l'effetto dell'attrito sul valore della SEA.

Una volta che lo strato di Teflon si è consolidato le colonnine vengono fissate al supporto e portate all'altezza designata, nel caso dei test volti si è deciso di lavorare con un'altezza di 10 mm rispetto alla base del supporto in funzione del lavoro svolto in parallelo dal team di ricerca di Instron sulla standardizzazione dei parametri del supporto. Successivamente il provino viene posizionato tra le due piastre di clampaggio e le viti vengono portate a battuta assicurandosi sempre che il provino sia centrato. A questo punto si utilizza una chiave dinamometrica per fissare la forza di clampaggio ad un valore ben definito, in questo caso la chiave è stata settata con una coppia di chiusura di **1 Nm** quindi la forza di clampaggio risultante è **4 kN**.

Fissato il provino tra le due piastre di clampaggio del supporto si procede montando il supporto nella camera di test, la quale può essere utilizzata anche come camera termostatica, mediante gli appositi bulloni.

Appena si è certi che il supporto sia ben saldo alla base della torre si chiude lo sportello della camera di test, si fornisce potenza alla macchina, la quale prima di poter effettuare misurazioni necessita di un'operazione di start che deve essere avviata manualmente dal pannello on board, e si apre il software bluehill.

Una volta terminata l'inizializzazione della torre (il carro sale sino ad arrivare a contatto con il sistema di alta energia e successivamente torna alla posizione definita di 'home') si può prendere il controllo da remoto ed impostare tutti i parametri di test mediante il software dedicato. Prima di impostare qualsiasi parametro bisogna caricare il carrello porta masse in modo che i calcoli che svolge il software siano coerenti con il sistema fisico reale.

L'operazione di caricamento delle masse è un'operazione molto delicata, infatti bisogna assicurarsi che ogni piatto si incastri perfettamente su quello sottostante e che il perno che le tiene insieme sia stretto opportunamente. Completate tali verifiche si può chiudere lo sportello di sicurezza e si può continuare con l'impostazione dei parametri, seguendo i seguenti step:

- **Creazione di un nuovo metodo oppure scegliendone uno già esistente che soddisfa le caratteristiche del test che si intende svolgere**
- **Impostare i parametri**

I parametri che maggiormente influenzano la prova e che quindi risulta fondamentale settare in maniera corretta sono:

- **Energia di impatto**
- **Velocità di impatto**
- **Range di lavoro del percussore**
- **Frequenza di campionamento**

Analizziamo perché è importante settare correttamente i parametri in modo da svolgere agevolmente il test e non danneggiare la torre. Partendo dall'energia e dalla velocità di impatto si può affermare che esse sono strettamente legate in virtù del principio di conservazione dell'energia meccanica; infatti, a parità di energia si può decidere se incrementare la massa del sistema oppure la velocità di impatto in funzione della tipologia di test che si intende eseguire. L'energia di impatto dipende strettamente dalla tipologia di provino che si vuole testare, ad esempio per provini molto sottili non si può pensare di lavorare con energie elevate in quanto si andrebbe ad impattare violentemente contro i deceleratori del supporto rischiando di danneggiarlo in maniera permanente.

Al contrario con provini molto spessi se si lavora con energie di impatto troppo esigue si rischia di ottenere una corsa utile troppo ridotta e quindi una curva con un tratto utile non adeguato al calcolo della SEA.

È molto importante valutare tutti questi fattori al fine di eseguire un test valido e non fratturare un provino senza ottenere informazioni utili dal test.

Prima di proseguire con la descrizione del test si fa una piccola digressione sul percussore e sulla torre utilizzata per questo lavoro di tesi. La torre utilizzata per i test discussi sino ad ora è il modello di Instron 9450 alimentata sia elettricamente sia ad aria compressa.

Per lo sviluppo di questo progetto di tesi è stato montato sulla torre un percussore da **220 kN** all'interno del quale è integrata una cella di carico estensimetrica in grado di misurare valori di forza sino al limite appena citato. In realtà nei test eseguiti il percussore è stato settato in modo da misurare forze sino al 50% della sua massima capacità in modo da rendere più precisa la misura. Si è potuta fare questa scelta in quanto le forze in gioco nei test crashworthiness difficilmente superano soglie di **110 kN**. La testa del percussore inoltre è stata ottimizzata in termini di geometria per ridurre al minimo i fenomeni di buckling del provino nella parte superiore ed ottenere quindi curve che siano utilizzabili per la campagna sperimentale [10]. Ultimo parametro da considerare, ma non meno importante, è la frequenza di campionamento del segnale, la quale scandisce ogni quanto la cella di carico integrata nel percussore esegue una misura. Solitamente per i test crashworthiness si imposta una frequenza di campionamento di **1 MHz**, ciò significa che per **1 ms** che scorre vengono misurati **1000** punti.

La frequenza di campionamento a prima vista può sembrare un parametro a sé stante, ma non è assolutamente corretto in quanto influenza tutte le operazioni di filtraggio che sono state spiegate al capitolo 2.

Dopo questa digressione si può procedere con l'avanzamento del test. Prima di eseguire il test di impatto vero e proprio è necessario svolgere ancora due ulteriori operazioni:

- **Portare a zero il percussore**
- **Posizionare correttamente la fotocellula**

Entrambe le operazioni e la loro funzione è stata spiegata in maniera dettagliata al capitolo 1, quindi non vengono nuovamente trattate dettagliatamente per non appesantire la trattazione.

I parametri impostati per gli 11 test sperimentali eseguiti in questo lavoro di tesi sono i seguenti:

- **Energia di impatto: 800 J**
- **Massa: 60.390 kg**
- **Velocità di impatto: 5.15 m/s**

Una volta che si è sicuri che tutto è pronto per il test e che la torre non mostri errori per quanto riguarda il setting dei parametri si può avviare il test.

Il carro dalla posizione di 'home', raggiunta dopo la posizione di zero, viene trascinato dal motore ad una certa altezza in funzione dei parametri di test e a questo punto viene lasciato cadere impattando sul provino.

Quanto detto sino ad ora accade quando le velocità di impatto impostate sono inferiori a **5 m/s**, in questo caso la velocità settata è di **5.15 m/s**, di conseguenza il carro si deve agganciare al sistema di alta energia affinché il percussore urti il provino con la velocità designata.

Tutti gli step descritti sino ad ora, se svolti in maniera corretta, permettono di fratturare correttamente il provino e di ottenere una curva di crash che sia utilizzabile per calcolare la SEA. Qui di seguito, in **Figura 52**, viene raffigurato uno dei provini testati durante l'impatto e dopo l'impatto:

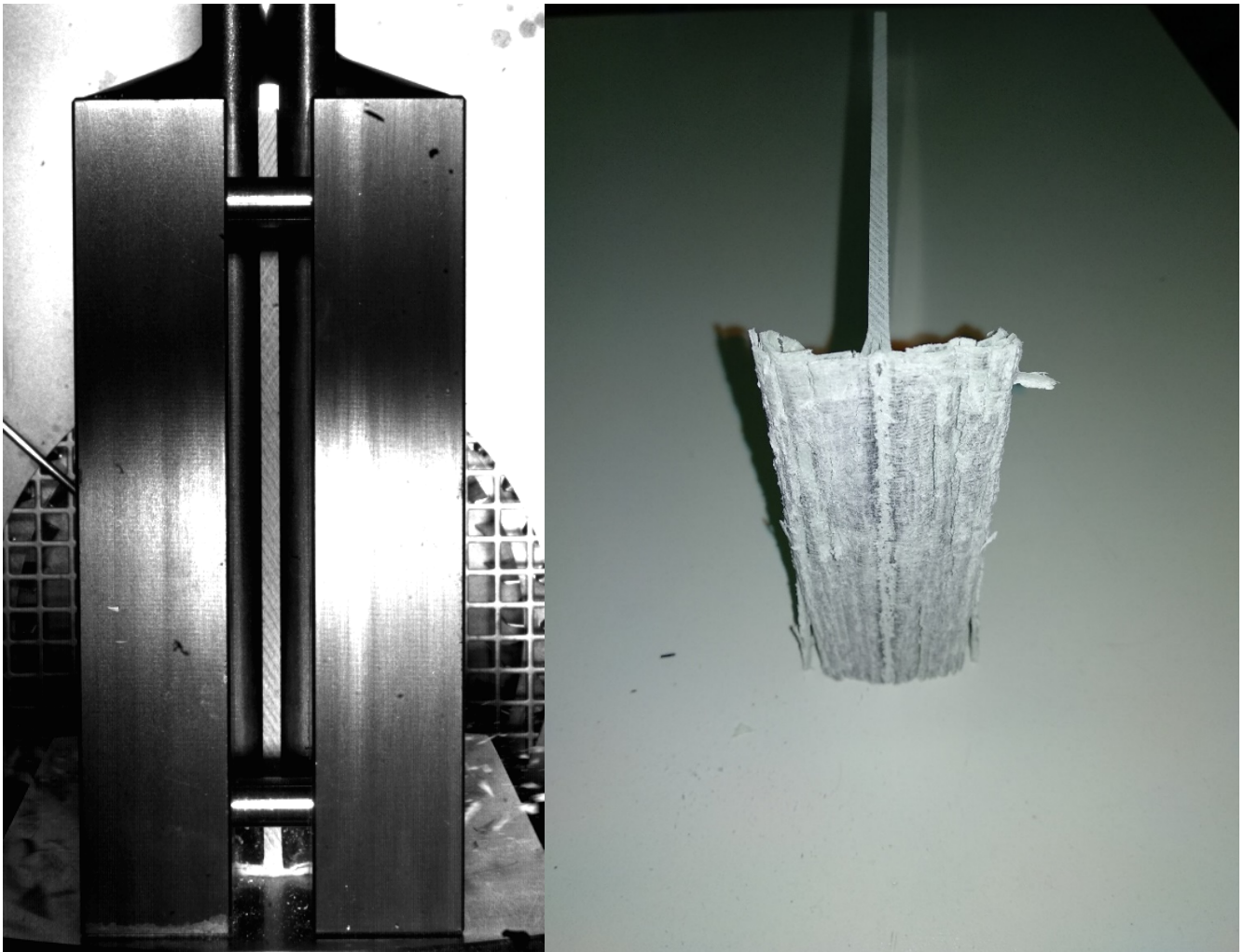


Figura 52: a) Provino 'glass-4' durante l'impatto || b) provino 'glass-4' dopo l'impatto

Osservando la **Figura 52** si evince come il provino si sia fratturato abbastanza coerentemente con quanto ci si aspetta da questa tipologia di test, questo implica che la scelta dei parametri di test è corretta e permette di ottenere curva utilizzabili per l'analisi della SEA dei vari provini. Ciò è confermato dalla curva registrata per questo test, riportata qui di seguito in **Figura 53**:

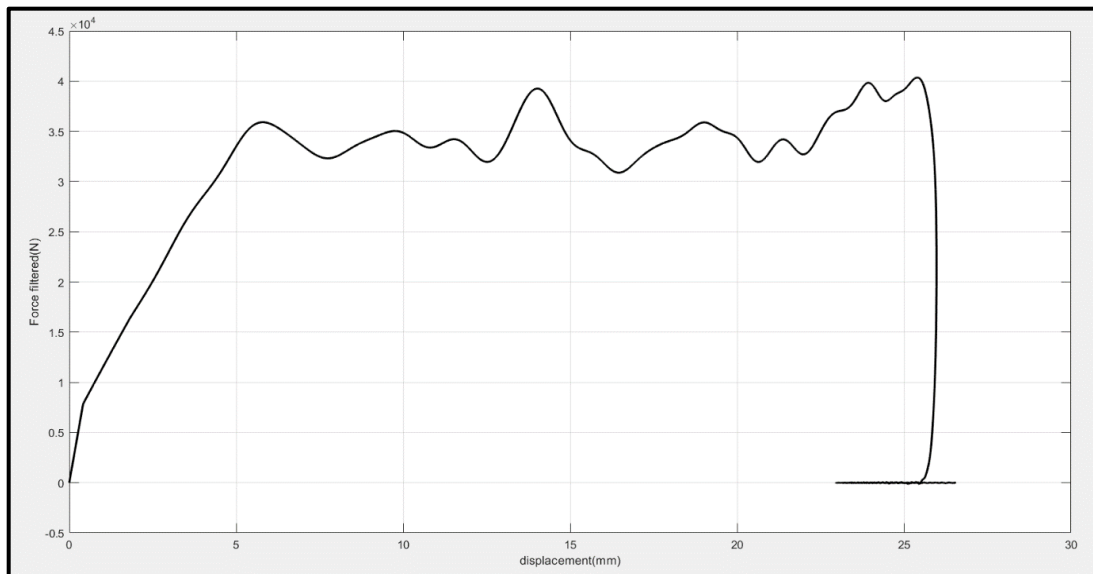


Figura 53: Curva di crash ottenuta testando il provino 'glass-4'

Il modus operandi analizzato sino ad ora è stato replicato fedelmente per tutti gli 11 test svolti con esito positivo, non è stata quindi riscontrata alcuna problematica nell'acquisizione delle varie curve di crash.

Con questo si conclude la sezione relativa al setup sperimentale; nel prossimo sotto capitolo verranno analizzate le curve ottenute in termini di SEA, permettendo così di confrontare i vari algoritmi proposti e portando alla luce l'approccio che minimizza la dispersione dei dati e garantisce la migliore precisione di calcolo.

4.2. Confronto degli algoritmi

I test analizzati e descritti nel precedente sotto capitolo sono stati eseguiti tutti nelle stesse condizioni, ovvero:

- Stessa tipologia di provino
- Stesso materiale
- Stessa altezza delle colonnine
- Stesse condizioni di attrito
- Stessa forza di clampaggio
- Stessa temperatura
- Stessa energia di impatto
- Stessa velocità di impatto

Nelle condizioni appena elencate, almeno a livello teorico, il valore di SEA calcolato dovrebbe essere identico per ogni provino testato. Quanto appena detto però, non si verifica a causa delle diversità derivanti dal processo produttivo, soprattutto in termini di distribuzione delle difettosità e delle fibre, presenti tra i vari provini. Ciò conduce inevitabilmente ad una differenza di SEA calcolata per le varie curve a parità di algoritmo. L'obiettivo primario è quello di individuare quale tra i tre approcci di calcolo proposti sia in grado di minimizzare questa dispersione e garantire la maggior precisione di calcolo possibile. Per evidenziare l'inefficacia dell'algoritmo che prende in input spostamenti imposti da un operatore è stato inserito nel confronto con gli altri tre approcci, al fine di rendere ancor più solida la necessità dell'oggettività del calcolo della SEA. Qui di seguito, in **Tabella 5**, **Tabella 6**, **Tabella 7** e **Tabella 8** viene riportato l'output in termini di SEA e spostamenti calcolati per ognuno dei provini e con ciascun algoritmo proposto:

Tabella 5: Tabella contenente i valori di SEA calcolati mediante l'algoritmo che considera gli spostamenti imposti da un operatore

Lista Curve sperimentali	SEA (kJ/kg)
2021_11_04 Prove tesi Enio_1_Export_Force_RawData_1.csv	45.3729
2021_11_04 Prove tesi Enio_2_Export_Force_RawData_2.csv	56.5013
2021_11_04 Prove tesi Enio_3_Export_Force_RawData_3.csv	52.6311
2021_11_04 Prove tesi Enio_4_Export_Force_RawData_4.csv	55.1630
2021_11_04 Prove tesi Enio_5_Export_Force_RawData_5.csv	47.0391
2021_11_04 Prove tesi Enio_6_Export_Force_RawData_6.csv	51.6438
2021_11_04 Prove tesi Enio_7_Export_Force_RawData_7.csv	52.2757
2021_11_04 Prove tesi Enio_8_Export_Force_RawData_8.csv	49.7998
2021_11_04 Prove tesi Enio_90_Export_Force_RawData_9.csv	56.3804
2021_11_04 Prove tesi Enio_91_Export_Force_RawData_10.csv	51.3265
2021_11_04 Prove tesi Enio_92_Export_Force_RawData_11.csv	50.5498

Tabella 6: Tabella contenente i valori di SEA calcolati mediante l'algoritmo che considera gli spostamenti imposti in percentuale

Lista curve	SEA (kJ/kg)	start displacement (mm)	final displacement (mm)
2021_11_04 Prove tesi Enio_1_Export_Force_RawData_1.csv	46.1985	6.3750	24.2250
2021_11_04 Prove tesi Enio_2_Export_Force_RawData_2.csv	56.6737	6.1298	23.2931
2021_11_04 Prove tesi Enio_3_Export_Force_RawData_3.csv	52.4055	6.3608	24.1709
2021_11_04 Prove tesi Enio_4_Export_Force_RawData_4.csv	54.1917	6.4915	24.6677
2021_11_04 Prove tesi Enio_5_Export_Force_RawData_5.csv	47.5304	7.1680	27.2384
2021_11_04 Prove tesi Enio_6_Export_Force_RawData_6.csv	50.5353	6.7268	25.5617
2021_11_04 Prove tesi Enio_7_Export_Force_RawData_7.csv	51.8524	6.6770	25.3726
2021_11_04 Prove tesi Enio_8_Export_Force_RawData_8.csv	49.1882	6.8648	26.0861
2021_11_04 Prove tesi Enio_90_Export_Force_RawData_9.csv	55.8966	6.2935	23.9153
2021_11_04 Prove tesi Enio_91_Export_Force_RawData_10.csv	50.4157	6.7195	25.5341
2021_11_04 Prove tesi Enio_92_Export_Force_RawData_11.csv	49.7346	6.7933	25.8144

Tabella 7: Tabella contenente i valori di SEA calcolati mediante l'algoritmo che considera l'area sottesa a tutta la curva

Lista curve	SEA (kJ/kg)	start displacement (mm)	final displacement (mm)
2021_11_04 Prove tesi Enio_1_Export_Force_RawData_1.csv	42.3518	0.0000	27.1460
2021_11_04 Prove tesi Enio_2_Export_Force_RawData_2.csv	53.3142	0.0000	24.5450
2021_11_04 Prove tesi Enio_3_Export_Force_RawData_3.csv	43.0955	0.0000	27.6770
2021_11_04 Prove tesi Enio_4_Export_Force_RawData_4.csv	52.1005	0.0000	25.9220
2021_11_04 Prove tesi Enio_5_Export_Force_RawData_5.csv	43.6132	0.0000	28.6710
2021_11_04 Prove tesi Enio_6_Export_Force_RawData_6.csv	49.3657	0.0000	26.8830
2021_11_04 Prove tesi Enio_7_Export_Force_RawData_7.csv	49.3539	0.0000	26.7190
2021_11_04 Prove tesi Enio_8_Export_Force_RawData_8.csv	46.5590	0.0000	27.4250
2021_11_04 Prove tesi Enio_90_Export_Force_RawData_9.csv	53.2192	0.0000	25.1560
2021_11_04 Prove tesi Enio_91_Export_Force_RawData_10.csv	49.6969	0.0000	26.8350
2021_11_04 Prove tesi Enio_92_Export_Force_RawData_11.csv	48.5066	0.0000	27.1190

Tabella 8: Tabella contenente i valori di SEA calcolati mediante l'algoritmo con spostamenti variabili

Lista curve	SEA (kJ/kg)	start displacement (mm)	final displacement (mm)
2021_11_04 Prove tesi Enio_1_Export_Force_RawData_1.csv	46.0887	6.5000	23.9980
2021_11_04 Prove tesi Enio_2_Export_Force_RawData_2.csv	56.6008	6.5020	23.0190
2021_11_04 Prove tesi Enio_3_Export_Force_RawData_3.csv	52.3231	6.5020	23.9430
2021_11_04 Prove tesi Enio_4_Export_Force_RawData_4.csv	54.0356	6.5010	24.4660
2021_11_04 Prove tesi Enio_5_Export_Force_RawData_5.csv	47.6263	6.5020	27.1720
2021_11_04 Prove tesi Enio_6_Export_Force_RawData_6.csv	50.4371	6.5010	25.4070
2021_11_04 Prove tesi Enio_7_Export_Force_RawData_7.csv	51.8265	6.5000	25.2080
2021_11_04 Prove tesi Enio_8_Export_Force_RawData_8.csv	49.3300	6.5010	25.9590
2021_11_04 Prove tesi Enio_90_Export_Force_RawData_9.csv	55.8229	6.5030	23.6740
2021_11_04 Prove tesi Enio_91_Export_Force_RawData_10.csv	50.2704	6.5020	25.3780
2021_11_04 Prove tesi Enio_92_Export_Force_RawData_11.csv	49.7427	6.5000	25.6730

Osservando le precedenti didascalie si nota come i valori di SEA calcolati con i differenti algoritmi sono diversi, ma questo lo si poteva prevedere vista la differente metodologia di calcolo degli spostamenti. Si noti però anche la diversità dei valori di SEA a parità di algoritmo, questo è appunto dovuto alle motivazioni precedentemente citate. A questo punto si posseggono tutti gli elementi per poter valutare i tre algoritmi dal punto di vista della dispersione dei risultati. Al fine di raggiungere tale obiettivo per ciascun set di valori di SEA ottenuti mediante un determinato approccio è stata calcolata la deviazione standard ed è stato selezionato come algoritmo con la maggior precisione di calcolo quello che minimizza tale valore. Qui di seguito, in **Tabella 9**, sono riportati i risultati ottenuti:

Tabella 9: Valori di SEA e relative deviazioni standard calcolati per ciascuno degli algoritmi discussi

	SEA Algoritmo con spostamenti imposti (kJ/kg)	SEA Algoritmo con spostamenti imposti in % (kJ/kg)	SEA Algoritmo curva completa (kJ/kg)	SEA Algoritmo con spostamenti variabili (kJ/kg)
	45.3729	46.1985	42.3518	46.0887
	56.5013	56.6737	53.3142	56.6008
	52.6311	52.4055	43.0955	52.3231
	55.1630	54.1917	52.1005	54.0356
	47.0391	47.5304	43.6132	47.6263
	51.6438	50.5353	49.3657	50.4371
	52.2757	51.8524	49.3539	51.8265
	49.7998	49.1882	46.5590	49.3300
	56.3804	55.8966	53.2192	55.8229
	51.3265	50.4157	49.6969	50.2704
	50.5498	49.7346	48.5066	49.7427
dev.standard	3.53527	3.29237	3.94321	3.25768

A questo punto si può proseguire con il confronto vero e proprio.

Osservando la **Tabella 8** si nota immediatamente come la minore deviazione standard risulta essere quella calcolata per l'algoritmo che considera gli spostamenti variabili. Questo risultato non è una sorpresa, infatti deriva dalla tipologia di controlli ed iterazioni implementate all'interno del codice, i quali sono in grado escludere dal calcolo della SEA i tratti di curva con andamenti anomali e considerare solamente il tratto di plateau della forza.

In **Figura 54** viene riportato un grafico a barre in modo da visualizzare sotto un'altra prospettiva i risultati mostrati nella precedente didascalia:

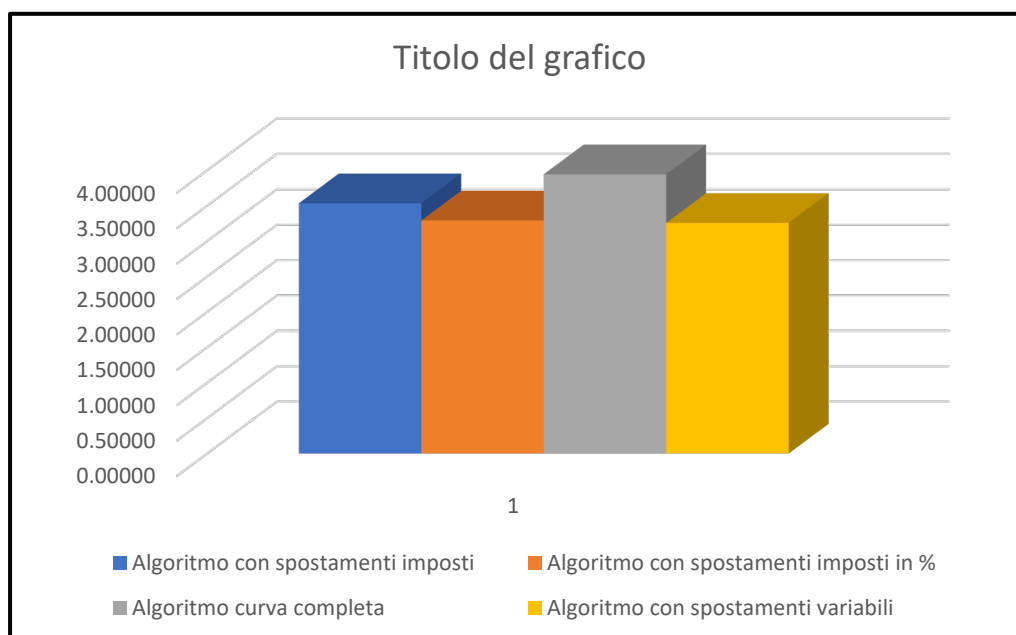


Figura 54: Grafico a barre: deviazione standard calcolata per ciascuno degli algoritmi proposti

Dai risultati appena mostrati ed analizzati si può arrivare alla conclusione che l'algoritmo che garantisce la maggiore precisione di calcolo è quello che considera gli spostamenti variabili; perciò, quest'ultimo verrà implementato nel software Bluehill Impact per il calcolo della SEA. Questo costituisce un passo in avanti verso la standardizzazione delle prove crashworthiness sui materiali compositi, anche se è solamente una piccola parte del lavoro attualmente condotto con tale obiettivo.

5. Calcolo della SEA mediante algoritmo con spostamenti variabili: effetto della tipologia di misura dei provini

Nel capitolo precedente si è arrivati a dire che l'algoritmo che considera gli spostamenti variabili risulta il migliore in termini di minimizzazione della dispersione dei risultati per quanto riguarda il calcolo della SEA. In questo progetto di tesi, come già accennato, il calcolo della SEA sugli 11 test sperimentali condotti è stato effettuato utilizzando le misure medie di tutto il set di provini. Come si può ben intuire però, effettuando il calcolo utilizzando le misure medie del singolo provino può portare a differenti valori di SEA calcolati, ci sono quindi due possibilità:

- Incremento della deviazione standard
- Riduzione della deviazione standard

Questo capitolo ha proprio lo scopo di indagare i risultati che si ottengono effettuando i calcoli della SEA sulle medesime curve, ma utilizzando le misure medie del singolo provino oppure dell'intero set.

Qui di seguito, in **Tabella 10**, è riportato tale confronto:

Tabella 10: Valutazione della differenza sui valori di SEA ottenuti utilizzando le misure medie

	SEA valori medi(kJ/kg)	SEA valori singoli(kJ/kg)	err perc SEA(%)
	46.0886	46.8677	1.6624
	56.6025	56.7600	0.2774
	52.3218	52.7501	0.8120
	54.0371	54.3586	0.5915
	47.6264	47.4613	0.3479
	50.4398	50.1626	0.5525
	51.8265	51.5272	0.5809
	49.3318	49.0766	0.5199
	55.8234	55.7368	0.1554
	50.2721	49.6906	1.1702
	49.7427	51.0417	2.5450
dev std(kJ/kg)	3.257869464	3.227444066	

di tutto il set di provini oppure le misure medie del singolo provino

Si analizza e si discute ora in dettaglio la tabella appena mostrata. Il primo fattore che è stato indagato è la differenza riscontrata tra i singoli valori di SEA calcolati mediante le due metodologie precedentemente citate. Per una valutazione oggettiva è stato determinato l'errore percentuale sui singoli valori di SEA, considerando come valori veri quelli ottenuti utilizzando le misure medie dei singoli provini. Osservando la **Tabella 10** si evince come l'errore percentuale calcolato per tutti i test eseguiti rimane sempre molto esiguo, solamente per l'ultimo test supera il **2 %**. Questo è un indizio che fa luce sulla piccola differenza che c'è nell'implementare il calcolo della SEA con i due diversi approcci. Un altro fattore molto importante valutato è la deviazione standard, che fornisce indicazioni riguardo la dispersione dei risultati.

Come riportato in **Tabella 10**, la deviazione standard calcolata risulta leggermente inferiore nel caso in cui il calcolo venga eseguito utilizzando le misure medie del singolo provino; ciò porterebbe a dire che sarebbe più opportuno calcolare la SEA con tale approccio piuttosto che l'altro in modo da rendere ancora inferiore la dispersione dei risultati. Quello che non è stato ancora tirato in ballo è però l'onerosità del calcolo, che viene ora spiegata più chiaramente.

Si supponga di voler analizzare **50** curve di crash e di voler calcolare la SEA utilizzando i valori medi del singolo provino; tale scelta implica necessariamente di dover inserire in input i dati relativi al provino ad ogni iterazione dell'algoritmo, in questo caso **50**, tante quante sono le curve. Si può ben immaginare che con un approccio del genere il tempo di analisi è notevolmente lungo e porta ad un calcolo oneroso.

Si immagini ora di utilizzare i dati medi del set di provini, in tal caso basta inserire i dati medi una volta sola e l'algoritmo eseguirà il calcolo della SEA sempre con i medesimi valori di densità, sezione ed altezza del trigger.

Non è difficile comprendere che con questa metodologia di calcolo il tempo di analisi è assai più ristretto e si è in grado di analizzare una mole di dati nettamente superiore nello stesso lasso di tempo.

Ponendo sui due piatti di una bilancia i due approcci si può affermare che eseguendo il calcolo utilizzando le misure del set di provini si riduce notevolmente il tempo di analisi ma si ha una dispersione dei risultati leggermente maggiore, mentre con l'altro approccio si riduce la dispersione a spese di un aumento esponenziale del tempo di analisi.

Valutando quanto detto solamente a livello letterale la scelta sembrerebbe ovvia, però osservando i risultati numerici si può arrivare alla seguente conclusione: l'errore che si commette effettuando il calcolo della SEA utilizzando le misure medie di tutto il set di provini è estremamente ridotto, di conseguenza accettabile a livello ingegneristico e tale da favorire questo approccio rispetto all'altro proposto, visto anche il notevole vantaggio in termini di tempo di analisi.

Pur essendo risultata più conveniente tale soluzione, per completezza, è stata sviluppata in parallelo una versione dell'algoritmo che lavora nella stessa identica maniera, l'unica sostanziale differenza sta nel fatto che questa versione alternativa è in grado di acquisire in input i dati del provino ad ogni iterazione e quindi di eseguire il calcolo utilizzando le misure medie del singolo campione.

6. Conclusioni

Come è ben noto, l'obiettivo primario di questo progetto di tesi è quello di sviluppare un algoritmo che sia in grado di analizzare una qualsiasi curva di crash e calcolarne il valore di SEA associato. All'avvio del lavoro non c'era alcuna certezza matematica riguardo la finalizzazione di tale traguardo, visti anche i numerosi studi già svolti da terzi sulla crashworthiness, i quali non hanno mai condotto ad un test standard per i compositi, questo non ha mai permesso un reale confronto oggettivo tra i diversi laminati impiegati in questo campo. Sino a questo momento il calcolo della SEA su tali materiali è stato sempre effettuato mediante analisi visiva della curva di crash, inglobando inevitabilmente la soggettività di chi osserva la curva stessa.

Questa tipologia di approccio non è ottimale quando si deve effettuare un confronto di proprietà meccaniche su una certa classe di materiali, soprattutto per la classe dei compositi, i quali presentano una serie di caratteristiche che rende difficile l'interpretazione e la previsione del loro comportamento meccanico:

- **Struttura ortotropa:** presentano differenti proprietà in termini di modulo elastico, coefficiente di Poisson e tensione di rottura, in funzione della direzione considerata
- **Differenza nella microstruttura, intrinseca ai processi produttivi:** diversa distribuzione delle fibre, difetti ecc.
- **Differente compatibilità tra fibre e matrice**

Tutte queste caratteristiche fanno sì che si possano sviluppare differenti meccanismi di frattura, soprattutto nelle prove crashworthiness, in cui si lavora con energie e velocità di impatto piuttosto elevate.

Vista la grande richiesta dei compositi nel campo dell'ingegneria, soprattutto per applicazioni in cui il peso e l'assorbimento di energia sono due fattori chiave, si rende necessaria una metodologia di calcolo che sia in grado di eliminare qualsiasi traccia di soggettività nel calcolo della SEA e rendere così prettamente oggettiva la caratterizzazione di questa classe di materiali.

L'algoritmo implementato con questo obiettivo ha preso forma in maniera graduale, come frutto di osservazioni e considerazioni fatte sulle curve di crash e sui test svolti durante tutta la campagna sperimentale.

Pur essendo già stato spiegato in dettaglio nel capitolo 3, si riassume in breve il funzionamento dell'algoritmo proposto in modo che il lettore abbia il quadro completo del lavoro svolto: l'algoritmo con spostamenti variabili prende come valori di spostamento iniziale e finale alla prima iterazione rispettivamente un valore pari alla profondità del trigger ed un valore identificato come lo spostamento oltre il quale i valori di forza divengono negativi.

Questo intervallo di spostamenti sancisce la porzione di curva all'interno della quale l'algoritmo si trova a svolgere le iterazioni che poi conducono al calcolo degli spostamenti iniziale e finale collocati sul tratto utile.

Definito tale intervallo, il codice itera confrontando ad ogni ciclo il valore i -esimo del vettore della forza con un valore limite, il cui calcolo è stato ampiamente discusso nel capitolo dedicato. Le iterazioni vengono effettuate sia per lo spostamento iniziale sia per lo spostamento finale, sino a che non viene trovato uno spostamento tale per cui il valore di forza associato sia maggiore o uguale al limite di forza. Il raggiungimento di tale condizione indica che gli spostamenti calcolati sono molto vicini al tratto utile, però per garantire una maggiore precisione si è applicato un margine su entrambi gli spostamenti. Il valore di margine tiene conto del fatto che nel tratto iniziale la forza non necessariamente cresce in maniera repentina, ma può anche subire una crescita graduale. Lo stesso ragionamento si può applicare per il decadimento dei valori di forza nel tratto finale della curva di crash.

Al termine delle iterazioni l'algoritmo ha individuato un range di spostamenti situato sul tratto utile, in cui si ha il plateau della forza. A questo punto prende i valori di forza all'interno del range di spostamenti appena determinato, ne calcola la forza media e di conseguenza la SEA utilizzando l'eq. (4).

Al fine di certificare la metodologia di calcolo proposta, essa è stata confrontata con altri due approcci al calcolo della SEA altrettanto validi; dal confronto è emerso che l'algoritmo proposto, come migliore per la finalizzazione dell'obiettivo del progetto sia effettivamente quello che garantisce la minor dispersione dei risultati e di conseguenza la minimizzazione della deviazione standard.

In conclusione, si può affermare che la solida collaborazione tra il Politecnico di Torino ed Instron ha reso possibile la realizzazione di questo progetto di tesi, permettendo di effettuare un salto di qualità nel campo della crashworthiness e portando alla risoluzione di una delle numerose problematiche legate alla definizione di uno standard per questa tipologia di test.

7. Appendice: codici degli algoritmi

Algoritmo con spostamenti imposti

```
%% IMPORT DELLE CURVE

lista_curve_da_analizzare= dir('*.csv');
[curve curva_analizzata
tabella_SEA]=import_curve_tabella_singolo_algoritmo_spostamenti_imposti(lista_curve_da_analizzare);
total=1;
v=2;

%% DATI IN INPUT

rho=input('rho='); % kg/m^3 densita' del provino
A=input('A='); %m^2 Sezione resistente del provino
start_displacement=input('start_displacement='); % mm
spostamento_iniziale imposto per calcolo forza media
final_displacement=input('final_displacement='); % mm
spostamento_finale imposto per calcolo forza media
%%
while total<=numel(lista_curve_da_analizzare)
    input=importdata(curve(total));
    Input=input.data;

    %% LETTURA DEI VETTORI TEMPO, FORZA E SPOSTAMENTO

    time=Input(:,2);
    time_cont=0;
    for z=1:length(time)
        if time(z)<0
            time_cont=time_cont+1;
        elseif time(z)==0
            time_cont=time_cont+1;
            break
        end
    end
    clear time
    time=Input(time_cont:end,2);
    force=Input(time_cont:end,3);
    displacement=Input(time_cont:end,6);
```

```

%% FILTRAGGIO DELLA CURVA

% rimozione delle frequenze che disturbano il segnale
force_frequency_filtered=bandstop(force,[1*10^3
40*10^3],1*10^6);
% arrotondamento dei picchi in modo che siano meno
spigolosi

force_filtered=smoothdata(force_frequency_filtered,'sgola
y',700);
%faccio partire la curva filtrata da zero(aspetto
grafico)
for i=1:80
    force_filtered(i)=0;
    displacement(i)=0;
    force(i)=0;
end
figure
subplot(3,1,1);
plot(time,force,'r','linewidth',1.8)
xlabel('time(ms) ');
ylabel('Force(N) ');
subplot(3,1,2);
plot(time,force_filtered,'k','linewidth',1.8)
xlabel('time(ms) ');
ylabel('Force filtered(N) ');
subplot(3,1,3);

plot(time,force,'r',time,force_filtered,'k','linewidth',1
.8)
xlabel('time(ms) ');
ylabel('Force(N) ');
figure
subplot(3,1,1);
plot(displacement,force,'r','linewidth',1.8)
xlabel('displacement(mm) ');
ylabel('Force(N) ');
subplot(3,1,2);
plot(displacement,force_filtered,'k','linewidth',1.8)
xlabel('displacement(mm) ');
ylabel('Force filtered(N) ');
subplot(3,1,3);

plot(displacement,force,'r',displacement,force_filtered,'
k','linewidth',1.8)
xlabel('displacement(mm) ');
ylabel('Force(N) ');

```

```

%% CALCOLO FORZA MEDIA: ALGORITMO CON SPOSTAMENTI
IMPOSTI

    %estraggo il numero di riga corrispondente allo
    spostamento iniziale imposto

[start_position]=trova_posizione_start(displacement,start
_displacement);
    % estraggo il numero di riga corrispondente allo
    spostamento finale imposto

[final_position]=trova_posizione_final(displacement,final
_displacement);
    % definisco un fattore sottrattivo che mi serve per
    riempire in maniera sequenziale il vettore force_final
    subtractive_factor= start_position-1;
    % inserisco all'interno del vettore force_final tutti
    i valori di
    % forza nell'intervallo di interesse
    for k=start_position:final_position
        force_final(k-subtractive_factor)=force(k);
    end
    force_average=mean(force_final);

%% CALCOLO SEA A SPOSTAMENTI IMPOSTI

delta=(final_displacement-start_displacement)*10^-3;
% m
mc=rho*A*delta;    % kg

SEA_spostamenti_imposti=((force_average*delta)/mc)/10^3;
% kJ/kg
    tabella_SEA.(v)(total)=SEA_spostamenti_imposti;
    clearvars -except rho A start_displacement
    final_displacement total v curve curva_analizzata
    lista_curve_da_analizzare tabella_SEA
    total=total+1;
end
%%
clearvars -except tabella_SEA

```

Algoritmo con spostamenti imposti in percentuale

```
%% IMPORT DELLE CURVE

lista_curve_da_analizzare= dir('*.csv');
[curve curva_analizzata
tabella_SEA]=import_curve_tabella_singolo_algoritmo_spostamenti_variabili(lista_curve_da_analizzare);
total=1;
for w=1:width(tabella_SEA)
    v(w)=w;
end

%% DATI IN INPUT

rho=input('rho='); % kg/m^3 densità del provino
A=input('A='); %m^2 Sezione resistente del provino
trigger=input('trigger='); %mm profondità del trigger
%%
while total<=numel(lista_curve_da_analizzare)
    input=importdata(curve(total));
    Input=input.data;

    %% LETTURA DEI VETTORI TEMPO, FORZA E SPOSTAMENTO

    time=Input(:,2);
    time_cont=0;
    for z=1:length(time)
        if time(z)<0
            time_cont=time_cont+1;
        elseif time(z)==0
            time_cont=time_cont+1;
            break
        end
    end
    clear time
    time=Input(time_cont:end,2);
    force=Input(time_cont:end,3);
    displacement=Input(time_cont:end,6);

    %% FILTRAGGIO DELLA CURVA
```

```

    % rimozione delle frequenze che disturbano il segnale
    force_frequency_filtered=bandstop(force,[1*10^3
40*10^3],1*10^6);
    % arrotondamento dei picchi in modo che siano meno
    spigolosi

force_filtered=smoothdata(force_frequency_filtered,'sgola
y',700);
    %faccio partire la curva filtrata da zero(aspetto
grafico)
    for i=1:100
        force_filtered(i)=0;
        displacement(i)=0;
        force(i)=0;
    end
    figure
    subplot(3,1,1);
    plot(time,force,'r','linewidth',1.8)
    xlabel('time(ms) ');
    ylabel('Force(N) ');
    subplot(3,1,2);
    plot(time,force_filtered,'k','linewidth',1.8)
    xlabel('time(ms) ');
    ylabel('Force filtered(N) ');
    subplot(3,1,3);

plot(time,force,'r',time,force_filtered,'k','linewidth',1
.8)
    xlabel('time(ms) ');
    ylabel('Force(N) ');
    figure
    subplot(3,1,1);
    plot(displacement,force,'r','linewidth',1.8)
    xlabel('displacement(mm) ');
    ylabel('Force(N) ');
    subplot(3,1,2);
    plot(displacement,force_filtered,'k','linewidth',1.8)
    xlabel('displacement(mm) ');
    ylabel('Force filtered(N) ');
    subplot(3,1,3);

plot(displacement,force,'r',displacement,force_filtered,'
k','linewidth',1.8)
    xlabel('displacement(mm) ');
    ylabel('Force(N) ');

```



```
%% CALCOLO FORZA MEDIA: ALGORITMO CON SPOSTAMENTI  
IMPOSTI IN PERCENTUALE
```

```
    %individuo la posizione corrispondente allo  
spostamento pari alla  
    %profondità del trigger  
start_displacement_1=trigger;  
position_start=0;  
for l=1:length(displacement)  
    if displacement(l)<=start_displacement_1  
        position_start=position_start+1;  
    elseif displacement(l)>=start_displacement_1  
        break  
    end  
end  
    %trovo il valore massimo del vettore degli  
spostamenti che in questo caso  
    %corrisponde al valore di spostamento tale per cui la  
forza assume  
    %valori negativi, e relativa posizione nel vettore  
'displacement'  
cont_force=0;  
for i=start_displacement_1:length(force)  
    if force_filtered(i)>0  
        cont_force=cont_force+1;  
    elseif force_filtered(i)<0  
        cont_force=cont_force+1;  
        break  
    end  
end  
displacement_max=displacement(cont_force);  
position_final=cont_force;
```

```

%determino il limite di forza e la relativa posizione
all'interno del
    %vettore 'Force'; prendo lo spostamento associato a
tale posizione nel
    %vettore 'displacement'
    half_displacement_max=(displacement_max-
trigger+0.4*trigger)/2;
    position_half_disp_max=0;
    for h=1:length(displacement)
        if displacement(h)<=half_displacement_max

position_half_disp_max=position_half_disp_max+1;
        elseif displacement(h)>=half_displacement_max
            break
        end
    end

force_limit=0.5*force_filtered(position_half_disp_max);
    for r=position_start:position_final
        if force_filtered(r)>=force_limit
            r=r+1;
        elseif force_filtered(i)<=force_limit
            r=r-1;
            break
        end
    end
    position_force_limit=r;
    displacement_max_1=
displacement(position_force_limit);
    % definisco lo spostamento iniziale e quello finale
come % rispetto allo
    % spostamento massimo calcolato precedentemente
    start_displacement_perc=0.25;
    final_displacement_perc=0.95;

start_displacement=start_displacement_perc*displacement_m
ax_1;

final_displacement=final_displacement_perc*displacement_m
ax_1;
    tabella_SEA.(v(3))(total)=start_displacement;
    tabella_SEA.(v(4))(total)=final_displacement;

```

```

%estraggo il numero di riga corrispondente allo
spostamento iniziale imposto

[start_position]=trova_posizione_start(displacement,start
_displacement);
    % estraggo il numero di riga corrispondente allo
spostamento finale imposto

[final_position]=trova_posizione_final(displacement,final
_displacement);
    % definisco un fattore sottrattivo che mi serve per
riempire in maniera sequenziale il vettore force_final
    subtractive_factor= start_position-1;
    % inserisco all'interno del vettore force_final tutti
i valori di
    % forza nell'intervallo di interesse
    for k=start_position:final_position
        force_final(k-subtractive_factor)=force(k);
    end
    force_average=mean(force_final);
    %% CALCOLO SEA A SPOSTAMENTO IMPOSTO

    delta=(final_displacement-start_displacement)*10^-3;
    %m
    mc=rho*A*delta;    %kg

SEA_spostamenti_imposti_in_percentuale=((force_average*de
lta)/mc)/10^3; % KJ/Kg

tabella_SEA.(v(2))(total)=SEA_spostamenti_imposti_in_perc
entuale;
    clearvars -except rho A start_displacement_perc
final_displacement_perc total v curve curva_analizzata
lista_curve_da_analizzare tabella_SEA trigger

    total=total+1;
end
%%
clearvars -except tabella_SEA

```

Algoritmo che considera l'area sottesa a tutta la curva

```
%% IMPORT DELLE CURVE

lista_curve_da_analizzare= dir('*.csv');
[curve curva_analizzata
tabella_SEA]=import_curve_tabella_singolo_algoritmo_spostamenti_variabili(lista_curve_da_analizzare);
total=1;
for w=1:width(tabella_SEA)
    v(w)=w;
end

%% DATI IN INPUT

rho=input('rho='); % kg/m^3 densità del provino
A=input('A='); %m^2 Sezione resistente del provino
s=input('s='); %mm spessore medio del provino
Larghezza=input('larghezza='); %mm larghezza media del provino
Lunghezza=input('lunghezza='); %mm lunghezza media del provino
trigger=input('trigger='); %mm profondità del trigger
%%
while total<=numel(lista_curve_da_analizzare)
    input=importdata(curve(total));
    Input=input.data;

    %% LETTURA DEI VETTORI TEMPO, FORZA E SPOSTAMENTO

    time=Input(:,2);
    time_cont=0;
    for z=1:length(time)
        if time(z)<0
            time_cont=time_cont+1;
        elseif time(z)==0
            time_cont=time_cont+1;
            break
        end
    end
    end
    clear time
    time=Input(time_cont:end,2);
    force=Input(time_cont:end,3);
    displacement=Input(time_cont:end,6);
    for p=1:numel(lista_curve_da_analizzare)
        displacement_matrix(:,p)=displacement;
    end
end
```

```

%% FILTRAGGIO DELLA CURVA

% rimozione delle frequenze che disturbano il segnale
force_frequency_filtered=bandstop(force,[1*10^3
40*10^3],1*10^6);
% arrotondamento dei picchi in modo che siano meno
spigolosi

force_filtered=smoothdata(force_frequency_filtered,'sgola
y',700);
%faccio partire la curva filtrata da zero(aspetto
grafico)
for i=1:100
    force_filtered(i)=0;
    displacement(i)=0;
    force(i)=0;
end
figure
subplot(3,1,1);
plot(time,force,'r','linewidth',1.8)
xlabel('time(ms) ');
ylabel('Force(N) ');
subplot(3,1,2);
plot(time,force_filtered,'k','linewidth',1.8)
xlabel('time(ms) ');
ylabel('Force filtered(N) ');
subplot(3,1,3);

plot(time,force,'r',time,force_filtered,'k','linewidth',1
.8)
xlabel('time(ms) ');
ylabel('Force(N) ');
figure
subplot(3,1,1);
plot(displacement,force,'r','linewidth',1.8)
xlabel('displacement(mm) ');
ylabel('Force(N) ');
subplot(3,1,2);
plot(displacement,force_filtered,'k','linewidth',1.8)
xlabel('displacement(mm) ');
ylabel('Force filtered(N) ');
subplot(3,1,3);

plot(displacement,force,'r',displacement,force_filtered,'
k','linewidth',1.8)
xlabel('displacement(mm) ');
ylabel('Force(N) ');

```

```

%% CALCOLO FORZA MEDIA: ALGORITMO CHE CONSIDERA
L'AREA SOTTESA A TUTTA LA CURVA

    %prendo nuovamente in input il vettore 'displacement'
in quanto quello
    %imprato originariamente è stato modificato per
scopi grafici (guardare
    %parte di filtraggio)
    displacement=displacement_matrix(:,total);
    %setto a zero mm lo spostamento iniziale e lo
inserisco in tabella
    start_displacement=0;
    tabella_SEA.(v(3))(total)=start_displacement;
    %determino la posizione relativa allo spostamento
iniziale nel vettore
    %'displacement'
    position_start=0;
    for l=1:length(displacement)
        if displacement(l)<=start_displacement+trigger
            position_start=position_start+1;
        elseif
displacement(l)>=start_displacement+trigger
            break
        end
    end
    %determino lo spostamento massimo del vettore,
definito come il valore
    %di spostamento oltre il quale i valori di forza
divengono negativi;
    %determino la sua posizione all'interno del vettore
'displacement'
    cont_force=0;
    for i=position_start:length(force)
        if force_filtered(i)>0
            cont_force=cont_force+1;
        elseif force_filtered(i)<0
            cont_force=cont_force+1;
            break
        end
    end
    displacement_max=displacement(cont_force);
    position_final=cont_force;

```

```

%calcolo il limite di forza e la relativa posizione nel
vettore 'Force'
    half_displacement_max=(displacement_max-
trigger+0.4*trigger)/2;
    position_half_disp_max=0;
    for h=1:length(displacement)
        if displacement(h)<=half_displacement_max

position_half_disp_max=position_half_disp_max+1;
        elseif displacement(h)>=half_displacement_max
            break
        end
    end

force_limit=0.3*force_filtered(position_half_disp_max);
    for r=position_start:position_final
        if force_filtered(r)>=force_limit
            r=r+1;
        elseif force_filtered(i)<=force_limit
            r=r-1;
            break
        end
    end
    position_force_limit=r;
    %assumo come spostamento finale lo spostamento
collocato alla posizione
    %del limite di forza, ma ovviamente nel vettore
'displacement'; lo
    %inserisco in tabella

final_displacement=displacement(position_force_limit);
    tabella_SEA.(v(4))(total)=final_displacement;
    %estraggo il numero di riga corrispondente allo
spostamento iniziale imposto

[start_position]=trova_posizione_start(displacement,start
_displacement);
    % definisco un fattore sottrattivo che mi serve per
riempire in maniera sequenziale il vettore force_final
    subtractive_factor= start_position-1;
    % inserisco all'interno del vettore force_final tutti
i valori di
    % forza nell'intervallo di interesse
    for k=start_position:position_force_limit
        force_final(k-subtractive_factor)=force(k);
    end
    force_average=mean(force_final);

```

```

%% CALCOLO SEA CURVA COMPLETA

delta=(final_displacement-start_displacement);
Vtot=(Lunghezza*Larghezza*s)-((10*10*5)/2); %volume
totale del provino
Vcrash=[Vtot-[Larghezza*(Lunghezza-delta)]*s]*10^-9;
%volume di provino crashato
SEA_curva_completa=[(force_average*delta*10^-
3)/(rho*Vcrash)]*10^-3;
tabella_SEA.(v(2))(total)=SEA_curva_completa;
clearvars -except rho A total v curve
curva_analizzata lista_curve_da_analizzare tabella_SEA s
Larghezza Lunghezza trigger
total=total+1;
end
%%
clearvars -except tabella_SEA

```


Algoritmo con spostamenti variabili: versione finale

```
%% IMPORT DELLE CURVE

lista_curve_da_analizzare= dir('*.csv');
[curve curva_analizzata
tabella_SEA]=import_curve_tabella_singolo_algoritmo_spostamenti_variabili(lista_curve_da_analizzare);
total=1;
for w=1:width(tabella_SEA)
    v(w)=w;
end

%% DATI IN INPUT

rho=input('rho='); % kg/m^3 densità del provino
A=input('A='); %m^2 Sezione resistente del provino
trigger=input('trigger='); %mm dimensione del trigger del provino
%%
while total<=numel(lista_curve_da_analizzare)
    input=importdata(curve(total));
    Input=input.data;

    %% LETTURA DEI VETTORI TEMPO, FORZA E SPOSTAMENTO

    % Gli export del software Bluehill Impact presentano valori temporali minori di
    % zero che indicano istanti prima dell'impatto.
    Questi dati non sono di
    % interesse per l'analisi della curva, di conseguenza la seguente parte di
    % script taglia tutti gli istanti temporali prima dell'impatto ed importa
    % solamente i dati a partire dall'istante zero.
    time=Input(:,2);
    time_cont=0;
    for z=1:length(time)
        if time(z)<0
            time_cont=time_cont+1;
        elseif time(z)==0
            time_cont=time_cont+1;
            break
        end
    end
    clear time
```

```

time=Input(time_cont:end,2);
force=Input(time_cont:end,3);
displacement=Input(time_cont:end,6);
%metto in tabella il valore massimo di forza grezza
tabella_SEA.(v(5))(total)=max(force);

%% FILTRAGGIO DELLA CURVA

% rimozione delle frequenze che disturbano il segnale
force_frequency_filtered=bandstop(force,[1*10^3
40*10^3],1*10^6);
% arrotondamento dei picchi in modo che siano meno
spigolosi

force_filtered=smoothdata(force_frequency_filtered,'sgola
y',700);
%metto in tabella il valore massimo di forza filtrata
tabella_SEA.(v(6))(total)=max(force_filtered);
%faccio partire la curva filtrata da zero(aspetto
grafico)
for i=1:80
    force_filtered(i)=0;
    displacement(i)=0;
end
displacement_1=Input(time_cont:end,6);
figure
subplot(3,1,1);
plot(time,force,'r','linewidth',1.8)
xlabel('Time(ms)');
ylabel('Force(N)');
grid on
subplot(3,1,2)
plot(time,force_filtered,'k','linewidth',1.8)
xlabel('Time(ms)');
ylabel('Force filtered(N)');
grid on

```

```

subplot(3,1,3);

plot(time,force,'r',time,force_filtered,'k','linewidth',1
.8)
    xlabel('Time(ms) ');
    ylabel('Force(N) ');
    grid on
figure
subplot(3,1,1);
plot(displacement_1,force,'r','linewidth',1.8)
    xlabel('displacement(mm) ');
    ylabel('Force(N) ');
    grid on
subplot(3,1,2);
plot(displacement,force_filtered,'k','linewidth',1.8)
    xlabel('displacement(mm) ');
    ylabel('Force filtered(N) ');
    grid on
subplot(3,1,3);

plot(displacement_1,force,'r',displacement,force_filtered
,'k','linewidth',1.8)
    xlabel('displacement(mm) ');
    ylabel('Force(N) ');
    grid on

%% CALCOLO FORZA MEDIA: ALGORITMO CON SPOSTAMENTI
VARIABILI

%come spostamento iniziale impongo l'altezza del
trigger
start_displacement=trigger;
%definisco i margini da aggiungere e togliere
rispettivamente allo spostamento iniziale e finale quando
si verifica la condizione di forza.
increase_start=1.5;
decrease_final=1.5;
%trovo la posizione associata a tale spostamento

[start_position]=trova_posizione_start(displacement,start
_displacement);

```

```

    %trovo il valore massimo del vettore degli
    spostamenti che in questo caso
    %corrisponde al valore di spostamento tale per cui si
    annulla la forza
    for i=start_position:length(force_filtered)
        if force_filtered(i)>0
            i=i+1;
        elseif force_filtered(i)<0
            cont_force=i;
            break
        end
    end
    displacement_max=displacement(cont_force);
    final_position=cont_force;

    % Una volta trovato lo spostamento massimo prendo uno
    spostamento sul
    % tratto utile della curva e ricavo in funzione del
    valore di forza
    % corrispondente a tale spostamento il limite di
    forza da utilizzare
    % nelle iterazioni.
    half_displacement_max=(displacement_max-
    trigger+0.4*trigger)/2;
    position_half_disp_max=0;
    for h=1:length(displacement)
        if displacement(h)<=half_displacement_max

position_half_disp_max=position_half_disp_max+1;
        elseif displacement(h)>=half_displacement_max
            break
        end
    end

    force_limit=0.5*force_filtered(position_half_disp_max);

```

```

    % A questo punto ricavo la posizione nel vettore
    force_filtered del limite
    % di forza. Se si ragionasse sulla forza grezza ci
    potrebbero essere dei
    % problemi nel caso in cui si verificano
    oscillazioni della forza del
    % 40%. Per questo motivo si ragiona sulla forza
    filtrata. Se ci fossero
    % oscillazioni di più del 40% sulla forza filtrata
    vorrebbe dire che sulla
    % forza grezza le oscillazioni sarebbero più del 100%
    e in questo caso il
    % test non risulta valido.
    for r=start_position:final_position
        if force_filtered(r)>=force_limit
            r=r+1;
        elseif force_filtered(i)<=force_limit
            r=r-1;
            break
        end
    end
    position_force_limit=r;
    % Ora verifico che dalla posizione del limite di
    forza sino alla
    % final_position non ci siano valori superiori al
    limite di forza. In caso
    % positivo(flag=0) l'algoritmo procede con le
    iterazioni sullo spostamento iniziale
    % ed iniziale normalmente, mentre in caso
    negativo(flag=1) prendo come
    % final_position la posizione successiva a quella del
    limite di forza e
    % parto ad iterare da lì in modo da evitare il tratto
    di curva non
    % desiderato e ricadere così nel tratto utile.
    flag=0;
    for a=(position_force_limit+1):final_position
        if force_filtered(a)>=force_limit
            flag=1;
            a=a-1;
            break
        elseif force_filtered(a)<=force_limit
            a=a+1;
        end
    end
end

```

```

    if flag==0
        %Eseguo delle iterazioni in maniera separata
        sullo spostamento finale e
        %sullo spostamento iniziale facendo un controllo
        puntuale sulla forza. Se
        %la forza calcolata ad una certa iterazione
        corrispondente ad un certo
        %spostamento è inferiore al limite allora
        l'algoritmo continua ad iterare
        % incrementando lo spostamento iniziale e
        riducendo quello finale fino
        %a convergenza.

        %Iterazione sullo spostamento iniziale
        for i=start_position:final_position
            if force_filtered(i)>=force_limit

start_displacement=displacement(i)+increase_start;
                break
            elseif force_filtered(i)<=force_limit
                i=i+1;
            end
        end
        %salvo lo spostamento iniziale calcolato ad ogni
        iterazione
        tabella_SEA.(v(3))(total)=start_displacement;

        %trovo la posizione relativa allo spostamento
        iniziale calcolato
        position_start=0;
        for l=1:length(displacement)
            if displacement(l)<start_displacement
                position_start=position_start+1;
            elseif displacement(l)>=start_displacement
                position_start=position_start+1;
                break
            end
        end
    end

```

```

%Iterazione sullo spostamento finale
    for j=final_position:-1:start_position
        if force_filtered(j)>=force_limit
            final_displacement=displacement(j)-
decrease_final;
            break
        elseif force_filtered(i)<=force_limit
            j=j-1;
        end
    end
    % eseguo un controllo ulteriore in quanto alcune
curve presentano un picco
    % alla fine, e quindi taglio a 45 mm in quanto
oltre questo spostamento vado
    % a toccare i deceleratori e questo provoca un
picco di forza.
    if final_displacement>45
        final_displacement=45;
    end
    %salvo lo spostamento finale calcolato ad ogni
iterazione
    tabella_SEA.(v(4)) (total)=final_displacement;
    %trovo la posizione relativa allo spostamento
finale dopo il controllo
    position_final=0;
    for h=1:length(displacement)
        if displacement(h)<final_displacement
            position_final=position_final+1;
        elseif displacement(h)>=final_displacement
            position_final=position_final+1;
            break
        end
    end
    subtractive_factor=position_start-1;
    for k=position_start:position_final
        force_final(k-subtractive_factor)=force(k);
    end
    force_average=mean(force_final);

```

```

        % calcolo lo stress medio in MPa dividendo la
forza media per la
        % sezione, il rapporto tra la forza media e
quella di picco sia
        % filtrata sia non filtrata e metto tutto in
tabella
        stress_average=force_average/(A*10^6);
        crash_force_efficiency=force_average/max(force);

crash_force_efficiency_filtered=force_average/max(force_f
iltered);
        tabella_SEA.(v(7))(total)=force_average;
        tabella_SEA.(v(8))(total)=stress_average;
        tabella_SEA.(v(9))(total)=crash_force_efficiency;

tabella_SEA.(v(10))(total)=crash_force_efficiency_filtere
d;

elseif flag==1
        clearvars displacement
        displacement=Input(time_cont:end,6);
        for i=1:100
                displacement(i)=0;
        end
        final_position=position_force_limit+1;

        for i=start_position:final_position
                if force_filtered(i)>=force_limit

start_displacement=displacement(i)+increase_start;
                        break
                elseif force_filtered(i)<=force_limit
                        i=i+1;
                end
        end
        %salvo lo spostamento iniziale calcolato ad ogni
iterazione
        tabella_SEA.(v(3))(total)=start_displacement;

```



```

        %trovo la posizione relativa allo spostamento
iniziale calcolato
        position_start=0;
        for l=1:length(displacement)
            if displacement(l)<start_displacement
                position_start=position_start+1;
            elseif displacement(l)>=start_displacement
                position_start=position_start+1;
                break
            end
        end
        %Iterazione sullo spostamento finale
        for j=final_position:-1:start_position
            if force_filtered(j)>=force_limit
                final_displacement=displacement(j)-
decrease_final;
                break
            elseif force_filtered(i)<=force_limit
                j=j-1;
            end
        end
        % eseguo un controllo ulteriore in quanto alcune
curve presentano un picco
        % alla fine, e quindi taglio a 45 mm in quanto
oltre questo spostamento vado
        % a toccare i deceleratori e questo provoca un
picco di forza.
        if final_displacement>45
            final_displacement=45;
        end
        %salvo lo spostamento finale calcolato ad ogni
iterazione
        tabella_SEA.(v(4)) (total)=final_displacement;
        %trovo la posizione relativa allo spostamento
finale dopo il controllo
        position_final=0;
        for h=1:length(displacement)
            if displacement(h)<final_displacement
                position_final=position_final+1;
            elseif displacement(h)>=final_displacement
                position_final=position_final+1;
                break
            end
        end
    end
end

```

```

        subtractive_factor=position_start-1;
        for k=position_start:position_final
            force_final(k-subtractive_factor)=force(k);
        end
        force_average=mean(force_final);
        % calcolo lo stress medio in MPa dividendo la
forza media per la
        % sezione, il rapporto tra la forza media e
quella di picco sia
        % filtrata sia non filtrata e metto tutto in
tabella
        stress_average=force_average/(A*10^6);
        crash_force_efficiency=force_average/max(force);

crash_force_efficiency_filtered=force_average/max(force_f
iltered);
        tabella_SEA.(v(7))(total)=force_average;
        tabella_SEA.(v(8))(total)=stress_average;
        tabella_SEA.(v(9))(total)=crash_force_efficiency;

tabella_SEA.(v(10))(total)=crash_force_efficiency_filtere
d;
    end
    %% CALCOLO SEA A SPOSTAMENTI VARIABILI

    delta=(final_displacement-start_displacement)*10^-3;
    %m
    mc=rho*A*delta;    %kg

    SEA_spostamenti_variabili_iterazioni_separate=((force_ave
rage*delta)/mc)/10^3; % KJ/Kg

    tabella_SEA.(v(2))(total)=SEA_spostamenti_variabili_itera
zioni_separate;
        clearvars -except lista_curve_da_analizzare curve
curve_analizzate rho A increase_start decrease_final
total v tabella_SEA trigger
        %%
        total=total+1;
    end
    clearvars -except tabella_SEA tabella_spostamenti

```

Algoritmo con spostamenti variabili: input dei dati ad ogni iterazione

```
%% IMPORT DELLE CURVE

lista_curve_da_analizzare= dir('*.csv');
[curve curva_analizzata
tabella_SEA]=import_curve_tabella_singolo_algoritmo_spostamenti_variabili(lista_curve_da_analizzare);
total=1;
for w=1:width(tabella_SEA)
    v(w)=w;
end
%%
while total<=numel(lista_curve_da_analizzare)
    input_1=importdata(curve(total));
    Input=input_1.data;

    %% DATI IN INPUT

    rho=input('rho='); % kg/m^3 densità del provino
    A=input('A='); %m^2 Sezione resistente del
provino
    trigger=input('trigger='); %mm dimensione del trigger
del provino

    %% LETTURA DEI VETTORI TEMPO, FORZA E SPOSTAMENTI
    % Gli export del software Bluehill Impact presentano
valori temporali minori di
    % zero che indicano istanti prima dell'impatto.
Questi dati non sono di
    % interesse per l'analisi della curva, di conseguenza
la seguente parte di
    % script taglia tutti gli istanti temporali prima
dell'impatto ed importa
    % solamente i dati a partire dall'istante zero.
    time=Input(:,2);
    time_cont=0;
    for z=1:length(time)
        if time(z)<0
            time_cont=time_cont+1;
        elseif time(z)==0
            time_cont=time_cont+1;
            break
        end
    end
end
```

```

clear time
time=Input(time_cont:end,2);
force=Input(time_cont:end,3);
displacement=Input(time_cont:end,6);
%metto in tabella il valore massimo di forza grezza
tabella_SEA.(v(5))(total)=max(force);

%% FILTRAGGIO DELLA CURVA

    % rimozione delle frequenze che disturbano il
segnale
    force_frequency_filtered=bandstop(force,[1*10^3
40*10^3],1*10^6);
    % arrotondamento dei picchi in modo che siano meno
spigolosi

force_filtered=smoothdata(force_frequency_filtered,'sgola
y',700);
    %metto in tabella il valore massimo di forza filtrata
tabella_SEA.(v(6))(total)=max(force_filtered);
    %faccio partire la curva filtrata da zero(aspetto
grafico)
    for i=1:80
        force_filtered(i)=0;
        displacement(i)=0;
    end
    displacement_1=Input(time_cont:end,6);
    figure
    subplot(3,1,1);
    plot(time,force,'r','linewidth',1.8)
    xlabel('Time(ms)');
    ylabel('Force(N)');
    grid on
    subplot(3,1,2)
    plot(time,force_filtered,'k','linewidth',1.8)
    xlabel('Time(ms)');
    ylabel('Force filtered(N)');
    grid on
    subplot(3,1,3);

plot(time,force,'r',time,force_filtered,'k','linewidth',1
.8)
    xlabel('Time(ms)');
    ylabel('Force(N)');
    grid on
    figure

```

```

subplot(3,1,1);
plot(displacement_1,force,'r','linewidth',1.8)
xlabel('displacement(mm) ');
ylabel('Force(N) ');
grid on
subplot(3,1,2);
plot(displacement,force_filtered,'k','linewidth',1.8)
xlabel('displacement(mm) ');
ylabel('Force filtered(N) ');
grid on
subplot(3,1,3);

plot(displacement_1,force,'r',displacement,force_filtered
,'k','linewidth',1.8)
xlabel('displacement(mm) ');
ylabel('Force(N) ');
grid on
%% CALCOLO FORZA MEDIA: ALGORITMO CON SPOSTAMENTI
VARIABILI (INPUT DEI DATI AD OGNI ITERAZIONE)

%come spostamento iniziale impongo l'altezza del
trigger
start_displacement=trigger;
%definisco i margini da aggiungere e togliere
rispettivamente allo spostamento iniziale e finale quando
si verifica la condizione di forza.
increase_start=1.5;
decrease_final=1.5;
%trovo la posizione associata a tale spostamento

[start_position]=trova_posizione_start(displacement,start
_displacement);
%trovo il valore massimo del vettore degli
spostamenti che in questo caso
%corrisponde al valore di spostamento tale per cui si
annulla la forza
for i=start_position:length(force_filtered)
    if force_filtered(i)>0
        i=i+1;
    elseif force_filtered(i)<0
        cont_force=i;
        break
    end
end
displacement_max=displacement(cont_force);
final_position=cont_force;

```

```

    % Una volta trovato lo spostamento massimo prendo uno
    spostamento sul
    % tratto utile della curva e ricavo in funzione del
    valore di forza
    % corrispondente a tale spostamento il limite di
    forza da utilizzare
    % nelle iterazioni.
    half_displacement_max=(displacement_max-
    trigger+0.4*trigger)/2;
    position_half_disp_max=0;
    for h=1:length(displacement)
        if displacement(h)<=half_displacement_max

position_half_disp_max=position_half_disp_max+1;
        elseif displacement(h)>=half_displacement_max
            break
        end
    end

    end

    force_limit=0.5*force_filtered(position_half_disp_max);
    % A questo punto ricavo la posizione nel vettore
    force_filtered del limite
    % di forza. Se si ragionasse sulla forza grezza ci
    potrebbero essere dei
    % problemi nel caso in cui si verificano
    oscillazioni della forza del
    % 40%. Per questo motivo si ragiona sulla forza
    filtrata. Se ci fossero
    % oscillazioni di più del 40% sulla forza filtrata
    vorrebbe dire che sulla
    % forza grezza le oscillazioni sarebbero più del 100%
    e in questo caso il
    % test non risulta valido.
    for r=start_position:final_position
        if force_filtered(r)>=force_limit
            r=r+1;
        elseif force_filtered(i)<=force_limit
            r=r-1;
            break
        end
    end

    end

    position_force_limit=r;
    % Ora verifico che dalla posizione del limite di
    forza sino alla
    % final_position non ci siano valori superiori al
    limite di forza.

```

```

In caso
    % positivo(flag=0) l'algoritmo procede con le
iterazioni sullo spostamento iniziale
    % ed iniziale normalmente, mentre in caso
negativo(flag=1) prendo come
    % final_position la posizione successiva a quella del
limite di forza e
    % parto ad iterare da lì in modo da evitare il tratto
di curva non
    % desiderato e ricadere così nel tratto utile.
flag=0;
for a=(position_force_limit+1):final_position
    if force_filtered(a)>=force_limit
        flag=1;
        a=a-1;
        break
    elseif force_filtered(a)<=force_limit
        a=a+1;
    end
end
if flag==0
    %Eseguo delle iterazioni in maniera separata
sullo spostamento finale e
    %sullo spostamento iniziale facendo un controllo
puntuale sulla forza. Se
    %la forza calcolata ad una certa iterazione
corrispondente ad un certo
    %spostamento è inferiore al limite allora
l'algoritmo continua ad iterare
    % incrementando lo spostamento iniziale e
riducendo quello finale fino
    %a convergenza.

    %Iterazione sullo spostamento iniziale
for i=start_position:final_position
    if force_filtered(i)>=force_limit

start_displacement=displacement(i)+increase_start;
        break
    elseif force_filtered(i)<=force_limit
        i=i+1;
    end
end
end

```

```

        %salvo lo spostamento iniziale calcolato ad ogni
iterazione
        tabella_SEA.(v(3))(total)=start_displacement;

        %trovo la posizione relativa allo spostamento
iniziale calcolato
        position_start=0;
        for l=1:length(displacement)
            if displacement(l)<=start_displacement
                position_start=position_start+1;
            elseif displacement(l)>=start_displacement
                break
            end
        end
        %Iterazione sullo spostamento finale
        for j=final_position:-1:start_position
            if force_filtered(j)>=force_limit
                final_displacement=displacement(j)-
decrease_final;
                break
            elseif force_filtered(i)<=force_limit
                j=j-1;
            end
        end
        % eseguo un controllo ulteriore in quanto alcune
curve presentano un picco
        % alla fine, e quindi taglio a 45 mm in quanto
oltre questo spostamento vado
        % a toccare i deceleratori e questo provoca un
picco di forza.
        if final_displacement>45
            final_displacement=45;
        end
        %salvo lo spostamento finale calcolato ad ogni
iterazione
        tabella_SEA.(v(4))(total)=final_displacement;
        %trovo la posizione relativa allo spostamento
finale dopo il controllo
        position_final=0;
        for h=1:length(displacement)
            if displacement(h)<=final_displacement
                position_final=position_final+1;
            elseif displacement(h)>=final_displacement
                break
            end
        end
    end
end

```



```

        subtractive_factor=position_start-1;
        for k=position_start:position_final
            force_final(k-subtractive_factor)=force(k);
        end
        force_average=mean(force_final);
        % calcolo lo stress medio in MPa dividendo la
        forza media per la
        % sezione, il rapporto tra la forza media e
        quella di picco sia
        % filtrata sia non filtrata e metto tutto in
        tabella
        stress_average=force_average/(A*10^6);
        crash_force_efficiency=force_average/max(force);

        crash_force_efficiency_filtered=force_average/max(force_f
        iltered);
        tabella_SEA.(v(7))(total)=force_average;
        tabella_SEA.(v(8))(total)=stress_average;
        tabella_SEA.(v(9))(total)=crash_force_efficiency;

        tabella_SEA.(v(10))(total)=crash_force_efficiency_filtere
        d;

    elseif flag==1
        clearvars displacement
        displacement=Input(time_cont:end,6);
        for i=1:100
            displacement(i)=0;
        end
        final_position=position_force_limit+1;

        for i=start_position:final_position
            if force_filtered(i)>=force_limit

start_displacement=displacement(i)+increase_start;
                break
            elseif force_filtered(i)<=force_limit
                i=i+1;
            end
        end
        %salvo lo spostamento iniziale calcolato ad ogni
        iterazione
        tabella_SEA.(v(3))(total)=start_displacement;

```

```

        %trovo la posizione relativa allo spostamento
        iniziale calcolato
        position_start=0;
        for l=1:length(displacement)
            if displacement(l)<=start_displacement
                position_start=position_start+1;
            elseif displacement(l)>=start_displacement
                break
            end
        end
        %Iterazione sullo spostamento finale
        for j=final_position:-1:start_position
            if force_filtered(j)>=force_limit
                final_displacement=displacement(j)-
decrease_final;
                break
            elseif force_filtered(i)<=force_limit
                j=j-1;
            end
        end
        % eseguo un controllo ulteriore in quanto alcune
        curve presentano un picco
        % alla fine, e quindi taglio a 45 mm in quanto
        oltre questo spostamento vado
        % a toccare i deceleratori e questo provoca un
        picco di forza.
        if final_displacement>45
            final_displacement=45;
        end
        %salvo lo spostamento finale calcolato ad ogni
        iterazione
        tabella_SEA.(v(4)) (total)=final_displacement;
        %trovo la posizione relativa allo spostamento
        finale dopo il controllo
        position_final=0;
        for h=1:length(displacement)
            if displacement(h)<=final_displacement
                position_final=position_final+1;
            elseif displacement(h)>=final_displacement
                break
            end
        end
        subtractive_factor=position_start-1;
        for k=position_start:position_final
            force_final(k-subtractive_factor)=force(k);
        end
        force_average=mean(force_final);

```

```

        % calcolo lo stress medio in MPa dividendo la
forza media per la
        % sezione, il rapporto tra la forza media e
quella di picco sia
        % filtrata sia non filtrata e metto tutto in
tabella
        stress_average=force_average/(A*10^6);
        crash_force_efficiency=force_average/max(force);

crash_force_efficiency_filtered=force_average/max(force_f
iltered);
        tabella_SEA.(v(7))(total)=force_average;
        tabella_SEA.(v(8))(total)=stress_average;
        tabella_SEA.(v(9))(total)=crash_force_efficiency;

tabella_SEA.(v(10))(total)=crash_force_efficiency_filtere
d;
    end
    %% CALCOLO SEA A SPOSTAMENTI VARIABILI

    delta=(final_displacement-start_displacement)*10^-3;
    %m
    mc=rho*A*delta;    %kg

SEA_spostamenti_variabili_iterazioni_separate=((force_ave
rage*delta)/mc)/10^3; % KJ/Kg

tabella_SEA.(v(2))(total)=SEA_spostamenti_variabili_itera
zioni_separate;
    clearvars -except lista_curve_da_analizzare curve
curve_analizzate increase_start decrease_final total v
tabella_SEA
    %%
    total=total+1;
end
clearvars -except tabella_SEA tabella_spostamenti

```

8. Bibliografia

- [1] D. H-JA Lukaszewicz, “5 Automotive Composite Structures for Crashworthiness,” *Advanced Composite Materials for Automotive Applications*, pp. 99–127, 2013.
- [2] G. L. Farley and R. M. Jones, “Energy-Absorption Capability of Composite Tubes and Beams.” [Online]. Available: <https://ntrs.nasa.gov/search.jsp?R=19890020120>
- [3] P. Feraboli, B. Wade, F. Deleo, and M. Rassaian, “Crush energy absorption of composite channel section specimens,” *Composites Part A: Applied Science and Manufacturing*, vol. 40, no. 8, pp. 1248–1256, Aug. 2009, doi: 10.1016/j.compositesa.2009.05.021.
- [4] C. Reuter, K. H. Sauerland, and T. Tröster, “Experimental and numerical crushing analysis of circular CFRP tubes under axial impact loading,” *Composite Structures*, vol. 174, pp. 33–44, Aug. 2017, doi: 10.1016/j.compstruct.2017.04.052.
- [5] P. Feraboli F. Deleo F. Gattoni, “Efforts in the standardization of composite materials crashworthiness,” *American Society for Composites*, 2007.
- [6] S. J. Pety *et al.*, “Effect of microchannels on the crashworthiness of fiber-reinforced composites,” *Composite Structures*, vol. 184, pp. 428–436, Jan. 2018, doi: 10.1016/j.compstruct.2017.09.105.
- [7] G. C. Jacob, J. M. Starbuck, S. Simunovic, and J. F. Fellers, “Energy Absorption in Glass-Reinforced Continuous Strand Mat Composites,” 2003.
- [8] R. A. Cutting, F. Rios-Tascon, and J. E. Goodsell, “Experimental investigation of the crush performance of prepreg platelet molding compound tubes,” *Journal of Composite Materials*, vol. 54, no. 27, pp. 4311–4324, Nov. 2020, doi: 10.1177/0021998320929418.
- [9] F. Mischo C. Goergen S. Schmeer P. Mitschang, “Use of recycled carbon staple fibers in an advanced thermoforming process and analysis of its crash performance,” *Advanced Manufacturing: Polymer & Composites Science*, pp. 48–56, Mar. 2020.
- [10] R. Destefanis, “POLITECNICO DI TORINO Corso di Laurea Magistrale in Ingegneria Meccanica,” 2021. Accessed: Jan. 21, 2022. [Online]. Available: <https://webthesis.biblio.polito.it/20208/1/tesi.pdf>

Ringraziamenti

Il mio percorso Universitario è iniziato con un ragazzo in possesso di ottime capacità, ma con scarsa fiducia in sé stesso. Al primo impatto con il Politecnico di Torino pensavo di non farcela, che per uno come me fosse uno scalino troppo alto da superare. Invece, come mi fossi teletrasportato nel tempo, eccomi qui che presento questo importante lavoro di tesi, nato dalla stretta collaborazione tra il Politecnico di Torino ed Instron. Tutto questo è stato reso possibile in primis dal Prof. Davide Paolino, che mi ha subito rassicurato e fatto sentire a casa pur conoscendo l'ardua impresa di trattare con i compositi.

Un ringraziamento va anche al Prof Giovanni Belingardi, al tutor aziendale Giuseppe Galizia e ad Andrea Calzolari, i quali mi hanno subito fatto sentire parte integrante del progetto crashworthiness, e quindi permesso di lavorare in un ambiente produttivo e stimolante.

Come posso non ringraziare Lorenzo, che ha avuto modo di seguire il mio percorso di tesi sin dagli albori, condividendo con me tutte le difficoltà del caso ma anche le belle soddisfazioni derivanti da questa strada percorsa insieme.

Per ultimi, ma credo che per qualsiasi individuo siano al primo posto, ringrazio la mia ragazza e i miei amici, che mi conoscono a fondo e sono stati capaci di far venire fuori la parte migliore di me, facendo capire a quel ragazzino spaesato che le proprie capacità vanno sempre valorizzate, a patto che ciò sia fatto con completa umiltà.