

# Laboratorio di Calcolo, Esercitazione 8, 1-5 dicembre 2025

Canale Pet-Z, Docenti: Shahram Rahatlou, Fabio Bellini, Sibilla Di Pace

**Interazione di particelle con un bersaglio di atomi:** Lo scopo di questa esercitazione è di simulare l'interazione di un fascio di particelle con un bersaglio di forma ellittica composto da atomi. È necessario implementare alcune funzioni ed utilizzare gli array per immagazzinare dati da passare alle funzioni.

## ► Cartella di lavoro

Fare login sulla postazione utilizzando le credenziali del vostro gruppo, `lcsrNNN`, dove NNN è il vostro numero di gruppo, ad esempio `098`. Creare una cartella `LCSR8` nella *home directory* con il comando `mkdir` in cui scriverei programmi di oggi. Spostarsi in questa cartella con il comando `cd`. Tutti i file di codice sorgente in C e in python dovranno trovarsi in questa cartella per essere visualizzati. **Le cartelle create sulla scrivania (Desktop) o in altre sotto-cartelle non verranno copiate né valutate.**

**Si consiglia di scrivere il programma in modo incrementale, verificando la corretta compilazione e l'esecuzione almeno dopo ciascuno dei passi indicati nel testo.**

---

## ► Prima parte

Quando una particella colpisce un bersaglio, interagisce con una certa probabilità  $p_{\text{int}}$  con gli atomi che compongono il bersaglio. Per simulare questo fenomeno si possono immaginare gli atomi del bersaglio come una serie di punti distribuiti a caso su una superficie  $S$  che rappresenta la sezione trasversale del bersaglio. Assumiamo che una particella che incide sul bersaglio nelle coordinate  $(x_p, y_p)$ , interne alla superficie  $S$ , abbia una probabilità di interazione  $p_{\text{int}} = 70\%$  di interagire con un atomo che si trova a una distanza  $R_{\text{int}}$  da essa.

Lo scopo del programma di oggi è di simulare l'interazione di un fascio di  $N_p$  particelle incidenti, lungo l'asse  $z$ , su un bersaglio ellittico con il semiasse maggiore  $A$  lungo l'asse  $x$  e il semiasse minore  $B$  lungo l'asse  $y$ , nel quale sono contenuti  $M_A$  atomi. Si ricorda che l'ellisse è descritta dall'equazione  $(x/A)^2 + (y/B)^2 = 1$  e quindi per i punti al suo interno si ha  $(x/A)^2 + (y/B)^2 \leq 1$ .

Creare un file `collision.c` nella cartella `LCSR8` utilizzando l'editor di testo `emacs`, per eseguire le seguenti operazioni:

1. chiedere all'utente di inserire i valori per le variabili  $N_p$ ,  $M_A$ ,  $R_{\text{int}}$ ,  $A$ , e  $B$  (le ultime tre variabili in unità arbitrarie *ua*), scegliendo variabili di tipo opportuno, ed assicurandosi che siano rispettate le condizioni  $R < 0.1$ ,  $B < A < 10$ , ed  $N_p < M_A < 10000$ . A tal fine implementare la funzione `inserisci(char* stringa, double a, double b)` di tipo `double`, che necessita come argomenti il nome della variabile e gli estremi inferiore e superiore. Utilizzare questa funzione per acquisire i valori delle variabili;
2. implementare una funzione `ellisse(double A, double B, double* x, double* y)` di tipo `void`, che generi in modo uniforme le coordinate  $x_A$  e  $y_A$  di un punto all'interno di un ellisse. Gli argomenti della funzione sono, rispettivamente, i due semiassi e le due coordinate da generare, passate come puntatori;

3. implementare una funzione `void atomi(double coord[][2], int M, double A, double B)` per generare le coordinate di  $M_A$  atomi contenuti all'interno dell'ellisse e memorizzare queste coordinate nell'array bidimensionale `bersaglio` di opportuna lunghezza. Al suo interno, questa funzione deve chiamare la funzione `ellisse()`, implementata al punto precedente, per generare i punti nell'ellisse;
  4. con un ciclo opportuno, generare le coordinate  $(x, y)$  di  $N_p$  particelle incidenti sul bersaglio. Per ciascun proiettile verificare se c'è un atomo a distanza  $R_{\text{int}}$ , e tenendo conto della probabilità  $p_{\text{int}}$  di interazione, determinare il numero  $N_{\text{int}}$  di atomi con cui interagisce.
  5. al termine del ciclo dei proiettili, stampare sullo schermo il numero medio di interazioni ed il numero massimo di interazioni avvenute per una particella incidente.
- 

## ► Seconda parte

1. Modificare `collision.c` per chiedere all'utente di inserire il nome del file in cui immagazzinare le posizioni  $x_A$  ed  $y_A$  degli atomi (un atomo per riga,  $M_A$  righe in totale);
2. scrivere un programma `collision.py` in python per leggere i valori dal file di dati e graficare la posizione degli  $M_A$  atomi usati nella simulazione; per visualizzare i punti con le coordinate  $(x, y)$  senza congiungerli con un segmento (scatter plot) potete usare la funzione

```

1  plt.scatter(xa,ya, marker='.', label='atomi nel bersaglio')
2

```

Listato 1: scatter plot di  $y$  verso  $x$

3. aggiungere opportuni titoli agli assi per comprendere il grafico;
4. disegnare un'ellisse con i valori di  $A$  e  $B$  inseriti.

```

1  # equazione parametrica per grafica l'ellisse
2  import numpy as np
3  theta = np.linspace(0, 2*np.pi, 100)
4  plt.plot(A*np.cos(theta), B*np.sin(theta), 'r-', label='bordo
5  bersaglio')

```

Listato 2: disegnare un'ellisse