



# Potenziamento STEAM e Pensiero Computazionale

---

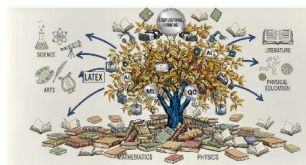
Liceo Scientifico tradizionale GALFER

## Cosa succede al Galfer?

La nostra scuola offre un curriculum in continua evoluzione, aperto alle innovazioni, alle differenti inclinazioni e talenti dei nostri studenti, alla complessità della contemporaneità.

Quindi una didattica mai arroccata nella tradizione, ma che da questa trova nuovi spunti di sperimentazione, nuove direzioni, con una maggiore enfasi sulla didattica laboratoriale dei programmi curriculari. In particolare (ma non solo) nei corsi di Matematica e Fisica, che caratterizzano il Liceo Scientifico.

Anche in questo contesto nasce la proposta di un Indirizzo Computazionale del Liceo Scientifico Tradizionale (di Ordinamento).



## Perché inserire nuovi percorsi?

1. perché il futuro è un bersaglio mobile, lo studente che si diploma viene proiettato in un contesto molto diverso dalla realtà di quando si era iscritto: la Scuola non deve rischiare di restare indietro
2. perché dobbiamo fornire allo studente le migliori possibilità di successo negli studi futuri

## Caratterizzazione del Percorso

La principale determinante è il **Pensiero Computazionale**: è pensare a più livelli di astrazione, è un processo logico-creativo per affrontare problemi, che si ispira a concetti fondamentali dell'Informatica; non si limita alla programmazione (coding), ma si inoltra nella risoluzione di problemi e nella comprensione di situazioni complesse.

Acquisire il Pensiero Computazionale è impadronirsi di una rete di strumenti cognitivi, oltre che tecnici, che espande l'approccio scientifico classico, e conduce a nuove possibilità di indagine. Innovare la didattica è quindi una necessità perché lo studente sia accompagnato nella sfida di questa trasformazione.

## Pensiero Computazionale - 4 capisaldi

Il Pensiero Computazionale si fonda su **quattro capisaldi**:

1. **Scomposizione**: ogni situazione viene ricondotta a parti più piccole, ciascuna di queste più facile da comprendere, analizzare, risolvere
2. **Riconoscimento**: cercare analogie e somiglianze con situazioni e schemi noti, e problemi già risolti, per poter guidare l'analisi e la risoluzione
3. **Astrazione**: guardare il quadro generale e le relazioni globali, filtrare le informazioni e le idee irrilevanti, concentrarsi su quelle rilevanti e organizzarle in strutture che permettano lo sviluppo di modelli e algoritmi
4. **Algoritmi**: astrazioni procedurali che lavorano sulle strutture e implementano possibili soluzioni utilizzando le risorse in maniera ottimale; verificare la correttezza delle soluzioni; generalizzare alla soluzione di problemi analoghi.

## Come lo mettiamo in pratica al Galfer?

Discipline e orario annuale	1°	2°	3°	4°	5°
Lingua e Letteratura italiana	4	4	4	4	4
Lingua e cultura latina	3	3	3	3	3
Lingua e cultura straniera	3	3	3	3	3
Storia e Geografia	3	3			
Storia			2	2	2
Filosofia			3	3	3
Matematica (con informatica)	4+1	4+1	3+1	3+1	3+1
Fisica (con informatica)	2+1	2+1	3+1*	3+1*	3+1*
Scienze naturali (Biologia, Chimica, Scienze della Terra)	2	2	3	3	3
Disegno e storia dell'arte	2	2	2	2	2
Scienze motorie e sportive	2	2	2	2	2
Religione cattolica o Attività alternative	1	1	1	1	1
TOTALE ORE	28	28	30*	30*	30*

\*L'indirizzo computazionale prevede 4 ore di fisica nel triennio, che non incidono sul monte orario settimanale, grazie ad una compensazione interna al curriculum che coinvolge a turno tutte le diverse discipline

La nostra proposta didattica prevede di **aggiungere** un'ora settimanale al corso di Fisica (per un totale di 3 ore nel biennio e 4 nel triennio), e che entrambe le discipline, tenute dal medesimo docente per ciascuna classe, riservino un'ora settimanale alle attività previste di tipo laboratoriale. Queste si articolano su più assi:

- **Informatica**: algoritmi e strutture dati, programmazione, sistemi operativi, architetture, reti e comunicazione - contesto privilegiato per acquisire le base del Pensiero Computazionale
- **Fablab**: Arduino, robotica educativa, stampanti 3D e laboratorio di making, realtà virtuale
- **Sviluppo del Pensiero Computazionale** e applicazione alle discipline, sia Scientifiche che Umanistiche
- **Applicazioni dell'Informatica** alla Matematica, alla Fisica, alle Scienze Naturali, modelli matematici, numerici e computazionali, simulazioni
- **Intelligenza Artificiale e Machine Learning**, fondamenti matematici e applicazione con utilizzo di modelli di linguaggio di grandi dimensioni (LLM)
- **Computazione Quantistica**: fondamenti matematici e fisici del Quantum Computing, applicazioni e Tecnologie Quantistiche (i 5 pilastri - calcolo, simulazione, comunicazione, sensoristica, ricerca di base)

## I nostri spazi, le nostre dotazioni

Il nostro Liceo ha potuto ampliare ulteriormente, anche grazie al PNRR Scuola 4.0, le dotazioni tecnologiche sia nelle aule che nei laboratori tra i quali, relativamente all'Indirizzo Computazionale:

- tutte le aule con LIM, 29 aule dotate di tablet e PC ad elevate prestazioni per sviluppo, grafica, VR, analisi dati
- Laboratorio di Fisica, Laboratorio di Chimica, 2 laboratori di Informatica, Laboratorio Linguistico, Laboratorio di Bioteologie, 2 FabLab (robotica/making/VR)

## Non solo STEM — STEAM!

I benefici del Pensiero Computazionale non si limitano a qualche applicazione di tecnologie digitali a problemi di Matematica e Fisica. Nelle ore **aggiuntive** dei corsi di Matematica e Fisica comincia un lavoro che conduce all'acquisizione degli strumenti base del Pensiero Computazionale, sia quelli (tecnici (digitali)), sia quelli cognitivi; ciò porta a ricadute in tutte le altre discipline:

Dal profilo dello studente del Liceo Scientifico

- **Area Logico-Argomentativa**: si sviluppa il ragionamento logico rigoroso, l'identificazione di problemi e l'individuazione di possibili soluzioni  
↳ tali obiettivi sono gli stessi del Pensiero Computazionale
- **Area Metodologica**: acquisizione di un metodo di studio autonomo e flessibile, e capacità di compiere le necessarie interconnessioni tra i metodi e i contenuti delle singole discipline  
↳ il Pensiero Computazionale fornisce strumenti mentali come la scomposizione (di un problema complesso) e l'astrazione (individuare la chiave del problema e i dettagli rilevanti) fondamentali per affrontare compiti complessi in qualsiasi ambito (analisi di un testo letterario complesso, ricostruzione di un evento storico)

I capisaldi del Pensiero Computazionale sono strumenti cognitivi universali:

- **Scomposizione**: pensare ai problemi in termini di parti e componenti - esempio: analisi di un'opera letteraria, o un periodo storico complesso, un un testo filosofico, scomponendo negli elementi costitutivi
- **Generalizzazione**: riconoscere pattern, individuare somiglianze e analogie, trasferire processi risolutivi - ad esempio collegare fatti analoghi nella storia, o temi ricorrenti in letteratura, o applicare modelli interpretativi a testi diversi
- **Astrazione**: analizzare i dati, individuare la chiave del problema, monitorare le informazioni rilevanti - per esempio comprendere l'essenziale di un complesso argomento filosofico, identificare l'idea principale di un testo, o distinguere le caratteristiche importanti di un periodo storico o un evento
- **Algoritmi**: pensare in termini di creazione di un algoritmo, una astrazione procedurale, verificare i risultati e poi generalizzare - ad esempio strutturare una argomentazione, pianificare una ricerca, organizzare un discorso in maniera logica e sequenziale; confrontare diverse interpretazioni, valutare l'affidabilità delle fonti storiche o letterarie, o scegliere la migliore strategia di presentazione di un argomento

## Il problema

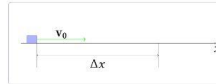
Un veicolo in moto alla velocità  $v_0$  frena bruscamente e le ruote si bloccano, quindi il veicolo viene frenato unicamente dall'attrito dinamico tra gli pneumatici e l'asfalto. Varianti del problema:

a. Quanto è intenso l'attrito?

b. Quanto spazio percorrerà ancora prima di arrestare la sua corsa?

## Il Modello Teorico

Normalmente i dispositivi di sicurezza evitano il blocco delle ruote, la frenata avviene in maniera controllata. Ma in caso di **malfunzionamento** le ruote si **bloccano**. Allora il modello è quello di un punto materiale di massa  $m$  che si muove su un piano (orizzontale) con una velocità iniziale nota  $v_0$ , frenato dalla forza di attrito dinamico  $F_d$ .



Ponendo l'asse delle ascisse lungo la direzione iniziale del moto dell'auto (quindi in direzione di  $v_0$ ) scriviamo la legge di Newton (lungo l'asse delle  $x$ ) in questo modo:

$$ma_x = -F_d = -\mu N = -\mu mg \quad (1)$$

laddove  $N$  è l'intensità della reazione vincolare, che, essendo il piano orizzontale, è pari alla forza peso  $N = mg$ . Per l'accelerazione:

$$a_x = -\mu g \quad (2)$$

Da cosa dipende il coefficiente di attrito dinamico  $\mu$ ?

Dalla teoria:  $\mu$  dipende da entrambi i materiali a contatto (in questo caso gomma/asfalto), ma le condizioni del contatto e dei materiali sono importanti:

- ci sono detriti?
- c'è terriccio/fango oppure foglie bagnate?
- acqua o ghiaccio?
- la gomma dello pneumatico è usurata?

In base a questi fattori  $\mu$  può variare in maniera notevole.

Calcolo del coefficiente di attrito  $\mu$  conoscendo  $v_0$  e  $\Delta x$

$$\mu = \frac{v_0^2}{2g\Delta x} \quad (3)$$

Calcolo dello spazio di arresto  $\Delta x$  conoscendo  $v_0$  e  $\mu$

Dalla cinematica del MRUA ricaviamo per lo spazio di frenata  $\Delta x$ :

$$\Delta x = \frac{v_0^2}{2g\mu} \quad (4)$$

## Approccio Sperimentale

Sul piano orizzontale del laboratorio prepariamo

- Un blocco di legno su un piano orizzontale
  - Un portale con fotocellule per determinare la velocità  $v_0$
  - Una scala graduata parallela al percorso del blocco per misurarne lo spostamento
- Il blocco viene lanciato attraverso il portale, in modo che possa proseguire parallelamente alla scala (talvolta può essere conveniente usare una riga al posto della scala fissa, nei casi di lancio non del tutto dritto).

Possiamo **misurare il coefficiente di attrito** misurando la velocità iniziale col portale e lo spazio di frenata con il metro, ed usando una procedura di fit (per esempio quadratico)

$$Y \pm \Delta x \quad X \pm v_0 \quad a = \frac{-1}{2g\mu}$$

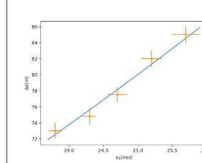
$$Y = aX^2$$

determinando il parametro  $a$  e con esso ricavare  $\mu$ .

Questo lavoro ci permette di scrivere un modello che permetta di prevedere lo spazio di frenata in funzione della velocità iniziale, utilizzando il valore del coefficiente di attrito dinamico  $\mu$  determinato sperimentalmente:

$$\Delta x = \frac{v_0^2}{2g\mu}$$

### Risultato



Il grafico rappresenta lo spazio di frenata in funzione della velocità iniziale  $\Delta x = \frac{v_0^2}{2g\mu}$  con  $\mu = 0.4$ . Sono rappresentate anche le barre di errore calcolate come nel riquadro sotto (cf. **Propagazione dell'errore**.)

Questo grafico rappresenta l'esito dell'applicazione del metodo sperimentale al problema posto. Il valore di  $\mu$  viene determinato come:

$$\mu = 0.398 \pm 0.002$$

### Propagazione dell'errore

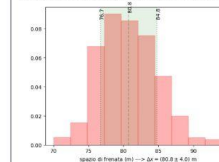
Usando le regole di propagazione dell'errore abbiamo per l'errore relativo nei due casi:

$$a. e_\mu = 2\frac{e_{v_0}}{v_0} + \frac{e_g}{g} + \frac{e_{\Delta x}}{\Delta x} = 2e_{v_0} + e_g + e_{\Delta x} \text{ - per determinare } \mu$$

$$b. e_{\Delta x} = 2\frac{e_{v_0}}{v_0} + \frac{e_g}{g} + \frac{e_\mu}{\mu} = 2e_{v_0} + e_g + e_\mu \text{ - per il modello predittivo } \Delta x(v_0)$$

## Approccio Computazionale

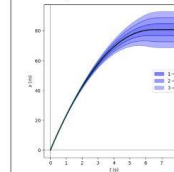
Un approccio diverso prevede di **simulare** un gran numero di volte il processo fisico da descrivere, per ogni simulazione vengono usati dei valori delle condizioni iniziali (posizione e velocità) e dei valori dei parametri del sistema (in questo caso  $\mu$  e  $g$ ) che vengono estratti casualmente, in modo che *in media*, su tutte le simulazioni, i valori medi siano pari ai valori che ci attendiamo queste variabili abbiano, e che la dispersione di questi valori casuali intorno alla media rispetti le incertezze delle misure di queste quantità (cioè gli errori assoluti su tali grandezze). Vengono poi calcolate le statistiche sugli esiti delle simulazioni, per esempio questo è l'istogramma degli spazi di frenata in un esempio svolto:



Il modello analitico ci dava la possibilità di determinare il valore medio dello spazio di frenata, e l'incertezza prevista tramite propagazione dell'errore. Invece il modello computazionale, oltre a darci le stesse informazioni di base, ci fornisce anche le probabilità di ciascun esito.

In questo modo abbiamo una informazione sulla robustezza delle nostre previsioni, cosa che non è così facile da ricavare in un modello analitico. Questo risultato lo otteniamo in maniera naturale tramite simulazioni, **scomponendo** il problema in un gran numero di ripetizioni: ciò permette di **determinare** le probabilità in maniera diretta. Qualunque variante sul processo fisico potrà essere inglobata nel modello introducendo i suoi effetti fisici in cascata simulazione.

La simulazione ci permette di monitorare gli istanti intermedi, possiamo determinare, nell'insieme di simulazioni, le probabilità con le quali l'automobile si trovi in *ciascun punto intermedio* del percorso:



In questo grafico abbiamo rappresentato con ombreggiature digradanti le traiettorie di minore probabilità. Notiamo che dopo circa 6 s la distanza percorsa dall'inizio della frenata smette di aumentare, per la maggior parte delle traiettorie.

## Un po' di codice

Qui a destra mostriamo il frammento di codice che si incarica di fare le simulazioni. Viene usato il linguaggio di programmazione Python, che è dotato di semplicità di sviluppo, leggibilità, e molte librerie contenenti funzionalità avanzate, spesso sviluppate da esperti dei vari settori, come analisi dati (Pandas), calcolo scientifico e numerico (NumPy e SciPy), machine learning (SKLearn, Orange) e così via.

```

import random
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.metrics import mean_squared_error

# Parametri fisici
mu = 0.4
g = 9.81

# Simulazioni
n_sim = 10000
spazio_frenata = []

for i in range(n_sim):
    v0 = random.uniform(0, 26)
    # Calcolo dello spazio di frenata
    dx = v0**2 / (2 * g * mu)
    spazio_frenata.append(dx)

# Statistiche
media_dx = np.mean(spazio_frenata)
std_dx = np.std(spazio_frenata)

# Plot
plt.hist(spazio_frenata, bins=10, density=True)
plt.title('Distribuzione dello spazio di frenata')
plt.xlabel('Spazio di frenata (m)')
plt.ylabel('Densità')
    
```

## Moto parabolico nella realtà

Nello studio del moto parabolico (moto di caduta libera) viene tipicamente trascurato l'attrito del punto materiale con l'aria. Naturalmente ciò è dovuto alla necessità di rendere il problema risolvibile facilmente con metodi matematici dei primi anni del Liceo, ma impedisce di analizzare il problema in un contesto realistico, come può essere il caso di una palla da gioco lanciata durante una partita. La matematica studiata all'ultimo anno permette invece la risoluzione del problema in presenza di attrito viscoso.

## Il problema

Il problema della traiettoria di un oggetto viene formulato in maniera fisicamente corretta come segue:

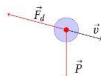
Un corpo di massa  $m$  viene lanciato con una velocità data  $\vec{v}_0$  (vettore noto). Supponiamo che la velocità resti bassa per tutta la durata del moto. Mentre il corpo è in volo alla velocità  $\vec{v}$  è soggetto a:

- la forza peso  $\vec{P} = m\vec{g}$
- l'attrito viscoso dell'aria  $\vec{F}_v = -\beta\vec{v}$



A velocità maggiore diventa più importante la resistenza aerodinamica, quindi il nostro modello prevede due forze:

- la forza peso  $\vec{P} = m\vec{g}$
- la resistenza aerodinamica (drag)  $\vec{F}_d = -\frac{1}{2}\rho C_d A \vec{v} = -c\vec{v}v$  (a velocità più alte la resistenza è proporzionale a  $v^2$ )



## Approccio Teorico - bassa velocità (attrito viscoso)

Come per tutti i problemi di dinamica lo studente scrive allora le equazioni di Newton per il corpo  $m\vec{a} = m\vec{g} - \beta\vec{v}$  in due dimensioni:

$$\begin{cases} m a_x = -\beta v_x \\ m a_y = -\beta v_y - mg \end{cases} \quad (1)$$

Dal punto di vista matematico il problema richiede la risoluzione di una coppia di equazioni differenziali non accoppiate

$$\begin{cases} m \frac{dv_x}{dt} = -\beta v_x \\ m \frac{dv_y}{dt} = -\beta v_y - mg \end{cases} \quad (2)$$

cosa che rientra nel programma della classe quinta, ma non in quello della classe terza. In assenza di mezzi computazionali siamo quindi costretti o a svolgere il problema nelle condizioni di assenza di attrito, oppure a limitarci ad un approccio sperimentale che in assenza di mezzi matematici mancherà di un modello teorico completamente svolto: nel modo tradizionale ci limiteremo a constatare che le due componenti del moto corrispondono al moto unidimensionale in un fluido viscoso, e quindi mostrare che entrambe tendono ad una velocità limite (nulla per il moto orizzontale,  $\frac{mg}{\beta}$  per il moto verticale).

## Approccio Teorico - velocità alta (resistenza aerodinamica)

Quando la velocità è più alta, però, il modello viscoso non descrive più bene le evidenze sperimentali. Le equazioni presentano allora un termine più difficile da trattare:

$$\begin{cases} m a_x = -c v_x \sqrt{v_x^2 + v_y^2} \\ m a_y = -c v_y \sqrt{v_x^2 + v_y^2} - mg \end{cases} \quad (3)$$

In questo caso il problema matematico è costituito da due equazioni differenziali accoppiate

$$\begin{cases} m \frac{dv_x}{dt} = -c v_x \sqrt{v_x^2 + v_y^2} \\ m \frac{dv_y}{dt} = -c v_y \sqrt{v_x^2 + v_y^2} - mg \end{cases} \quad (4)$$

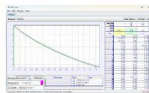
ed è molto più complesso. Per questo è necessario avere una risoluzione computazionale del modello teorico.

## Approccio Sperimentale: Tracker

Per enfatizzare il ruolo dell'attrito conviene svolgere l'esperimento con proiettili molto leggeri, in modo che la resistenza dell'aria (viscosa o aerodinamica) sia maggiormente evidente. Un esempio qui riportato prevede l'utilizzo di un volano per il Badminton.

La presa dati viene qui effettuata tramite un video, che può essere realizzato in palestra: uno studente lancia un volano in modo che il suo moto si svolga in un piano perpendicolare alla linea di vista della telecamera. Successivamente viene analizzato il video tramite il programma Tracker, che ci mette in condizione di fare l'analisi dei dati ricavati dal video. Per esempio possiamo analizzare separatamente le componenti del moto, determinando tramite Tracker posizioni, velocità e accelerazioni. Se conosciamo le previsioni del modello teorico (se possibile determinate analiticamente, altrimenti in maniera computazionale - vedi affianco) sono possibili diverse analisi:

- usare i dati sulle accelerazioni e sulle velocità e verificare se la forza contiene un termine proporzionale alla velocità o al suo quadrato (caso viscoso/caso drag)
- usare i dati delle velocità e quelli sulle posizioni e decidere se i tracciati corrispondono, all'interno dell'incertezza sperimentale, alle previsioni del modello.



Inoltre Tracker permette di esportare i dati per successive analisi, per esempio usando Python. Un possibile sviluppo è quello di usare simulazioni Monte Carlo tramite la libreria PyMC per determinare i parametri fisici del modello: in questo modo è possibile usare modelli più complessi di quanto sia possibile usando il solo Tracker, che usa un approccio classico all'analisi dati anziché bayesiano.

## Approccio Computazionale

Un modello più complesso per descrivere il problema potrebbe avere **entrambi** i termini visti precedentemente, terremo conto sia dell'attrito viscoso sia del drag aerodinamico:

$$\begin{cases} m a_x = -\beta v_x - c v_x \sqrt{v_x^2 + v_y^2} \\ m a_y = -mg - \beta v_y - c v_y \sqrt{v_x^2 + v_y^2} \end{cases} \quad (5)$$

Come anticipato il modello più completo è molto complesso, e per affrontarlo usiamo le possibilità offerte dalla programmazione in Python; le traiettorie vengono simulate numericamente a partire dal modello, il valore dei parametri viene determinato tramite una procedura probabilistica, in modo da scegliere i valori che meglio riproducono i dati osservati, utilizzando il teorema di Bayes:

$$P(c, \beta | \text{Dati}) \propto P(\text{Dati} | c, \beta) P(c, \beta) \quad (6)$$

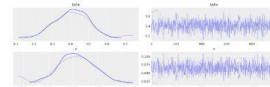
I dati in ingresso vengono raccolti tramite Tracker (vedi riquadro a sinistra), e letto dalla libreria di analisi dati Pandas: in questo modo tutta la programmazione viene effettuata in Python, che risulta essere quindi strumento di elezione per le attività computazionali didattiche dell'indirizzo.

Il metodo Monte Carlo prevede che venga effettuato un gran numero di simulazioni per valutare le probabilità richieste dal teorema di Bayes, in questo modo si ottiene una previsione probabilistica dei possibili valori dei parametri.

A tale scopo viene utilizzata la libreria PyMC, che permette di ridurre un compito così oneroso ad un problema di programmazione: anche se i dati generati sono numerosissimi, l'astrazione della libreria PyMC rende possibile maneggiarli con agilità.

```
def model(c, beta):
    """Model function for the projectile motion"""
    # Initial conditions
    x0 = 0.0
    y0 = 0.0
    vx0 = 1.0
    vy0 = 0.0
    # Time steps
    dt = 0.01
    t = 0.0
    # Trajectory
    x = x0
    y = y0
    vx = vx0
    vy = vy0
    while t < 1.0:
        # Accelerations
        ax = -beta*vx - c*vx*sqrt(vx**2 + vy**2)
        ay = -g - beta*vy - c*vy*sqrt(vx**2 + vy**2)
        # Velocities
        vx = vx0 + ax*dt
        vy = vy0 + ay*dt
        # Positions
        x = x0 + vx*dt
        y = y0 + vy*dt
        t = t + dt
    return x, y, vx, vy
```

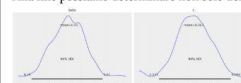
## I tracciati delle simulazioni



## Valori dei parametri e probabilità

Alla fine possiamo determinare non solo delle previsioni per i valori dei

parametri fisici  $c$  e  $\beta$ , ma le probabilità per ciascun valore di essi, come per esempio nel grafico affianco.





# Presentazione sintetica del calcolo del MCD nei due Licei scientifici.

prof. di Matematica e di Fisica Abracadabrus<sup>1</sup>

<sup>1</sup>Liceo Scientifico Statale Galileo Ferraris di Torino

## Introduzione

Il Massimo Comune Divisore (MCD) rappresenta uno dei concetti fondamentali dell'aritmetica, un pilastro su cui si costruisce la comprensione della teoria dei numeri. Nel percorso del liceo scientifico, lo studio del MCD segna un passaggio cruciale dall'aritmetica elementare all'algebra, fornendo strumenti essenziali per la scomposizione di polinomi e la semplificazione di espressioni.

Nel liceo scientifico tradizionale, l'MCD viene introdotto attraverso il metodo della scomposizione in fattori primi. Gli studenti imparano un procedimento algoritmico basato sulla fattorizzazione dei numeri per identificare i divisori comuni e, tra questi, il più grande. Questo approccio, sebbene efficace, è strettamente legato al calcolo manuale e rafforza le abilità di scomposizione e analisi numerica.

Il concetto assume una dimensione nuova e più dinamica nel liceo scientifico con potenziamento informatico. Qui, la solida base matematica si arricchisce della potenza del pensiero computazionale. Gli studenti non si limitano a calcolare l'MCD, ma imparano a "insegnare" a una macchina come farlo. Grazie alla conoscenza del linguaggio di programmazione Python, possono esplorare metodi più efficienti, come l'antico ma potentissimo algoritmo euclideo. Implementare l'algoritmo di Euclide in Python trasforma una nozione teorica in un'applicazione pratica e interattiva. Gli studenti non solo comprendono la logica matematica sottostante, ma sviluppano anche competenze di problem solving e coding, estendendo il concetto di MCD dal dominio della matematica pura a quello della programmazione e dell'efficienza algoritmica.

## MCD NEL LICEO SCIENTIFICO TRADIZIONALE (sintesi)

Il Massimo Comune Divisore (MCD) è il più grande numero intero positivo che può dividere esattamente due o più numeri interi, senza lasciare resto.

In altre parole, se prendi due numeri, l'MCD è il maggiore tra tutti i loro divisori in comune.

Il metodo più comune per calcolare l'MCD è la scomposizione in fattori primi. La procedura consiste nei seguenti passaggi:

- Scomporre ogni numero nei suoi fattori primi.
- Scegliere solo i fattori primi che sono comuni a tutti i numeri.
- Prendere ciascuno di questi fattori comuni con l'esponente più piccolo con cui compare.
- Moltiplicare tra loro i fattori scelti.

### 1. Scomposizione:

$$36 = 2^2 \cdot 3^2 \\ 48 = 2^4 \cdot 3^1$$

### 2. Fattori comuni: I fattori in comune sono il 2 e il 3.

### 3. Esponente più piccolo:

- Per il fattore 2, l'esponente più piccolo è 2 (da  $2^2$ ).
- Per il fattore 3, l'esponente più piccolo è 1 (da  $3^1$ ).

### 4. Moltiplicazione:

$$\text{MCD}(36, 48) = 2^2 \cdot 3^1 = 4 \cdot 3 = 12$$

Il numero 12 è il numero più grande che può dividere sia 36 (infatti  $36 \div 12 = 3$ ) sia 48 (infatti  $48 \div 12 = 4$ ).

A Cosa Serve?

L'MCD è un concetto fondamentale in matematica, usato principalmente per:

- Semplificare le frazioni: Per ridurre una frazione ai minimi termini, si dividono il numeratore e il denominatore per il loro MCD.

Esempio: Per semplificare la frazione  $\frac{48}{36}$ , si dividono entrambi i termini per  $\text{MCD}(48, 36) = 12$ .

$$\frac{48}{36} = \frac{48 \div 12}{36 \div 12} = \frac{4}{3}$$

## MCD NEL LICEO SCIENTIFICO TRADIZIONALE CON POTENZIAMENTO INFORMATICO (sintesi)

Per gli studenti che conoscono Python, lo studio del MCD si trasforma da un semplice esercizio di calcolo a un'esplorazione del pensiero computazionale e dell'efficienza algoritmica. L'obiettivo non è più solo trovare il risultato, ma progettare un metodo che un computer possa eseguire in modo rapido ed efficace.

Il primo passo è analizzare criticamente il metodo tradizionale. La scomposizione in fattori primi funziona bene per numeri piccoli, ma cosa succede con numeri molto grandi?

Gli studenti scoprono rapidamente che fattorizzare numeri di decine o centinaia di cifre è un compito estremamente lento e difficile, anche per un computer. Questo crea la necessità di un algoritmo più intelligente.

Tabella 1. Tempi di esecuzione approssimativi per il calcolo del MCD di due numeri tramite il metodo della scomposizione in fattori primi, al variare del numero di cifre.

Numero di Cifre	Ordine di Grandezza	Tempo di Esecuzione Stimato
10	$10^{10}$	$\approx 0.001$ secondi
12	$10^{12}$	$\approx 0.04$ secondi
14	$10^{14}$	$\approx 2 - 5$ secondi
15	$10^{15}$	$\approx 30 - 60$ secondi
16	$10^{16}$	1 - 5 minuti
18	$10^{18}$	Impraticabile (> 30 minuti)

Come la tabella dimostra chiaramente, il tempo di esecuzione non cresce in modo lineare, ma espone esponenzialmente. Passare da numeri di 14 cifre a numeri di 16 cifre aumenta il tempo di calcolo di quasi 100 volte.

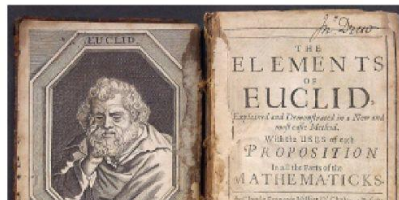
Perché Accade Questo? Il punto debole del metodo è la scomposizione in fattori primi. Trovare i fattori primi di un numero molto grande è un problema computazionalmente "difficile" (su cui si basa, per esempio, la sicurezza della crittografia RSA). Non esiste un algoritmo veloce conosciuto per farlo.

Il Confronto con l'Algoritmo di Euclide Se avessimo eseguito lo stesso test usando l'algoritmo di Euclide implementato in Python, il risultato sarebbe stato radicalmente diverso:

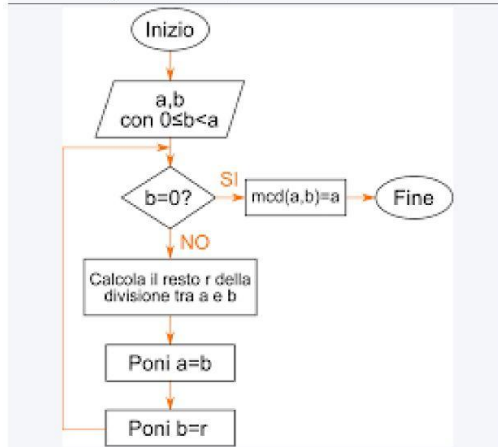
- Il calcolo dell'MCD per tutti i numeri nella tabella, inclusi quelli a 18 cifre (e anche molto più grandi), sarebbe stato praticamente istantaneo (nell'ordine dei milionesimi di secondo).

Questo esperimento non è solo un esercizio di programmazione, ma una lezione fondamentale sull'importanza della scelta dell'algoritmo giusto. Un buon algoritmo può trasformare un problema impraticabile in un compito banale per un computer.

Presentiamo, allora, nella prossima sezione l'algoritmo euclideo per la determinazione del MCD attraverso un diagramma di flusso.



## L'algoritmo euclideo per la determinazione del MCD



## L'algoritmo euclideo in Python

```
def mcd_euclide(a, b):
    """Calcola l'MCD usando l'algoritmo di Euclide (versione iterativa)."""
    while b != 0:
        # La riga seguente qui: a prende il valore di b,
        # e b prende il valore del resto di a / b.
        a, b = b, a % b
    return a

# Esempio di utilizzo
num1 = 48
num2 = 36
risultato = mcd_euclide(num1, num2)
print(f"l'MCD tra {num1} e {num2} è: {risultato}")
# Output: l'MCD tra 48 e 36 è: 12
```

Naturalmente, è possibile poi estendere il calcolo del MCD a più di due numeri, utilizzando ad esempio le liste e poi "ciclando" sulla function che definisce il MCD di due elementi. Al termine presentiamo la soluzione pythonica.

```
import math

# Il modo più semplice e veloce in Python
risultato_nativo = math.gcd(48, 36) # più via per Greatest Common Divisor
print(f"l'MCD usando la funzione nativa di Python è: {risultato_nativo}")
# Output: l'MCD usando la funzione nativa di Python è: 12
```

# Epidemiologia +Modello matematico+Simulazione in Python

prof.ssa di Scienze Naturali Aldebaran <sup>1</sup> – prof. di Matematica e di Fisica Rigel <sup>2</sup>

<sup>1</sup> Liceo Scientifico Statale Galileo - Ferraris di Torino

## Introduzione

Questo progetto interdisciplinare unisce scienze, matematica e informatica per analizzare la dinamica delle epidemie.

Dopo un ciclo di lezioni di scienze sulle basi biologiche delle malattie infettive, il docente di matematica e fisica introduce il modello epidemiologico SIR (Suscettibili, Infetti, Rimossi). Questo modello, descritto da un sistema di equazioni differenziali, permette di tradurre i meccanismi di contagio in un linguaggio matematico.

Successivamente, gli studenti utilizzano Python e le librerie scientifiche (come SciPy e Matplotlib) per implementare il modello. L'obiettivo è simulare e analizzare un caso di studio reale: un'epidemia di influenza avvenuta in un collegio inglese nel 1978. Confrontando i dati storici con i risultati della simulazione, gli studenti possono stimare i parametri chiave del modello, come il tasso di trasmissione ( $\beta$ ) e il tasso di guarigione ( $\gamma$ ), comprendendone il significato fisico.

Infine, il progetto introduce generalizzazioni del modello base, come il modello SEIR (che aggiunge lo stato di "Esposto") o modelli che includono nascite e morti, mostrando come la matematica possa adattarsi per descrivere sistemi biologici più complessi. L'attività sviluppa competenze di problem solving e coding, collegando la teoria astratta a un'applicazione concreta.

## Le Basi Biologiche delle Malattie Infettive: Un'Introduzione al Modello SIR

Le malattie infettive rappresentano una costante interazione tra agenti patogeni e i loro ospiti. Comprendere i fondamenti biologici di questa dinamica è essenziale per poterla poi descrivere con modelli matematici come il SIR (Suscettibili, Infetti, Rimossi). Alla base di ogni epidemia c'è un agente patogeno – un virus, un batterio, un fungo o un protozoo – capace di invadere un organismo ospite, replicarsi e causare una malattia.

Il processo infettivo può essere descritto dalla "catena del contagio", un ciclo che definisce come il patogeno si sposta da un individuo all'altro. Il primo anello è l'agente stesso, che risiede in un serbatoio (reservoir), come un'altra persona, un animale o l'ambiente. Da qui, deve trovare una via di uscita (es. goccioline respiratorie, sangue) per poi essere trasmesso a un nuovo ospite. La modalità di trasmissione è un concetto chiave: può essere diretta, tramite contatto fisico o starnuti, o indiretta, attraverso superfici contaminate (fomiti) o vettori come le zanzare. Questa interazione tra chi è infetto e chi non lo è, è il motore biologico che nel modello SIR viene catturato dal tasso di trasmissione ( $\beta$ ).

Una volta che il patogeno supera le barriere dell'ospite attraverso una via d'ingresso (es. mucose del naso o della bocca), incontra un individuo che può essere suscettibile. Un ospite suscettibile è colui che non possiede un'immunità specifica contro quel determinato agente. Qui inizia la battaglia con il sistema immunitario. Se il patogeno riesce a superare le difese iniziali e a replicarsi, l'individuo diventa infetto e, spesso, anche infettivo, cioè in grado di trasmettere a sua volta l'agente patogeno. Questo stato corrisponde al compartimento "I" (Infetti) del modello SIR.

La risposta dell'ospite è cruciale. Il sistema immunitario si attiva per combattere l'infezione, producendo anticorpi e cellule specializzate. L'esito di questa battaglia determina il destino dell'individuo. Se il sistema immunitario ha successo, l'infezione viene debellata. L'individuo guarisce e, in molti casi, sviluppa una memoria immunologica che lo protegge da future infezioni dello stesso patogeno. Questo stato di immunità acquisita è esattamente ciò che definisce il compartimento "R" (Rimossi o Guariti) nel modello SIR. L'individuo non è più suscettibile né infettivo. Il tempo necessario per passare dallo stato di infetto a quello di rimosso è determinato dalla biologia dell'infezione e dalla competenza del sistema immunitario, un concetto catturato dal tasso di guarigione ( $\gamma$ ).

In sintesi, il modello SIR è un'elegante semplificazione matematica di processi biologici complessi. I compartimenti S, I, e R non sono altro che una fotografia della popolazione basata sul suo stato immunologico e infettivo. La transizione da S a I è governata dalla biologia della trasmissione, mentre il passaggio da I a R riflette la biologia della risposta immunitaria e della guarigione.

## Il modello SIR

Il **modello SIR** è uno dei modelli compartimentali più semplici e fondamentali in epidemiologia matematica. Esso descrive la diffusione di una malattia infettiva in una popolazione suddividendola in tre gruppi, o "compartimenti", in base allo stato di salute degli individui.

1. **S (Suscettibili):** Rappresenta il numero di individui sani che non hanno ancora contratto la malattia ma che possono essere infettati.
2. **I (Infetti):** Indica il numero di individui che hanno contratto la malattia e sono in grado di trasmetterla ai suscettibili.
3. **R (Rimossi/Guariti):** Include gli individui che sono guariti dall'infezione e hanno acquisito un'immunità permanente, oppure sono deceduti. In entrambi i casi, non partecipano più al processo di infezione.

Il modello assume che la popolazione totale,  $N = S + I + R$ , rimanga costante. La dinamica dell'epidemia è descritta dal flusso di individui da un compartimento all'altro.

Il passaggio degli individui tra i compartimenti è descritto da un sistema di tre equazioni differenziali ordinarie, che esprimono la variazione di ciascun gruppo nel tempo ( $t$ ).

I suscettibili possono solo diminuire di numero, poiché si trasformano in infetti. La velocità con cui questo accade dipende dal numero di incontri tra individui infetti e suscettibili. Questa transizione è regolata da un parametro  $\beta$  (beta), il *tasso di trasmissione*. L'equazione è:

$$\frac{dS}{dt} = -\frac{\beta SI}{N} \quad (1)$$

- $\frac{dS}{dt}$  è la variazione del numero di suscettibili nel tempo.
- Il segno negativo indica che la popolazione suscettibile diminuisce.
- Il termine  $S \cdot I$  rappresenta le possibili interazioni tra i due gruppi.
- $N$  è la popolazione totale, quindi  $\frac{SI}{N}$  è la frazione di infetti.

Il numero di infetti aumenta a causa dei nuovi contagi provenienti dai suscettibili e diminuisce mano che le persone guariscono. La guarigione è governata dal parametro  $\gamma$  (gamma), il *tasso di guarigione*. L'inverso di gamma ( $1/\gamma$ ) rappresenta la durata media del periodo infettivo. L'equazione è:

$$\frac{dI}{dt} = \frac{\beta SI}{N} - \gamma I \quad (2)$$

- $\frac{\beta SI}{N}$  è il flusso di nuovi infetti (lo stesso termine che fa diminuire S).
- $\gamma I$  è il flusso di individui che guariscono, proporzionale al numero di infetti e al tasso di guarigione.

Il compartimento dei rimossi può solo aumentare, poiché accoglie gli individui che escono dai gruppi degli infetti. La sua variazione è uguale al termine di uscita dai compartimenti I.

$$\frac{dR}{dt} = \gamma I \quad (3)$$

- $\frac{dR}{dt}$  è la variazione del numero di rimossi nel tempo.
- Il termine  $\gamma I$  rappresenta il numero di individui che guariscono nell'unità di tempo.

Queste tre equazioni formano un sistema che, una volta impostate le condizioni iniziali (il numero di persone in S, I, e R al tempo  $t = 0$ ), descrive l'evoluzione temporale di un'epidemia.

## Scheda di Analisi: Modello SIR e Influenza in un Collegio Inglese (1978)

- **Evento:** Epidemia di influenza (virus H1N1).
- **Periodo:** Gennaio - Febbraio 1978.
- **Luogo:** Un collegio maschile nel nord dell'Inghilterra.
- **Popolazione:** Considerata "chiusa", ideale per l'applicazione del modello SIR.

- **Popolazione Totale (N):** 763 ragazzi.
- **Condizioni Iniziali** (al giorno  $t = 0$ ):
  - Suscettibili ( $S_0$ ): 738
  - Infetti ( $I_0$ ): 25
- **Inizio osservazione:** 22 gennaio 1978.

Basandosi sui dati dell'epidemia, sono stati stimati i parametri chiave del modello SIR:

- **Tasso di Guarigione ( $\gamma$ ):** 0.476 (corrispondente a un periodo infettivo medio di circa 2.1 giorni).
- **Tasso di Trasmissione ( $\beta$ ):** 0.002342 (calcolato dalla relazione  $\beta = 0.004918 \times \alpha$ ).

Il picco dell'epidemia si verifica quando il numero di nuovi contagi eguaglia il numero di guarigioni ( $\frac{dI}{dt} = 0$ ).

- **Picco Osservato dai Dati:** 296 ragazzi infetti (registrato tra il 6° e il 7° giorno).
- **Picco Calcolato dal Modello ( $I_{max}$ ):** 298 ragazzi infetti.

L'accordo tra il valore teorico e quello sperimentale è quasi perfetto, dimostrando l'elevata capacità predittiva del modello.

- **Accordo Modello-Dati:** Eccellente. La curva teorica degli infetti  $I(t)$  generata dal modello SIR riproduce fedelmente l'andamento dei dati reali, sia nella fase di crescita rapida che in quella di decrescita.
- **Efficacia:** Questo caso di studio è un esempio classico che dimostra come il semplice modello SIR di Kermack-McKendrick sia uno strumento potente e accurato per descrivere la dinamica di un'epidemia in una popolazione chiusa e omogenea.

## Generazione della curva $I(t)$ con la matplotlib di Python

Per ragioni di brevità e per mantenere il focus sull'analisi del modello, il codice Python utilizzato per la simulazione del caso di studio non è stato riportato in questo articolo.

