



Lezione 1

In questa lezione inizieremo a parlare dei limiti dei computer.

Partiamo osservando una linea retta.



Una **retta** è un'ente geometrico fondamentale costituito da un insieme **infinito** di punti, che possiamo rappresentare sul piano cartesiano utilizzando tre differenti equazioni: l'equazione implicita della retta, l'equazione esplicita della retta e l'equazione della retta passante per due punti:

Equazione implicita

$$ax + by + c = 0$$

(con a , b e c numeri reali (a , b non contemporaneamente uguali a zero))

Equazione esplicita


$$y = mx + q$$

(con m coefficiente angolare)

Retta passante per due punti

$$\frac{x-x_1}{x_2-x_1} = \frac{y-y_1}{y_2-y_1}$$

(dove i due punti sono (x_1, y_1) e (x_2, y_2))

 Ma come possiamo disegnare una retta sullo schermo di un computer ?

La retta è un'entità *infinita*, mentre lo schermo di un computer è un dispositivo con dimensioni *finite*. Quindi, se vogliamo disegnare una linea retta sullo schermo di un computer dovremmo **approssimarla** con un segmento formato da un insieme di **pixel**!

La **trasformazione** di una forma in una **immagine costituita da pixel** è detta **rasterizzazione**.

Esercizio 01

La seguente griglia (Figura 1) di dimensioni 20×20 **pixel** (*quadrantini*) rappresenta un (*piccolissimo*) schermo dove l'**origine delle coordinate di riferimento (0,0)** è posizionato nel pixel in alto a sinistra.

- Disegnate il segmento con estremi A e B, annerendo i pixel che vi sembrano più opportuni (la prima coordinata dei punti indica le ascisse, la seconda coordinata indica le ordinate):
A = (2, 7)
B = (18, 14)
- Quando avete finito, dividetevi in gruppi da due o tre persone e confrontate le rette disegnate.
Sono tutte uguali?
- Esistono dei casi particolari in cui le linee possono essere disegnate in modo semplice e non ambiguo?

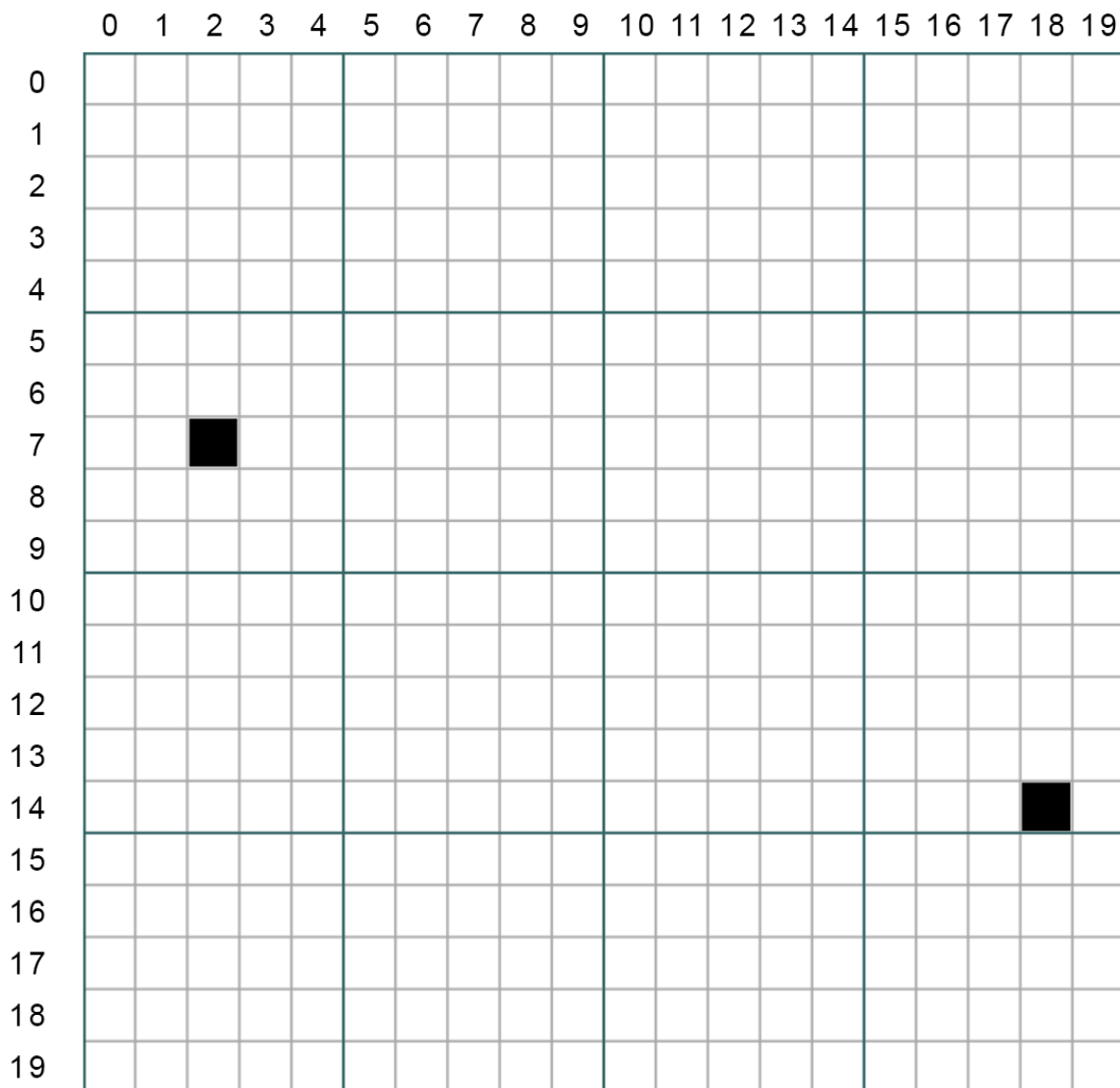


Figura 1: schema per l'esercizio 01.



Perché l'origine delle coordinate di questa griglia sono in alto a sinistra? Perché imitano le coordinate degli schermi.

Esercizio 01 +

Osservando la tua retta, prova a scrivere lo pseudo codice che ne permette la realizzazione.

E' utilizzabile per disegnare altre rette?

Una possibile soluzione al problema della rappresentazione delle linee rette in uno schermo attraverso i pixel, sfrutta l'**equazione esplicita della retta**. Dati i due punti A e B, possiamo ricavare il valore di m e q :

$$m = \frac{y_1 - y_2}{x_1 - x_2} \qquad q = y_1 - \frac{y_1 - y_2}{x_1 - x_2} x_1$$

Tramite l'equazione esplicita della retta, per ogni valore di x compreso tra i due punti A e B, si calcola il valore corrispondente della y , che ci indica, una volta arrotondato a numero intero, le coordinate del pixel da colorare.

Ecco la procedura:

```
INPUT: due punti (x1, y1), (x2, y2)

m = (y1 - y2) / (x1 - x2)
q = y1 - m * x1
per ogni x tra min(x1, x2) e max(x1, x2)
    y = m*x + q
    colora il pixel (x, round(y))
```

Esercizio 02

Esiste almeno un caso particolare in cui questo algoritmo non funziona come desiderato?

▼ *Suggerimento*

Come viene disegnata il segmento con estremi (2, 18) e (5, 5)?

Esercizio 02 +

Scrivi, in Python, un programma che calcola e stampa le coordinate dei pixel che

l'algoritmo *Naive line-drawing* (pseudo codice presentato sopra) andrebbe a disegnare.

Esegui il codice sui punti di estremi (2, 7) e (18, 14) (come nell'esercizio 1), e confrontali con la linea che avevi disegnato!

In tutti i casi in cui il coefficiente angolare della retta m è maggiore di 1, l'algoritmo *Naive line-drawing* non funziona come desiderato: lascia troppi spazi bianchi!

Per questo motivo sono stati sviluppati altri algoritmi per la rasterizzazione di linee, tra cui uno dei più utilizzati è l'**algoritmo di Bresenham**:

```
INPUT: due punti (x1, y1), (x2, y2)

a = 2 * (y2 - y1)
b = a - 2 * (x2 - x1)
p = a - (x2 - x1)

colora il pixel iniziale
per ogni x tra min(x1, x2)+1 e max(x1, x2)
  se p < 0:
    colora il pixel di ascissa x sulla stessa linea del pixel precedente
    p = p + a
  altrimenti:
    colora il pixel di ascissa x con ordinata incrementata rispetto al pixel precedente
    p = p + b
```

Oltre ad essere più corretto, l'algoritmo di Bresenham è più **efficiente**, perché le somme sono più semplici rispetto a moltiplicazioni e divisioni (usate, invece, nel ciclo dell'algoritmo *naive*).

Per fare altri due nomi di algoritmi di rasterizzazione di linee: algoritmo DDA e algoritmo della linea di Xiaolin Wu (che introduce l'effetto "sfumato" nelle linee, detto *antialiasing*).

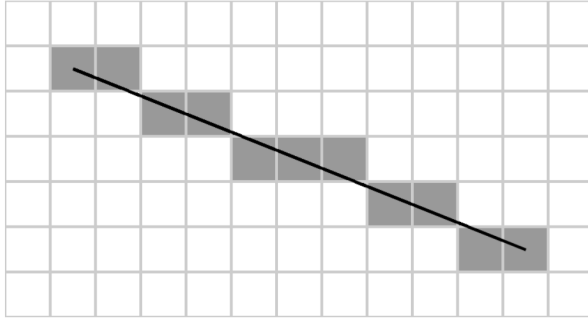


Figura 2: Esempio di linea disegnata con l'algoritmo di Bresenham.

Credits: Dntrlthmmnm_(1), (2).

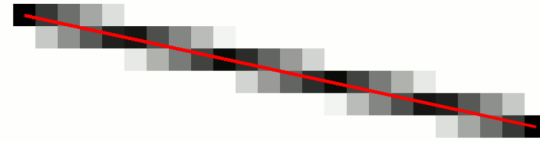


Figura 3: Esempio di linea disegnata con l'algoritmo di Xiaolin Wu.



Approfondimento - per una ulteriore spiegazione del metodo di Bresenham, leggete questo paragrafo: <https://www.csfieldguide.org.nz/en/chapters/computer-graphics/drawing-lines-and-circles/#bresenham-s-line-algorithm>

Cosa possiamo ricavare dall'esempio della rasterizzazione delle linee?

I computer hanno dei **limiti**!

Così come abbiamo incontrato i limiti nella rappresentazione grafica, li troviamo anche nella rappresentazione dei **numeri** e nei **calcoli** numerici.

1. Abbiamo visto che le linee rette, che sono infinite, devono essere approssimate da segmenti finiti. Allo stesso modo non tutti i numeri possono essere rappresentati dai computer, ma devono essere **approssimati**. Infatti, i computer hanno un **spazio di memoria finita** per i dati!
2. Alcuni problemi possono essere risolti con **algoritmi differenti**, e la rasterizzazione delle linee ne è un esempio. Il problema della rasterizzazione è un problema di approssimazione, e metodi differenti portano a soluzioni finali leggermente diverse le une dalle altre, più o meno vicine alla soluzione ideale.
3. Il tempo di calcolo di un problema dipende da quante operazioni bisogna eseguire. Vorremmo trovare le soluzioni più veloci (o comunque eseguibili in *tempo finito*!).

Cos'è il calcolo numerico?



Il **calcolo numerico** è la disciplina matematica che studia i metodi per risolvere problemi matematici tramite algoritmi computazionali.

I metodi del calcolo numerico possono essere suddivisi in due categorie:

1. **Metodi diretti**: risolvono i problemi ottenendo la soluzione esatta, applicando un numero finito di operazioni (ad esempio risolvere in modo esatto un sistema di equazioni lineari);
2. **Metodi iterativo**: risolvono i problemi tramite successive approssimazioni, che portano ad ottenere una soluzione anch'essa approssimata.

Il concetto di **approssimazione** è fondamentale nel calcolo numerico. Perché accontentarsi di una soluzione approssimata?

Perché alcuni problemi non hanno soluzioni analitiche (come il *problema dei tre corpi*, Figura 4), oppure perché calcolare la soluzione esatta richiederebbe troppo tempo (ad esempio calcolare il valore di una funzione molto complessa).

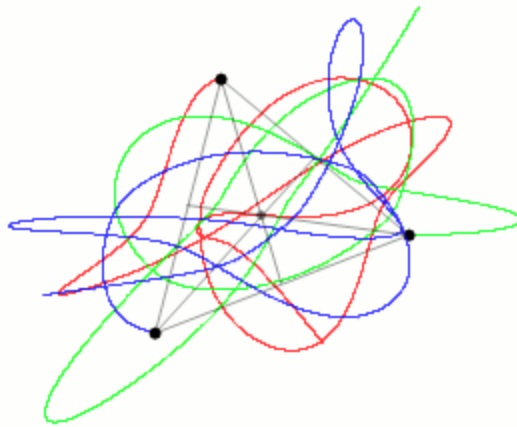


Figura 4: Approssimazione delle traiettorie di tre corpi soggetti all'attrazione gravitazionale reciproca (detto "problema dei tre corpi").

Credits: [Dntllthmmnm](#)

🔌 Il calcolo numerico nasce molto prima dell'avvento dei computer! Perché i metodi del calcolo numerico permettevano di facilitare i calcoli a mano. Per esempio, pensate ad Archimede: fu il primo a definire un metodo per approssimare scientificamente il valore di Pi greco.

Dopo questa breve descrizione di cos'è il calcolo numerico, quali potrebbero essere, secondo voi, le sue applicazioni?

Scrivetele qui: http://scrumblr.ca/applicazioni_del_calcolo_numerico0822

Le applicazioni del calcolo numerico

🪐 Astronomia

- Il calcolo numerico è fondamentale per la **ricostruzione delle immagini** di **telescopi** e **satelliti**. Questi strumenti, compresi i famosi *Hubble Space Telescope* (<https://hubblesite.org/>) e il più recente *Webb Space Telescope* (<https://webb.nasa.gov/>), producono immagini affette da **rumore** (perturbazioni nei segnali e nei dati). Questo rumore

deve essere rimosso (tramite metodi del calcolo numerico) per ottenere immagini che possono essere studiate dagli scienziati. [1]

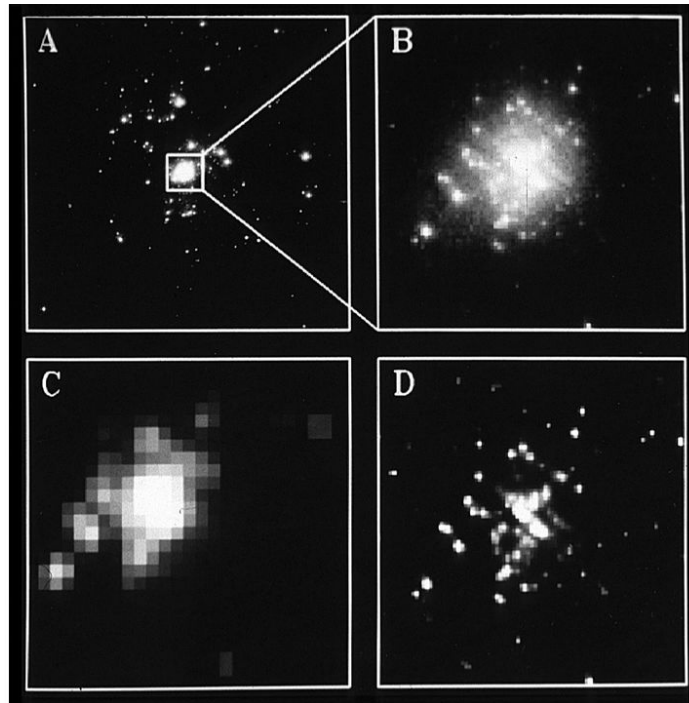
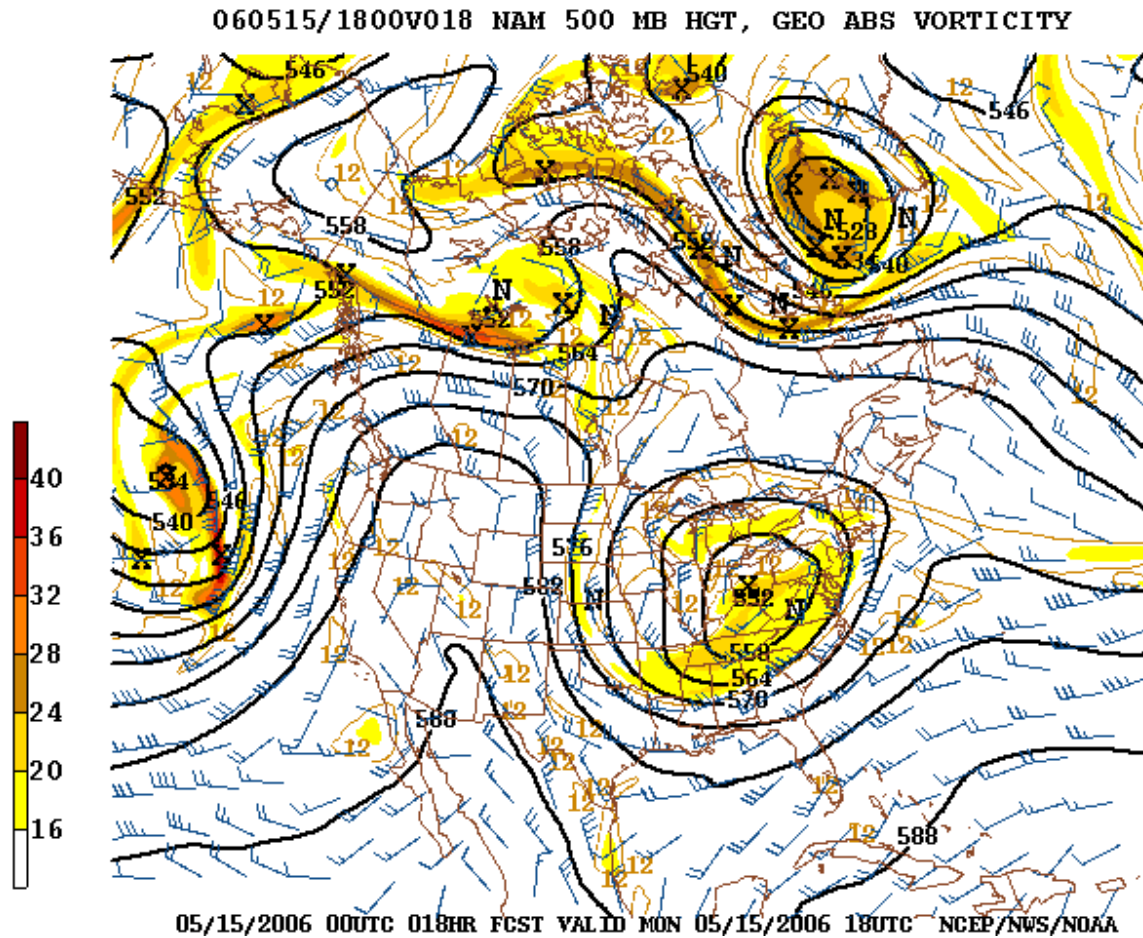


Immagine originale (A) tratta dalla telecamera *Wide Field / Planetary Camera* del telescopio spaziale Hubble, e il risultato di tre diverse tecniche di ricostruzione di immagini (B), (C), (D).

Credits: [NASA](#), [ESA](#), [STScI](#)

Meteorologia

- Le previsioni del tempo prevedono molti calcoli complicati, tra cui la risoluzione di equazioni differenziali (equazioni che lega una funzione incognita alle sue derivate) che avviene tramite **metodi di approssimazione numerica**, oggetto di studio del Calcolo Numerico.

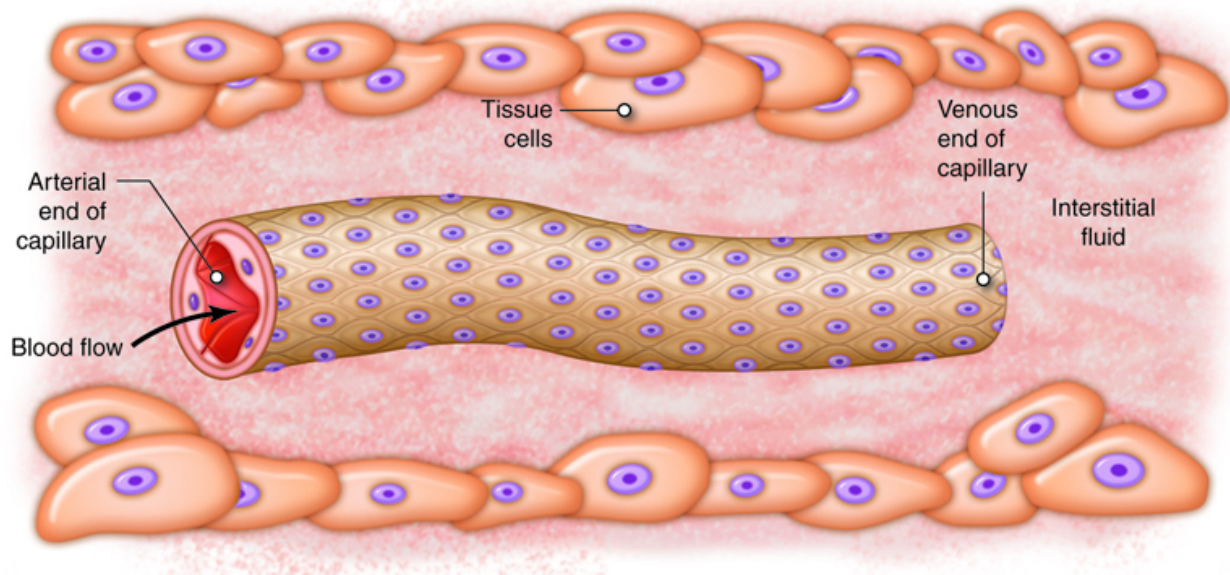


Carta meteorologica.

Credits: [NWS](#)

Biologia

- In modo simile a quanto avviene per le immagini astronomiche, le tecniche di calcolo numerico vengono utilizzate per la ricostruzione delle immagini provenienti da **microscopi a fluorescenza** (microscopi ottici ottici che permettono l'osservazione di campioni precedentemente marcati con una molecola fluorescente: https://it.wikipedia.org/wiki/Microscopio_a_fluorescenza).
- Inoltre, i metodi del calcolo numerico vengono utilizzati per modellare vari fenomeni nel campo della medicina, come il trasporto di sostanze chimiche nel corpo umano [2], o la dinamica di una popolazione di cellule staminali.

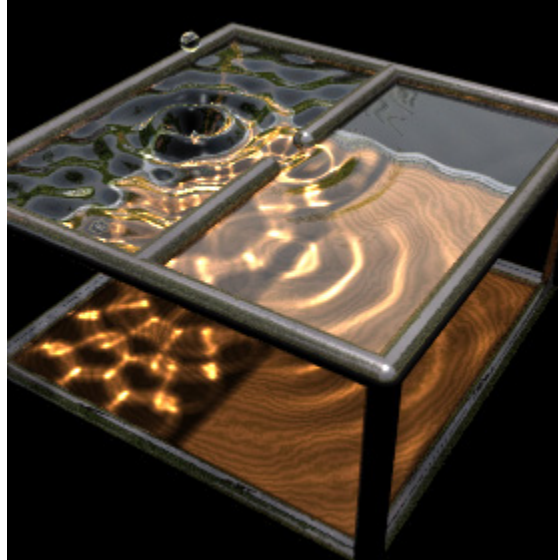


Flusso sanguigno capillare.

Credits: CCCS Open Textbooks

Simulazioni fisiche e computer grafica

- Le simulazioni di fenomeni fisici prevedono spesso l'utilizzo di equazioni differenziali. Il calcolo numerico mette a disposizione dei metodi per realizzare, tramite la **computer grafica, simulazioni scientifiche** che ci aiutano a capire i fenomeni del nostro mondo! Questi stessi metodi ci permettono anche di realizzare (e vedere) incredibili film di animazione



Simulazione dell'ondulazione dell'acqua.

Credits: [Paul Nylander](#)



Avete visto il film “***Il diritto di contare***” ([pagina Wikipedia](#))?

Se non lo avete ancora visto, consideratelo il compito per casa! Un bellissimo film che racconta una storia vera (con qualche licenza artistica) che tratta di matematica, calcolatori, integrazione e diritti femminili.

Sentirete nominare il metodo di Eulero, una procedura dell'**analisi numerica** per la risoluzione delle equazioni differenziali!

| Concludiamo la lezione 1:

Avete dei dubbi su quello che abbiamo visto in questa lezione? Quale argomento avete capito meglio? E quale peggio? **Scrivetelo qui:** http://scrumbler.ca/lezione01_calcolo_numerico0822

Immagine di testata: Webb Space Telescope (NASA, ESA, CSA, STScI)