

# Elementi di Bioinformatica

Gianluca Della Vedova

Univ. Milano-Bicocca  
<http://gianluca.dellavedova.org>

31 ottobre 2019

Allineamento

# Allineamento di 2 sequenze

## Allineamento

- Input: 2 sequenze  $s_1$  e  $s_2$
- Aggiunta di **indel** in  $s_1$  e  $s_2$
- sequenze estese = stessa lunghezza
- NO colonne di indel

# Allineamento: esempio

## Input

ABRACADABRA

BANANA

## sequenze allineate 1

ABRACADABRA

-B-ANA---NA

## sequenze allineate 2

ABR-AC-ADABRA

---B-ANA---NA

## sequenze allineate 3

ABRACADABRA

-BANA----NA

# Allineamento: costo o valore?

## Problema di ottimizzazione

- Istanza: insieme infinito di casi
- Soluzioni ammissibili: ammissibilità verificabile in tempo polinomiale
- Funzione obiettivo: soluzione ammissibile  $\mapsto \mathbb{Q}$
- Massimizzazione o minimizzazione

## costo o valore?

- Costo da minimizzare
- Valore da massimizzare

# Valore

## Valore di un allineamento

- Somma dei valori delle singole colonne

# Valore

## Valore di un allineamento

- Somma dei valori delle singole colonne
- Valore di una colonna =

# Valore

## Valore di un allineamento

- Somma dei valori delle singole colonne
- Valore di una colonna =
- valore in ingresso

# Valore

## Valore di un allineamento

- Somma dei valori delle singole colonne
- Valore di una colonna =
- valore in ingresso

## Istanza

- due sequenze  $s_1$  e  $s_2$
- matrice di score  $d : (\Sigma \cup \{-\}) \times (\Sigma \cup \{-\}) \mapsto \mathbb{Q}$  (normalmente interi)
- problema di massimizzazione = massima omologia



# Needleman-Wunsch: Equazione di ricorrenza

## Definizione

$M[i, j]$  = ottimo su  $s_1[:i]$ ,  $s_2[:j]$

$$M[i, j] = \max \begin{cases} M[i-1, j-1] + d(s_1[i], s_2[j]) \\ M[i, j-1] + d(-, s_2[j]) \\ M[i-1, j] + d(s_1[i], -) \end{cases}$$

## Condizione al contorno

- $M[0, 0] = 0$
- $M[i, 0] = M[i-1, 0] + d(s_1[i], -)$
- $M[0, j] = M[0, j-1] + d(-, s_2[j])$

# Allineamento locale

## Allineamento globale

Si allineano le sequenze intere

## Allineamento locale

- 1 Input:  $s_1, s_2$ , matrice di score  $d$
- 2 Individuare sottostringhe  $t_1$  di  $s_1$  e  $t_2$  di  $s_2$  tale che
- 3  $All[t_1, t_2] \geq All[u_1, u_2]$  per ogni coppia di sottostringhe  $u_1, u_2$  di  $s_1, s_2$ .
- 4 Algoritmo banale: calcolo tutte le sottostringhe di  $s_1, s_2$  e ne calcolo allineamento globale
- 5 Tempo  $O(n^3m^3)$

# Smith-Waterman

## Osservazione 1

- 1 Matrice  $M[i, j]$  memorizza allineamento di tutte le coppie di **prefissi** di  $s_1, s_2$
- 2 Allineamento massimo fra coppie di prefissi = valore massimo in  $M$

## Osservazione 2

- 1  $M[0, 0] = 0$
- 2 quindi non si prendono sottostringhe con allineamento negativo

# Equazione di ricorrenza

## Definizione

$M[i, j]$  = ottimo fra tutte le stringhe  $s_1[k : i]$ ,  $s_2[h : j]$

$$M[i, j] = \max \begin{cases} M[i-1, j-1] + d(s_1[i], s_2[j]) \\ M[i, j-1] + d(-, s_2[j]) \\ M[i-1, j] + d(s_1[i], -) \\ 0 \end{cases}$$

## Condizione al contorno

- $M[0, 0] = M[i, 0] = M[0, j] = 0$
- punto finale = valore massimo
- si risale nell'allineamento fino a uno 0.
- Tempo ( $nm$ )

# Gap

## Definizione

- 1 Sequenza contigua di indel in un **allineamento**

## Esempio

ABR-AC-ADABRA: 2 gap

---B-ANA---NA: 3 gap

## Osservazione

- 1 Un gap sposta il frame di lettura
- 2 no indel  $\neq$  1 indel  
item 1 indel  $\approx$  2 indel

# Allineamento con gap generici

- costo gap lungo  $l$ :  $P(l)$

# Allineamento con gap generici

- costo gap lungo  $l$ :  $P(l)$
- Come descrivo l'allineamento ottimo?

# Allineamento con gap generici

- costo gap lungo  $l$ :  $P(l)$
- Come descrivo l'allineamento ottimo?
- Come è fatta l'ultima colonna?



# Allineamento con gap generici

- costo gap lungo  $l$ :  $P(l)$
- Come descrivo l'allineamento ottimo?
- Come è fatta l'ultima colonna?
- Come è fatto l'ultimo gap?

# Gap generico

## Definizione

$M[i, j] = \text{ottimo su } s_1[:i], s_2[:j]$

$$M[i, j] = \max \begin{cases} M[i-1, j-1] + d(s_1[i], s_2[j]) \\ \max_{l>0} M[i, j-l] + P(l) \\ \max_{l>0} M[i-l, j] + P(l) \end{cases}$$

## Condizione al contorno

- $M[0, 0] = 0$

# Gap generico

## Definizione

$M[i, j] = \text{ottimo su } s_1[:i], s_2[:j]$

$$M[i, j] = \max \begin{cases} M[i-1, j-1] + d(s_1[i], s_2[j]) & \text{no gap} \\ \max_{l>0} M[i, j-l] + P(l) & \text{gap in } s_1 \\ \max_{l>0} M[i-l, j] + P(l) & \text{gap in } s_2 \end{cases}$$

## Condizione al contorno

- $M[0, 0] = 0$
- $M[i, 0] = P(i), M[0, j] = P(j)$

# Gap generico

## Definizione

$M[i, j] = \text{ottimo su } s_1[:i], s_2[:j]$

$$M[i, j] = \max \begin{cases} M[i-1, j-1] + d(s_1[i], s_2[j]) \\ \max_{l>0} M[i, j-l] + P(l) \\ \max_{l>0} M[i-l, j] + P(l) \end{cases}$$

## Condizione al contorno

- $M[0, 0] = 0$
- $M[i, 0] = P(i), M[0, j] = P(j)$
- Tempo  $O(nm(n+m))$

# Allineamento con gap affine

- costo gap lungo  $l$ :  $P_o + lP_e$

# Allineamento con gap affine

- costo gap lungo  $l$ :  $P_o + lP_e$
- $P_o$ : costo apertura gap

# Allineamento con gap affine

- costo gap lungo  $l$ :  $P_o + lP_e$
- $P_o$ : costo apertura gap
- $P_e$ : costo estensione gap

# Allineamento con gap affine

- costo gap lungo  $l$ :  $P_o + lP_e$
- $P_o$ : costo apertura gap
- $P_e$ : costo estensione gap
- $P_e, P_o > 0$



# Allineamento con gap affine

- costo gap lungo  $l$ :  $P_o + lP_e$
- $P_o$ : costo apertura gap
- $P_e$ : costo estensione gap
- $P_e, P_o > 0$
- Come descrivo l'allineamento ottimo?

# Allineamento con gap affine

- costo gap lungo  $l$ :  $P_o + lP_e$
- $P_o$ : costo apertura gap
- $P_e$ : costo estensione gap
- $P_e, P_o > 0$
- Come descrivo l'allineamento ottimo?
- Come è fatta l'ultima colonna?

# Allineamento con gap affine

- costo gap lungo  $l$ :  $P_o + lP_e$
- $P_o$ : costo apertura gap
- $P_e$ : costo estensione gap
- $P_e, P_o > 0$
- Come descrivo l'allineamento ottimo?
- Come è fatta l'ultima colonna?
- Come è fatto l'ultimo gap?

# Gap affine

## Definizione

- $M[i, j] =$  ottimo su  $s_1[:i], s_2[:j]$
- $E_1[i, j] =$  ottimo su  $s_1[:i], s_2[:j]$ , con estensione di gap finale in  $s_1$
- $E_2[i, j] =$  ottimo su  $s_1[:i], s_2[:j]$ , con estensione di gap finale in  $s_2$
- $N_1[i, j] =$  ottimo su  $s_1[:i], s_2[:j]$ , con apertura di gap alla fine di  $s_1$
- $N_2[i, j] =$  ottimo su  $s_1[:i], s_2[:j]$ , con apertura di gap alla fine di  $s_1$

# Gap affine

$$M[i, j] = \max \begin{cases} M[i-1, j-1] + d(s_1[i], s_2[j]) \\ E_1[i, j], E_2[i, j] \\ N_1[i, j], N_2[i, j] \end{cases}$$

$$E_1[i, j] = \max \begin{cases} E_1[i, j-1] + P_e \\ N_1[i, j-1] + P_e \end{cases}$$

$$E_2[i, j] = \max \begin{cases} E_2[i-1, j] + P_e \\ N_2[i-1, j] + P_e \end{cases}$$

$$N_1[i, j] = M[i, j-1] + P_o + P_e, \quad N_2[i, j] = M[i-1, j] + P_o + P_e$$

# Allineamento multiplo

$k$  sequenze

- Input: insieme di sequenze  $\{s_1, \dots, s_k\}$

# Allineamento multiplo

## $k$ sequenze

- Input: insieme di sequenze  $\{s_1, \dots, s_k\}$
- Aggiunta di **indel** nelle sequenze

# Allineamento multiplo

## $k$ sequenze

- Input: insieme di sequenze  $\{s_1, \dots, s_k\}$
- Aggiunta di **indel** nelle sequenze
- sequenze estese = tutte stessa lunghezza



# Allineamento multiplo

## $k$ sequenze

- Input: insieme di sequenze  $\{s_1, \dots, s_k\}$
- Aggiunta di **indel** nelle sequenze
- sequenze estese = tutte stessa lunghezza
- NO colonne di indel

# Valore di una colonna

## SP: sum of pairs

- $\{s_1, \dots, s_k\} \mapsto \{s_1^*, \dots, s_k^*\}$  allineate

## Complessità

# Valore di una colonna

## SP: sum of pairs

- $\{s_1, \dots, s_k\} \mapsto \{s_1^*, \dots, s_k^*\}$  allineate
- Valore  $\{s_1^*[h], \dots, s_k^*[h]\}$

## Complessità

# Valore di una colonna

## SP: sum of pairs

- $\{s_1, \dots, s_k\} \mapsto \{s_1^*, \dots, s_k^*\}$  allineate
- Valore  $\{s_1^*[h], \dots, s_k^*[h]\}$
- $\sum_{i < j} d(s_1^*[i], s_k^*[j])$

## Complessità

# Valore di una colonna

## SP: sum of pairs

- $\{s_1, \dots, s_k\} \mapsto \{s_1^*, \dots, s_k^*\}$  allineate
- Valore  $\{s_1^*[h], \dots, s_k^*[h]\}$
- $\sum_{i < j} d(s_i^*[i], s_k^*[j])$

## Complessità

- se  $k$  è arbitrario  $\Rightarrow$  NP-completo

# Valore di una colonna

## SP: sum of pairs

- $\{s_1, \dots, s_k\} \mapsto \{s_1^*, \dots, s_k^*\}$  allineate
- Valore  $\{s_1^*[h], \dots, s_k^*[h]\}$
- $\sum_{i < j} d(s_i^*[i], s_j^*[j])$

## Complessità

- se  $k$  è arbitrario  $\Rightarrow$  NP-completo
- se  $k$  è fissato  $\Rightarrow$  tempo  $O(n^k)$

# Matrici di sostituzione

- 1 Utilizzate per valutare un allineamento
- 2 Implicitamente probabilità di transizione
- 3 Mutazioni ricorrenti
- 4 Allineamenti di proteine

# PAM: unità di misura

- 1 PAM: point/percent accepted mutation
- 2 due sequenze  $s_1$  e  $s_2$ : quanto sono distanti?
- 3 distanza 1PAM  $\Rightarrow$  numero mutazioni =  $\frac{1}{100}|s_1|$
- 4 semplice in assenza di indel
- 5 Mutazioni ricorrenti  $\Rightarrow$  misura affidabile solo per piccoli valori
- 6  $s_1$  e  $s_2$  distanti 100 PAM  $\Rightarrow$  una singola base ha 36% di probabilità di non essere mutata



# Matrici PAM

- 1 dipende dalla distanza attesa
- 2 PAM250, PAM200, PAM1

## Calcolo PAM $k$

- 1 Costruzione PAM $k$
- 2 Si prendono varie sequenze distanti  $k$ PAM
- 3 si allineano le sequenze
- 4 si calcolano le frequenze  $f(i)$ ,  $f(i, j)$  di tutti i singoli caratteri e le coppie di caratteri
- 5 
$$\text{PAM}k(i, j) = \log \frac{f(i, j)}{f(i)f(j)}$$

# Log odds ratio

## Odds ratio

- 1  $\frac{p}{1-p}$ ,  $p$  è la probabilità dell'evento interessante (target)
- 2  $\frac{f(i,j)}{f(i)f(j)}$
- 3  $f(i,j)$ : frequenza della mutazione misurata
- 4  $f(i)f(j)$ : ipotesi nulla (caratteri indipendenti)

# Matrici PAM

## Calcolo PAM $k$ nella realtà

- Problema: come allineare se non si conosce la matrice
- Allineate sequenze molto simili
- no indel
- $M_k(i, j) = \log \frac{f(i)M_1^k(i, j)}{f(i)f(j)} = \log \frac{M_1^k(i, j)}{f(j)}$
- valori moltiplicati per 10
- arrotondati all'intero più vicino
- si somma un intero a tutti i valori

# Matrici BLOSUM

## Confronto con PAM

- PAM allinea sequenze vicine
- ma viene usata per allineare sequenze lontane
- regioni conservate e non conservate hanno stessa importanza

## BLOCKS

- blocchi di regioni conservate
- scelte “a mano”
- $B(i, j) = \log \frac{f(i, j)}{f(i)f(j)}$

# Matrici BLOSUM

## BLOSUM<sub>x</sub>

- le sequenze che sono simili più di  $x\%$  vengono clusterizzate
- cluster = rimuovere tutte tranne una
- scopo: evitare di sovrappesare parti sovrarappresentate nel campione
- BLOSUM62: più usata per gli allineamenti

# Statistiche Karlin-Altschul

## Ricerca in un database

- Punteggio positivo possibile
- Punteggio medio negativo
- Simboli indipendenti e equiprobabili
- Sequenze infinitamente lunghe
- Allineamenti senza gap

# Equazione Karlin-Altschul

$$E = kmne^{-\lambda S}$$

- $E$ : numero allineamenti
- $k$ : costante
- $n$ : numero caratteri in database
- $m$ : lunghezza stringa query
- $\lambda S$ : punteggio normalizzato

# BLAST

## Basic Local Alignment Search Tool

- Ricerca seed
- seed = pattern matching con sottostringa di lunghezza 3
- Costruzione high-scoring segment pair (HSP) = estensione seed
- Filtro seed tenuti solo HSP con alta significatività
- Fusione HSP vicine
- Smith-Waterman sulle regioni



# Licenza d'uso

Quest'opera è soggetta alla licenza Creative Commons: Attribuzione-Condividi allo stesso modo 4.0. (<https://creativecommons.org/licenses/by-sa/4.0/>).

Sei libero di riprodurre, distribuire, comunicare al pubblico, esporre in pubblico, rappresentare, eseguire, recitare e modificare quest'opera alle seguenti condizioni:

- **Attribuzione** — Devi attribuire la paternità dell'opera nei modi indicati dall'autore o da chi ti ha dato l'opera in licenza e in modo tale da non suggerire che essi avallino te o il modo in cui tu usi l'opera.
- **Condividi allo stesso modo** — Se alteri o trasformi quest'opera, o se la usi per crearne un'altra, puoi distribuire l'opera risultante solo con una licenza identica o equivalente a questa.