Alignement de deux séquences

Une petite faute

Remarque : La visite de suivi effectuée par l'auditeur/l'inspecteur responsable à la suite d'une non-conformité majeure doit avoir lieu le plus rapidement possible après la « date d'exécution de l'action corrective » indiquée à la partie B de la DAC (voir la section 6.3.3 du présent docment).

Lorsque l'auditeur/l'inspecteur responsable constate que l'action corrective a été exécutée et qu'elle est efficace, la DAC peut être classée.

(Source: Agence canadienne d'inspection des aliments. Manuel de mise oeuvre du Programme d'amélioration de la salubrité des aliments.)

«docment» : document, dorment, doucement, . . .

Distance entre deux mots

Distance d'édition : nombre d'opérations pour transformer un mot à un autre.

Opérations : sur caractères
- insertion (I *insert*)
- suppression (D *delete*)
- substitution (S *substitute*)
- identité (M *match*)

Calcul de distance

But : calculer le nombre *minimal* d'opérations pour la transformation.

Minimal : autant de M (identité) que possible.

Exemples : docment-document, docment-dorment, docment-doucement, docment-moments

On peut avoir plusieurs suites de la même taille minimale : abbc-axc

Calcul de distance 2

Idée : calculer les distances entre les préfixes successivement

Nos notions formelles :
- alphabet fini Σ (p.e. $\Sigma = \{\mathtt{a}, \ldots, \mathtt{z}\}$ ou $\Sigma = \{\mathtt{T}, \mathtt{G}, \mathtt{A}, \mathtt{C}\}$)
- mot ou séquence : une suite de caractères de Σ

Soit S une séquence.
- taille de S, denotée par $|S|$ = nombre de caractères
- caractère en position i : $S[i]$
- sous-mot $S[i..j]$: le mot formé par les caractères $S[i]$, $S[i+1]$,..., $S[j]$
- préfixe : sous-mot de forme $S[1..i]$.

Calcul de distance 3

Def. Distance d'édition entre deux séquences S_1 et S_2 : nombre minimal de I, D, S dans une suite d'opérations qui transforme S_1 en S_2.

Thm. La distance d'édition est une fonction symétrique.
Preuve. I $\Leftrightarrow$ D.

Un fait intéressant :) la distance d'édition est parfois appellée distance de Levenshtein.

Calcul de distance 4

Def. Soit S et T deux séquences. On définit $D(i,j)$ par

$$D(i,j) = \begin{cases} \text{distance entre } S[1..i] \text{ et } T[1..j] & \text{si } i > 0 \text{ et } j > 0, \\ i & \text{si } j = 0 \\ j & \text{si } i = 0. \end{cases}$$

Donc $D(i,j)$ est la distance entre les deux préfixes de tailles i et j.

Thm. Si $i, j > 0$, on a

$$D(i,j) = \min\Big\{ D(i-1,j) + 1, \\ D(i,j-1) + 1, \\ D(i-1,j-1) + \{S[i] \neq T[j]\} \Big\}.$$

Calcul de distance 5

Preuve. 1. La dernière opération pour achever $D(i, j)$ doit être I, D, ou S/M.
2. Tous les trois sont possibles.

Donc on peut calculer $D(|S|, |T|)$ par récursion. . . pas une bonne idée. (La même valeur sera calculée plusieurs fois.)

Quand même on n'a que $(1 + |S|) \times (1 + |T|)$ appels récursifs possibles.

Au lieu d'explorer l'arbre de récursions en descendant, il est mieux de le faire de manière ascendante.

Tableau de calculs

Les cases contiennent les $D(i, j)$.

Parcours : ligne par ligne.

Exemple : $\texttt{AAAC} \rightarrow \texttt{AGC}$

Temps de calcul : $O(mn)$ (on remplit chaque case du tableau en un temps constant : trois comparaisons et deux ou trois additions)

Comment trouver une suite d'opérations qui correspond à $D(i, j)$? Enregistrer dans chaque case la direction du $\min$ de la recurrence.

Programmation dynamique

PD : récurrence+tableau+retrouver la solution optimale (en retraçant les pas à partir du case Sud-Est)

PD est utilisée pour resoudre des problèmes d'optimisation

1. sous-structures optimales ($\Rightarrow$ récurrence)
2. sous-problèmes superposés ($\Rightarrow$ tableau)

Mais pourquoi ça nous intéresse ?

L'alignement de séquences est une méthode utilisée très souvent en biologie moléculaire d'aujourd'hui.

Idée : similarité de séquences $\Rightarrow$ similarité de structures et fonctions (évolution de gènes/protéines : duplication+modification)

Régions conservées : importance pour la fonction/structure.

Exemple

protéine trypsine : souris (P07146 de SWISS-PROT) et grenouille (P70059 de SWISS-PROT)

```
souris      MSALLILALVGAAVAFPVDDDDKIVGGYTCRESSVPYQVSLNAGYHFCGGSLINDQWVVSAAHC
grenouille  MKFLVILVLLGAAVAFEDDD--KIVGGFTCAKNAVPYQVSLNAGYHFCGGSLINSQWVVSAAHC
```

Alignement des deux sequences : représente leur similarité.

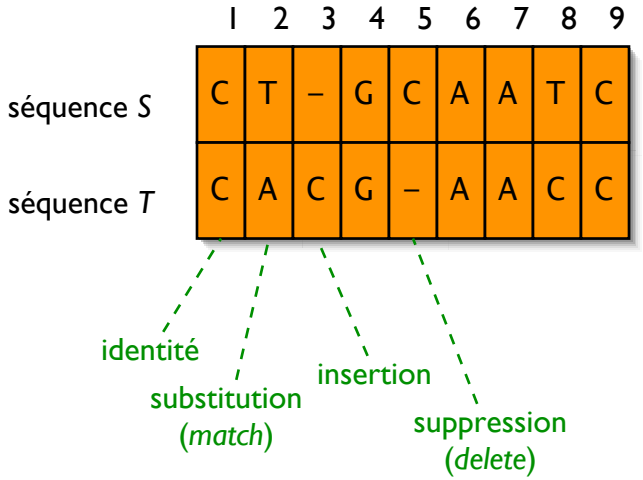
Alphabet d'alignement : $\Sigma \cup \{-\}$.
Appariement : un couple de $\Sigma \cup \{-\}$ (p.e. $(\mathtt{C}, -)$, ou $(\mathtt{C}, \mathtt{G})$)
Alignement : suite d'appariements.

Relation entre opérations d'éditions et alignements (procédure et produit).

Alignement

Déf. alignement=vecteur d'appariements

Types d'appariements :
- (a, a) : occurence (*match*)
- (a, b) : erreur (*mismatch/substitution*)
- $(a, -)$ et $(-, a)$: indel



Alignement pondéré

Tableau $\mathbf{C}$ de pondération d'appariements

	A	C	G	T	–
A	1	−3	−3	−3	−5
C	−3	1	−3	−3	−5
G	−3	−3	1	−3	−5
T	−3	−3	−3	1	−5
–	−5	−5	−5	−5	X

ou :

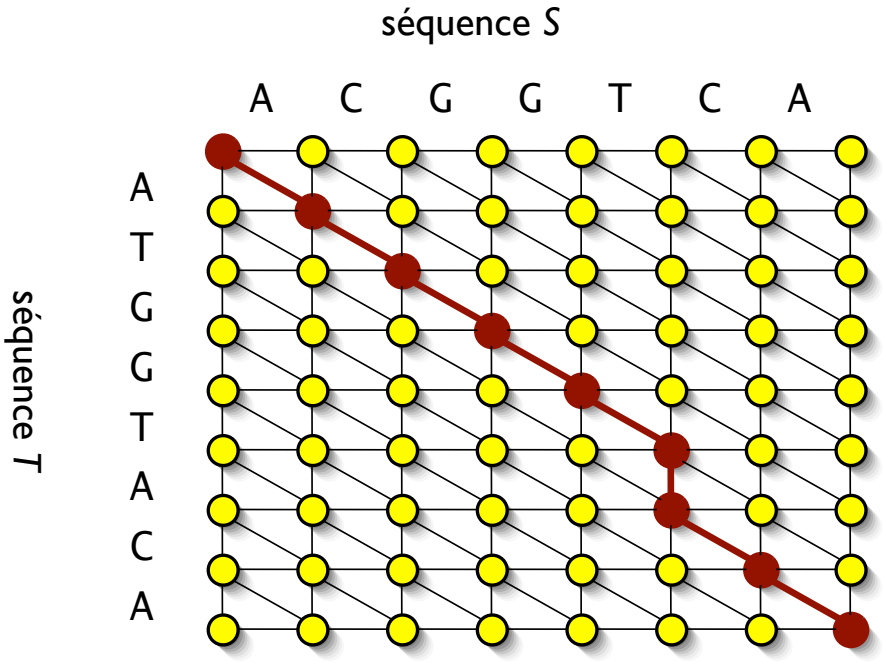
	A	C	G	T	–
A	91	−114	−31	−123	−300
C	−114	100	−125	−31	−300
G	−31	−125	100	−114	−300
T	−123	−31	−114	91	−300
–	−300	−300	−300	−300	

Score ou valeur de l'alignement : somme des valeurs des appariements

Problème : trouver l'alignement avec le score maximal

PD pour maximiser la similarité

Graphe d'édition



ACGGT-CA
ATGGTACA

Graphe d'édition 2

pondération des arêtes :

$$\begin{aligned}
\mathsf{poids}(v_{i-1,j-1}v_{i,j}) &= \mathbf{C}\Big[S[j], T[i]\Big];\\
\mathsf{poids}(v_{i-1,j}v_{i,j}) &= \mathbf{C}\Big[-, T[i]\Big];\\
\mathsf{poids}(v_{i,j-1}v_{i,j}) &= \mathbf{C}\Big[S[j], -\Big].
\end{aligned}$$

alignement global : trouver le chemin de $v_{0,0}$ à $v_{|T|,|S|}$ avec poids maximal.

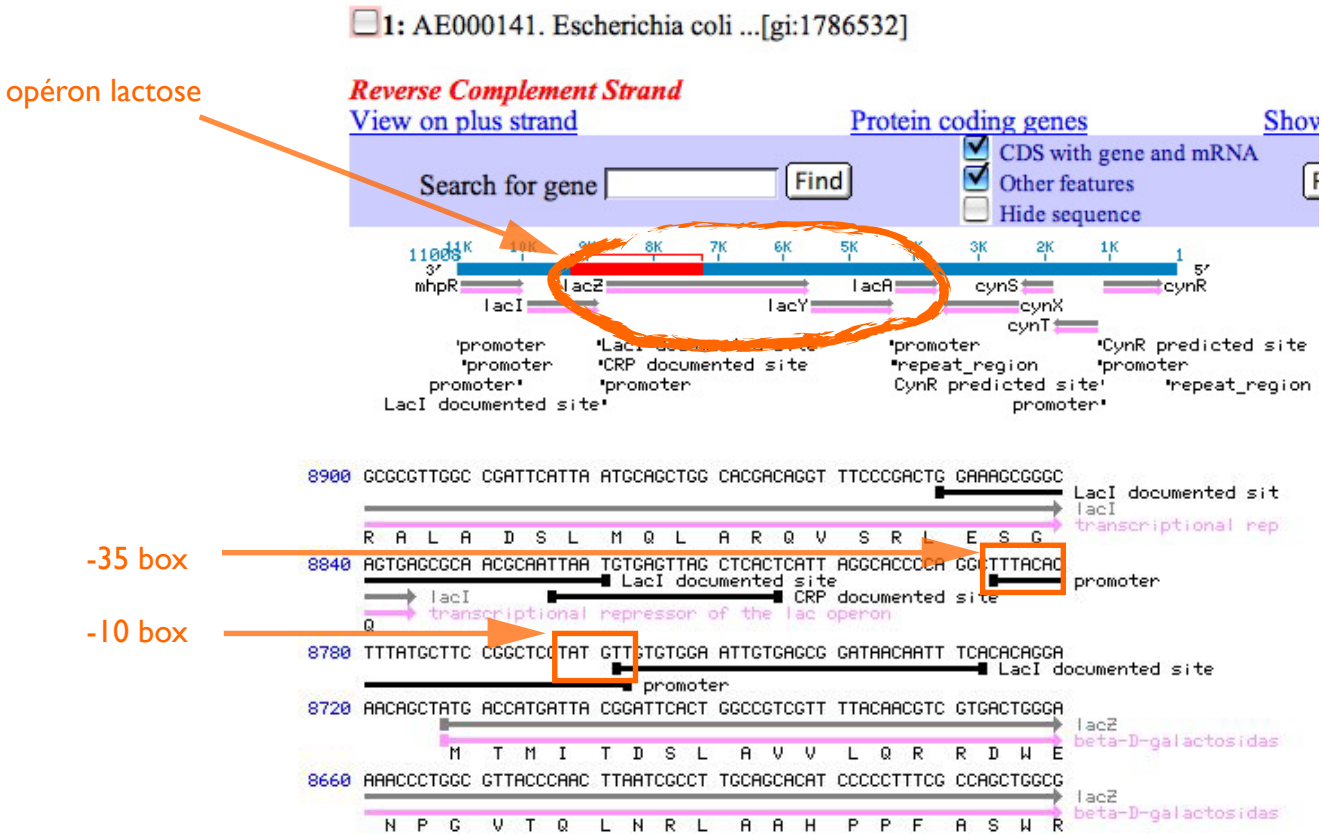
Alignement : variation 1

Trouver l'occurrence plus similaire d'une (courte) séquence dans une autre (longue)

Idée : calculer l'alignement pour les suffixes

Exemple 1 : séquences répétées (p.e. de type retrovirus)

Exemple 2 : promoteur sigma A : TATAAT...TTGACA



Alignement : variation 2

Trouver la région de similarité maximale entre deux séquences — alignement local

Exemple : domaines conservés (p.e. homeobox)

Les noms :
Needleman-Wunsch : problème de l'alignement global
Smith-Waterman : algorithme de l'alignement local

Alignement : variation 3

Exemple 1 : Alignement de ADNg et ADNc
(ADNc : ADN complémentaire pour ARNm transcrit)

Exemple 2 : pseudogènes

Interprétation probabiliste

probabilité d'identité : p ; probabilité de substitution : q ; probabilité d'indel r.

Alignement avec n_{id} identités, n_{sub} substitutions et n_{indel} indels : probabilité $P = p^{n_{\text{id}}} q^{n_{\text{sub}}} r^{n_{\text{indel}}}$.

log-vraisemblance : $\log_2 P = n_{\text{id}}(\log_2 p) + n_{\text{sub}}(\log_2 q) + n_{\text{indel}}(\log_2 r)$.

Soit $s = p/2$. Alors
$\log_2 P = n_{\text{id}} - \mu n_{\text{sub}} - \delta n_{\text{indel}} + (\log_2 s)(n_{\text{id}} + n_{\text{sub}} + \frac{n_{\text{indel}}}{2})$,
avec $\mu = \log_2 \frac{s}{q}$ et $\delta = \log_2 \frac{r}{\sqrt{s}}$.
Or, $(n_{\text{id}} + n_{\text{sub}} + \frac{n_{\text{indel}}}{2}) = \frac{|S|+|T|}{2}$, une constante.
On cherche l'alignement de S et T qui maximise $n_{\text{id}} - \mu n_{\text{sub}} - \delta n_{\text{indel}}$ où μ et δ correspondent à la pondération de subsitutions et indels.

Génération d'une matrice de pondération

Modèle probabiliste pour une séquence : caractères aléatoires iid (indépendants et identiquement distribués) : $\mathbb{P}\{S[i] = \sigma\} = \pi_\sigma$

P.e., $\pi_{\mathtt{A}} = \pi_{\mathtt{T}} = 32\%$, $\pi_{\mathtt{C}} = \pi_{\mathtt{G}} = 18\%$: taux de $(\mathtt{G} + \mathtt{C})$ à 36%.

Modèle probabiliste pour évolution $S \to T$: substitutions aléatoires indépendantes : $\mathbb{P}\Big\{T[i] = \sigma' \;\Big|\; S[i] = \sigma\Big\} = p_{\sigma \to \sigma'}$ (pas de trous !)

Matrice de subsitutions : $\mathbf{M} = \Big[p_{\sigma \to \sigma'}\Big]_{\sigma, \sigma' \in \Sigma}$

Matrice de pondération 2

Probabilité d'un «vrai» alignement : $P_1 = \mathbb{P}\{S, T\}$

Probabilité d'un alignement au hasard : $P_0 = \mathbb{P}\{S\}\mathbb{P}\{T\}$

$$P_1 = \prod_{i=1}^{n} \mathbb{P}\{S[i], T[i]\} = \prod_{i=1}^{n} \pi_{S[i]} p_{S[i] \to T[i]};$$

$$P_0 = \prod_{i=1}^{n} \mathbb{P}\{S[i]\}\mathbb{P}\{T[i]\} = \prod_{i=1}^{n} \pi_{S[i]} \pi_{T[i]}.$$

Matrice de pondération 3

Rapport de P_0 et P_1 : LODS (logarithmes des chances) $\log \frac{P_1}{P_0}$.

On a

$$\mathrm{LODS} = \sum_{i=1}^{n} \log \frac{\pi_{S[i]} p_{S[i] \to T[i]}}{\pi_{S[i]} \pi_{T[i]}} = \sum_{i=1}^{n} \log \frac{p_{S[i] \to T[i]}}{\pi_{T[i]}}.$$

Score de substitution $\sigma \to \sigma'$:

$$\mathrm{C}[\sigma, \sigma'] = \left\lceil \alpha \log \frac{p_{\sigma \to \sigma'}}{\pi_{\sigma'}} \right\rceil,$$

où α est une facteur d'échelle (notez qu'on utilise le plafond pour valeurs entières)

Matrice de pondération 4

Problème : comment estimer π_σ et $p_{\sigma \to \sigma'}$?

Solution : prendre de «bons alignements»

Exemples : BLOSUMnn (nn=45,62,80 pour nn% d'identité) ;
PAMxxx (xxx=100,250,. . . , mesure divergence) ;
Chiaromonte (pour ADN)

Trous

Pondérations de trou par sa taille :
- quelconque : $\delta(\text{longueur})$
- constante
- linéaire : $\delta(\ell) = \delta_{\text{ouvrir}} + \ell\delta_{\text{cont}}$

Récurrence pour pondération «quelconque» :

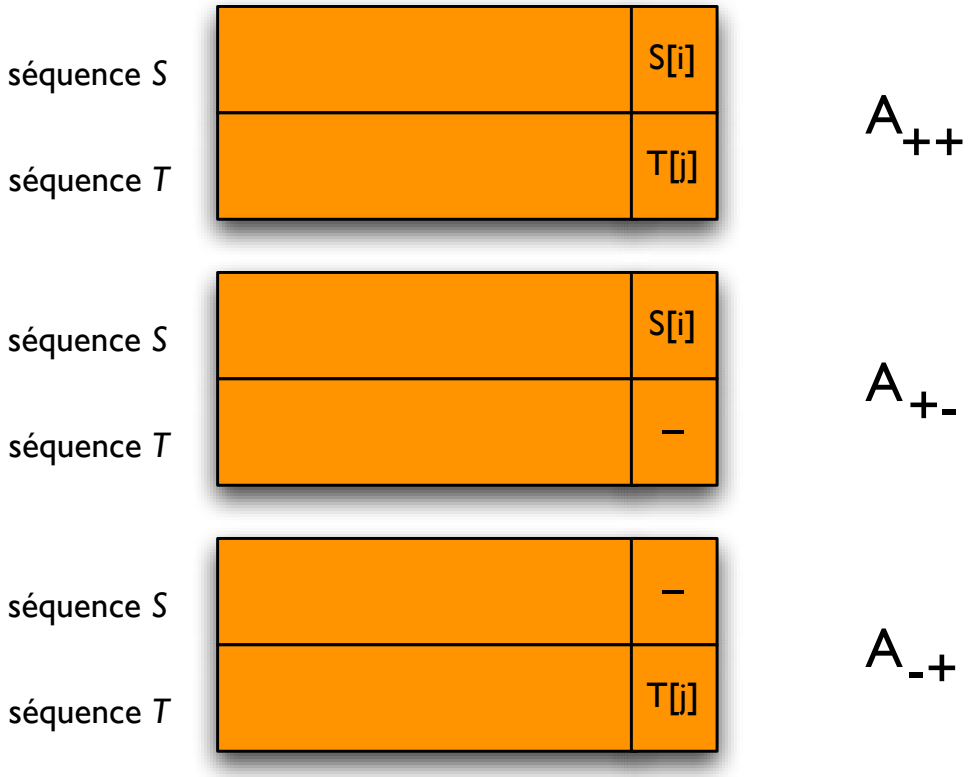
$$A(i,j) = \max\Big\{ A(i-1,j-1) + \mathbf{C}[S[i],T[j]],$$
$$\max_{\ell=1,\ldots,|S|}\{A(i-\ell,j)+\delta(\ell)\}, \max_{\ell=1,\ldots,|T|}\{A(i,j-\ell)+\delta(\ell)\}\Big\}.$$

Cas de base : $A(i,0) = \delta(i)$; $A(0,j) = \delta(j)$.

Temps de calcul : $O(nm^2 + mn^2)$ pour $|S| = n, |T| = m$.

Pondération linéaire

Récurrences pour $A^{+-}(i,j)$, $A^{-+}(i,j)$ et $A^{++}(i,j)$



Pondération linéaire 2

$$A^{++}(i,j) = \mathrm{C}[S[i], T[j]] + \max\Big\{ A^{++}(i-1, j-1),$$
$$A^{+-}(i-1, j-1), A^{-+}(i-1, j-1) \Big\};$$
$$A^{+-}(i,j) = \max\Big\{ A^{+-}(i-1, j) + \delta_1, A^{++}(i-1, j) + \delta_0 \Big\};$$
$$A^{-+}(i,j) = \max\Big\{ A^{-+}(i, j-1) + \delta_1, A^{++}(i, j-1) + \delta_0 \Big\};$$
$$A(i,j) = \max\Big\{ A^{++}(i,j), A^{-+}(i,j), A^{+-}(i,j) \Big\},$$

où $\delta_1 = \delta_{\mathsf{cont}}$ et $\delta_0 = \delta_{\mathsf{ouvrir}} + \delta_{\mathsf{cont}}$.

[on ignore $A^{+-}(i,j) = A^{-+}(i-1, j) + \delta_1$ ici]

Pondération linéaire 3

En fait, on peut éliminer A^{++} :

$$A^{+-}(i,j) = \max\Big\{A^{+-}(i-1,j) + \delta_1, A(i-1,j) + \delta_0\Big\};$$

$$A^{-+}(i,j) = \max\Big\{A^{-+}(i,j-1) + \delta_1, A(i,j-1) + \delta_0\Big\};$$

$$A(i,j) = \mathrm{C}[S[i],T[j]] + \max\Big\{A(i-1,j-1), A^{+-}(i,j), A^{-+}(i,j)\Big\}.$$

Cas de base : $A(0,0) = A^{+-}(0,0) = A^{-+}(0,0) = 0; A(i,0) = A^{+-}(i,0) = \delta_0 + (i-1)\delta_1; A(0,j) = A^{-+}(0,j) = \delta_0 + (j-1)\delta_1.$

Pondération linéaire 4