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.

Thm. La distance d'édition satisfait l'inégalité du triangle

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.

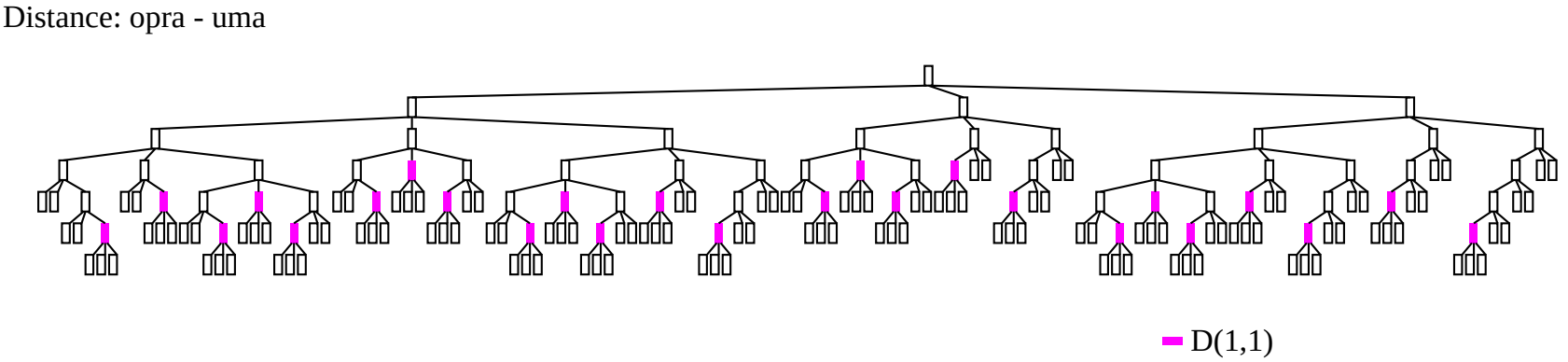
Thm. Si $i, j > 0$, on a

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

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 : `AAAC` $\rightarrow$ `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.

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)

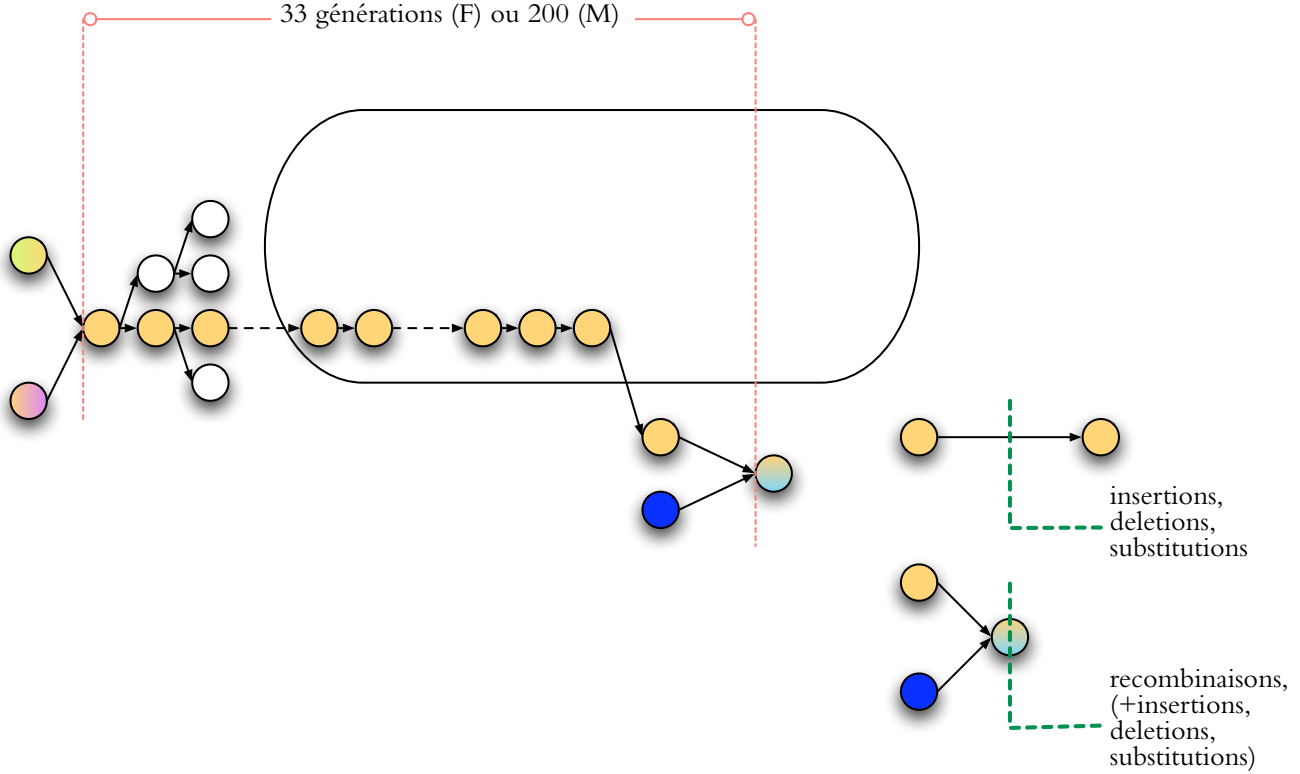
```
souris      MSALLILALVGAAVAFPVDDDDKIVGGYTCRESSVPYQVSLNAGYHFCGGSLINDQWVVSAAHC
grenouille  MKFLVILVLLGAAVAFEDDD--KIVGGFTCAKNAVPYQVSLNAGYHFCGGSLINSQWVVSAAHC
```

Alignement des deux séquences : 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).

MUTATIONS



les mutations s'accumulent pendant les années → divergence de séquences d'origin commun

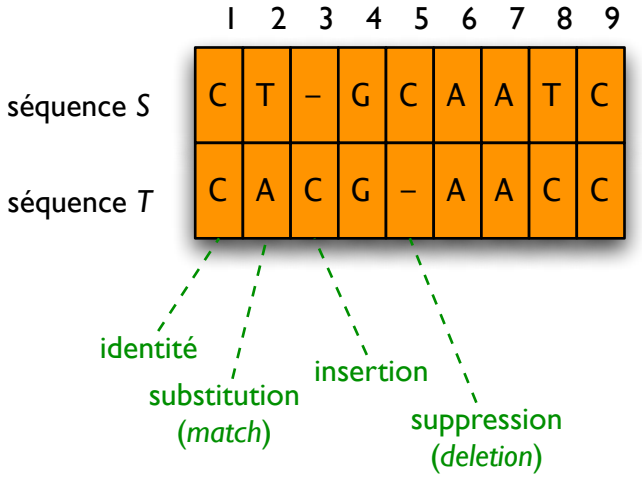
grand divergence ⇒ plus de temps et/ou moins de sélection négative

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 **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

Chiaromonte et al. PSB 2002.

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

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

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

$$\mathbf{C}\begin{bmatrix} \sigma \\ \sigma' \end{bmatrix} = \left\lceil \alpha \log \frac{p_{\sigma \to \sigma'}}{\pi_{\sigma'}} \right\rceil,$$

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

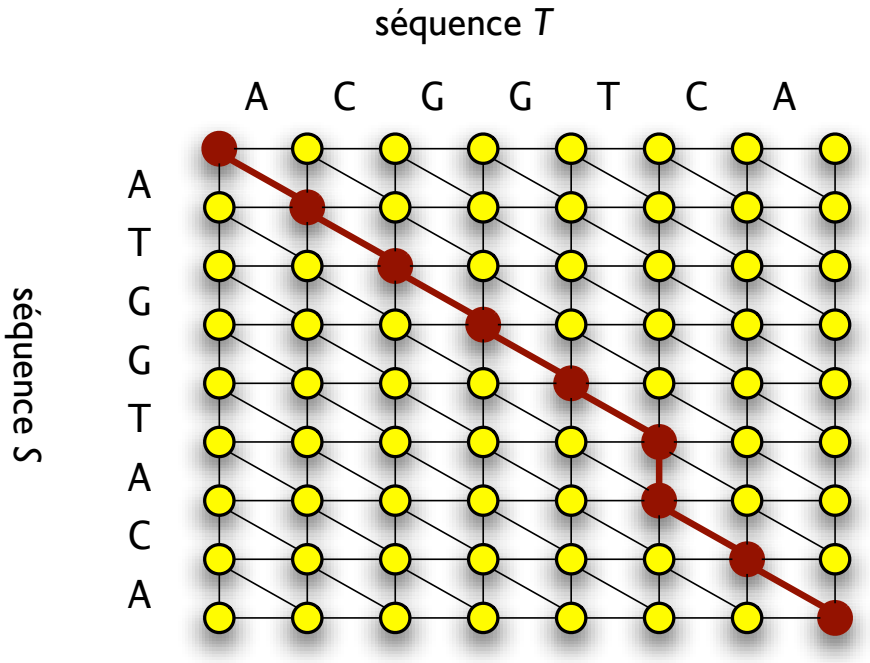
Matrice de pondération 4

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, protéines) ;
PAMxxx (xxx=100,250,. . . , mesure divergence, protéines) ;
Chiaromonte (pour ADN)

Graphe d'édition



ACGGT–CA
ATGGTACA

Graphe d'édition 2

pondération des arêtes :

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

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

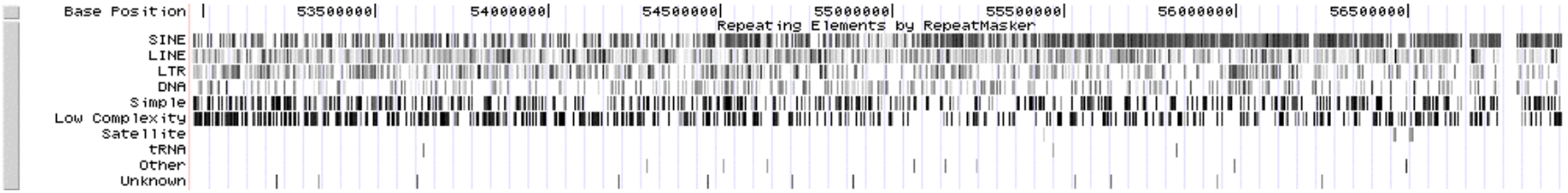
Alignement : variation 1

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

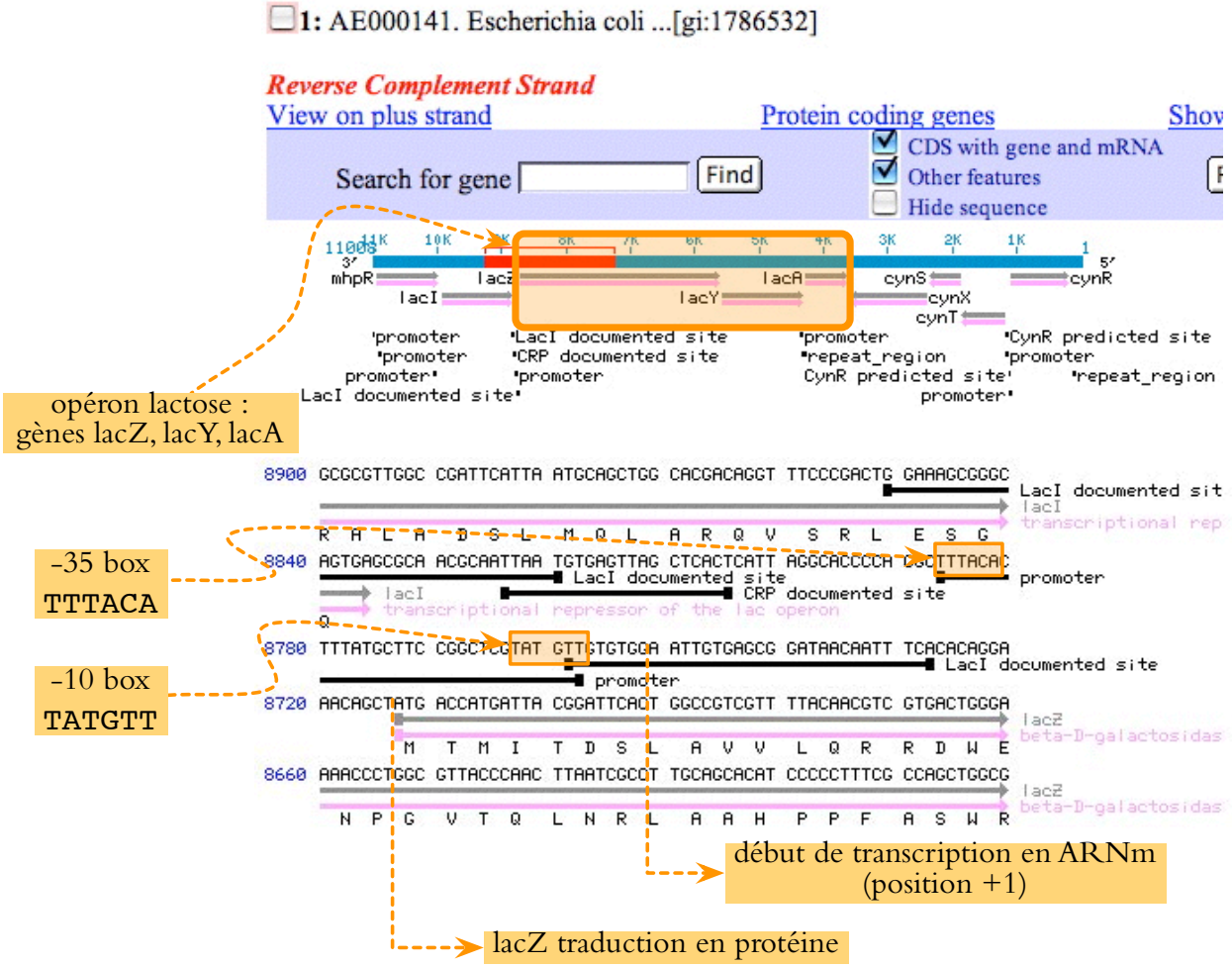
Idée : PD pour les suffixes (ou préfixes) — récurrences similaires

Dans le graphe d'édition : trouver le chemin de $v_{i,0}$ à $v_{i',|T|}$ avec poids maximal pour $i' > i$ quelconques (occurrence de T en S)

Exemple 1 : séquences répétées (p.e., SINE - *Short Interspersed Nuclear Elements*)



Exemple 2 : promoteur sigma A : `TTGACA...TATAAT` [à -35 et -10]



Brown *Genomes*, p. 180, Wiley 1999

Alignement : variation 2

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

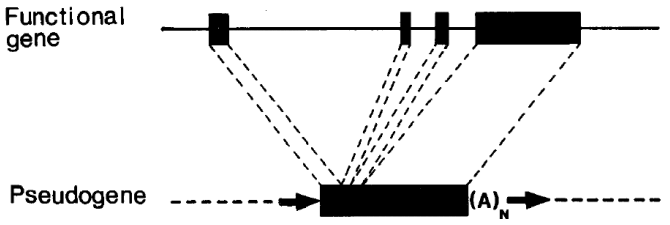
Dans le graphe d'édition : trouver le chemin de $v_{i,j}$ à $v_{i',j'}$ avec poids maximal pour $i' \geq i, j' \geq j$ quelconques

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

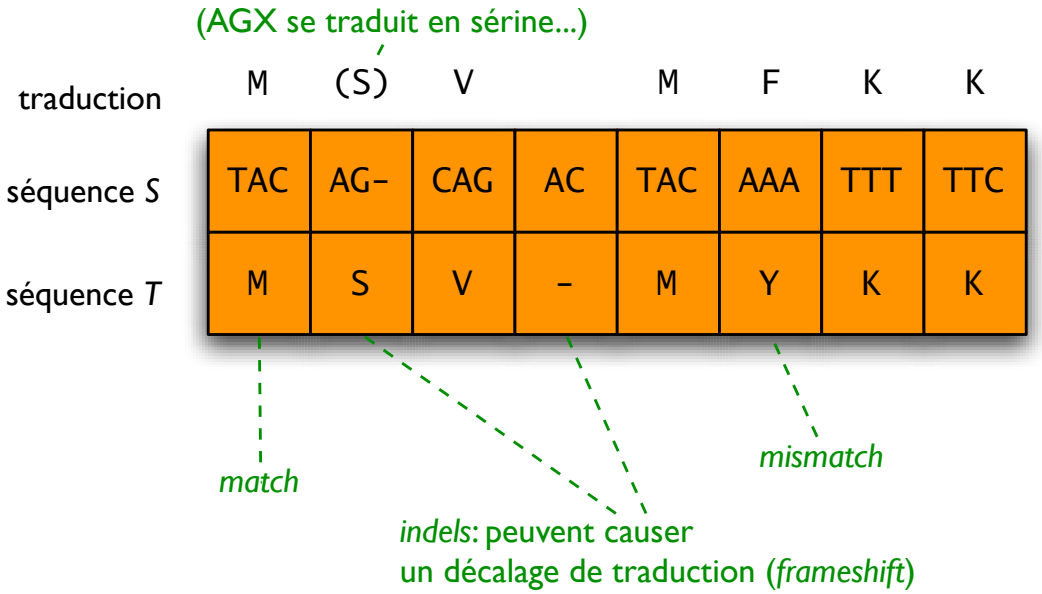
Exemple 1 : Alignement de ADN génomique et ADN codant — pseudogènes




```
                A  D  Q  T  S  G  D  Q  S  P  L  P     P  C  T  P  T  P  P
SHP2        GCG GAC CAG ACG AGT GGA GAT CAG AGC CCT CTC  CC--G CCT TGT ACT CCA ACG CCA CCC
SHP2-P3     .T. ... ... ... ..G ... ... ... ..C ...  .T--. ... .A. ... ... .T. ... ...
```

Vanin *Annu Rev Genet* 19 :253 ; Andersen & al *FASEB J* 18 :8

Alignement ADN-protéine



Trous

Pénalisation d'un trou par sa taille :
- quelconque : $\delta(\text{longueur})$
- constante
- linéaire : $\delta(\ell) = \delta_{\text{ouvrir}} + \ell\delta_{\text{cont}}$ (avant on avait le cas spécial $\delta_{\text{ouvrir}} = 0$)

Récurrence pour pondération «quelconque» :

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

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 — exemple

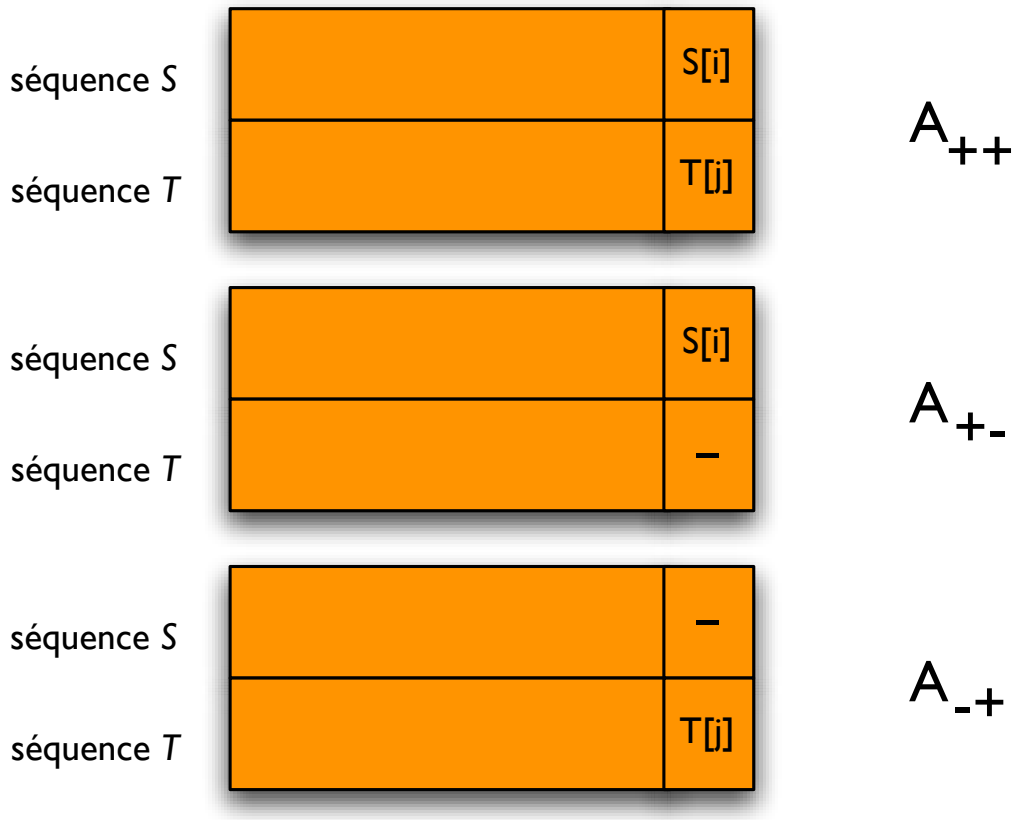
Trous longs (disons $\ell \geq 5$) en S avec pénalisation constante δ_{long}, trous courts avec pénalisation quelconque $\delta(\ell)$; trous en T avec pénalisation simple $\delta'\ell$.

Définir graphe d'alignement avec deux genres de sommets : $v_{i,j}$ pour alignements de préfixes qui finissent pas par trous longs et $t_{i,j}$ pour ceux qui finissent par trous longs.

Arêtes : $v_{i,j} \searrow v_{i+1,j+1}$ ponderée par $\mathbf{C}\begin{bmatrix} S[i] \\ T[j \end{bmatrix}$, $v_{i,j} \downarrow v_{i+k,j}$ pour $k = 1, \ldots, 4$ ponderée par $\delta(k)$, $v_{i,j} \rightarrow v_{i,j+1}$ ponderée par δ', $u_{i,j} \rightarrow v_{i,j}$ ponderée par 0, $v_{i,j} \downarrow u_{i+5,j}$ ponderée par δ_{long}, $u_{i,j} \downarrow u_{i+1,j}$ ponderée par 0.

Pondération linéaire

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



Pondération linéaire 2

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

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

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

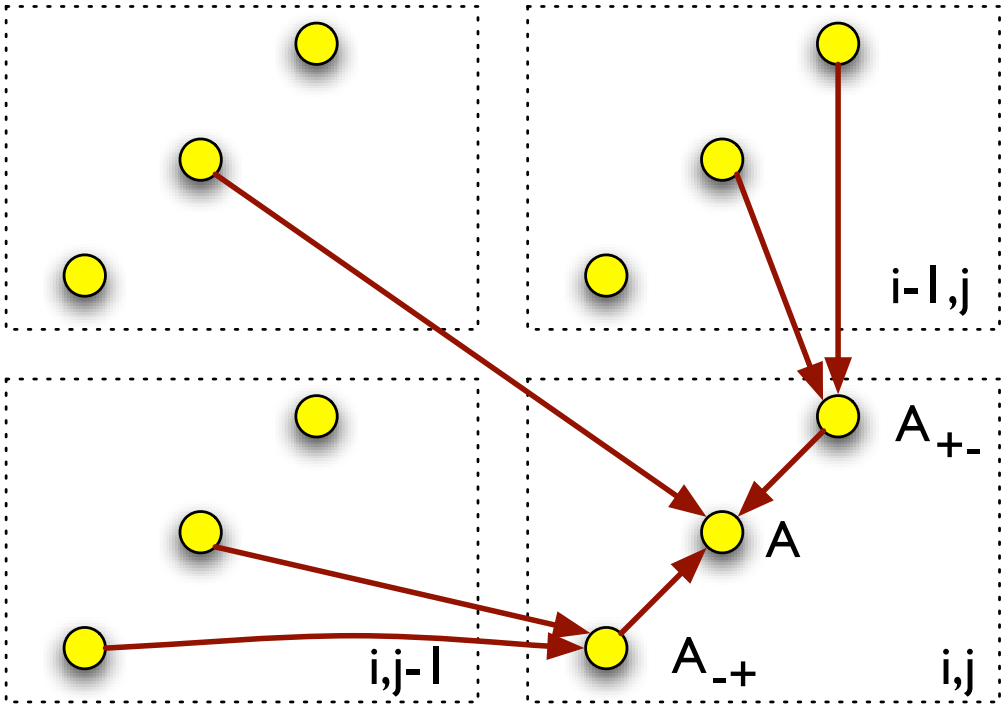
Pondération linéaire 3

En fait, on peut éliminer A^{++} :

$$A^{+-}(i,j) = \max\Big\{A^{+-}(i-1,j) + \delta_1, A(i-1,j) + \delta_0\Big\};$$
$$A^{-+}(i,j) = \max\Big\{A^{-+}(i,j-1) + \delta_1, A(i,j-1) + \delta_0\Big\};$$
$$A(i,j) = \max\Big\{\mathbf{C}\left[\begin{smallmatrix} S[i] \\ T[j] \end{smallmatrix}\right] + A(i-1,j-1), A^{+-}(i,j), A^{-+}(i,j)\Big\}.$$

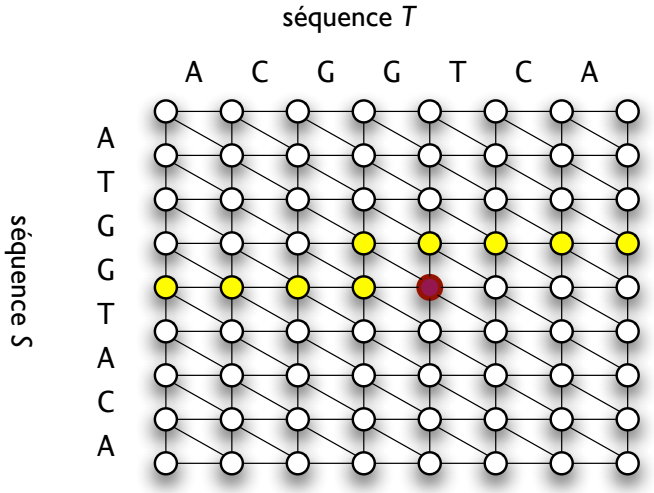
Cas de base : $A(0,0) = A^{+-}(0,0) = A^{-+}(0,0) = 0; A(i,0) = A^{+-}(i,0) = \delta_0 + (i-1)\delta_1; A(0,j) = A^{-+}(0,j) = \delta_0 + (j-1)\delta_1$.

Pondération linéaire 4



Calcul en espace linéaire

Calcul de score $V(i,j)$ rangée par rangée de gauche à droite (trou de taille ℓ pondéré par $\delta\ell$)



On n'a besoin que de $V(i-1,j')\colon j' \geq j$ et de $V(i,j'')\colon j'' \leq j$.

Espace linéaire 2

Algo Ali-LinEspace
Entrée : séquences S, T avec $n = |S|$ et $m = |T|$; matrice de coûts $\mathbf{C}$

L1 $U[0] \leftarrow 0$; **for** $j \leftarrow 1..m$ **do** $U[j] \leftarrow U[j-1] + \mathbf{C}\begin{bmatrix} - \\ T[j] \end{bmatrix}$

L2 **for** $i \leftarrow 1..n$ **do**

L3 $\quad x \leftarrow U[0]$; $U[0] \leftarrow U[0] + \mathbf{C}\begin{bmatrix} S[i] \\ - \end{bmatrix}$

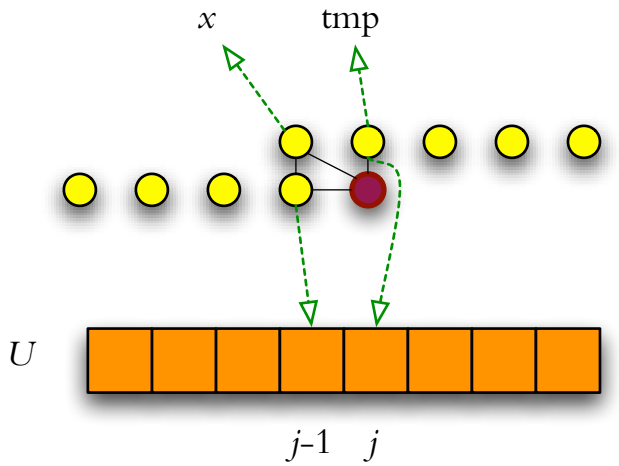
L4 $\quad$ **for** $j \leftarrow 1..m$ **do**

L5 $\quad\quad$ tmp $\leftarrow U[j]$

L6 $\quad\quad U[j] \leftarrow \max\left\{U[j-1] + \mathbf{C}\begin{bmatrix} S[i] \\ - \end{bmatrix}, x + \mathbf{C}\begin{bmatrix} S[i] \\ T[j] \end{bmatrix}, U[j] + \mathbf{C}\begin{bmatrix} - \\ T[j] \end{bmatrix}\right\}$

L7 $\quad\quad x \leftarrow$ tmp

Affectations en Ligne L6



Espace linéaire 3

Mais comment trouver le meilleur alignement (et pas seulement son score) en espace linéaire ?

Idée de clé : trouver le noeud en ligne i du graphe par lequel le chemin du meilleur alignement doit passer

1. calculer $\forall j : V_{\text{pre}}(i, j)$, le score du meilleur alignment de $S[1..i]$ et $T[1..j]$ ($V_{\text{pre}}(i, j) = U[j]$ en Ali-LinEspace si appellé avec $S[1..i]$ et T)

2. calculer $\forall j : V_{\text{suf}}(i, j)$, le score du meilleur alignement de $S[i + 1..n]$ et $T[j + 1..m]$.

3. le meilleur alignement entre S et T maximise $V(i, j^*) = V_{\text{pre}}(i, j^*) + V_{\text{suf}}(i, j^*)$

On peut calculer V^* en espace linéaire avec un algorithme similaire à Ali-LinEspace

Espace linéaire 4 — Hirschberg

Algo Ali-Hirschberg
Entrée : séquences S, T avec $n = |S|$ et $m = |T|$; matrice de coûts $\mathbf{C}$

H1 **if** $m = 0$ **then return** alignement $\begin{bmatrix} S[1] & \dots & S[n] \\ - & \dots & - \end{bmatrix}$
H2 **if** $n = 1$ **then**
H3 trouver $j \in 1, \dots, m$ t.q. $\mathbf{C}\begin{bmatrix} S[1] \\ T[j] \end{bmatrix}$ is maximal
H4 **return** alignement $\begin{bmatrix} - & \dots & - & S[1] & - & \dots & - \\ T[1] & \dots & T[j-1] & T[j] & T[j+1] & \dots & T[m] \end{bmatrix}$
H5 soit $i \leftarrow \lfloor n/2 \rfloor$
H6 calculer $U[j] = V_{\text{pre}}(i, j)$ et $W[j] = V_{\text{suf}}(i, j)$ pour tout $j = 0, \dots, m$
H7 soit $M \leftarrow \max_{j=0,\dots,m}\{U[j] + V[j]\}$
H8 soit $j^* \leftarrow \min\{j : U[j] + W[j] = M\}$
H9 calculer l'alignement $A_1 \leftarrow$ Ali-Hirschberg$(S[1..i], T[1..j^*], \mathbf{C})$
H10 calculer l'alignement $A_2 \leftarrow$ Ali-Hirschberg$(S[i+1..n], T[j^*+1..m], \mathbf{C})$
H11 **return** la concaténation de $A_1 \cdot A_2$

(En lignes H9–H10, $T[1..0] = \varepsilon$ et $T[m+1..m] = \varepsilon$)

Hirschberg, *Comm. ACM*, 18 : 341

Alignement plus rapide

1. méthodes heuristiques : hachage, arbres de suffixe, PD limitée (taille totale de trous bornée)

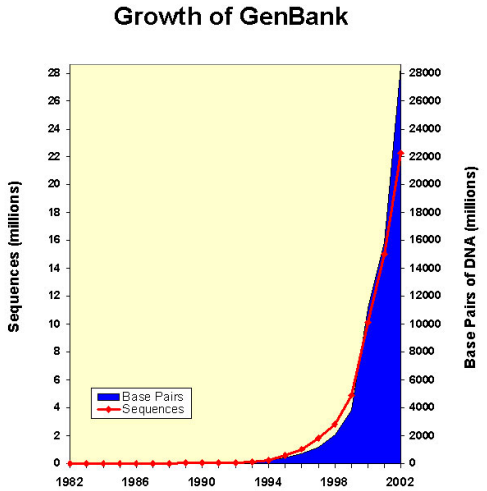
2. PD éparse (pour sous-séquence commune ou *chaînage* en alignement global heuristique)

Banques de séquences

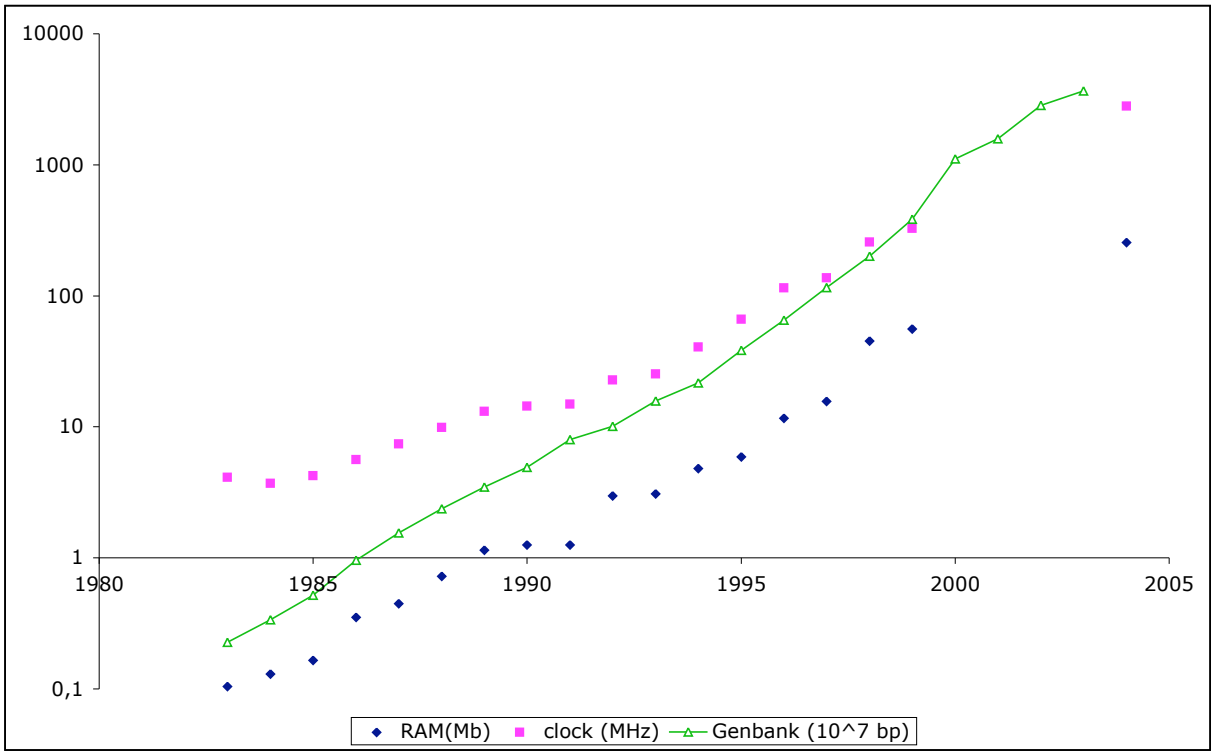
Bases de données de séquences : beaucoup d'information.

Exemple : GenBank

- 38 milliards de nucléotides ; 32.5 millions de séquences
- croissance exponentielle (taille doublée tous les 14 mois)



Need for speed



Banques de données

NCBI : «National Center for Biotechnology Information» — États-Unis

Interface [Entrez] à plusieurs bases de données :

- séquences d'acides nucléiques
- séquences protéiques
- PubMed : publications
- structures
- taxonomie
- ...

GenBank

Séquences d'ADN : GenBank (É-U), DDBJ (Japon), EMBL (Europe)

GenBank «flatfile» : exemple (HUMXIHB) :

```
LOCUS       HUMXIHB                  458 bp    mRNA    linear   PRI 14-JAN-1995
DEFINITION  Human zeta hemoglobin mRNA, complete cds.
ACCESSION   M24173
VERSION     M24173.1  GI:340391
KEYWORDS    zeta-globulin.
SOURCE      Homo sapiens (human)
  ORGANISM  Homo sapiens
            Eukaryota; Metazoa; Chordata; Craniata; Vertebrata; Euteleostomi;
            Mammalia; Eutheria; Primates; Catarrhini; Hominidae; Homo.
REFERENCE   1  (bases 1 to 458)
  AUTHORS   Cohen-Solal,M.M., Authier,B., deRiel,J.K., Murnane,M.J. and
            Forget,B.G.
  TITLE     Cloning and nucleotide sequence analysis of human embryonic
            zeta-globin cDNA
  JOURNAL   DNA 1 (4), 355-363 (1982)
  MEDLINE   83182021
   PUBMED   6963223
COMMENT     Original source text: Human erythroleukemia cell line K562, cDNA to
            mRNA, clones 1 (1g7-8), 2 (4p7-7), and 3 (5a3-3).
```

GenBank - champs 1

`LOCUS`

- 1–10 caractères alphanumériques ; jadis l'identificateur de la séquence (p.e. l'abbreviation du gène), préservée pour compatibilité seulement.
- longueur et type de la séquence (DNA, mRNA, tRNA, rRNA)
- code de la division (p.e. PRI) et date de dernière modification.

`DEFINITION` «sommaire» de la séquence : espèce et le nom de la séq

`ACCESSION` nombre d'accession : clé dans la base de donnée. Identificateur unique parmi les BDs. Forme AA999999. L'accno est généré automatiquement lors de la soumission d'une séquence à la BD.

GenBank - champs 2

`KEYWORDS` et `SOURCE` : moins d'importance (pour nous)

`VERSION` donne ⟨accno⟩.⟨version⟩ et gi : identificateur de GenInfo. Ce sont des identificateurs des *séquences* (qui peut changer pour le même accno).

`REFERENCE`

GenBank — exemple cont.

```
FEATURES             Location/Qualifiers
     source          1..458
                     /organism="Homo sapiens"
                     /db_xref="taxon:9606"
                     /map="16p13.3"
     gene            1..458
                     /gene="HBZ"
     CDS             30..458
                     /gene="HBZ"
                     /note="zeta hemoglobin"
                     /codon_start=1
                     /protein_id="AAA61306.1"
                     /db_xref="GI:340392"
                     /db_xref="GDB:G00-119-302"
                     /translation="MSLTKTERTIIVSMWAKISTQADTIGTETLERLFLSHPQTKTYF
                     PHFDLHPGSAQLRAHGSKVVAAVGDAVKSIDDIGGALSKLSELHAYILRVDPVNFKLL
                     SHCLLVTLAARFPADFTAEAHAAWDKFLSVVSSVLTEKYR"
BASE COUNT       80 a    173 c    127 g     78 t
ORIGIN      79 bp upstream of BglII site.
        1 actccagtgc agctgcccac cctgccgcca tgtctctgac caagactgag aggaccatca
          .......
      421 cggtcgtatc ctctgtcctg accgagaagt accgctga
//
```

GenBank - champs 3

`FEATURES` annotation de la séquence : un «feature» comprend un mot-clé, sa position, et des qualifieurs

position : sous-mot [p.e., `2..280`], entre deux bases [p.e., `91^92`], ...,
et opérations : `complement`(·), `join`(·,...,·)

mots-clé :
- `source` information taxonomique
- `CDS` partie traduite en une séquence protéique
- `exon`, `intron`, `gene`
- `repeat_region`
- ...

Exemple : U96726

GenBank - entrées virtuelles

Exemple : U00089

```
LOCUS       U00089                816394 bp    DNA     circular CON 06-DEC-2002
DEFINITION  Mycoplasma pneumoniae M129, complete genome.
...
CONTIG      join(AE000016.2:1..19313,AE000015.2:59..17535,AE000014.2:22..12521,
            AE000013.2:53..10328,AE000012.2:59..10228,AE000011.2:59..15387,
            ... [plusieurs lignes]
            AE000019.2:59..10270,AE000018.2:59..11147,AE000017.2:62..15963)
//
```

FASTA

Un autre format très répandu, utilisé originalement par les logiciels du package FASTA.

```
>CRA|agCP11170 /len=264 /protein_uid=197000044174854 /org=Anopheles_gambiae
ATSFTMPQNEYIERHIKLYGRRLDYEERKRKREAREPKKRAAMARKLRGMKAKLFQKQRR
NEKIQMKRKIQAHEEKKVKKTTEKVEDGALPPYLMDRGIQSNAKVLSNMIKQKRKEKAGK
WDVPIPKVRAQADAEVFKVIRSGKTKRKAWKRMVTKVTYVGENFTRKPPKYERFIRPMAL
RMNKAHVTHPELKATFHLPIIGVKKNPSSPMYTSLGVITKGTVIEVNISELGLVTQSGKV
VWGKYAQVTNNPENDGCINAVLLV
>gi|30697195|ref|NP_200732.2|gnl|TIGR|At5g59240 structural constituent of ribosome [A. thaliana]
MGISRDSIHKRRATGGKQKMWRKKRKYELGRQPANTKLSSNKTVRRIRVRGGNVKWRALR
LDTGNFSWGSEAVTRKTRILDVAYNASNNELVRTQTLVKSAIVQVDAAPFKQGYLQHYGV
DIGRKKKGEAVTTEEVKKSNHVQRKLEMRQEGRALDSHLEEQFSSGRLLACIASRPGQCG
RADGYILEGKELEFYMKKLQKKKGKNAGAA
...
```

defline

une ou plusieurs séquences (ADN ou protéine)

en-tête pour chaque séquence (ligne `>...`)

syntaxe souvent utilisé : `/propriété=valeur`

defline : références à des banques de séquences

fomat général : ⟨db⟩ | ⟨ident⟩

NCBI

GenBank «flatfile» généré automatiquement à partir des bases de données.

Entrez : interface integré : recherche par identificateurs, mots clés, auteurs, etc.

BLAST : famille d'outils pour trouver des occurrences inexactes d'une séquence S dans le «texte» T

choix de T : `nr`, `est`, `month`, etc.

BLAST

BLAST : recherche par hachage + théorie de probabilités pour alignements locaux

Hachage — idée principale :
pour alignement local rapide entre S et T

1. fixer $k > 0$
2. comparer chaque sous-mot de longueur k (k-mer) de S avec ceux de T
3. extension des matches pour obtenir un alignement local entre S et T

$\Rightarrow$ on trouvera rapidement les alignements qui contiennent k matches consécutifs

Hachage

Deux séquences S, T
Fonction de hachage : $h\colon \{\mathtt{A}, \mathtt{C}, \mathtt{G}, \mathtt{T}\}^k \mapsto \mathcal{H}$
hit : (i, j) avec $h(S[i..i+k-1]) = h(T[j..j+k-1])$
Technique : listes $\mathsf{Occ}(u)$ de positions où $h^{-1}(u)$ apparaît

1. **pour** $i \leftarrow 1, \ldots, |S| - k + 1$ **faire**
2. $\quad \mathsf{clé} \leftarrow h(S[i..i+k-1])$
3. $\quad$ ajouter i à la fin de la liste $\mathsf{Occ}(\mathsf{clé})$
4. **pour** $j \leftarrow 1, \ldots, |T| - k + 1$ **faire**
5. $\quad \mathsf{clé} \leftarrow h(T[j..j+k-1])$
6. $\quad$ traitement [extension ?] des *hits* $(i, j)\colon i \in \mathsf{Occ}(\mathsf{clé})$

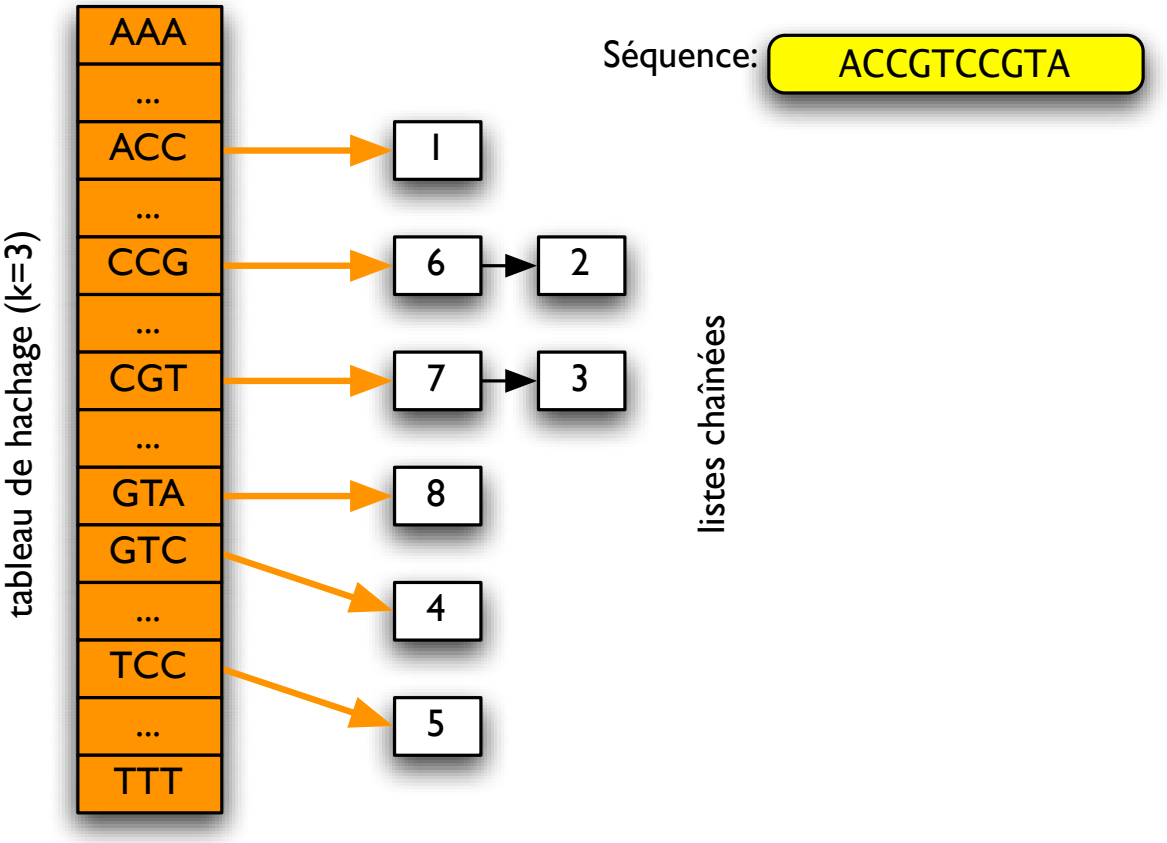
Hachage — structure de données

Trouver les clés partagés : stocker les occurrences (Occ) de tous les clés de S en un tableau de hachage

Implantation facile en Java : on peut utiliser les sous-mots w comme clés directement, `Hashtable` calcule des clés de hachage automatiquement, liste chaînée pour chaque $\mathsf{Occ}(w)$

(utilise plus de mémoire que nécessaire...)

TABLEAU DE k-MERS



Ma, Tromp et Li. *Bioinformatics* 18 :440.

IMPLANTATION

1. Encodage des k-mers en $2k$ bits :
`A` $\rightarrow 00$, `C` $\rightarrow 01$, `G` $\rightarrow 10$, `T` $\rightarrow 11$.

Java `int` : 32 bits ($k \leq 16$) ; `long` : 64 bits ($k \leq 32$).

2. Encodage des listes chaînées :
chaque position de la séquence n'apparaît qu'une fois !
Définir un tableau `int[] successeur` où `successeur`$[i]$ donne la position qui suit i dans une des listes chaînées ou égale à -1 si i est la dernier objet dans une liste.
Tête de chaque liste est trouvée par un tableau `int[] tete` où `tete`$[i]$ donne la première position ou le k-mer encodé par i se trouve.

Mémoire : $(4^k + |S|)$ fois taille de `int` (4 octets).

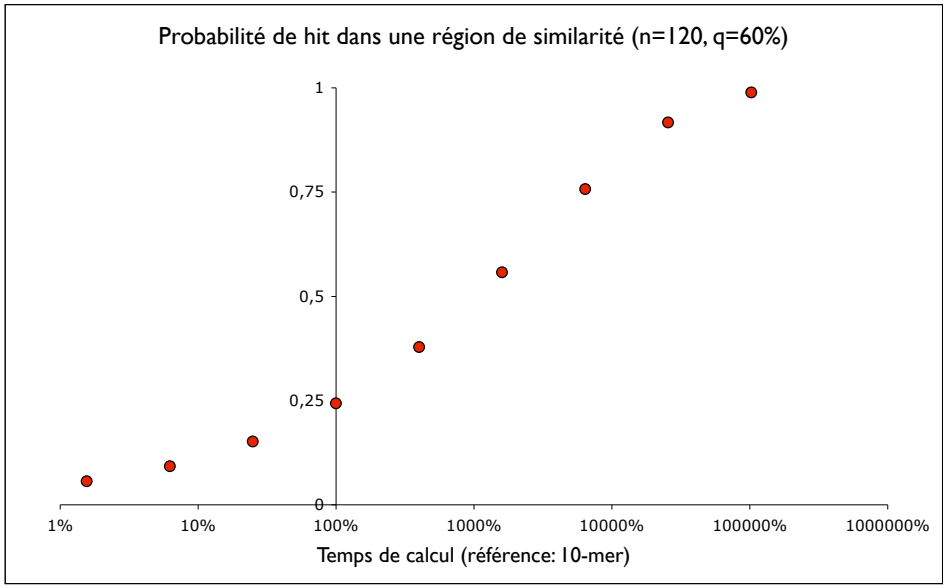
IMPLANTATION — JAVA

```
HT1  int[] tete=new int[1<<(2*k)];
HT2  int[] successeur=new int[S.length()-k+1];
HT3  pour tout i, tete[i] ← −1
HT4  for (int i=0; i<S.length()-k+1; i++){
HT5      calcul de l'encodage w pour le sous-mot S[i..i+k−1];
HT6      successeur[i]=tete[w];
HT7      tete[w]=i;
HT8  }
HT9  for (int j=0; j<T.length()-k+1; j++){
HT10     calcul de l'encodage w pour le sous-mot T[j..j+k−1];
HT11     int i=tete[w];
HT12     while (i != -1){
HT13         extension du hit (i, j)
HT14         i=successeur[i];
HT15     }
HT16 }
```

Hachage — performance

Spécificité : mesurée par nombre de *hits* entre deux séquences sans homologies (p.e., aléatoires)

Sensibilité : mesurée par la probabilité de *hit* dans une région de homologie



Hachage — nombre de *hits*

Modèle : S aléatoire, avec nucléotides iid selon p ; T aléatoire, avec nucléotides iid selon q : $\mathbb{P}\{S[i] = c\} = p_c$ et $\mathbb{P}\{T[j] = c'\} = q_{c'}$.

Fonction de hachage : identité $h(u) = u$, $\mathcal{H} = \Sigma^k$ ($\Sigma = \{\mathtt{A}, \mathtt{C}, \mathtt{G}, \mathtt{T}\}$)

Thm. Soit $\beta = \sum_{c \in \Sigma} p_c q_c$. Alors le nombre de *hits* en espérance est $st\beta^k$ où $s = |S| - k + 1$ et $t = |T| - k + 1$.

Détour — nombre de hits

L'espérance de nombre de hits est la même pour tous les sous-mots $w \in \Sigma^k$. Est-ce que la distribution est la même aussi ? Non !

Exemple : «pas tous les mots sont créés égaux»

Exemple : T est une séquence de longueur n «au hasard»
au hasard : chaque caractère de T est 0 ou 1 avec probabilités $\frac{1}{2}$-$\frac{1}{2}$.

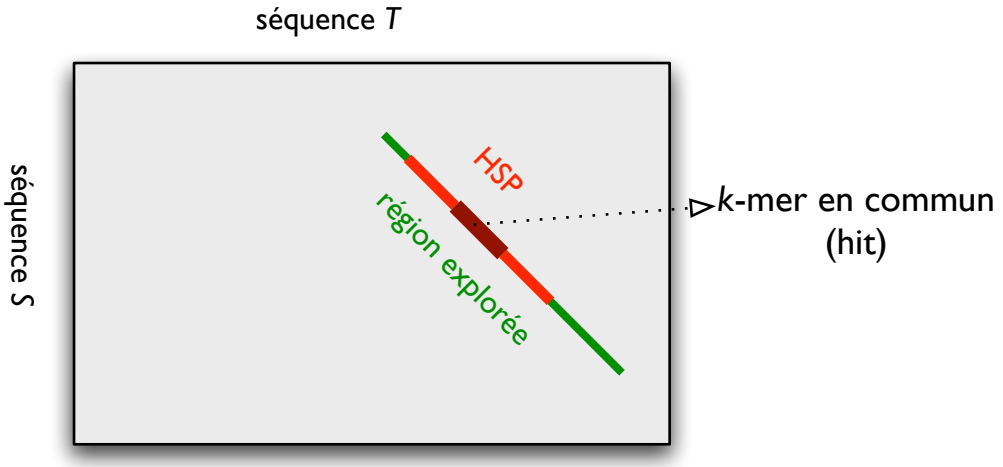
Quelle est la probabilité de voir $w = \texttt{00}$ ou $w = \texttt{01}$?

Extension d'un hit

Techniques :

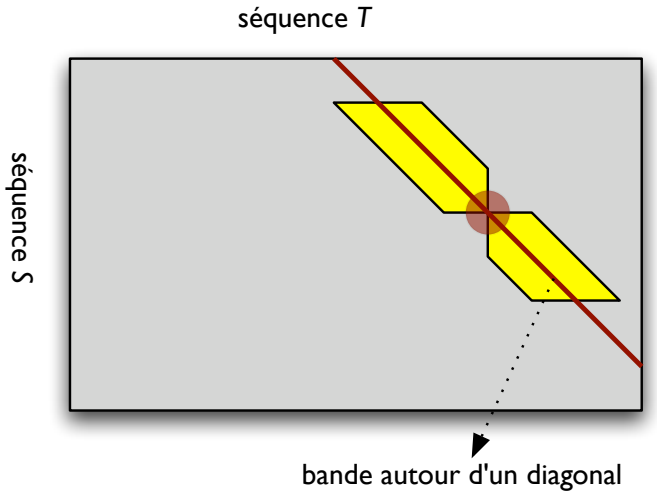
- extension rapide sur une diagonale
- X-drop
- programmation dynamique dans une bande

Extension rapide



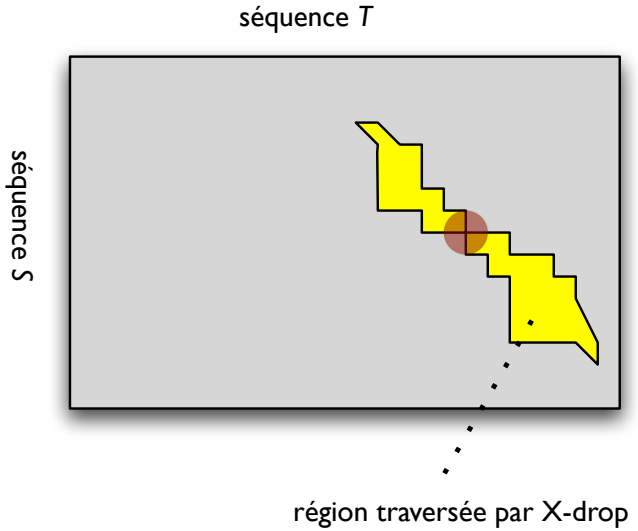
Rester sur la même diagonale ; explorer jusqu'à ce que le score devienne 0, prendre le meilleur segment (*high-scoring segment pair*, HSP)

PD DANS UNE BANDE



Bande de $\pm d$ sommets proches de la diagonale D : $\{v_{i,j} : |(i-j) - D| \leq d\}$

X-DROP



à partir d'une case initiale, explorer vers $v_{0,0}$ et $v_{|S|,|T|}$; arrêter si le score tombe par X

(exploration de toute une région ou quelques [même 1] diagonales)

Altschul et al, *Nucleic Acids Res.* 25 : 3389.

Détails : extension rapide [vers sud-est]

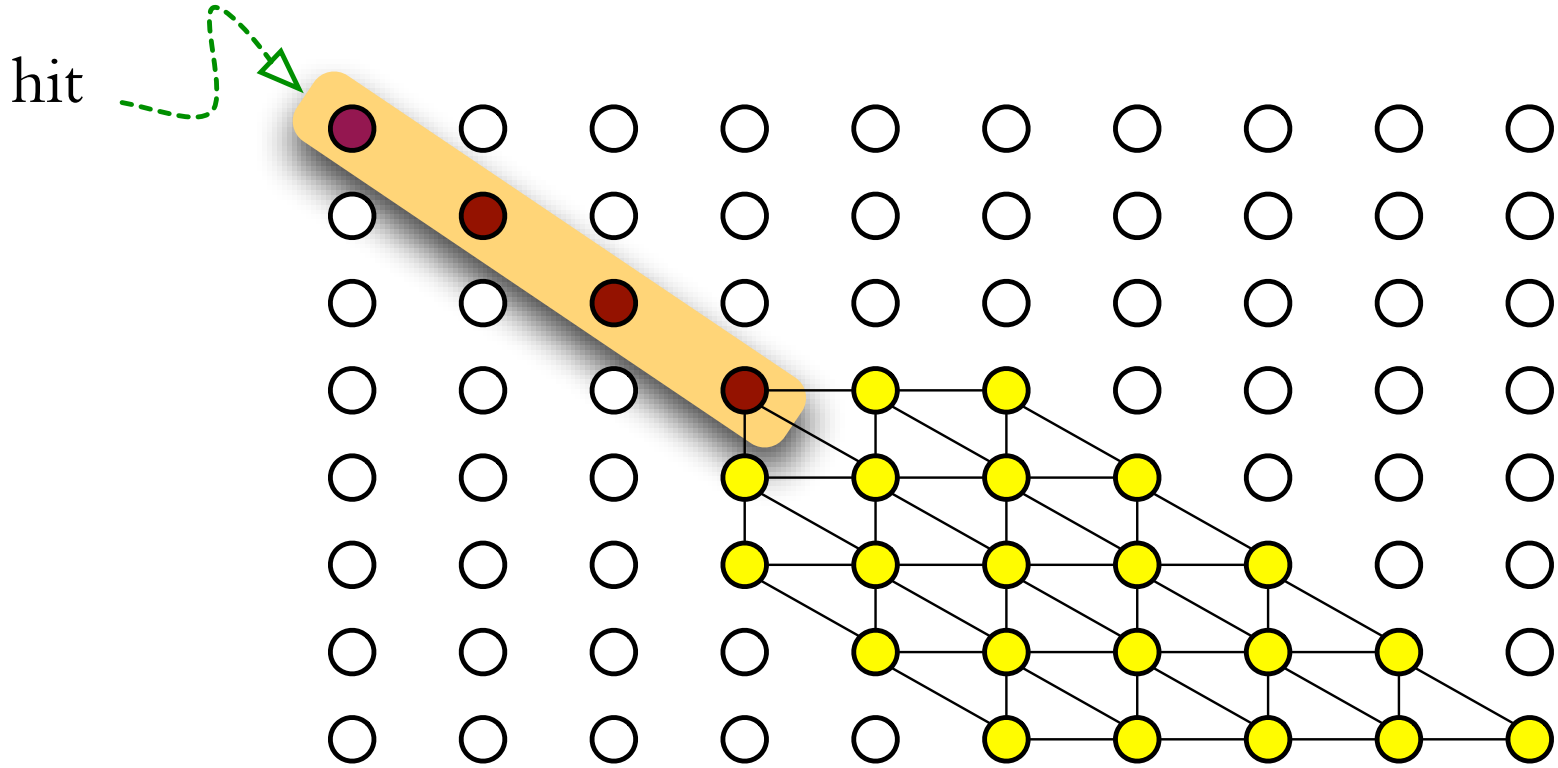
ER1 **Entrée** i_0, j_0 départ de l'extension, s_0 score initial
ER2 meilleur $\leftarrow s_0$; extension $\leftarrow 0$
ER3 $i \leftarrow i_0 + 1$; $j \leftarrow j_0 + 1$; score $\leftarrow s_0$
ER4 **tant que** $i \leq |S|, j \leq |T|$, score ≥ 0
ER5 score $\leftarrow$ score $+ \mathbf{C}\begin{bmatrix} S[i] \\ T[j] \end{bmatrix}$
ER6 **si** score $\geq$ meilleur **alors** meilleur $\leftarrow$ score, extension $\leftarrow j - j_0$
ER7 $i \leftarrow i + 1, j \leftarrow j + 1$
ER8 **retourner** meilleur, extension

Détails : bande [vers sud-est]

B1 **Entrée** i_0, j_0 départ de l'extension, s_0 score initial, d épaisseur
B2 $A^* \leftarrow s_0, i \leftarrow i_0, D \leftarrow i_0 - j_0$
B3 **tant que** $i \leq |S|$
B4 $\quad j \leftarrow \max\{j_0, i - D - d\}$
B5 $\quad$ **tant que** $j \leq \min\{|T|, i - D + d\}$
B6 $\quad\quad$ calculer $A(i,j)$; **si** $A^* < A(i,j)$ **alors** $A^* \leftarrow A(i,j)$
B7 $\quad\quad j \leftarrow j + 1$
B8 $\quad$ **si** $\forall j : A(i,j) \leq 0$ **alors** sauter à Ligne B10.
B9 $\quad i \leftarrow i + 1$
B10 reporter A^*

Détails bande 2 — graphe

Calcul en ligne B6 ($d = 2$)



Détails bande 2 — code

Calcul en ligne B6 : pondération par $\mathbf{C}$

- si $i = i, j = j_0$, alors $A(i, j) = s_0$
- si $i = i_0$, et $j > j_0$, alors $A(i, j) = A(i, j-1) + \mathbf{C}\begin{bmatrix} - \\ T[j] \end{bmatrix}$
- si $i > i_0$ et $j = j_0$, alors $A(i, j) = A(i-1, j) + \mathbf{C}\begin{bmatrix} S[i] \\ - \end{bmatrix}$
- si $i > i_0 + d$ et $j = i - D - d$, alors
  $A(i, j) = \max\left\{ A(i-1, j-1) + \mathbf{C}\begin{bmatrix} S[i] \\ T[j] \end{bmatrix}, A(i-1, j) + \mathbf{C}\begin{bmatrix} S[i] \\ - \end{bmatrix} \right\}$

Détails bande 2 — code (cont.)

- si $i > i_0$ et $\max\{j_0, i-D-d\} < j < \min\{|T|, i-D+d\}$, alors $A(i,j) = \max\Big\{A(i-1,j)+\mathbf{C}\Big[\begin{smallmatrix}S[i]\\-\end{smallmatrix}\Big], A(i-1,j-1)+\mathbf{C}\Big[\begin{smallmatrix}S[i]\\T[j]\end{smallmatrix}\Big], A(i,j-1)+\mathbf{C}\Big[\begin{smallmatrix}-\\T[j]\end{smallmatrix}\Big]\Big\}$
- si $i > i_0$ et $j = \min\{|T|, i-D+d\}$, alors $A(i,j) = \max\Big\{A(i-1,j-1)+\mathbf{C}\Big[\begin{smallmatrix}S[i]\\T[j]\end{smallmatrix}\Big], A(i,j-1)+\mathbf{C}\Big[\begin{smallmatrix}-\\T[j]\end{smallmatrix}\Big]\Big\}$

Détails X-drop

Idée : maintenir A^* score du meilleur alignement et ne pas continuer l'extension si $A(i,j) < A^* - X$

Stocker col_g, col_d : colonnes de la dernière rangée que l'on a explorée.

(Ou code plus simple si l'exploration est dans une bande seulement : on n'a pas besoin de col_g, col_d)

[Code pour extensions vers sud-est seulement]

Détails X-drop 2

XD1 **Entrée** i_0, j_0 départ de l'extension, s_0 score initial, X
XD2 $A^* \leftarrow s_0$, $\mathsf{col}_\mathrm{g} \leftarrow j_0$, $\mathsf{col}_\mathrm{d} \leftarrow |T|$
XD3 $i \leftarrow i_0$
XD4 **tant que** $i \leq |S|, \mathsf{col}_\mathrm{g} \leq \mathsf{col}_\mathrm{d}$
XD5 $\quad j \leftarrow \mathsf{col}_\mathrm{g}$
XD6 $\quad$ **tant que** $j \leq \min\{\mathsf{col}_\mathrm{d} + 1, |T|\}$
XD7 $\quad\quad$ calculer $A(i, j)$
XD8 $\quad\quad$ **si** $A(i, j) > A^*$ **alors** $A^* \leftarrow A(i, j)$
XD9 $\quad\quad$ **si** $A(i, j) < A^* - X$ **alors** $A(i, j) \leftarrow -\infty$
XD10 $\quad\quad$ $j \leftarrow j + 1$
XD11 $\quad$ **tant que** $\mathsf{col}_\mathrm{g} \leq \mathsf{col}_\mathrm{d}$ et $A(i, \mathsf{col}_\mathrm{g}) = -\infty$, $\mathsf{col}_\mathrm{g} \leftarrow \mathsf{col}_\mathrm{g} + 1$
XD12 $\quad$ $\mathsf{col}_\mathrm{d} \leftarrow \mathsf{col}_\mathrm{d} + 1$; **tant que** $\mathsf{col}_\mathrm{d} \geq \mathsf{col}_\mathrm{g}$ et $A(i, \mathsf{col}_\mathrm{d}) = -\infty$, $\mathsf{col}_\mathrm{d} \leftarrow \mathsf{col}_\mathrm{d} - 1$
XD13 $\quad$ $i \leftarrow i + 1$
XD14 retourner A^*

Détails X-drop 3

Calcul en ligne XD7 : pondération par $\mathbf{C}$

- si $i = i_0$ et $j > j_0$, alors $A(i,j) \leftarrow A(i,j-1) + \mathbf{C}\left[\begin{smallmatrix} - \\ T[j] \end{smallmatrix}\right]$
- si $i > i_0$ et $j = \mathsf{col}_{\mathrm{g}}$ alors $A(i,j) \leftarrow A(i-1,j) + \mathbf{C}\left[\begin{smallmatrix} S[i] \\ - \end{smallmatrix}\right]$
- si $i > i_0$ et $j = \mathsf{col}_{\mathrm{d}}+1$ alors $A(i,j) \leftarrow \max\left\{A(i-1,j-1)+\mathbf{C}\left[\begin{smallmatrix} S[i] \\ T[j] \end{smallmatrix}\right], A(i,j-1) + \mathbf{C}\left[\begin{smallmatrix} - \\ T[j] \end{smallmatrix}\right]\right\}$
- sinon $A(i,j) \leftarrow \max\left\{A(i-1,j)+\mathbf{C}\left[\begin{smallmatrix} S[i] \\ - \end{smallmatrix}\right], A(i-1,j-1)+\mathbf{C}\left[\begin{smallmatrix} S[i] \\ T[j] \end{smallmatrix}\right], A(i,j-1) + \mathbf{C}\left[\begin{smallmatrix} - \\ T[j] \end{smallmatrix}\right]\right\}$