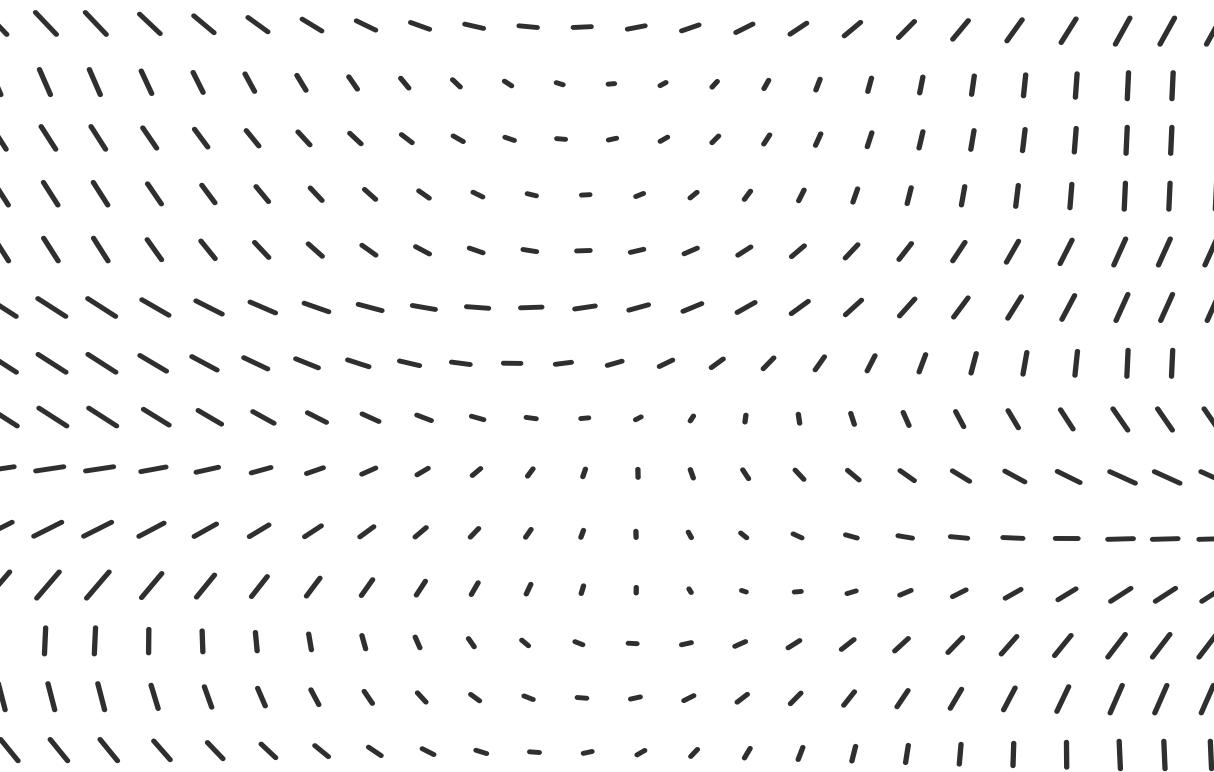


Narombra Condé : 20251772
Aissatou Ndiaye: 202175732



Prochainement des séquences :

- match : +4
- mismatch : -4
- ident : -8

1) Différence entre un alignement global et celui traité :

- Dans cet exercice, on cherche le meilleur chevauchement $x_i \rightarrow x_j$, donc, on veut aligner un suffixe de x_i avec un préfixe de x_j , et on ne pénalise pas ce qui sort hors de cette zone (ce qui correspond à un alignement local).
- Alors, qu'un alignement global, utilise les deux (2) séquences d'un bout à l'autre en pénalisant les gap aux deux (2) extrémités.

2) Valeurs d'initialisation:

$$V(i, 0) = 0$$

$$V(0, j) = 0$$

Justification

On met $V(i, 0) = V(0, j) = 0$ car dans un alignement local, les parties non alignées ne sont pas pénalisées. Ça permet à l'algorithme de redémarrer l'alignement à 0 dès qu'un score devient négatif pour pouvoir repérer la zone la plus similaire entre deux (2) séquences.

3) Équation de récurrence :

$$V(i, j) = \max \begin{cases} 0 \\ V(i-1, j) - 8 \\ V(i, j-1) - 8 \\ V(i-1, j-1) + 4 & \text{si } x_i = x_j \text{ (match)} \\ V(i-1, j-1) - 4 & \text{si } x_i \neq x_j \text{ (mismatch)} \end{cases}$$

4) Procédure entière pour retrouver l'alignement :

Étape 1 :

Après avoir rempli la table de programmation dynamique V , on trouve la case ayant le score maximal (le score le plus élevé du tableau). C'est le point de fin du meilleur chevauchement.

Étape 2 :

On effectue le traceback à partir de cette case :

- Si le mouvement vient de la diagonale, c'est un match (ou un mismatch si le score diffère),
- Si le mouvement vient du haut, c'est une insertion (gap dans la 2^e séquence),
- Si le mouvement vient de gauche, c'est une suppression (gap dans la 1^{re} séquence);

En cas de plusieurs chemins possibles, on peut choisir n'importe lequel des meilleurs scores.

On continue à remonter jusqu'à atteindre une case où $V(i,j) = 0$: cela marque le début de l'alignement.

Résultat :

les caractères parcourus entre cette case et la case maximale représentent le meilleur chevauchement suffixe-prefixe entre les deux séquences.

5) Implémentation de l'algorithme

Voir l'algorithme de l'alignement global dans le fichier question_1.s.py qui regroupe les fonctions de calcul de chevauchement.

Assemblage de fragments

1) Pour chaque paire de reads, on a calculé le score de l'alignement correspondant au chevauchement maximal entre deux séquences.

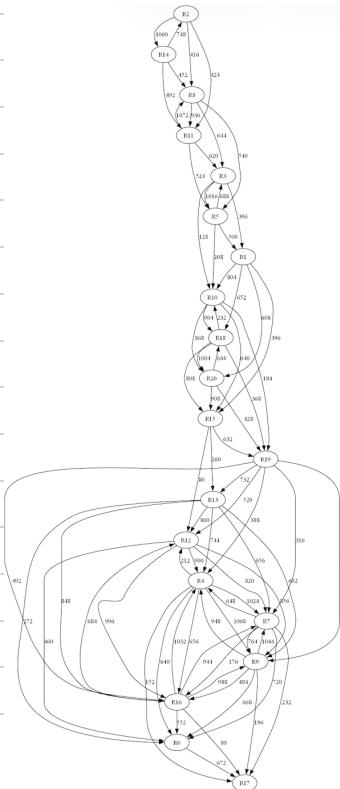
Voir l'algorithme à la suite du fichier question_1_s.py.

2)

a) L'exécution du programme indique qu'après filtrage, 76 arêtes dépassent le seuil.

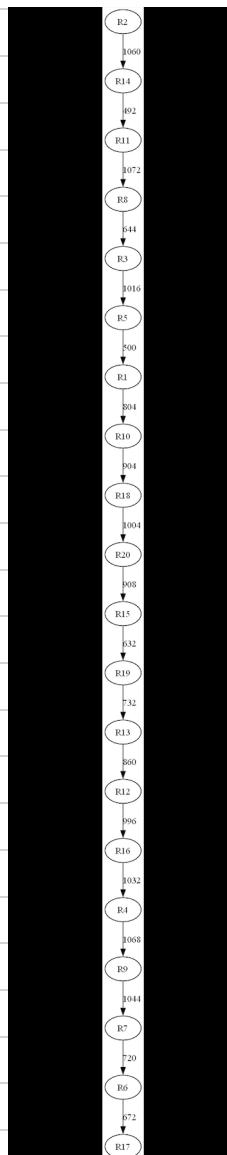
```
Nombre d'arêtes avec score >= 80 : 76  
Graph exported to overlap_graph.dot  
Après réduction transitive et suppression des 2-cycles, nombre d'arêtes : 19  
Graph exported to overlap_graph_reduced.dot
```

b) Graph orienté de chevauchement utilisant un seuil minimal de 80 pour filtrer correctement :



Le graphe se trouve dans le fichier overlap-graph.pmg

c) Réduisons la séquence du fragment génomique séquéncé et sa longueur:



Le graphe se trouve dans le fichier overlap-graph-reduced2.png

Recherche d'introns et Blast :

1-a) Cadre de lecture du codon start de la séquence de protéine:

- listons les cadres possibles

* Cadre 1 : lecture à partir de la première base (A) :

AGG ACA AAG TTT GCC TTA CCA TAT GTT TCC T

AGG : R

GCC : A

GTT : V

ACA : T

TTA : L

TCC : F

AAG : K

CCA : P

TTT : F

TAT : Y

⇒ Cadre 1 : RTKFALPY VF

* Cadre 2 : lecture à partir de la deuxième base (G)

GGA CAA AGT TTG CCT TAC CAT ATG TTT CCT

GGA : G

CCT : P

TTT : F

CAA : Q

TAC : Y

CCT : P

AGT : S

CAT : H

TTG : L

ATG : M

⇒ Cadre 2 : GQSLPYHMF P

* Cadre 3 : lecture à partir de la troisième base (G)

GAC AAA GTT TGC CTT ACC ATA TGT TTC CT

GAC : D

CTT : L

TTC : F

AAA : K

ACC : T

GTT : V

ATA : I

TGC : C

TGT : C

⇒ Cadre 3 : DKVCLTICF

Alors, ici, seul le cadre 2 contient un codon start ATG (M)

suivi d'une suite d'acide aminés qui pourrait être cohérente avec la protéine geneX.fasta.

On en déduit que la protéine X est codée sur le cadre de lecture 2 du brin codant.

1-b) Algorithme de programmation dynamique pour retrouver le gène en minimisant les introns:

Données :

* Protéine cible : $P = p_1 \dots p_m$ (issu de geneX.fasta)

* Séquence d'ADN génomique, dans le cadre trouvé en 1-a (cadre 2, à partir de l'ATG)

Modèle :

Signification: match / mismatch

Diagonale : { Action : avancer dans les deux(?) séquences
Score : +1 (match) / -1 (mismatch)

Signification : saut de l'ADN (intron)

Horizontale : { Action : on reste sur le même AA mais on saute les codons
Score : go (ouverture), ge (extension)

Signification : Interdit

Verticale : { Action : on ne saute pas AA
Score : N/A

Conditions initiales :

$E(i)(j) = 0$ (on part sans alignement ni score au début)

$t(i)(j) = -\infty$ (impossible d'aligner des AA sans codons)

$I(i)(j) = -\infty$ (on ne peut pas commencer directement dans un intron sans position dans la séquence)

$I(i)(j) = go + ge * j$ (un intron peut exister avant le premier exon, pénalisé selon sa longueur).

Réurrence :

$$E(i)(j) = \max \begin{cases} E(i-1)(j-1) + S(p_i, aa(c_j)), \\ I(i-1)(j-1) + S(p_i, aa(c_j)) \end{cases}$$

$$I(i)(j) = \max \begin{cases} E(i)(j-1) + g_0 \\ I(i)(j-1) + g_e \end{cases}$$

avec, E : état "exon"

I : état "intron"

Backtracking

- une fois la matrice remplie :

- On part de la dernière case ($\max(E(m)(m), I(m)(m))$),
- On remonte :
 - les diagonales \Rightarrow segments d'exons,
 - les horizontales \Rightarrow segments d'introns.

• Cela permet de reconstituer :

- la position des exons dans la séquence d'ADN,
- les zones d'introns entre eux,
- l'alignement optimal entre la protéine et le gène.

Résultat attendu

À la fin de l'algorithme, on obtient :

- les positions des exons et des introns dans la séquence d'ADN,
- l'alignement optimal entre la protéine et la séquence génomique,
- le score d'alignement maximal qui mesure la cohérence du gène trouvé.

2) Identifions le nom et la fonction de la protéine X :

Les deux (2) premiers résultats correspondent à des protéines hypothétiques (non caractérisées) donc, sans fonction connues.

La première correspondance fonctionnelle est la protéine ribosomal 40S 5'27'-like (avec 95.65% d'identité et une t-value de 5×10^{-57}).

Fonction: La protéine ribosomale 40S S97-like. Elle agit comme un constituant structural du ribosome, se liant à un ion Zinc (Zn^{2+}), ce qui stabilise sa structure.

Sa fonction principale est de participer à la traduction, c'est à dire, la synthèse des protéines en facilitant la traduction de l'ARN messager en acides aminés.

Cluster Composition	Cluster Ancestor	Cluster Representative Sequence	Max Score	Total Score	Query Cover	E value	Per. Ident	Acc. Len	Accession
Click the to see the cluster contents									
<input checked="" type="checkbox"/> 4 member(s), 4 organism(s)	crab-eating macaque	hypothetical protein EGM_16037 [Macaca fascicularis]	188	188	95%	1e-59	96.84%	113	EHH63131.1
<input checked="" type="checkbox"/> 1 member(s), 1 organism(s)	giant pangolin	hypothetical protein MC885_007588 [Smutsia gigantea]	183	183	91%	1e-57	95.60%	114	KAK2505348.1
<input checked="" type="checkbox"/> 1 member(s), 1 organism(s)	Damara mole-rat	40S ribosomal protein S27-like [Fukomys damarensis]	181	181	90%	5e-57	95.65%	105	XP_010639513.1

Functionⁱ

Cofactorⁱ

Zn²⁺ (UniProtKB | Rhea | CHEBI:29105) Automatic Annotation

Note: Binds 1 zinc ion per subunit. Automatic Annotation

Keywordsⁱ

Molecular function	#Ribonucleoprotein	Automatic Annotation
Ligand	#Ribosomal protein	Automatic Annotation
	#Metal-binding	Automatic Annotation

Definition

Protein of the ribosome, large ribonucleoprotein particles where the translation of messenger RNA (mRNA) into protein occurs. They are both free in the cytoplasm and attached to membranes of eukaryotic and prokaryotic cells. Ribosomes are also present in all plastids and mitochondria, where they translate organelle-encoded mRNA.

Category

Molecular function

Repliement d'ARN

1) Dessinons un repliement 2D qui illustre les sous-structures principales de la molécule:

Notation: (((((...)))))) ou ((.(((...).).))))



a a c g g a c c u a c a g g u g u a c c u

Bonus:

Données :

* Géquence d'ARN $S = s_1 \dots s_n$

* Appariements permis : A-U, G-C (on peut aussi accepter U-A et C-G)

* Empilement : un appariement (i, j) tel que $(i+1, j-1)$ est aussi apparié.

Modèle :

* $P[i, j]$: maximum d'empilement sur le segment $s_i \dots s_j$ (structure libre)

* $C[i, j]$: maximum d'empilement sur $s_i \dots s_j$ en imposant que i est apparié avec j (structure formée par (i, j)).

NB: si i et j ne sont pas compatibles alors $C[i, j] = -\infty$

Conditions initiales:

* Pour tout i : $P[i, i] = 0$, $P[i, i-1] = 0$

* Pour tout $i > j$: $P[i,j] = 0$, $C[i,j] = -\infty$

* Pour tout $i < j$ non compatibles : $C[i,j] = -\infty$

Récurrence

Table fermée C :

- Si s_i et s_j sont compatibles :

$$C[i,j] = \max \begin{cases} P[i+1,j-1] & \text{si } (i,j) \text{ apparié dans empilements immédiats} \\ 1 + C[i+1,j-1] & \text{empilement si } (i+1,j-1) \text{ est aussi apparié} \end{cases}$$

Table libre P :

$$P[i,j] = \max \begin{cases} P[i+1,j] & \text{si } i \text{ non apparié} \\ P[i,j-1] & \text{si } j \text{ non apparié} \\ \max_{k: s_i \text{ compatible avec } s_k} (C[i,k] + P[k+1,j]) & \text{si } i \text{ apparié à } k \end{cases}$$

Backtracking :

On part de $P[1,n]$.

A chaque cellule, mémoriser un pointeur pendant le remplissage

Dans $P[i,j]$

* Si $P[i,j] = P[i+1,j]$, marquer i comme apparié, aller sur $(i+1,j)$.

* Sinon, si $P[i,j] = P[i,j-1]$ => marquer j comme non apparié, aller sur $(i,j-1)$.

* Sinon, il existe k compatible avec i , tel que :

$$P[i,j] = C[i,k] + P[k+1,j]$$

- ajouter la paire (i,k)

- on backtrack d'abord dans $C[i,k]$ puis dans $P[k+1,j]$

Dans $C[i,j]$: on sait que (i,j) est apparié :

* Si $C[i,j] = P[i+1,j-1]$ alors (i,j) apparié sans

empilement immédiat, on va dans $P[i+1, j-1]$.

* Sinon, on a : $C[i, j] = 1 + C[i+1, j-1]$ alors (i, j) et $(i+1, j-1)$ forment un empilement.

- Enregistrer la paire $(i+1, j-1)$ au backtracking récursif;

- Aller dans $[i+1, j-1]$ et continuer.

A la fin, l'ensemble des paires reconstituées donnent le repliement, le nombre d'empilement est $P[1, m]$.

Résultat attendu :

- Score optimal d'empilement $P[1, m]$.

- liste des paires (i, j) du repliement.

NB :

les sections recherche d'introns et BLAST (question 1.b) et Repliement d'ARN (question bonus) ont été réalisées par nous même, à partir des explications fournies par ChatGPT qui ont permis de bien comprendre et d'appliquer les algorithmes.