M1 - BIF
TP4 - Alignement de séquences par programmation dynamique

Victor Levallois, Claire Lemaitre, Pierre Peterlongo

2023-2024

Pour implémenter l'algorithme d'alignement global de deux séquences, nous proposons d'utiliser la programmation orientée objet de Python. Nous allons définir une classe `DynamicMatrix` qui contient comme attributs :

— les deux séquences à aligner, S et T,
— la matrice de programmation dynamique,
— le système de score : les valeurs des scores d'un $match$, d'un $mismatch$ et d'un gap.

Vous trouverez dans le fichier `dynamicProg.py` sous Moodle, le début de l'implémentation de cette classe (en Python3) : le constructeur de la classe (définition des attributs et initialisation de la matrice avec des 0) ainsi qu'une méthode pour afficher la matrice. Il vous reste à implémenter les méthodes de cette classe qui permettent d'obtenir l'alignement global optimal entre deux séquences. Note : on n'utilisera aucune bibliothèque extérieure, telle que `numpy` ou `BioPython`.

1 Préliminaires

Q 1. Quel type d'objet python est utilisé pour stocker la matrice de programmation dynamique ? Avec quelle commande accède-t-on à la valeur de la cellule en face de la i-ème lettre de S et la j-ième de T ?

Q 2. Écrire une méthode `score` qui prend en argument 2 caractères et qui renvoie le score d'un *match* si les deux caractères sont égaux et le score d'un *mismatch* sinon.

2 Alignement global

Q 3. Écrire une méthode `initGlobal` qui initialise la matrice pour l'alignement global (première ligne et première colonne).

Q 4. Écrire une méthode `fill` qui remplit la matrice selon l'algorithme de Needleman-Wunsch (formule de récurrence avec les trois cases voisines en haut et à gauche), et renvoie le score du meilleur alignement global des deux séquences.

Q 5. Proposer une méthode `printGlobalAln` qui affiche **un** alignement de meilleur score de S contre T comme dans l'exemple qui suit (dans cet exemple, le système de score suivant est utilisé : $match = 2, mismatch = -1, gap = -2$) et qui renvoie son pourcentage d'identité.

Exemple :

		A	A	T	G	A	A	T	C
	0	-2	-4	-6	-8	-10	-12	-14	-16
G	-2	-1	-3	-5	-4	-6	-8	-10	-12
G	-4	-3	-2	-4	-3	-5	-7	-9	-11
A	-6	-2	-1	-3	-5	-1	-3	-5	-7
T	-8	-4	-3	1	-1	-3	-2	-1	-3
A	-10	-6	-2	-1	0	1	-1	-3	-2
G	-12	-8	-4	-3	1	-1	0	-2	-4
C	-14	-10	-6	-5	-1	0	-2	-1	0

```
1 alignement optimal :
--GGATAGC
AATGA-ATC

pcId = 44.4 %
```

3 Heuristique : alignement contraint dans une bande

Q 6. Implémenter une heuristique de l'alignement global qui consiste à contraindre l'alignement autour de la diagonale dans une *bande*. Dans cette heuristique, on gagne du temps puisqu'on ne remplit pas la matrice en entier, mais seulement une sous-partie : les cellules situées autour de la diagonale (bande). C'est une heuristique car on n'explore pas tout l'espace de recherche, on peut donc manquer un alignement de score maximal qui "deborderait" de la bande. La largeur horizontale de la bande sera déterminée par le paramètre `width`, qui représente le nombre de cellules à gauche (et à droite) des cellules (i, i) à considérer dans la bande, soit une largeur de bande égale à $2\times$`width`$+1$. On se restreindra au cas où les 2 séquences sont de même taille et on se contentera de calculer le score de l'alignement global (sans afficher l'alignement).

4 Tests de l'heuristique

Dans cette section nous utiliserons $match = 2, mismatch = -1, gap = -2$.

Q 7. Générer des couples de séquences d'ADN aléatoires de taille 500, 1000, et 2000. Pour chacune de ces tailles, tester l'alignement global exact et l'alignement global heuristique (score et temps de calcul), avec différentes valeurs de `width` : 1, 5, 10, 20 et 40. Vous présenterez les résultats sous la forme de tableaux, un tableau par taille de séquence (cf. exemple ci-dessous). Quelles conclusions tirer des résultats obtenus ?

	taille =	500pb
width	score	temps
1		
5		
10		
20		
40		
Exact		

Q 8. Effectuer les mêmes tests avec des données plus réalistes : une séquence biologique réelle ainsi que 3 versions mutées de cette-ci, fichier `sequences_puceron.fasta` sur moodle. Les conclusions sont-elles différentes avec ces données ?

5 Bonus : alignement en mémoire linéaire

Q 9. Implémenter la version heuristique de telle façon que la complexité en mémoire soit linéaire avec la largeur de la bande : $\mathcal{O}($`width`$)$. On ne cherchera pas à afficher l'alignement, seulement le meilleur score.

Pour cela, vous créerez une deuxième classe `DMLinearMem` avec les méthodes nécessaires.