Master Bioinformatique parcours MISO 2021-22

**Méthodes pour l'Analyse des Séquences en Bioinformatique**

---

# Exercices de programmation dynamique

## Exercice 0: Découverte de la récursivité

Vous pouvez trouver des exercices corrigés simples sur le site
`https://developpement-informatique.com/article/102/exercices-corriges-de-recursivite-en-python--serie-12`

## Exercice 1: Pratique de l'alignement

Construisez la matrice de programmation dynamique pour l'alignement semi-global optimal des deux séquences `TTACTGTG` et `ACTGAGAC` avec le système de score: identité +2, substitution -1, insertion ou délétion -1. Quel est l'alignement associé? Retracez-le sur la matrice de programmation dynamique.

## Exercice 2: Dissimilarité versus similarité

Pour comparer deux séquences, au lieu de chercher à maximiser la *similarité*, on peut minimiser la *distance*, en comptant le nombre de différences. Le système de score est alors du type :

`Indel`$(x) > 0$, coût de l'insertion ou de la délétion du nucléotide $x$,
`Diff`$(x,y) > 0$ si $x \neq y$, et 0 si $x = y$.

**Question 1.** Donnez les formules de programmation dynamique pour l'alignement global de deux séquences en minimisant la distance entre celles-ci.

**Question 2.** Que pensez-vous du problème de l'alignement local ?

## Exercice 3: Les palindromes génétiques

Sur l'ADN, un *palindrome génétique* est un mot de la forme $u\bar{u}$ où $u$ est un mot et $\bar{u}$ est obtenu à partir de $u$ en l'inversant et en le complémentant : $A$ est le complément de $T$, $C$ est le complément de $G$. Par exemple, si $U = ACCTGC$, en l'inversant on obtient $CGTCCA$, et en complémentant $\bar{U} = GCAGGT$. Le mot $ACCTGCGCAGGT$ est donc un palindrome génétique.

Le problème du *palindrome maximal* est le suivant :

Donnée : une séquence $S$ d'ADN
Question : quel est le plus long sous-mot de $S$ qui soit un palindrome génétique?

Ainsi, dans $ACCGGATT$, le palindrome maximal est $CCGG$, qui est de longueur 4.

Le professeur *Archibalde* propose une solution par programmation dynamique à l'aide d'un

$$P(i,j) = j - i + 1, \text{ si } S(i..j) \text{ est un palindrome, 0 sinon.}$$

**Question 1.** Donnez la formule de récurrence qui permet de calculer $P(i,j)$, en traitant les cas de base et le cas général.

**Question 2.** En déduire un algorithme pour trouver le palindrome le plus long dans une séquence. Quelle est sa complexité en temps et en mémoire ?

## Exercice 4: Réplication hasardeuse

Une bactérie est atteinte par un virus qui affecte la machinerie de la réplication de la sorte

- chaque `A` peut être remplacé par 2 `A`,
- chaque `C` peut être remplacé par 2 `C`,

Notez que la multiplication des bases n'est pas systématique, et il se peut qu'à une position donnée la copie soit correcte.

**Question 1.** Donnez un algorithme qui pour deux séquences $U$ et $V$ détermine si $U$ peut être une version infectée de $V$.

Le virus a muté, et en plus d'induire une copie multiple d'une position, il est également possible que la base, `A`, `C`, `G` ou `T`, soit oubliée, provoquant une délétion.

**Question 2.** Modifiez l'algorithme précédent pour prendre en compte ce nouveau phénomène.

## Exercice 5: Tous les alignements locaux

On veut aligner les deux séquences *tacgcgtggattgatc* et *acgatcgatgata* avec l'algorithme d'alignement local, de Smith et Waterman. On obtient la matrice de programmmation dynamique suivante.

| | | $t$ | $a$ | $c$ | $g$ | $c$ | $g$ | $t$ | $g$ | $g$ | $a$ | $t$ | $t$ | $g$ | $a$ | $t$ | $c$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $a$ | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| $c$ | 0 | 0 | 0 | 2 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| $g$ | 0 | 0 | 0 | 1 | 3 | 2 | 2 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| $a$ | 0 | 0 | 1 | 0 | 2 | 1 | 1 | 0 | 0 | 0 | 2 | 1 | 0 | 0 | 2 | 1 | 0 |
| $t$ | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 2 | 1 | 0 | 1 | 3 | 2 | 1 | 1 | 3 | 2 |
| $c$ | 0 | 0 | 0 | 1 | 0 | 2 | 1 | 1 | 0 | 0 | 0 | 2 | 1 | 0 | 0 | 2 | 4 |
| $g$ | 0 | 0 | 0 | 0 | 2 | 1 | 3 | 2 | 2 | 1 | 0 | 1 | 0 | 2 | 1 | 1 | 3 |
| $a$ | 0 | 0 | 1 | 0 | 1 | 0 | 2 | 1 | 1 | 0 | 2 | 1 | 0 | 1 | 3 | 2 | 2 |
| $t$ | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 3 | 2 | 1 | 1 | 3 | 2 | 1 | 2 | 4 | 3 |
| $g$ | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 2 | 4 | 3 | 2 | 2 | 1 | 3 | 2 | 3 | 2 |
| $a$ | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 3 | 2 | 4 | 3 | 2 | 2 | 4 | 3 | 2 |
| $t$ | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 2 | 1 | 3 | 5 | 4 | 3 | 3 | 5 | 4 |
| $a$ | 0 | 0 | 2 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 2 | 4 | 3 | 2 | 4 | 4 | 3 |

Le poids d'une identité est 1, d'une substitution est $-1$ et d'une insertion ou d'une délétion est de $-3$

**Question 1.** Dans un premier temps, on considère les alignements locaux réduits à une seule région de forte similarité. Dans ce cas, combien il a-t-il d'alignements locaux optimaux ? Quel est leur score ? Quels sont-ils ?

**Question 2.** On s'intéresse maintenant aux alignements locaux pouvant être composés de plusieurs zones similaires *colinéaires*, c'est-à-dire sans croisement ni chevauchement. Donnez un tel alignement dont le score total est égal à 8. Le score total est obtenu en prenant la somme des scores des zones qui le composent.

**Question 3.** Expliquez comment ce problème peut être modélisé comme une recherche de chemin dans un graphe orienté acyclique.

## Exercice 6: Alignements co-optimaux

Niveau expert

On considère le problème de l'alignement global entre deux séquences, avec l'algorithme de Needleman et Wunsh. Des alignements sont *co-optimaux* s'ils ont le même score et que ce score est maximal.

**Question 1.** Pour le jeu de scores -2 pour une insertion ou une délétion, -1 pour une substitution et +1 pour une identité, construisez un exemple de deux séquences acceptant au moins deux alignements optimaux distincts.

**Question 2.** Ecrire un algorithme qui à partir de la matrice de programmation dynamique de l'algorithme de Needleman et Wunsh détermine le nombre d'alignements co-optimaux.

## Exercice 7: Alignement de séquences codantes

Niveau expert

Le problème est le suivant : on souhaite écrire un algorithme qui permette d'aligner deux séquences d'ADN non pas en fonction des nucléotides `A`, `C`, `G` et `T`, mais en fonction des séquences protéiques obtenues après traduction. Par exemple, les deux séquences `TCACGATTCCTG` et `TAGCAGGTTA` s'alignent ainsi

```
  T C A C G A T T C C T G
  | | | | | |       | | |
T A G C A G G - - - T T A
```

car `TCACGATTCCTG` se traduit en `SRFL` (à partir de la première position) et `TAGCAGGTTA` en `SRL` (à partir de la deuxième position).

Le point délicat est que la phase de lecture n'est pas connue pour les séquences à aligner. Il y a donc 3 phases possibles par séquence (on ne considère pas les phases correspondant au brin complémentaire). Il faut également tenir compte des codons STOP, qui marquent l'arrêt de la traduction. Par convention, on décide qu'au premier codon STOP rencontré dans la phase, le score de similarité est $-\infty$.

**Question 1.** Écrivez une formule de récurrence pour calculer la similarité `Sim` de deux séquences d'ADN pour ce problème. Pour cela, vous diposez d'une fonction `Sub` qui à un couple de triplets de nucléotides associe le score de similarité entre les deux acides aminés correspondants, d'une fonction `Indel` qui à un triplet de nucléotides associe le coût de l'insertion ou de la délétion de l'acide aminé correspondant, ainsi que d'une fonction booléenne `STOP` qui permet de tester si un triplet de nucleotides est un codon STOP.

**Question 2.** En déduire un algorithme quadratique qui permet de construire un alignement optimal. Comment retrouver le score correspondant à l'alignement optimal ?

## Exercice 8: Pas d'insertion après une délétion, s'il vous plait

Niveau expert

On souhaite modifier l'algorithme d'alignement global avec pénalités de gaps affines (celui avec les trois matrices $a$, $b$ et $c$) de manière à interdire une insertion à la suite d'une délétion, ou une délétion à la suite d'une insertion. Les deux configurations suivantes sont donc proscrites.

```
A  -          - A
-  G          T -
```

Comment modifier les formules de récurrence pour calculer $a$, $b$ et $c$ pour prendre en compte cette contrainte ?

## Exercice 9: Algorithme de Hirschberg

Niveau expert

Soient $U$ et $V$, deux séquences d'ADN de longueur respective $m$ et $n$. Vous avez vu en cours qu'il était possible de construire l'alignement global optimal entre $U$ et $V$ en temps et en mémoire quadratique, grâce à l'algorithme de Needleman et Wunsh. Dans cet exercice, nous allons voir comment une méthode alternative permet de construire l'alignement en espace linéaire, tout en restant en temps quadratique.

On note $\mathcal{A}$ l'alignement global optimal entre deux séquences et $+$ la concaténation entre alignements.

**Question 1.** Soit $\mathcal{A}(U,V)$, un alignement global optimal entre $U$ et $V$, et soit $i$, une position $U$ $(0 < i \leq m)$. Montrez qu'il existe une position $j$ de $V$ $(0 < j \leq n)$ telle que l'un des deux cas ci-dessous est satisfait:

- Cas 1

$$\mathcal{A}\begin{pmatrix} U \\ V \end{pmatrix} = \mathcal{A}\begin{pmatrix} U(1..i-1) \\ V(1..j-1) \end{pmatrix} + \begin{pmatrix} U(i) \\ V(j) \end{pmatrix} + \mathcal{A}\begin{pmatrix} U(i+1..m) \\ V(j+1..n) \end{pmatrix}$$

- Cas 2

$$\mathcal{A}\begin{pmatrix} U \\ V \end{pmatrix} = \mathcal{A}\begin{pmatrix} U(1..i-1) \\ V(1..j-1) \end{pmatrix} + \begin{pmatrix} U(i) \\ - \end{pmatrix} + \mathcal{A}\begin{pmatrix} U(i+1..m) \\ V(j..n) \end{pmatrix}$$

**Question 2.** Vrai ou faux ? Pour toute valeur de $i$, $0 \leq i < m$, l'algorithme de Needleman et Wunsh permet de calculer l'ensemble des scores d'alignements globaux entre $U(1..i-1)$ et tous les préfixes de $V$ en espace linéaire et en temps quadratique. Pourquoi ?

**Question 3.** Montrez, en décrivant l'algorithme, que pour chaque $i$ on peut calculer l'ensemble des scores d'alignements globaux entre $U(i+1..m)$ et l'ensemble de tous les suffixes de $T$.

**Question 4.** A partir des deux questions précédentes, expliquez comment on peut écrire une fonction `Score` qui pour une valeur de $i$ donnée détermine le cas (1 ou 2) et la valeur de $j$ et de la question 1. Quelle est sa complexité ?

**Question 5.** En fixant $i = m/2$, déduisez en un algorithme *récursif* qui calcule l'alignement global optimal entre $U$ et $V$. Quelle est sa complexité ?