Informatique Génomique — Analyse de Séquences

TD #1 **07 Mars 2011**

Rédacteur: *Stéphane Vialette*

Il s'agit de programmer l'algorithme de Needleman-Wunsch (alignement global) avec retour arrière. Nous utiliserons un système de score paramétrable : coût d'un *match*, coût d'un *mismatch*, et coût d'un *gap*.

Pour simplifier, nous supposerons dans un premier temps que ce système de score est spécifié de la façon suivante : b

```
match    =  1.
mismatch = -1.
gap      = -1.
s = {
    'AA': match,    'AG': mismatch, 'AC': mismatch, \
    'AT': mismatch, 'GA': mismatch, 'GG': match,    \
    'GC': mismatch, 'GT': mismatch, 'CA': mismatch, \
    'CG': mismatch, 'CC': match,    'CT': mismatch, \
    'TA': mismatch, 'TG': mismatch, 'TC': mismatch, \
    'TT': match,\
    }
```

Question 1

Écrire la fonction `do_global_alignment` qui prend en en argument deux chaînes `s1` et `s2` (sur l'alphabet `A, C, G, T`) et qui retourne la table de programmation dynamique associée.

Question 2

Écrire la fonction `give_one_optimal_alignment` qui prend en argument la table de programmation dynamique et qui retourne un alignement optimal. Celui-ci doit pouvoir être affiché. L'alignement global peut être donné sous la forme d'une paire de chaînes de caractères ou via un objet dédié.

Par exemple :

```
$ python global_align --oneoptimal
Calculating....

# Using:  match=1.0; mismatch=-1.0; gap=-1.0
GAGACCGCCATGGCGACCCTGGAAAAGCTGATGAAGGCCCT
AGACCCAATGCGACCCTGAAAAAGCTGATGAAGGCCTTTTT

# One optimal global alignment is
GAGACCGCCATGGCGACCCTGGAAAAGCTGATGAAGGCC___CT
_AGACC_CAAT_GCGACCCTGAAAAAGCTGATGAAGGCCTTTTT

G=====G=C==G=========G================___C=
_=====_=A==_=========A================TTTT=

# 35  matches;  3  mismatches;  6  gaps.
# score 26.0 / 41
$
```

Question 3

Plus difficile est la construction de toute les alignments optimaux. Écrire la fonction `give_all_optimal_alignments` qui prend en argument la table de programmation dynamique et qui retourne la liste de tous les alignements optimaux. Une option de commande (ici `--short`) permet de limiter la sortie.

Par exemple,

```
$ python global_align --alloptimal --short
Calculating....

# Using:  match=1.0; mismatch=-1.0; gap=-1.0
GAGACCGCCATGGCGACCCTGGAAAAGCTGATGAAGGCCCT
AGACCCAATGCGACCCTGAAAAAGCTGATGAAGGCCTTTTT

# 20 optimal alignments
1 .
GAGACCGCCATGGCGACCCTGGAAAAGCTGATGAAGGCC___CT_
_AGACC_CAAT_GCGACCCTGAAAAAGCTGATGAAGGCCTTTTTT

2 .
GAGACCGCCATGGCGACCCTGGAAAAGCTGATGAAGGCC___CT_
_AGACC_CAATG_CGACCCTGAAAAAGCTGATGAAGGCCTTTTTT

3 .
GAGACCGCCATGGCGACCCTGGAAAAGCTGATGAAGGCC__C_T_
_AGACC_CAAT_GCGACCCTGAAAAAGCTGATGAAGGCCTTTTTT
```

4 .
GAGACCGCCATGGCGACCCTGGAAAAGCTGATGAAGGCC__C_T_
_AGACC_CAATG_CGACCCTGAAAAAGCTGATGAAGGCCTTTTTT

5 .
GAGACCGCCATGGCGACCCTGGAAAAGCTGATGAAGGCC_C__T_
_AGACC_CAAT_GCGACCCTGAAAAAGCTGATGAAGGCCTTTTTT

6 .
GAGACCGCCATGGCGACCCTGGAAAAGCTGATGAAGGCC_C__T_
_AGACC_CAATG_CGACCCTGAAAAAGCTGATGAAGGCCTTTTTT

7 .
GAGACCGCCATGGCGACCCTGGAAAAGCTGATGAAGGCCC___T_
_AGACC_CAAT_GCGACCCTGAAAAAGCTGATGAAGGCCTTTTTT

8 .
GAGACCGCCATGGCGACCCTGGAAAAGCTGATGAAGGCCC___T_
_AGACC_CAATG_CGACCCTGAAAAAGCTGATGAAGGCCTTTTTT

9 .
GAGACCGCCATGGCGACCCTGGAAAAGCTGATGAAGGCC__CT__
_AGACC_CAAT_GCGACCCTGAAAAAGCTGATGAAGGCCTTTTTT

10 .
GAGACCGCCATGGCGACCCTGGAAAAGCTGATGAAGGCC__CT__
_AGACC_CAATG_CGACCCTGAAAAAGCTGATGAAGGCCTTTTTT

11 .
GAGACCGCCATGGCGACCCTGGAAAAGCTGATGAAGGCC_C_T__
_AGACC_CAAT_GCGACCCTGAAAAAGCTGATGAAGGCCTTTTTT

12 .
GAGACCGCCATGGCGACCCTGGAAAAGCTGATGAAGGCC_C_T__
_AGACC_CAATG_CGACCCTGAAAAAGCTGATGAAGGCCTTTTTT

13 .
GAGACCGCCATGGCGACCCTGGAAAAGCTGATGAAGGCCC__T__
_AGACC_CAAT_GCGACCCTGAAAAAGCTGATGAAGGCCTTTTTT

14 .
GAGACCGCCATGGCGACCCTGGAAAAGCTGATGAAGGCCC__T__
_AGACC_CAATG_CGACCCTGAAAAAGCTGATGAAGGCCTTTTTT

15 .
GAGACCGCCATGGCGACCCTGGAAAAGCTGATGAAGGCC_CT___

```
_AGACC_CAAT_GCGACCCTGAAAAAGCTGATGAAGGCCTTTTTT

16 .
GAGACCGCCATGGCGACCCTGGAAAAGCTGATGAAGGCC_CT___
_AGACC_CAATG_CGACCCTGAAAAAGCTGATGAAGGCCTTTTTT

17 .
GAGACCGCCATGGCGACCCTGGAAAAGCTGATGAAGGCCC_T___
_AGACC_CAAT_GCGACCCTGAAAAAGCTGATGAAGGCCTTTTTT

18 .
GAGACCGCCATGGCGACCCTGGAAAAGCTGATGAAGGCCC_T___
_AGACC_CAATG_CGACCCTGAAAAAGCTGATGAAGGCCTTTTTT

19 .
GAGACCGCCATGGCGACCCTGGAAAAGCTGATGAAGGCCCT____
_AGACC_CAAT_GCGACCCTGAAAAAGCTGATGAAGGCCTTTTTT

20 .
GAGACCGCCATGGCGACCCTGGAAAAGCTGATGAAGGCCCT____
_AGACC_CAATG_CGACCCTGAAAAAGCTGATGAAGGCCTTTTTT
$
```