

## TP: ADN Mining – Expressions Régulières

Vous trouverez le nécessaire ici : <http://www.math-info.univ-paris5.fr/~lomn/Cours/BC/>

### Exemple 1

```
import os
import sys
import re

#os.system("wget http://www.gutenberg.org/cache/epub/20/pg20.txt")

file = open(sys.argv[1] + ".txt")

for line in file:
    if re.match('people', line):
        print(line) #affiche la ligne

#ou os.system("more " + sys.argv[1] + ".txt | grep -o -E [a-zA-Z]{15} | wc -w")

findall = re.findall("kiki", "Le kiki de mon kiki est mon kiki")
print(findall)

search = re.search("kiki", "Le kiki de mon kiki est mon kiki")
print(search)

file.close()
```

### Exemple 2

```
def searchpattern(target, search_string) :
    matches = []
    for i in range(0, len(search_string)) :
        #print("Compare %i and %i-long strings."%(len(target),
len(search_string[i:i+len(target)])))
        if target == search_string[i : i+len(target)] :
            matches.append(i)
    return matches

def searchstop(search_string) :
    rslt = []
    #stoppatterns = ["UAA", "UCA", "UAC"]
    stoppatterns = ["TAA", "TCA", "TAC"]

    for pat in stoppatterns :
        rslt = rslt+searchpattern(pat, search_string)
    return rslt

search_string =
"CCAAATTCTTCGGGACCCTGGGGGGTTAAATTACCCTGACCCTGATGCATGGTATGTACAGTAGACTAGGACAACCC
TGGGGTAGA"

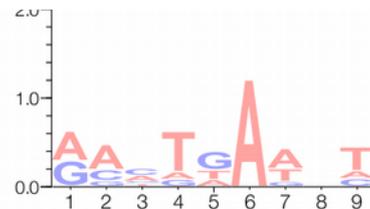
matches = searchstop(search_string)

print("STOP codons were found at : %s.\nPlease take position in account for
eventual frameshifts."%str(matches))
```

### Exercice 1 . (p. 110 du cours de F. Dardel)

Les bactéries n'ont pas de nucléosomes, mais possèdent néanmoins des protéines qui condensent l'ADN génomique, pour former ce que l'on appelle un nucléoïde. Une de ces protéines, H-NS, se fixe fortement partout à l'ADN, mais plus sélectivement sur certaines séquences contenant un motif d'environ 9 nucléotides de long. Ci-dessous est représenté un «logo» de ces séquences préférées par H-NS, obtenue par une analyse statistique. A la position 8, il n'y a aucune préférence, c'est pourquoi la position du logo est vide.

Trouvez puis codez une expression régulière qui puisse reconnaître des séquences qui correspondent à des motifs préférés par H-NS.



### Exercice 2 .

Écrire et coder une expression régulière qui reconnaisse simultanément les deux séquences nucléotidiques suivantes `CCTCTAAAAAATTATT` et `CCTCTAAAAAAGTATT` (le nucléotide souligné est variable). Essayez d'écrire l'expression la plus compacte possible.

### Exercice 3 .

On aligne des séquences d'une famille d'enzymes, des glutathion-S-transférases. Une région de l'alignement des séquences protéiques correspondantes est montrée ci-dessous (les positions conservées ou semi-conservées sont indiquées en gras) :

```
...FLQN--KAFLTGPHISLADL... GST Homme
...FLQD--KAFIIGSEISLADL... GST Poulet
...WLKNGNGQYLLGGLRPSIADL... GST Soja
...FLEG--QEYAAGNDLTIADL... GST Drosophile
...WLRD--REYVCGDEVSYADL... GST Methylobacterium
```

Écrire et coder une expression régulière qui reconnaisse le motif conservé ci-dessus ?

### Exercice 4.

L'arginine est codée par 6 codons : CGA, CGC, CGT, CGC, AGA et AGG. Écrivez et codez une expression régulière qui reconnaisse exactement ces six codons et seulement ceux-ci. Trouvez une forme concise de cette expression.

### Exercice 5.

Le motif suivant reconnaît un type de séquence particulier dans le génome des vertébrés :

`CG ([AGT]* ou C C* [AT])* C* CG`

Analysez ce motif et expliquez ce qu'il reconnaît exactement.

**Remarque tirée du cours:** Malgré leur richesse, les expressions régulières ont certaines limites. Elles ne permettent par exemple pas de reconnaître des motifs définis par une règle, comme par exemple des palindromes ou des séquences répétées inversées de longueur quelconque.

ANNEXE

## Expressions régulières et Recherche de Motifs

<http://www.shellunix.com/regexp.html>

<http://www.catonmat.net/download/perl1line.txt>

- ▶ codon de démarrage bactérien : [AG]TG
- ▶ codon stop : T(GA ou A[AG])
- ▶ codon codant (non-stop) : [ACG]. . ou T(CT. ou G[CGT] ou A[CT])
- ▶ cadre ouvert de lecture : [AG]TG ([ACG]. . ou T(CT. ou G[CGT] ou A[CT]))\* T(GA ou A[AG])
- ▶ Shine-Dalagarno (RBS) : AGGA ou GGAG ou GAGG
- ▶ Démarrage de la traduction (*E. coli*) : (AGGA ou GGAG ou GAGG) . {6,12} [AG]TG
- ▶ ADN courbe : (AAAA.....){4,} ou (TTTT.....){4,}

*p. 109-110 Dardel*

### Recherche de motif simple

<code>\$dna =~ /GAATTC/ ;</code>	
<code>\$dna =~ /GGG[GATC]CCC/ ;</code>	
<code>\$dna =~ /GAATTC AAGCTT/ ;</code>	

Métacaractères (on utilise \ pour indiquer qu'on veut utiliser le symbole d'écriture . par exemple et pas le métacaractère . )

.	Tout caractère sauf nouvelle ligne	\d (tout chiffre) et \D le contraire
^	Le début d'une ligne	\w (tout caractère hors ponctuation et invisible) \W le contraire
\$	La fin d'une ligne	\s (tout caractères invisibles) et \S le contraire

Le quantifieur {}. A{2} représente AA. A{2,4} représente AA ou AAA ou AAAA. {2,} signifie un nombre minimum de A accolé de 2 et {2} au ) plus de 2. A+ signifie A répété un nombre indéfini de fois.

?		0 ou 1 occurrence
+		1 ou plus d'occurrences
*		0 or more occurrences
{N,M}		Entre N et M occurrence
{N,}		Au moins N occurrences
{,M}		Pas plus que M occurrences

Exemples :

<code>\$french_postal_code =~ /\d\d\d\d\d/ ; \$frenc_postal_code =~ /\d{5}/;</code>	75004
<code>\$sequence_id =~ /\w+\.\d+/ ;</code>	M588200.2
<code>\$sequence =~ /^[GATCN]gatcn]+\$;/;</code>	<code>\$sequence =~ /^[GATCN]+\$;/i;</code>

<pre>/^&gt;\S+\s*.*\$/</pre> <p>Cette expression régulière cherche le motif M18580 isolé par un &gt; en début de ligne et un caractère de type espace blanc ensuite, puis n'importe quel chaîne jusque fin de ligne.</p> <p><b>Usage des parenthèses :</b></p> <pre>/^&gt;(\S+)\s*(.*)\$/</pre> <p>\$id=\$1 ; \$description=\$2 ;</p>	<p>&gt;M18580 Clone 305A4,complete sequence</p> <p>« Parse » la ligne pour segmenter en éléments essentiels.</p> <p><b>Affecte à des variables :</b></p> <p>\$id ou \$1 valent M18580 \$description ou \$2 valent «Clone 305A4,complete sequence»</p>
<p>[^...] classe complétée Tous les caractères sauf ceux énumérés</p> <pre>1. #!/usr/bin/perl 2. #check for non DNA character 3. my \$string='CCACATGCCAT*' ; 4. if (\$string =~ /^[^AGCT])/i { 5.     print 'Warning ! Found : \$1' ; 6. }</pre>	<p>Cherche si la chaîne de caractères contient des caractères autres que A,G,C,T</p>