

# 3I005

## Projet 3 : Analyse de séquences génomiques

Ariana Carnielli  
Yasmine Ikhelif

### Table des matières

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Préliminaires : données et lecture des fichiers</b>	<b>1</b>
<b>3</b>	<b>Annotation des régions promoteurs</b>	<b>2</b>
3.1	Description Empirique, préliminaires . . . . .	2
3.2	Simulation de séquences aléatoires . . . . .	3
3.3	Modèles de dinucléotides et trinucéotides . . . . .	7

## 1 Introduction

Les valeurs présentées dans ce document ont été calculées à l'aide des codes en Python dans le fichier `projet.py`, qui contient l'implémentation des fonctions demandées, et dans le Jupyter Notebook `Projet3.ipynb`, qui contient les appels de ces fonctions et l'affichage des résultats.

## 2 Préliminaires : données et lecture des fichiers

### Question 1

Quatre fichiers ont été donnés pour ce projet. Leurs noms et les nombres de séquences et de nucléotides dans chacun d'entre eux sont présentés dans le tableau ci-dessous.

Nom du fichier	Nombre de séquences	Total de nucléotides
<code>regulatory_seq_PH0.fasta</code>	5	4000
<code>regulatory_seqs_GAL.fasta</code>	7	5608
<code>regulatory_seqs_MET.fasta</code>	9	7200
<code>yeast_s_cerevisae_genomic_chr1-4.fna</code>	4	2515853

Les chromosomes de *S. cerevisae* donnés dans le fichier `yeast_s_cerevisae_genomic_chr1-4.fna` contiennent ainsi 2515853 nucléotides.

## Question 2

L'application de la fonction `nucleotide_frequency` à la liste des nucléotides de *S. cerevisiae* donnés dans le fichier `yeast_s_cerevisiae_genomic_chr1-4.fna` donne les fréquences présentées dans le tableau ci-dessous.

Nucléotide	Fréquence
A	0.29984
C	0.20110
G	0.20000
T	0.29907

## Questions 3 et 4

Les fonctions `logproba` et `logprobafast` ont été codées dans `projet.py` et testées dans `Projet3.ipynb`.

# 3 Annotation des régions promoteurs

## 3.1 Description Empirique, préliminaires

### Question 1

Les fonctions `code`, `inverse` et `comptage` ont été codées dans `projet.py` et testées dans `Projet3.ipynb`.

### Question 2

On fixe un mot  $w = w_0w_1\cdots w_{k-1}$  de longueur  $k$  et on considère une séquence aléatoire de longueur  $\ell$ ,  $S_0S_1\cdots S_{\ell-1}$  avec  $\ell \geq k$ . On suppose que les variables aléatoires  $S_0, \dots, S_{\ell-1}$  correspondant à chaque lettre sont indépendantes et identiquement distribuées (leur loi étant estimée par la fonction donnée `nucleotide_frequency`). Soit  $X_i$  la variable aléatoire Bernoulli qui vaut 1 si et seulement si le mot  $w$  apparaît dans  $S$  à la case  $i$ , c'est-à-dire  $S_i = w_0$ ,  $S_{i+1} = w_1$ ,  $\dots$ ,  $S_{i+k-1} = w_{k-1}$ . Remarquons que cela n'est possible que si  $i + k - 1 \leq \ell - 1$ , donc  $i \leq \ell - k$ . Comme les variables  $S_j$  sont indépendantes, on a

$$P(X_i = 1) = P(S_i = w_0)P(S_{i+1} = w_1)\cdots P(S_{i+k-1} = w_{k-1}).$$

De plus, comme les variables  $S_j$  sont identiquement distribuées,  $P(X_i = 1)$  ne dépend pas de  $i$  (mais dépend de  $w$ ). Notons  $p = P(X_i = 1)$ . Remarquons que  $p$  n'est rien d'autre que l'exponentielle de la valeur rendue par la fonction `logproba` lorsque son argument `liste_entiers` vaut  $w$  et son argument `m` contient la loi des  $S_j$ .

Soit  $X$  la variable aléatoire donnant le nombre d'occurrences de  $w$  dans  $S$ . On a alors

$$X = \sum_{i=0}^{\ell-k} X_i.$$

On remarque que, sauf dans le cas  $k = 1$ , les variables  $X_i$  ne sont pas indépendantes, donc  $X$  n'est pas forcément une variable aléatoire binomiale. Cependant, comme l'espérance est linéaire et  $\mathbb{E}(X_i) = P(X_i = 1) = p$  car  $X_i$  suit une loi de Bernoulli, on peut calculer l'espérance de  $X$  par

$$\mathbb{E}(X) = \sum_{i=0}^{\ell-k} \mathbb{E}(X_i) = \sum_{i=0}^{\ell-k} p = p(\ell - k + 1).$$

Cette formule a été implémentée dans la fonction `comptage_attendu` codée dans le fichier `projet.py` et testée dans le fichier `Projet3.ipynb`.

### Question 3

Les graphiques, obtenus dans le fichier `Projet3.ipynb`, sont données dans la Figure 1. On y remarque qu'il y a un écart important entre les nombres d'occurrence attendus et observés, qui peut être vu par la distance entre les points tracés et la diagonale, représentée en gris sur les figures. Cet écart est d'autant plus important que la longueur du mot recherché est grand. Ainsi, le modèle utilisé jusqu'à présent, qui suppose que deux lettres à des positions différentes sont des variables aléatoires indépendantes, ne permet pas d'expliquer les observations des occurrences des mots. Certains mots sont beaucoup plus fréquents que d'autres et cela ne peut pas être expliqué simplement par les fréquences de leurs lettres.

## 3.2 Simulation de séquences aléatoires

### Question 1

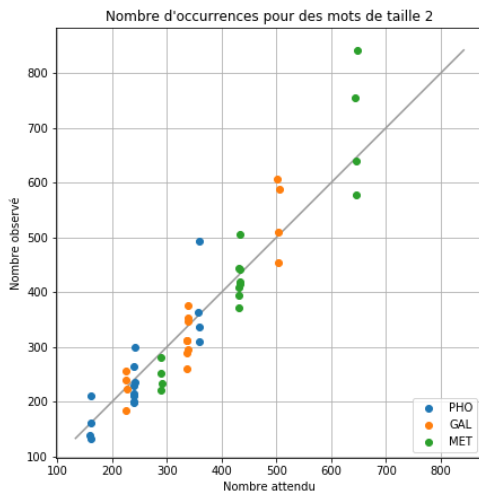
La fonction `simule_sequence` a été codée dans `projet.py` et testée dans `Projet3.ipynb`. Pour la création de la séquence, on suppose que les lettres à des positions différentes sont indépendantes.

### Question 2

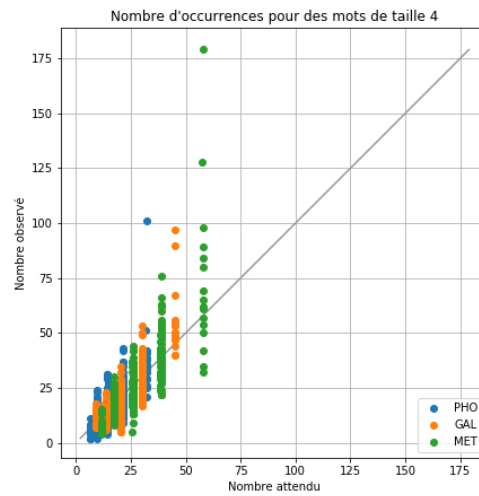
On a simulé des séquences de grande longueur et tracé le graphique avec les nombres attendus et observés de chaque mot de longueurs  $k$  données. Les résultats pour une séquence de taille 1000000 et  $k = 6$  sont donnés dans la Figure 2. On y observe que les points sont assez proches de la diagonale, comme attendu, car le calcul du nombre d'occurrences attendu se base sur l'indépendance des lettres de la séquence, utilisée pour la création de la séquence aléatoire. Ces points sont d'autant plus proches de la diagonale que la taille de la séquence générée est grande mais leur variance, et donc leur dispersion autour de la diagonale, augmente rapidement avec  $k$ .

### Question 3

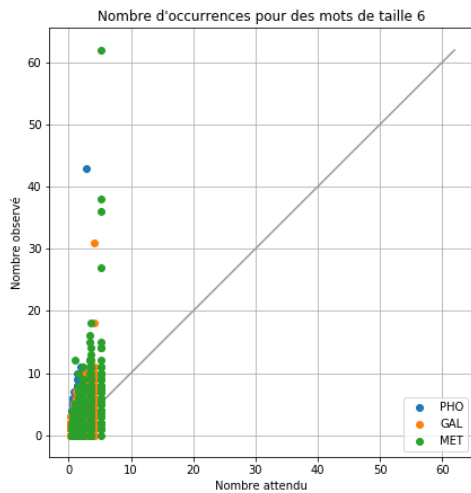
La fonction `proba_empirique` calculant la probabilité empirique d'un mot a été codée dans `projet.py` et testée dans `Projet3.ipynb`. Elle prend en argument



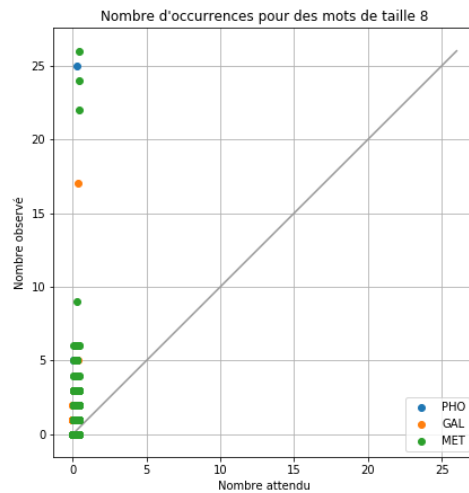
(a)



(b)



(c)



(d)

FIGURE 1 – Nombre d'occurrences attendu et observé pour (a)  $k = 2$ , (b)  $k = 4$ , (c)  $k = 6$  et (d)  $k = 8$ .

un mot, la taille  $1g$  de la séquence à simuler, les probabilités de chaque nucléotide et un nombre de simulations à faire et renvoie un dictionnaire tel que l'élément de clé  $i$  contient la probabilité estimée que ce mot apparaisse exactement  $i$  fois dans une séquence de taille  $1g$ .

#### Question 4

La Figure 3 donne les histogrammes avec les probabilités estimées du nombre d'occurrences des mots ATCTGC, ATATAT, TTTAAA et AAAAAA pour des suites de longueur 10000 en faisant 1000 simulations pour chaque mot. Le code pour créer ces figures est donné dans le fichier `Projet3.ipynb`.

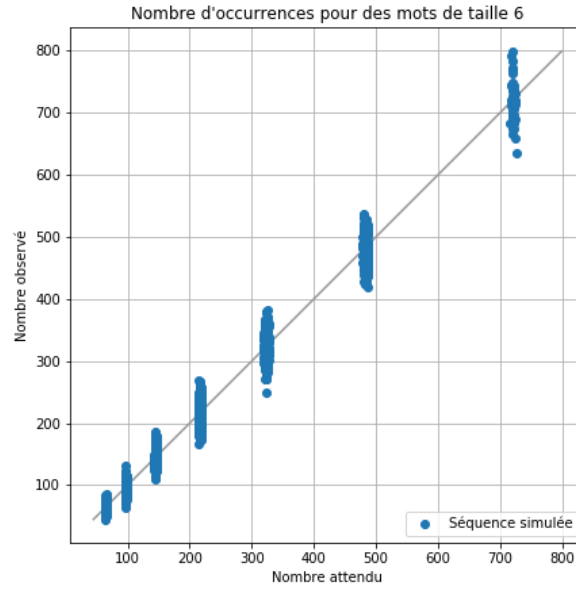
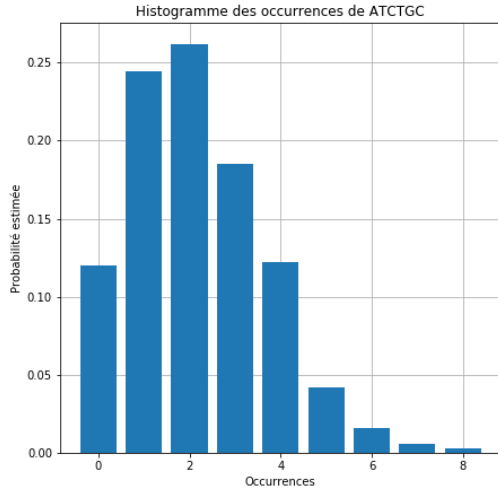


FIGURE 2 – Nombre d’occurrences attendu et observé pour une séquence de taille 1000000 générée aléatoirement.

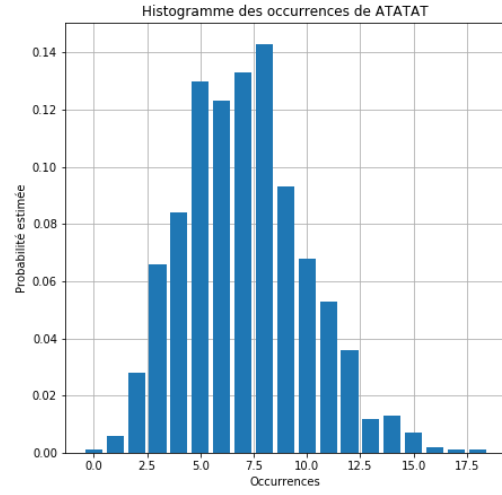
On observe dans la figure que les probabilités d’occurrence dépendent du mot en question. Cela dépend des lettres qui composent ce mot, car A et T sont plus fréquentes que C et G, ce qui explique la différence de la Figure 3(a) avec les trois autres, mais les probabilités dépendent aussi du mot en soi. En effet, certains mots peuvent se chevaucher plus facilement, en on espère ainsi avoir des nombres d’occurrence plus grands de ces mots. Par exemple, en supposant les lettres A et T comme équiprobables, les séquences à 12 lettres ATATATATATAT, TTTAAATTTAAA et AAAAAAAAAAAA ont toutes la même probabilité d’apparaître, mais la première donne lieu à 4 occurrences du mot ATATAT, la deuxième donne lieu à 2 occurrences du mot TTTAAA et la troisième donne lieu à 7 occurrences du mot AAAAAA, cette dernière étant la séquence qui s’auto-chevauche le plus facilement. En particulier, dans une séquence de longueur  $\ell$  avec  $\ell$  un multiple de 6, le nombre maximal d’occurrences du mot AAAAAA est  $\ell - 6 + 1$ , le nombre maximal d’occurrences de ATATAT est  $\frac{\ell}{2} - 2$  et le nombre maximal d’occurrences de TTTAAA est  $\frac{\ell}{6}$ , ce qui indique déjà que les lois de probabilité du nombre d’occurrences de ces mots sont différents car leurs supports sont différents.

### Question 5

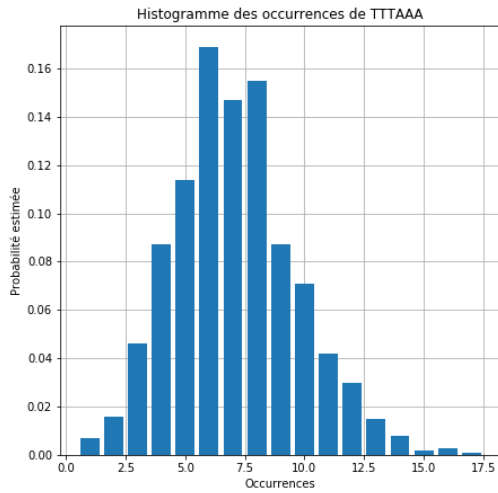
Fixons un mot  $w$  et deux entiers  $n$  et  $\ell$ . Soit  $N$  la variable aléatoire représentant le nombre d’occurrences de  $w$  dans des séquences aléatoires de longueur  $\ell$ . Pour estimer  $p = P(N = n)$ , on peut considérer que  $p$  est le paramètre d’une variable aléatoire de loi Bernoulli qui vaut 1 si  $N = n$  et 0 sinon. On peut alors utiliser



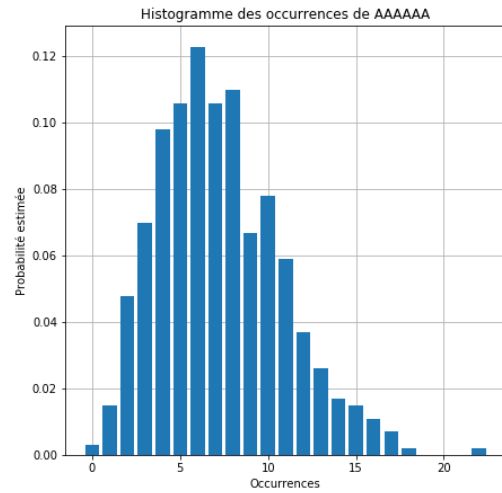
(a)



(b)



(c)



(d)

FIGURE 3 – Histogrammes avec les probabilités estimées du nombre d’occurrences des mots (a) ATCTGC, (b) ATATAT, (c) TTATAA et (d) AAAAAA.

l’intervalle de confiance pour une loi de Bernoulli, donné par

$$I = \left[ \hat{p} - t_{\alpha} \sqrt{\frac{\hat{p}(1 - \hat{p})}{N}}, \hat{p} + t_{\alpha} \sqrt{\frac{\hat{p}(1 - \hat{p})}{N}} \right]$$

où  $\hat{p}$  est la valeur estimée de  $p$  et  $t_{\alpha}$  est calculé à partir du niveau de confiance  $\alpha$ , avec  $t_{\alpha} \approx 1,96$  si  $\alpha = 0,05$ . La Figure 4 représente l’histogramme du nombre d’occurrences du mot ATATAT avec des barres représentant les intervalles de confiance de chaque probabilité calculés à l’aide de la formule ci-dessus et  $\alpha = 0,05$ . Les histogrammes avec les intervalles de confiance pour les autres mots sont donnés dans le fichier `Projet3.ipynb`.

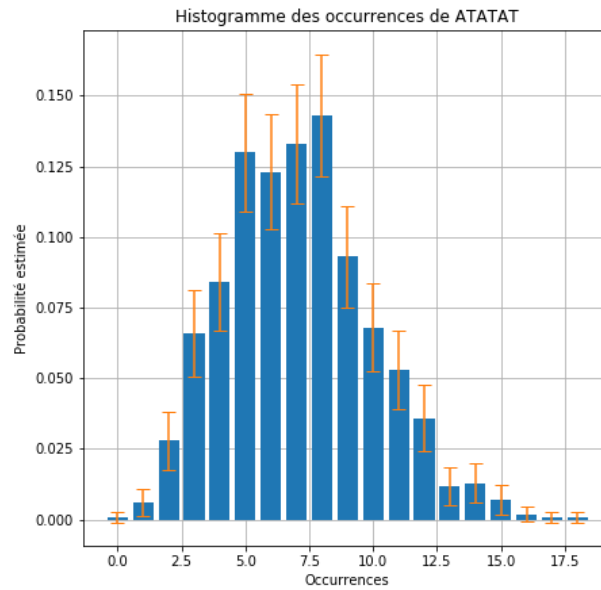


FIGURE 4 – Histogrammes avec les probabilités estimées et les intervalles de confiance pour le nombre d’occurrences du mot ATATAT.

### 3.3 Modèles de dinucléotides et trinucléotides

#### Question 1

Il s’agit d’une chaîne de Markov d’ordre 1 car seul le nucléotide précédent est utilisé pour déterminer la loi du prochain nucléotide de la séquence. Comme on s’attend à ce que tous les coefficients de la matrice  $M$  soient strictement positifs (car il n’y a pas de paire de nucléotides qui ne peuvent pas se suivre dans une séquence), la chaîne de Markov correspondant est ergodique. Alors la probabilité stationnaire peut être estimée par la fréquence des lettres dans une séquence obtenue par l’utilisation de la chaîne de Markov. En supposant que la séquence du fichier `yeast_s_cerevisae_genomic_chr1-4.fna` provient d’une chaîne de Markov, la probabilité stationnaire peut être estimée alors par la table de nucléotides obtenue à partir de l’application de la fonction `nucleotide_frequency` aux données de ce fichier.

#### Questions 2 et 3

Les fonctions `estim_M` et `simule_Markov` pour estimer la matrice  $M$  et pour simuler une séquence produite par la chaîne de Markov correspondante ont été codées dans `projet.py` et testées dans `Projet3.ipynb`. La matrice  $M$  estimée à

partir des données du fichier `yeast_s_cerevisae_genomic_chr1-4.fna` est

$$M = \begin{pmatrix} 0.32505 & 0.18552 & 0.20059 & 0.28884 \\ 0.33571 & 0.20925 & 0.15756 & 0.29747 \\ 0.31321 & 0.20245 & 0.20774 & 0.27659 \\ 0.24149 & 0.21034 & 0.22275 & 0.32541 \end{pmatrix}.$$

On observe en particulier que le modèle précédent où toutes les lettres de la séquence étaient considérées comme indépendantes n'est pas très adapté, puisque les lignes de la matrice  $M$  ne sont pas toutes identiques. On vérifie aussi, conformément à la question précédente, que le vecteur

$$m = (0.29984 \quad 0.20110 \quad 0.20000 \quad 0.29907)$$

obtenu dans la Question 2 de la Partie 2 est bien une probabilité invariante pour  $M$  car  $mM = m$ .

#### Question 4

Soit le mot  $w = w_0 w_1 \dots w_{k-1}$  de longueur  $k$  et une séquence  $S_0 S_1 S_2 \dots$  obtenue à partir d'une chaîne de Markov de matrice de transition  $M$  et probabilité invariante  $m$ . La probabilité pour que  $w$  apparaisse à l'indice  $i$  de cette séquence est:

$$\begin{aligned} & P(S_i = w_0 \cap S_{i+1} = w_1 \cap \dots \cap S_{i+k-1} = w_{k-1}) \\ &= P(S_i = w_0) P(S_{i+1} = w_1 | S_i = w_0) \dots P(S_{i+k-1} = w_{k-1} | S_{i+k-2} = w_{k-2}) \\ &= m(w_0) M(w_0, w_1) \dots M(w_{k-2}, w_{k-1}). \end{aligned}$$

Le calcul de cette probabilité a été fait dans la fonction `logproba_mot_Markov`, qui renvoie le logarithme de cette probabilité car elle peut parfois être très petite.

#### Question 5

Les mêmes arguments qu'à la Question 2 de la Partie 3.1 montrent que, en fixant un mot  $w$  de longueur  $k$  et une taille  $\ell$  de séquence, la variable aléatoire  $X$  donnant le nombre d'occurrences de ce mot dans une séquence de taille  $\ell$  satisfait

$$\mathbb{E}(X) = p(\ell - k + 1),$$

la seule différence étant que désormais on a, d'après la question précédente,

$$p = m(w_0) M(w_0, w_1) \dots M(w_{k-2}, w_{k-1}).$$

On a ainsi codé une fonction `comptage_attendu_Markov` qui est très similaire à `comptage_attendu`, mais remplace l'appel à `logproba` par un appel à `logproba_mot_Markov`.



### Question 6

On a comparé les nombre d'occurrences attendus dans le modèle de nucléotides où toutes les lettres sont indépendantes et dans le modèle de dinucléotides par chaîne de Markov d'ordre 1. Les résultats obtenus pour des mots de longueur 2 sur des séquences de taille 100 sont présentés dans le tableau ci-dessous :

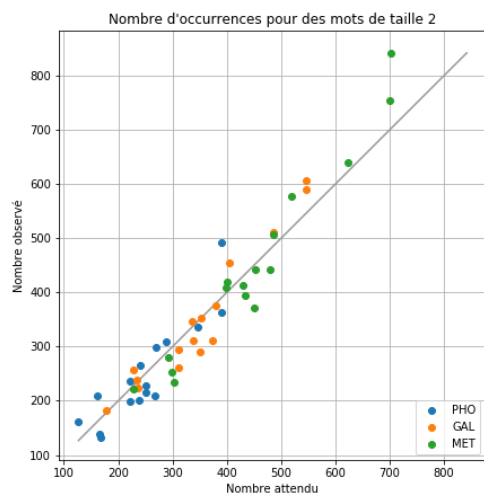
Mot	Nucléotide	Dinucléotide
AA	8,9002	9,6486
AC	5,9695	5,5069
AG	5,9367	5,9543
AT	8,8774	8,5740
CA	5,9695	6,6836
CC	4,0038	4,1661
CG	3,9818	3,1369
CT	5,9541	5,9224
GA	5,9367	6,2015
GC	3,9818	4,0084
GG	3,9599	4,1133
GT	5,9214	5,4765
TA	8,8774	7,1500
TC	5,9541	6,2277
TG	5,9214	6,5952
TT	8,8546	9,6346

On remarque ainsi qu'il y a des différences : certains mots, comme AA, CA, GA ou TT, sont plus fréquents ; d'autres, comme AT, CG, GT ou TA, sont moins fréquents, et d'autres, comme AG, CT ou GC, ont eu leur fréquence quasiment inchangée. Les différences sont en général assez petites, car, même si la matrice  $M$  n'a pas toutes ses lignes identiques, elles sont néanmoins assez similaires au vecteur  $m$  contenant la probabilité invariante, ce qui indique que ce modèle de dinucléotide n'est pas très loin du modèle de nucléotide.

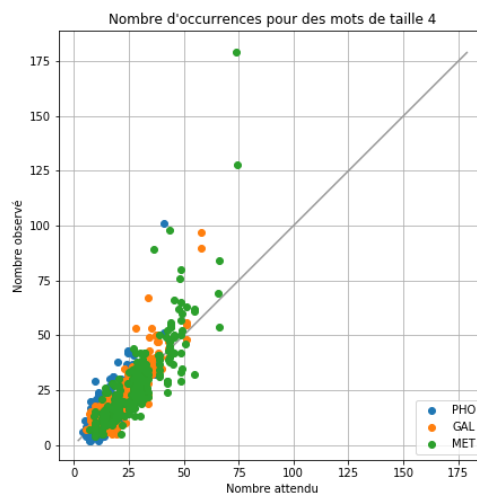
### Question 7

Pour un mot  $w$  de taille  $k$  qui ne se chevauche pas avec lui-même, le nombre maximal d'occurrences de ce mot dans une séquence de longueur  $\ell$  vaut  $\left\lfloor \frac{\ell}{k} \right\rfloor$ . On divise ainsi la séquence de longueur  $\ell$  en  $\left\lfloor \frac{\ell}{k} \right\rfloor$  positions, chacune ayant une probabilité  $p$  de contenir le mot  $w$ . En supposant que l'apparition de ce mot dans ces différentes positions sont indépendantes, on obtient alors que le nombre d'occurrences de ce mot suit une loi binomiale de paramètres  $p$  et  $\left\lfloor \frac{\ell}{k} \right\rfloor$ .

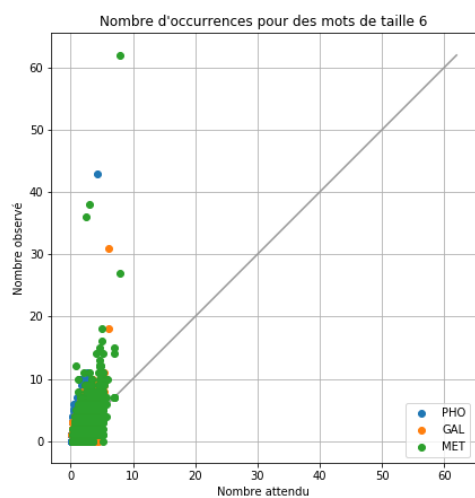
On a aussi recalculé les graphes représentant les nombres d'occurrences observés et attendus par le modèle de chaîne de Markov, qui sont représentés sur la Figure 5. On observe des comportements similaires à ceux de la Figure 1.



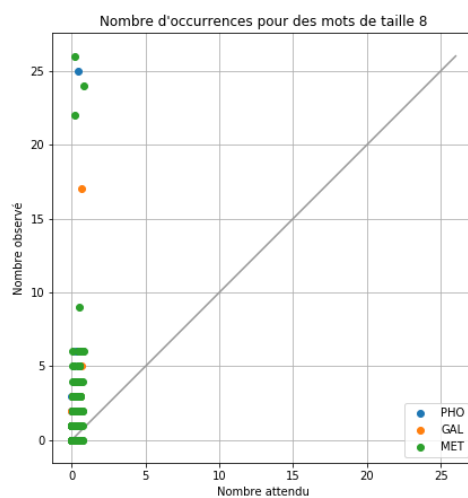
(a)



(b)



(c)



(d)

FIGURE 5 – Nombre d'occurrences attendu dans le modèle de chaîne de Markov et observé pour (a)  $k = 2$ , (b)  $k = 4$ , (c)  $k = 6$  et (d)  $k = 8$ .