Telecom Nancy
Université de Lorraine
Groupe 27

Cyrielle LACRAMPE–DITER, Hind LATRACHE,Gisel RODRIGUEZ-BAIDE,Céline ZHU

PROJET TAPS

Villers-lès-Nancy
Janvier 2021

Introduction

"Nous sommes convaincus que le génie humain et l'intelligence collective au service d'une Science d'excellence, rigoureuse, ouverte et tournée vers le bien de l'humanité, vaincront ce virus et nous amèneront vers des jours meilleurs."

-Olivier Festor
(Voeux de fin d'année scolaire 2020 à Telecom Nancy)

La cartographie du génome du Sars-COV2 et l'étude par la programmation de fonctions classiques du domaine de la Bioinformatique sont une partie du travail effectués dans le cadre du contrôle de l'évolution du virus à l'origine de la pandémie du COVID-19.

La réalisation de ce projet dans le cadre du module de TAPS (*Techniques and Algorithms for Problem Solving*) n'a pas pour ambition d'apporter notre contribution au domaine de la génomique, ensemble des activités qui ont pour but l'étude des génomes, mais de construire des outils permettant l'analyse descriptive d'un virus.

Ainsi les objectifs de ce projet sont celles définies dans le sujet, c'est-à-dire (1) la gestion de projet, (2) l'état de l'art et la compréhension d'algorithme, (3) le développement de fonctions avancées en Python, (4) la conception et réalisation de tests, (5) la documentation et (6) la présentation orale des résultats et des démonstrations.

Table des matières

Chapitre 1

Etat de l'art de la génomique

Contents

1.1 Généralités

Discipline de la biologie moderne, la génomique se scinde en deux branches principales :

— *génomique structurale* ou encore analyses des structures de génomes ;
— *génomique fonctionnelle* ou encore recherche de la fonction des séquences informatives identifiées.

Dans le cadre de notre projet, nous nous situons donc dans la première catégorie puisque nous allons procéder à l'analyse descriptive de la séquence d'ARNm de la COVID-19.

Dans cette perspective nous allons clarifier quelques termes nécessaires à la bonne compréhension de notre étude.

1.2 Analyse descriptive

L'analyse descriptive est un sous-domaine des méthodes quantitatives (méthodes de recherches scientifiques utilisant l'analyse mathématique et scientifique). Dans la perspective d'appliquer cette science à notre base de données, nous allons par la suite nous inspirer de la statistique descriptive [8] et la définitions des briques fondamentales qui construisent l'institut du séquençage génomique.

1.2.1 Statistique descriptive et séquençage génomique

"Descriptive statistics are used to describe the basic features of the data in a study. They provide simple summaries about the sample and the measures. Together with simple graphics analysis, they form the basis of virtually every quantitative analysis of data." est la définiton donnée par William M. k dans son article *Research Methods Knowledge Base* ou encore en français, la statistique descriptive est un outil utilisé pour décrire les caractéristiques fondamentales d'un ensemble de données dans une étude. Elle fournit un résumé simple de l'ensemble des données et des mesures. Ensemble avec une analyse graphique, elles forment la fondation de toutes les analyses quantatives des données.

Dans le cadre de l'étude du COVID-19, avec la base de données fournie par le *National Center for Biology Information* [5], nous avons accès aux séquences génomiques depuis le début de la pandémie provenant de de plusieurs pays. De cette manière, du fait de la nature de l'information, nous étudions l'évolution du virus à travers *le temps* et *l'espace*.

Par ailleurs, l'acide désoxyribonucléique (ou ADN) est principalement caractérisé par la quantité de nucléotides formées de la base azotée : adénine (A), cytosine (C), guanine (G) ou thymine (T). Nous constatons donc que les quatres données principales pour traiter l'ensemble de nos informations sont **le temps**, **l'espace**, **la quantité de chaque base azotées au sein d'une séquence** ainsi que **la longueur résultante de chaque séquence**.

On note par ailleurs, que le projet demandait l'utilisation de l'acide ribonucléique messager (ARNm) du COVID-19, ou encore une copie transitoire d'une portion de l'ADN formées de ribonucléotides : l'adénine (A), l'uracile (U),la guanine (G) ou cytosine (C). De la même manière, les codons dont l'existence a été démontrée par l'expérience de Crick, Brenner, Barnett et Watts-Tobin, est une séquence de trois nucléotides sur un acide ribonucléique messager (ARNm) et sont séparés par plusieurs catégories comme l'illustre la figure ci-dessous [7].

LE CODE GENETIQUE

ARN messager Codon : première base azotée	ARN messager Codon : deuxième base azotée U	C	A	G	ARN messager Codon : troisième base azotée
U	Phe	Ser	Tyr	Cys	U
	Phe	Ser	Tyr	Cys	C
	Leu	Ser	STOP	STOP	A
	Leu	Ser	STOP	Trp	G
C	Leu	Pro	His	Arg	U
	Leu	Pro	His	Arg	C
	Leu	Pro	Gln	Arg	A
	Leu	Pro	Gln	Arg	G
A	Ile	Thr	Asn	Ser	U
	Ile	Thr	Asn	Ser	C
	Ile	Thr	Lys	Arg	A
	Met	Thr	Lys	Arg	G
G	Val	Ala	Asp	Gly	U
	Val	Ala	Asp	Gly	C
	Val	Ala	Glu	Gly	A
	Val	Ala	Glu	Gly	G

Figure1 : Tableau des codons ARN

Dans le cadre de notre projet, l'approfondissement de la définition précise de ces termes n'est pas nécessaire. Cependant, des défintions plus précises ainsi qu'une littérature adaptée sera fournie dans l'Annexe pour potentiellement approfondir notre réflexion et satisfaire notre curiosité scientifique [7].

1.2.2 Algorithmes et explication pédagogiques

Afin de choisir les outils statistiques pertinentes à la génomique, nous allons utiliser celles définies par le CNRS de Montpellier dans son programme éducatif *Montpellier Genomix* [3] .

1.2.2.1 Dispersion : Variance

La dispersion représente la variabilité des différentes valeurs que peut prendre une variable. En statistiques, il existe différentes mesures de la dispersion. Les plus courantes sont la variance, l'écart-type ou encore l'intervalle inter-quartile. C'est une mesure peu influencée par la présence de valeurs extrêmes.

Le terme de dispersion est notamment employé dans les méthodes d'analyse différentielle en RNA-Seq pour parler de la variabilité des données.

La variance est la moyenne des carrés des écarts à la moyenne :

$$\sigma^2 = \frac{\sum_{i=1}^{n}(x_i - \mu)^2}{n}$$

Il s'agit alors d'appliquer l'algorithme ci-dessous :

Algorithm 1: Variance

```
Result: Variance
l=list();
for i=0 jusqu'à la longueur de la liste l avec un pas de 1 do
    compteur=0;
    for j=0 jusqu'à la longueur de la liste l avec un pas de 1 do
        if l(i)(j)=n then
            compteur=compteur+1 ;
        end
    end
    l=l+compteur ;
end
compteur=0 ;
for i=0 jusqu'à la longueur de la liste l avec un pas de 1 do
    compteur=compteur+l(i) ;
end
moyenne=compteur/longueur(l) ;
compteur=0 ;
for i=0 jusqu'à la longuer de la liste l avec un pas de 1 do
    compteur=compteur+(l(i)-moyenne)^2
end
Variance=compteur/longueur(l) ;
```

1.2.2.2 Dispersion : Ecart-type

L'Ecart-type est la moyenne quadratique des écarts par rapport à la moyenne ou plus simplement la racine carrée de la Variance :

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \mu)^2}{n}}$$

En admettant la viabilité de l'algorithme précédent, on définie une unique instruction : l'attribution de la racine carrée de la valeur absolu du résultat

obtenu par l'algorithme précédent.

Algorithm 2: Ecart-type

Result: Et
Et=$\sqrt{|Variance|}$;

1.2.2.3 Médiane

La médiane d'un ensemble de valeurs (échantillon, population, distribution de probabilités) est une valeur m qui permet de couper l'ensemble des valeurs triées en deux parties égales : mettant d'un côté une moitié des valeurs, qui sont toutes inférieures ou égales à m et de l'autre côté l'autre moitié des valeurs, qui sont toutes supérieures ou égales à m (s'il y a un nombre pair de valeurs, la médiane sera la moyenne des 2 valeurs "centrales" de la distribution) :

Si l'effectif total de la série est impair : $N = 2p + 1$, la médiane est la $(p + 1)^{\text{ème}}$ valeur.

Si l'effectif est pair : $N = 2p$, on prend en général pour médiane la moyenne de la $p^{\text{ème}}$ et de la $(p + 1)^{\text{ème}}$ valeur.

Il s'agit alors d'appliquer l'algorithme ci-dessous :

Algorithm 3: Mediane

Result: Mediane
Med=list() ;
for *i=0 jusqu'à la longuer de la liste l avec un pas de 1* **do**
  med=med+liste(i);
end
N=entier((longueur(med)+1/2) ;
med=sorted(med) ;
mediane=0 **if** *longueur(Mediane)*$\equiv 0 \mod 2$ **then**
  mediane=med(longueur(med)+1/2)+longueur((med)-1/2)/2 ;
else
  mediane=med(N) ;
end

1.2.2.4 Premier Quartile

Les quartiles sont les 3 valeurs qui divisent les données triées en 4 parts égales, de sorte que chaque partie représente 1/4 de l'échantillon depopulation. Il existe donc trois quartiles : Q1, Q2 (égal à la médiane) et Q3. Par exemple, Q1 est la valeur telle que 25 % des valeurs de l'échantillon lui sont inférieures, 75 % supérieures.

Algorithm 4: Premier Quartile

Result: Premier Quartile
premquart=list() ;
for *i=0 jusqu'à la longueur de la liste avec un pas de 1* **do**
  premquart=prequart+liste(i);
end
N=entier((longueur(premquart))//4) premquart=sorted(premquart) ;
Premier Quartile=0 **if** *longueur(premquart)*$\equiv 0 \mod 4$ **then**
  Premier Quartile=premquart(longueur(premquart)//4-1) ;
else
  Premier Quartile=premquart(N) ;
end

1.2.2.5 Fold Change

Le fold-change [6] est le rapport du niveau moyen d'expression d'un gène dans une condition par rapport à une autre. Il est généralement exprimé en log (logarithme en base 2) afin de rendre symétriques les rapports par rapport à 1.

Il s'agit alors d'appliquer l'algorithme ci-dessous :

Algorithm 5: Fold Change

Result: Fold Change
A= f(list1[chars],*args) ;
B= f(list2[chars],*args) ;
if *B=0* **then**
  Fold Change=0;
else
  Fold Change=A/B
end

1.3 Levenshtein

La distance de Levenshtein entre deux mots [10] est le nombre minimum de modifications d'un seul caractère (insertions, suppressions ou substitutions) nécessaires pour changer un mot dans l'autre. Ou encore :

$$d_{a,b}(i,j) = \min \begin{cases} 0 & \text{if } i = j = 0 \\ d_{a,b}(i-1,j) + 1 & \text{if } i > 0 \\ d_{a,b}(i,j-1) + 1 & \text{if } j > 0 \\ d_{a,b}(i-1,j-1) + 1_{(a_i \neq b_j)} & \text{if } i,j > 0 \\ d_{a,b}(i-2,j-2) + 1 & \text{if } i,j > 1 \text{ and } a[i] = b[j-1] \text{ and } a[i-1] = b[j] \end{cases}$$

Il s'agit alors d'appliquer l'algorithme ci-dessous :

Algorithm 6: Levenshtein

```
Result: Distance
for i=0 à longueur(Chaine1) avec un pas de 1 do
 |  d(i, 0) = i
end
for j=0 à longueur(Chaine2) avec un pas de 1 do
 |  d(0, j) = j
end
for  i=1 à longueur(Chaine1) avec un pas de 1 do
 |  for  j=1 à longueur(Chaine2) do
 |  |  if  chaine1(i) = chaine2(j) then
 |  |  |  coûtSubstitution = 0
 |  |  else
 |  |  |  coûtSubstitution = 1
 |  |  end
 |  |  d[i, j] = minimum( d(i-1, j ) + 1, d(i, j-1) + 1, d(i-1, j-1) +
 |  |   coûtSubstitution)
 |  end
end
Distance=d(longueurChaine1,longueurChaine2)
```

1.4 Needleman-Wunsch

L'algorithme Needleman-Wunsch [9],calcule l'alignement global de deux chaînes de carctères en utilisant la programation dynamique. Pour cela elle divise un grand problème (par exemple la séquence complète) en une série de problèmes plus petits, et il utilise les solutions aux plus petits problèmes pour trouver une solution optimale au problème plus vaste. L'algorithme s'effectue en deux temps, d'une part la détermination de la matrice de similarité puis déterminer les alignements correspondant à ce score. Soit encore :

Algorithm 7: Needleman-Wunsch

```
Result: AlignementA, AlignementB
for i=0 to length(A)-1 do
    F(i, 0) ← d*i
end
for j=0 to length(B)-1 do
    F(0,j) ← d*j
end
for i=1 to length(A)-1 do
    for j = 1 to length(B)-1 do
        Choice1 ← F(i-1,j-1) + S(A(i), B(j));
        Choice2 ← F(i-1, j) + d;
        Choice3 ← F(i, j-1) + d;
        F(i, j) ← max(Choice1, Choice2, Choice3)
    end
end
AlignmentA ← "";
AlignmentB ← "";
i ← length(A) - 1;
j ← length(B) - 1;
while (i > 0 AND j > 0) do
    Score ← F(i, j);
    ScoreDiag ← F(i - 1, j - 1);
    ScoreUp ← F(i, j - 1);
    ScoreLeft ← F(i - 1, j);
    if (Score == ScoreDiag + S(A(i), B(j))) then
        AlignmentA ← A(i) + AlignmentA;
        AlignmentB ← B(j) + AlignmentB;
        i ← i - 1;
        j ← j - 1;
    end
    else if (Score == ScoreLeft + d) then
        AlignmentA ← A(i) + AlignmentA;
        AlignmentB ← "-" + AlignmentB;
        i ← i - 1
    else
        (Score == ScoreUp + d)
    end
    AlignmentA ← "-" + AlignmentA;
    AlignmentB ← B(j) + AlignmentB;
    j ← j - 1
end
while (i > 0) do
    AlignmentA ← A(i) + AlignmentA;
    AlignmentB ← "-" + AlignmentB;
    i ← i - 1;
end
while (j > 0) do
    AlignmentA ← "-" + AlignmentA;
    AlignmentB ← B(j) + AlignmentB;
    j ← j - 1
end
```

Chapitre 2

Développement et conception des fonctions Python

Contents

2.1 Généralités

Nous allons dans cette section expliquer le processus de conception et de développement des fonctions Python (oui oui exactement comme indiqué dans le titre)en respectant le processus présenté dans le module. nous pouvons relever trois spécialité dû à la nature de nos données et de nos outils :

les fichiers fasta propre à la bibliothèque Biopython et une tolérance des types/classes propre à Python.

2.2 (Bio)python

Maintenant que nous avons définfies les données fondamentales définissant notre projet ainsi que les algorithmes nécessaire à une analyse descriptive, il s'agit alors d'appliquer les différents algorithmes aux deux derniers descripteurs de notre données- **le nombre d'acides aminés de chaque type dans la séquence** et **la longueur de la séquence**- comme établi précedemment.

2.2.1 Données

Les données sont présentés sous un format, standard à la bioinformatique, dit *FASTA*, c'est à dire un fichier texte composé d'au moins deux lignes :

- Une ligne décrivant la séquence commençant par le signe ">"
- Une deuxième ligne contenant des lettres représentant les acides nucléiques ou les acides aminés de la séquence

Dans cette perspective, nous allons donc extraire la séquence sous forme de liste.Nous allons donc coder une fonction *fas2ch*

La fonction *fas2ch* permet d'extraire les séquences génétiques du fichier fasta et les renvoie sous forme de liste. Elle prend en paramètres : une chaîne de caractères avec des guillemets contenant le nom du fichier de la forme ("nomdufichier.fasta") , Elle renvoie la liste de listes de caractères individuels : (list[list[char]]). Il s'agit alors d'appliquer l'algorithme ci-dessous :

Algorithm 8: fas2ch

Result: liste l
l=list() ;
for *tout seq_ record dans le fichier "nomdufichier.fasta" de type fasta* **do**
| Ajouter la séquence sous forme de liste l.
end

2.2.2 Traitement

On applique alors les algorithmes définis dans la partie précédente sur les deux descripteurs.

	Quantité de bases azotées	Longueur de la séquence
Dispersion	*var_ n, et_ n*	*var_ leng, et_ leng*
Médiane		*mediane*
Quartile		*premquartil*
Fold Change	*fold_ change*	*fold_ change*

Fonctions	Description
var_ n	La fonction *var_ n* permet de calculer la variance du nombre d'occurrence d'un nucléotide sur un certain nombre de séquences sous forme de liste. Elle prend en paramètres : La liste des séquences sous forme de liste (list[list[char]]) , on la note liste ; le nucléotide (char), on le note n. Elle renvoie la variance (int).
var_ leng	La fonction *var_ leng* permet de calculer la variance de la longueur d'une liste de séquences. Elle prend en paramètre une liste des séquences : list[char] et elle renvoie la variance : float.
et_ n	La fonction *et_ n* permet de calculer l'écart type du nombre d'occurrence d'un nucléotide sur un certain nombre de séquences sous forme de liste. Elle prend en paramètres : La liste des séquences sous forme de liste (list[list[char]]) , on la note liste ; le nucléotide (char), on le note n. Elle renvoie l'écart type (int).
et_ leng	La fonction *et_ leng* permet de calculer la racine carrée de la variance de la longueur d'une liste de séquences. Elle prend en paramètre une liste de séquences : list[chars] et elle renvoie l'écart-type : float.
premquartil	La fonction *premquartil* permet de calculer le premier de la longueur des séquences représentées par des listes. Elle prend en paramètres : La liste des séquences sous forme de liste (list[list[char]]) , on la note liste Elle renvoie la valeur entière : (int).
mediane	La fonction *mediane* permet de calculer la médiane de la longueur des séquences représentées par des listes. Elle prend en paramètres : La liste des séquences sous forme de liste (list[list[char]]) , on la note liste Elle renvoie la valeur entière : (int).
fold_ change	La fonction *fold_ change* permet de calculer le rapport du niveau moyen d'expression d'une gène dans une condition par rapport à une autre .elle prend en paramètre les deux listes de séquences qu'on souhaite comparer : list[chars] , et une fonction qui sera la norme qui nous permettra de comparer les deux liste.

2.3 Un type de problème : problème de type

Plusieurs des questions clés font appel à l'ARNm et à un typage *list* cela permet de souligner plusieurs problèmes que nous allons présenter, puis les solutions proposées et utilisées.

2.3.1 Uracile ou Thymine telle est la question

Une carctéristique de l'ARNm est l'utilisation de la base nucléique Uracile, plutôt que celle de la Thymine qui elle est propre à l'ADN ; Cependant, les fichiers FASTA contiennent la Thymine plutôt que l'Uracile.

Qu'en est-il ?

En réalité, le problème est moins compliqué qu'il n'y paraît puisque, ce remplacement peut être retracé à une conventionde 1985 par le *Nomenclature Committee of the International Union of Biochemistry* qui justifie la convention par sa compatibilité avec les expressions ADN. Nous pouvons donc retenir que dans le cadre de notre étude Thymine(T)==Uracile(U).

Ainsi la fonction *codons* qui suit le principe du tableau présenté dans le chapitre 1, sera :

2.3.2 Tolérance chez Python : une dangereuse pente

Un autre problème recontrée lors de l'étude de notre projet est l'adaptation de nos fonctions d'analyse descriptive aux séquences modifiés par la séquence sont principalement un problème de typage. or python fait l'amalgame pratique des classes chaînes de caractères et listes : elles répondent alors aux mêmes commandses, ainsi les fonctions de l'analyse descriptive peut être adaptée facilement en implémentant l'appel de la fonction codons puis de la fonction de l'analyse descriptive correspondante.

2.4 Représentation des résultats le continuum espace-temps

Le titre ici est bien plus impressionant que son contenu, mais une question fondamentale se pose bien : comment représenter les résultats en prenant en compte des quatres données principales :**le temps**,**l'espace**,**la quantité de**

Algorithm 9: *codons*

```
Result: L_acides
L_codons=list()
L_acides=list()
tableau=[['F','TTT','TTC'],['L','TTA','TTG','CTT','CTC','CTA','CTG'],['I','ATT','ATC','ATA']
,['M','ATG'],['V','GTT','GTC','GTA','GTG'],['S','TCT','TCC','TCA','TCG','AGT','AGC']
,['P','CCT','CCC','CCA','CCG'],['T','ACT','ACC','ACA','ACG'],['A','GCT','GCC','GCA','GCG']
,['Y','TAT','TAC'],['Stop ocre','TAA'],['Stop ambre','TAG']
,['H','CAT','CAC'],['Q','CAA','CAG'],['N','AAT','AAC'],['K','AAA','AAG']
,['D','GAT','GAC'],['E','GAA','GAG'],['C','TGT','TGC'],['Stop
 opale','TGA']
,['W','TGG'],['R','CGT','CGC','CGA','CGG','AGA','AGG'],['G','GGT','GGC','GGA','GGG']]
if longueur(seq)%3= !0 : then
    return []
else
    Sinon : for  i = 0 jusqu'à la longueur de seq avec un pas de 3 do
        On ajoute à L_codons seq[i]+seq[i+1],seq[i+2]
    end
    for chaque codon dans la liste Lcodons do
        for chaque case 'acide' dans le tableau : do
            for c=1 jusqu'à longueur de la case 'acide' do
                if codon == acide[c] then
                    on ajoute acide[0] à la liste L_ acide
                end
            end
        end
    end
end
```

chaque base azotées au sein d'une séquence et **la longueur résultante de chaque séquence.**

Si les deux derniers ont déjà été pris en compte et le premier est une représentation classique des données en fonction du temps ;comment alors représenter les données à la fois dans l'espace et le temps ? On peut imaginer plusieurs solutions que nous allons présenter par la suite.

2.4.1 Temps

Une remarque intéressante sur la façon dont est organisée le site **NCBI** est que les séquences sont classés par ordre chronologique. La découpe des données collectées en leurs mois respectifs s'est donc fait pour deux raisons principales : un soucis de temps d'exécution des fonctions et une volonté de représenter l'aspect temporel du virus du COVID-19.

Ainsi, les séquences regroupées, nous allons établir une fonction qui nous permet d'avoir accès à n'importe lequel des ces mois. Nous allons appeler cette fonction, la fonction *date*.

La fonction *date* permet de récupérer les séquences collectées durant un certain mois. Elle prend en paramètres : l'année (int) ; le mois (int). Elle renvoie une liste de séquences, les séquences étant sous forme de listes de caractères (list[list[char]]).

Algorithm 10: *date*

```
Result: liste l
if annee != 2020 OU mois < 1 OU mois > 11 then
  | l=[]
end
if mois < 10 then
  | mois_str = "0" + str(mois)
else
  | mois_str = str(mois)
end
l= fas2ch("Sequences_" + str(annee) + "_" + mois_str + ".fasta")
```

2.4.2 Espace

La section 2.2 nous a introduit au concept de fichier **FASTA**, dont la première ligne contient un nombre d'informations par rapport à la séquence, dont notamment le pays d'origine sous forme d'abréviation **OTAN**. Nous

allons donc dans cette prespective définir une fonction *geog* en deux temps.

La fonction geog permet de récupérer les séquences collectées dans un certain pays pendant une certaine période de temps. Elle prend donc en paramètre : le pays sous forme de l'abréviation en trois lettres définie par l'**OTAN** (str) ; le mois à partir duquel on effectue notre recherche (int) ; le mois après lequel on s'arrête de chercher (int). Elle renvoie une liste de séquences, les séquences étant sous forme de listes de caractères (list[list[char]]).

Algorithm 11: is_from

```
Result: b
n = len(mot) ;
for i allant de 0 à len(phrase) - n do
    if phrase[i :i+n] == mot then
        Retourner b=True
    else
        b=False
    end
end
```

Algorithm 12: geog

```
Result: l
L = []
for mois allant de max(1, mois1) à min(12, mois2 + 1) do
    if mois < 10 then
        mois_str = "0" + str(mois)
    else
        mois_str = str(mois)
    end
end
for seq_record appartenant à SeqIO.parse("Sequences_2020_" +
  mois_str + ".fasta","fasta") do
    if is_from("/" + pays + "/", seq_record.description) then
        L1 ← [ ] ;
        for k appartenant à seq_record.seq do
            Ajouter k à L1 Ajouter L1 à L
        end
    end
end
```

Certes : mais comment exprimer cette spatialité ?

Nous pouvons nous inspirer du journal LE MONDE et proposer deux solutions différentes et complémentaires :

— une représentation temporelle avec des courbes différentes pour chaque pays
— une animation de l'évolution temporelle sur un planisphère(représentation géospatial)[4]

Seule une des deux solutions nous est accessible à notre niveau de formation, ce qui nous empêche pas d'apprendre, Internet étant une merveilleuse invention.

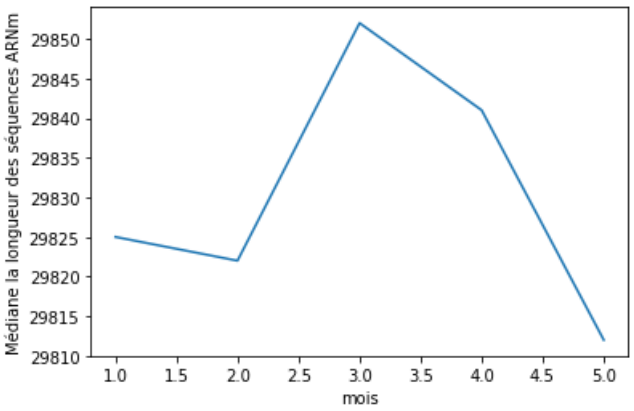
2.5 Interprétation des résultats

Nous avons donc obtenu bon nombre d'information au sujet des séquences génomiques du COVID-19 : mais que signifient-elles ? Nous allons donc en former une opinion, non de biologiste, mais de mathématicien.L'ensemble des interprétations de cette partie sera faite avec l'hypothèse évidente mais non négligeable que l'ensemble de nos algorithmes sont corrects.

2.5.1 Analyse descriptive

Médiane

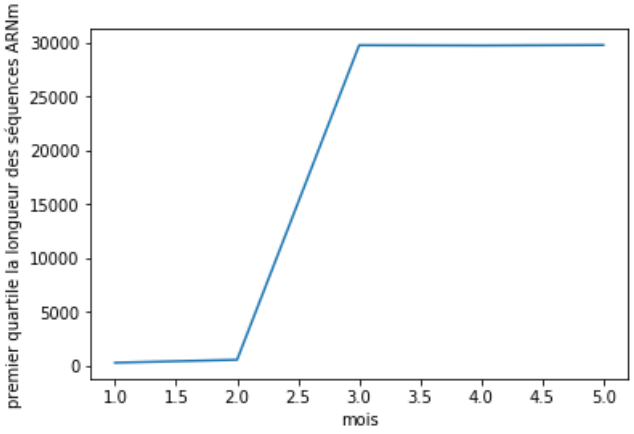
Comme nous l'avons établi antérieurement, la médiane représente la tendance centrale de l'ensemble des données étudiées : ici la longueur des séquences ARNm au cours du temps. Ici on s'aperçoit que cette médiane qui semble à première vue très instable, représente en réalité une faible évolution au vu de la longueur usuelle des séquences.



Médiane de la longueur des séquences ARNm en fonction du temps

Premier Quartile

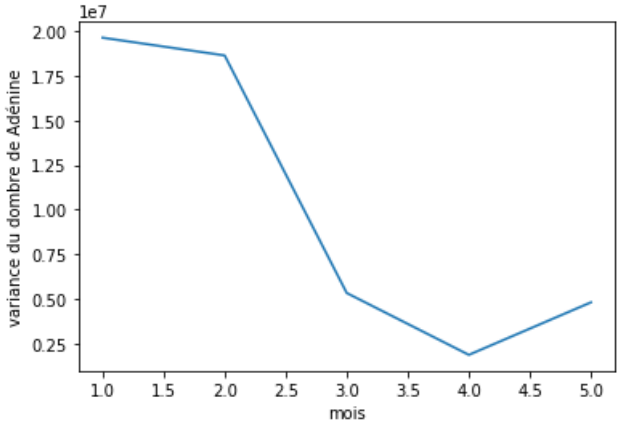
Au contraire, le premier quartile lui semble suivre une évolution intéressante, avec un drôle de phénomène de palier. Mais l'augmentation de ce premier quartile signifie un regroupement de la dispersion des données : en d'autre mots : le virus se "standardise".



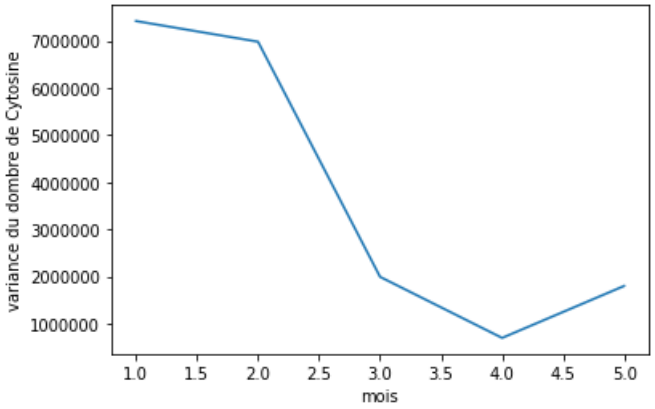
Premier quartile de la longueur des séquences ARNm en fonction du temps

Variance de la quantité d'Adénine et de Cytosine

Il est visuellement évident que ces deux courbes suivent une même tendance, ce qui est logique, puisque comme on l'a montré précdemment, la dispersion se rapproche, ce qui exactement ce que représente la variance.

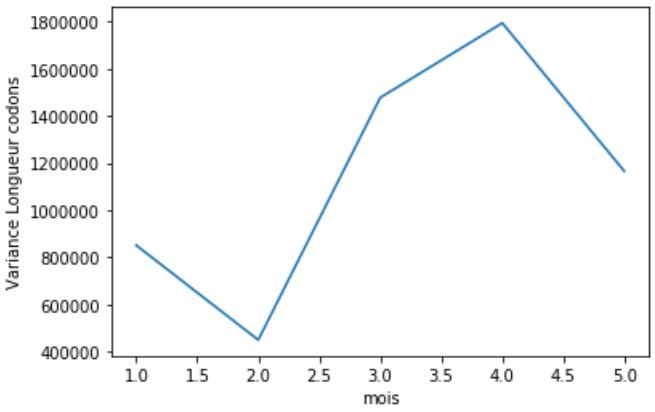


Variance du nombre de Adénine des séquences ARNm en fonction du temps



Variance du nombre de Cytosine des séquences ARNm en fonction du temps

Variance acides aminés Pourtant, quand on arrive aux acides aminés, on a une image très différente du Virus, et c'est là qu'interviendrai l'expertiste d'un biologiste : comment une standardisation de l'ARNm peut-elle menée à une évolution disproportionnée des acides aminés ? Peut-être une modification minuscule d'une partie cruciale de l'ARNm peut tout changé. En tout cas une chose est sûr, grâce la nouvelle connaissance acquise en codant la fonction codons, c'est que le moindre décalage.



Variance de la longueur des séquences d'acides aminés en fonction du temps

2.5.2 Levenshtein

Un aveux, un peu honteux, doit se faire en cette partie et la partie suivante : les expériences menées ne peuvent pas être concluante : dû au faible échantillon traité. En effet, on en reparlera dans le chapitre sur les performances, mais les deux fonctions Distance de Levenshtein et Needleman-Wunsch tournent lentement sur nos ordinateurs.

Cependant si on compare quelques séquences savamment choisies (Amérique /Chine ou Janvier/Novembre) on s'aperçoit que la Distance semblerait être croissante.

2.5.3 Needleman-Wunsch

De la même façon, on peut argumenter de façon plus intuitif que scientifique que le score maximal est de plus en plus grand.

Chapitre 3

Tests et Performances

Contents

3.1 Tests

Dans le but de simplifier et de faciliter le travail, on a décidé de nommer les fichiers comme les fonctions.

3.1.1 Le principe Right-BICEP

Afin de vérifier que les fonctions s'exécutent conformément au cahier des charges, l'écriture de tests est indispensable. Il a été choisi de suivre les principes « Right-BICEP » [2] afin d'écrire des tests les plus exhaustifs possibles.

Il s'agit tout d'abord de vérifier si les valeurs retournées par une fonction sont celles qui sont attendues (« Right »). Il est ensuite nécessaire de vérifier son comportement aux conditions limites (« Boundary »), lorsque les paramètres d'entrée atteignent des valeurs particulières, car beaucoup de problèmes sont susceptibles de s'y trouver. De plus, si un fonction faisant le calcul inverse existe, nous pouvons vérifier que l'on retrouve bien les paramètres donnés au départ (« Inverse »), et s'il existe un autre moyen d'obtenir les mêmes valeurs que celles retournées par la fonction testée, nous pouvons croiser les résultats pour les vérifier (« Cross-check »).

Il peut ensuite être possible de vérifier les cas où la fonction renvoie une erreur (« Error »), comme des paramètres invalides ou des contraintes de l'environnement. Il est enfin nécessaire de s'assurer que le temps d'exécution de la fonction soit convenable (« Performance »).

En plus de suivre le principe Right-BICEP, des tests à l'aveugle ont été réalisés en prarallèles des tests écrits par le programmeur de chaque fonction, Ainsi, ces tests écrits par une personne n'ayant pas étudié l'algorithme de la fonction permettent de tester des cas de figure auxquels le programmeur n'aurait pas forcément pensé.

3.1.2 Application des tests de performance

Concernant les tests de performance, nous avons choisi de vérifier que nos fonctions s'exécutent en des temps conformes aux complexités que l'on a calculées. Pour cela, nous avons utilisé la fonction *timeit* de la librairie *timeit*, qui prend en paramètre le code à exécuter sous forme de chaîne de caractères ainsi que le nombre de fois qu'il doit être exécuté, et retourne le temps d'exécution, que l'on peut ensuite comparer avec nos valeurs théoriques.

Ces valeurs théoriques de temps d'exécution sont calculées à partir des complexités des fonctions et nous avons choisi de considérer qu'une machine peut effectuer 10^9 opérations par secondes. Nous effectuons donc les mêmes calculs que ceux qui ont été nécessaires pour remplir la table 3.1 [1].

TABLE 3.1 – Temps nécessaire à 10^9 op/s

Complexité	**n=10**	**n=100**	**n=1000**
n	10ns	100ns	$1\mu s$
n^2	100ns	$10\mu s$	1ms
n^3	$1\mu s$	1ms	1s
2^n	$>1\mu s$	$3 \cdot 10^{13}$ans	10^{284}ans
$\sqrt{n}$	3.3ns	10ns	31ns
$\sqrt[3]{n}$	2.15ns	4.64ns	10ns
$\log n$	3.3ns	6.6ns	9.9ns

3.2 Complexité

Le calcul des complexités de chacune des fonctions est nécessaire afin d'étudier leurs performances. Ces calculs sont regroupés dans la table suivante :

Fonction	Complexité
date	La fonction *date* effectue des opérations en temps constant pour construire le nom du fichier puis fait appel à la fonction *fas2ch*. Elle a donc la même complexité que la fonction *fas2ch*, c'est-à-dire en $O((nombre\ de\ sequences) \cdot (longueur\ max\ d'une\ sequence))$.
var_leng	Dans la fonction *var_leng*, on a deux boucles *for* séparées dont la complexité de chacune est $O(nombre\ de\ sequences)$ donc la complexité de la fonction *var_leng* est en $O(nombre\ de\ sequences)$.
et_leng	La fonction *et_leng* appelle la fonction *var_leng* et effectue une opération en temps constant donc elle a la même complexité que la fonction *var_leng*, soit $O(nombre\ de\ sequences)$.
fas2ch	La fonction *fas2ch* possède deux boucles *for* dont une sous-entendue. Cela donne une complexité de $O((nombre\ de\ sequences)^2)$.
mediane	La fonction *mediane* possède une boucle *for* et une fonction de tri. En notant n le nombre de séquences contenues dans la liste passée en paramètres, cela donne une complexité en $O(n \cdot log(n))$.
premquartil	La fonction *prequartil* possède une boucle *for* et une fonction de tri. En notant n le nombre de séquences contenues dans la liste passée en paramètres, cela donne une complexité en $O(n \cdot log(n))$.
geog	Notons : — *nseq* le nombre de séquences collectées dans la période de temps qu'on l'on étudie ; — *npays* le nombre de séquences que l'on va retourner ; — *ldes* la longueur maximale de la description d'une séquence ; — *lseq* la longueur maximale d'une séquence. La fonction *geog* fait *nseq* boucles au cours desquelles elle appelle à chaque fois la fonction auxiliaire *is_from* de complexité en $O(ldes)$. Parmi ces *nseq* boucles, il y a *npays* boucles où elle effectue *lseq* boucles pour récupérer les séquences en listes de caractères. La complexité de la fonction *geog* est donc en $O(nseq \cdot ldes + npays \cdot lseq) = O(nseq \cdot lseq)$.
var_n	La fonction *var_n* possède quatre boucles *for* dont deux sont imbriquées. Cela donne une complexité en $O((nombre\ de\ sequences)^2)$.

et_ n	La fonction *et_ n* appelle la fonction *var_ n* et effectue une opération en temps constant donc elle a la même complexité que la fonction *var_ n*, soit $O(nombre\ de\ sequences)$.
fold_ change	La fonction *fold_ change* exécute deux fois la fonction passée en paramètres et effectue des opérations à temps constant donc sa complexité est la même que celle de la fonction passée en paramètre.
codons	La fonction *codons* contient une boucle *for* qui parcourt la liste passée en paramètres donc sa complexité est en $O(nombre\ de\ sequences)$.
var_ aa	La fonction *var_ aa* construit la liste *Nb_ aa* en en faisant *longueur de L* fois *longueur d'une séquence* opérations, puis calcule *esp2* en faisant *longueur de L* fois opérations, donc sa complexité est en $O((longueur\ de\ L) \cdot (longueur\ d'une\ squence))$.
et_ aa	La fonction *et_ aa* appelle la fonction *var_ aa* et effectue une opération en temps constant donc elle a la même complexité que la fonction *var_ aa*, soit $O((longueur\ de\ L) \cdot (longueur\ d'une\ squence))$.
mediane_ aa	Dans la fonction *mediane_ aa*, on a deux boucles *for* incrémentées dont la complexité de la boucle for externe est O(N) alors la complexité de la boucle for interne est O(len(list[i])) avec N la taille de la liste et N non nulle sinon la complexité de mediane-aa est O(1) (meilleur cas) et len(list[i]) est la taille de la i-iéme séquence. Soit m=max(len(list[i]) , avec i un entier entre 0 et N), alors la complexité de la fonction $\text{mediane}_a aest O(N * m)$
var_ leng_ aa	La complexité est la somme des deux complexités des deux fonctions utilisées ainsi la complexité est de O(2*n) avec n la longueur de la liste de caractères.
et_ leng_ aa	La complexité est la somme des deux complexités de la fonction utilisées ainsi la complexité est de O(2*n) avec n la longueur de la liste de caractères.
premquartil_ aa	Dans la fonction premquartil-aa, on a deux boucles 'for' incrémentées dont la complexité de la boucle for externe est O(N) alors la complexité de la boucle for interne est O(len(list[i])) avec N la taille de la liste et N non nulle sinon la complexité de premquartil-aa est O(1) (meilleur cas) et len(list[i]) est la taille de la i-iéme séquence. Soit m=max(len(list[i]) , avec i un entier entre 0 et N), alors la complexité de la fonction premquartil-aa est O(N*m).

Chapitre 4

Gestion de Projet

Contents

4.1 Généralités

Ce projet se développe dans le cadre du module de TAPS, il vise la mobilisation des connaissances acquises en matière de Techniques and Algorithms for Problem Solving de même qu'en Gestion de Projet.

Les algorithmes et leur étude développés lors de ce projet doivent répondre à l'énoncé fourni par le Professeur de TAPS, Olivier Festor. Tous les élèves participant au projet doivent travailler dans le déroulement de chacune de ses parties, autant comme développeurs que managers. Le projet se déroule par jalons définis à partir des sous parties de l'énoncé.

Le projet vise à livrer un compte rendu qui sera noté.

4.2 Outils de gestion de projet

Le bon outil pour le bon usage.

Pour bien gérer notre projet et pour respecter les consignes de l'énoncé, on s'appuie sur plusieurs outils de gestion de projet :

- Charte de projet rédigée par Gisel Rodríguez
- Cahier de charge rédigé par Céline Zhu
- Matrice SWOT rédigée par Cyrielle Lacrampe–Diter
- Diagramme WBS rédigé par Hind Latrache
- Diagramme de Gantt rédigé par Céline Zhu
- Matrice RACI rédigée par Hind Latrache
- Système de points proposé par Cyrielle Lacrampe–Diter
- Comptes-rendus rédigés lors de chaque réunion par l'une des intégrantes de l'équipe

4.2.1 Charte de projet

- **Cadrage/Finalités et importance du projet**

  Ce projet cherche à répondre à la totalité des questions de son énoncé. On s'intéresse à l'étude des algorithmes courants de bio-informatique et de les appliquer sur des cartes génétiques du SARS-COV 2. Le développement et l'étude des fonctions algorithmiques ainsi que la gestion du projet seront conçus par toutes les participantes et le compte rendu final comportera un document rédigé sur Latex ainsi qu'une présentation orale accompagnée d'une démonstration. À l'issue du compte rendu, une note sera donnée aux élèves.

- **Objectifs et résultats opérationnels**

Les livrables de ce projet sont les compte rendus finales : un document sur Latex et une présentation orale suivie d'une démonstration.

— **Déroulement du projet/Organisation/ressources**

— Dans le déroulement de ce projet, les participantes sont :

Céline ZHU
Cyrielle Lacrampe—Diter
Hind Latrache
Gisel Rodríguez

Chacune d'entre elles aura le même rôle en tant que développeur et manager.

— Les moyens à disposition sont multiples :
- Logiciels de traitement de texte tel Latex
- Logiciels permettant travaille en équipe tel Gitlab
- Connaissances acquises lors du module de TAPS pour le développement et l'étude d'algorithmes
- Connaissances acquises lors du module de MOOC pour la gestion de projet
- Éventuelle aide des professeurs de TÉLÉCOM Nancy

— **Jalons : échéancier / événements importants**

Jalon	Description
Étape 1 : exigences opérationnelles	Validation des spécifications : cahier des charges techniques.
Étape 2 : préparation pour démarrer le projet	Réalisation de la matrice SWOT, diagrammes RACI, WBS et Gantt.
Étape 3 : Développement du livrable	Récupération des données partie 5.1
Étape 4 : Développement du livrable	Création et étude des fonctions spécifiées dans la partie 5.2 de même que leur rédaction dans le compte rendu.
Étape 5 : Développement du livrable	Création et étude des fonctions spécifiées dans la partie 5.3 de même que leur rédaction dans le compte rendu.
Étape 6 : Développement du livrable	Création et étude des fonctions spécifiées dans la partie 5.4 de même que leur rédaction dans le compte rendu.
Étape 7 : Développement du livrable	Création et étude des fonctions spécifiées dans la partie 5.5 de même que leur rédaction dans le compte rendu.
Étape 8 : compte rendu	Valider compte rendu final et présentation orale.

— **Risques et opportunités**
Après analyse du déroulement du projet, on dégage :

Plusieurs risques tels :
- Un mauvais accès à internet
- Des difficultés de communication dues au confinement
- Un prolongement possible du confinement

Plusieures opportunités telles :
- Le déconfinement
- La possibilité de solliciter une personne extérieure au projet pour s'entraîner à la présentation orale
- La possibilité de demander de l'aide à un professeur en cas de blocage

4.2.2 Cahier de charge

On a réalisé pour bien illustrer ce qui demandé de nous dans l'énoncé et pour être efficace dans notre travail.

Le cahier de charge est illustré par la figure 4.1 et 4.2

4.2.3 Matrice SWOT

	Forces	Faiblesses
Origine interne	-Les membres de l'équipe savent programmer en python -Les membres de l'équipe sont motivés et travailleurs -Les membres de l'équipe sont familiers aux termes du projet grâce aux cours de science et vie de la terre enseignés au lycée	-Les membres de l'équipe ne savent pas encore utiliser LateX -Les membres de l'équipe ne s'y connaissent pas énormément en génétique -L'équipe du projet est affectée négativement par le confinement
Origine externe	Opportunités -Le déconfinement -La possibilité de solliciter une personne extérieure au projet -La possibilité de demander de l'aide à un professeur en cas de blocage	Menaces -Un mauvais accès à internet -Des difficultés de communication dues au confinement -Un prolongement possible du confinement

4.2.4 Diagramme WBS

À partir de la décomposition hiérarchique de chaque section du projet, le diagramme WBS a été créé pour déterminer quelles questions pouvaient être traitées en parallèle ou en priorité.

Le diagramme WBS pour les sections 5.1 jusqu'à 5.5 sont illustrés par les figures 4.3 jusqu'à 4.7 présentes dans l'annexe.

4.2.5 Diagramme de Gantt

Le découpage du projet en lots de travail et la définition des différents jalons ont permis le développement d'un diagramme de Gantt. Cet outil a aidé au suivi de l'avancement du projet dans le temps qui été délimité par un délai.

Le diagramme de Gantt est illustré par la figure 4.8 .

4.2.6 Matrice RACI

On a réalisé une matrice RACI dans le but de formaliser le rôle de chacune lors des différents lots de travail et avoir une idée claire de nos responsabilités.

La matrice RACI et illustrée par la figure 4.9.

4.2.7 Système de points

Le système de points mis en place pour contribuer à la motivation de l'équipe consiste à : partir toutes les membres de l'équipe à 1 point et lors de chaque réunion, si l'une des membres n'a pas fini le travail qui lui a été désigné, elle récoltera une quantité de points proportionnelle à celle du travail inachevé. De même, des points pourront être enlevés en cas d'accomplissement exceptionnel. Ainsi le but est d'avoir le moins de points possible !

4.2.8 Comptes-rendus

À l'issue de chaque réunion, un compte-rendu était rédigé pour résumer les propos traités, formaliser les décisions prises et la liste des responsabilités désignées à chaque membre de l'équipe pour la prochaine réunion. La rédaction était prise en charge par chacune d'entre nous de manière cyclique.

Chaque compte-rendu a été rédigé sous le modèle de celui représenté aux figures 4.10, 4.11 et 4.12 .

4.3 Organigramme projet

Selon leur localisation dans l'organigramme, il existe 3 types de projets :

— Au sein d'un même service : c'est un *projet local.*

— Impliquant plusieurs services différents : des *projets transversaux.*

— *projets sortis* : de taille importante et font appel à des intervenants détachés spécialement.

Notre projet est clairement un projet tranversal au sein d'une structure fonctionnelle.

En effet, la structure fonctionnelle est une structure où y'en n'a pas un chef de projet ce qui encourage l'initiative, favorise l'adaptation aux évolutions de l'environnement et facilite la circulation des informations car elle divise la division du travail se fait par fonctions ce qui favorise l'efficacité du travail.

Cette structure a permis à chacune d'entre nous de décider quelles tâches leur a été confiées :

— Cyrielle Lacrampe–Diter : responsable de la rédaction de la matrice SWOT, de la fonction *geog* ,*date*, *et_ aa*, *var_ aa* et de la partie 3.1 du LaTeX.

— Hind Latrache : responsable de la rédaction du diagramme WBS, de la matrice RACI, de la fonction *var_ leng*, *et_ leng*, *fold_ change*, *mediane_ aa* et *premquartil_ aa* et de la partie 3.2, 4.1, 4.2.1, 4.2.2, 4.2.3 et de la partie 4.3 du LaTeX.

— Gisel Rodríguez : responsable de la rédaction du diagramme WBS, de la matrice RACI, de la fonction *var_ n*, *et_ n*, *codons*, *date_ aa* et *geog_ aa* et de la partie 3.2, 4.2.5, 4.2.6, 4.2.7, et de la partie 4.2.8 du LaTeX.

— Céline Zhu : responsable de la rédaction du cahier de charge, du diagramme de Gantt, de la fonction *mediane*, *premquartil*, *fas2ch*, *var_ leng_ aa* , *et_ leng_ aa*, *lev_ dis*, *needwu*, et *motfind*, de l'introduction, de la couverture et du chapitre 1 et 2 du LaTeX.

Conclusion

"Les avancées sur les vaccins et les traitements, les innovations technologiques et l'ingénierie associée sont extrêmement prometteuses." nous a dit le Directeur. Après ce projet, nous pouvons apprécier l'ampleur du travail nécessaire à la construction des données nécessaire à l'élaboration d'un vaccin. La rapidité à laquelle l'Humanité y est parvenue est impressionante. La capacité d'évolution du COVID-19 semble être hors du commun, avec une modification importante des protéines pour une faible modification de son ARNm. Il est donc notre devoir à tous, d'empêcher l'exacerbation de cette évolution. Protégeons-nous, protégeons les autres.

4.4 Conclusion personnelle

Pour conclure, voici le ressenti du projet de la part de chacune des intégrantes de l'équipe.

4.4.1 Cyrielle Lacrampe–Diter

Ce projet m'a permis de développer des compétences variées : c'était la première fois que j'utilisais LateX et que je mettais git en pratique pour un travail de groupe. Mais la plus grosse différence avec les précédents travaux en groupe que j'ai pu faire au cours de ma scolarité, c'est le côté gestion de projet qui était totalement nouveau. Le fait d'avoir des outils et des méthodes prédéfinis sur lesquels se reposer a apporté un cadre et une organisation appréciable.

Cependant, je pense cela a aussi un peu été la raison pour laquelle on a démarré un pau lentement. Nous avons passé le début du projet à vouloir bien suivre la gestion de projet et nous avons mis du temps à commencer à coder réellement, ce qui s'est accumulé avec la situation de confinement dans laquelle il m'est au fur et à mesure devenu plus difficile de me concentrer et de travailler régulièrement. Au final, notre temps n'a pas forcément été gérée

de façon optimale, et il est dommage que nous ayons pu passer moins de temps sur les fonctions difficiles que sur les premières que nous avons codé.

De plus, bien que notre groupe ait bien fonctionné, le confinement a fait que l'on pense moins souvent à communiquer sur nos avancées respectives, ce qui n'a pas arrangé notre retard.

4.4.2 Hind Latrache

Personnellement, ce projet de TAPS m'a appris beaucoup de choses. Ces choses incluent l'utilisation de ce que j'ai appris en TOP et en MOOC et se sentir à l'aise de travailler avec d'autres personnes.

En effet, ce projet m'a permis d'appliquer les techniques de gestion de project apprises en MOOC et de voir l'importance du management en réalisant un projet avec d'autres personnes avec une date limite.

Ce projet m'a également aidé à me familiariser avec le travail d'équipe car je n'ai personnellement jamais vraiment eu la chance de travailler aussi longtemps avec d'autres personnes.

J'ai également appris à utiliser le LaTeX et à gérer le temps et les tâches sans me sentir dépassé même si nous avons perdu beaucoup de temps au début sur des choses qui n'étaient pas très importantes.

4.4.3 Gisel Rodríguez

Pour ma part, ce projet de TAPS m'a permis d'acquérir et d'implémenter plusieurs connaissances.

D'une part, on a mis en place les outils d'organisation de projet vus lors du MOOC. Ceci m'a permis de constater pas seulement comment les utiliser mais aussi à quel points ils ont facilité le déroulement du projet.

D'autre part, le développement d'algorithmes et leurs tests ont aussi contribué à mieux m'approprier des connaissances acquises lors du module de TOP. De même, la rédaction d'une partie du compte rendu sur LaTeX m'as permis de me familiariser plus avec ce logiciel.

Enfin, la dynamique du travail en équipe m'a permis de me familiariser plus avec les concepts d'autonomie, de responsabilité et de communication lors des projets réalisés à plusieurs.

4.4.4 Céline Zhu

La différence entre théorie et pratique m'a parfois surprise durant la réalisation de ce projet que ce soit en gestion de projet ou en informatique. Qu'est-ce qu'on pourrait améliorer ? Beaucoup de choses, mais je citerai trois

points clés : un démarrage en gestion de projet plus rapide, une sauvegarde plus automatique des fonctions codées et enfin être moins optimiste quant aux dates butoires.

Ce premier projet d'informatique m'a permis de vérifier la bonne acquisitions des compétences acquise durant ce premier trimestre à TELECOM NANCY, ainsi d'entrevoir ne serait ce brièvement le monde la bioinformatique, domaine dans lequel je serai professionellement intéressée.

En conclusion, ce fut une expérience enrichissante qui s'est très bien passée dû à la synergie de travail du groupe.

Bibliographie

[1] Td1 : Complexité algorithmique.

[2] Olivier Festor Martin Quinson, Sébastien da Silva. Algorithmic thinking and problem solving techniques.

[3] MGX. Glossaire de statistique pour la génomique.

[4] Paco Nathan. Visualizing geospatial data in python, 2020.

[5] U.S. National Library of Medicine. Site web du ncbi sars-cov-2 resources.

[6] Wikipedia contributors. Fold change — Wikipedia, the free encyclopedia, 2020. [Online ; accessed 5-January-2021].

[7] Wikipédia. Codon — wikipédia, l'encyclopédie libre, 2019. [En ligne ; Page disponible le 2-décembre-2019].

[8] Wikipédia. Statistique descriptive — wikipédia, l'encyclopédie libre, 2019. [En ligne ; Page disponible le 17-juin-2019].

[9] Wikipédia. Algorithme de needleman-wunsch — wikipédia, l'encyclopédie libre, 2020. [En ligne ; Page disponible le 20-janvier-2020].

[10] Wikipédia. Distance de levenshtein — wikipédia, l'encyclopédie libre, 2020. [En ligne ; Page disponible le 27-décembre-2020].

Annexes

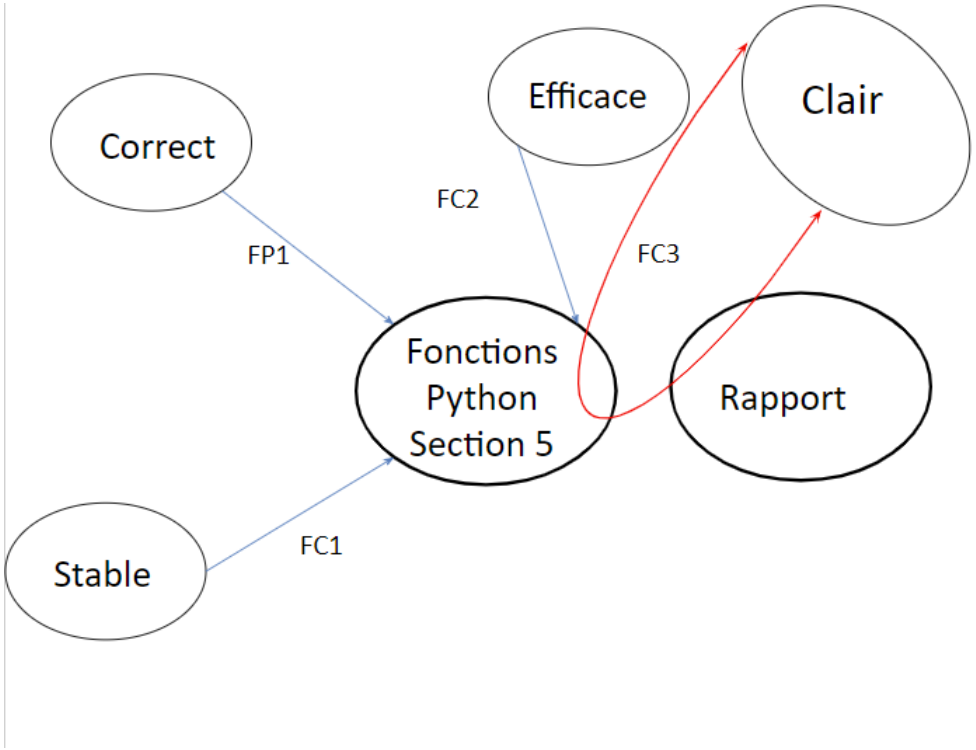


FIGURE 4.1 – Schéma du cahier de charge

Fonctions	Utilisations
FP1	Respecter les résultats attendus de la section 5.
FC1	Comportement robuste , pour un pas de discrétisation tendant vers 0.
FC2	Rapide et utilise peu de mémoire: somme totale du temps d'exécution en dessous de 6 minutes et mémoire totale en dessous de celle des ordinateurs utilisés.
FC3	Compréhensible à tous utilisateurs ayant des connaissances générales du code Python en moins de 5 minutes. (Commentaires et rapport)

FIGURE 4.2 – Schéma du cahier de charge

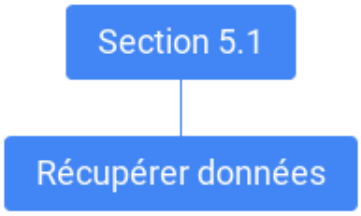


FIGURE 4.3 – Diagramme WBS section 5.1

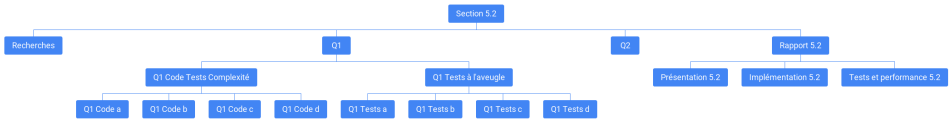


FIGURE 4.4 – Diagramme WBS section 5.2

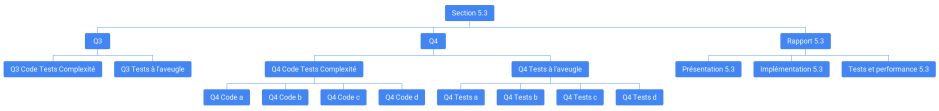


FIGURE 4.5 – Diagramme WBS section 5.3

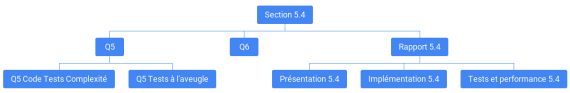


FIGURE 4.6 – Diagramme WBS section 5.4

FIGURE 4.7 – Diagramme WBS section 5.5

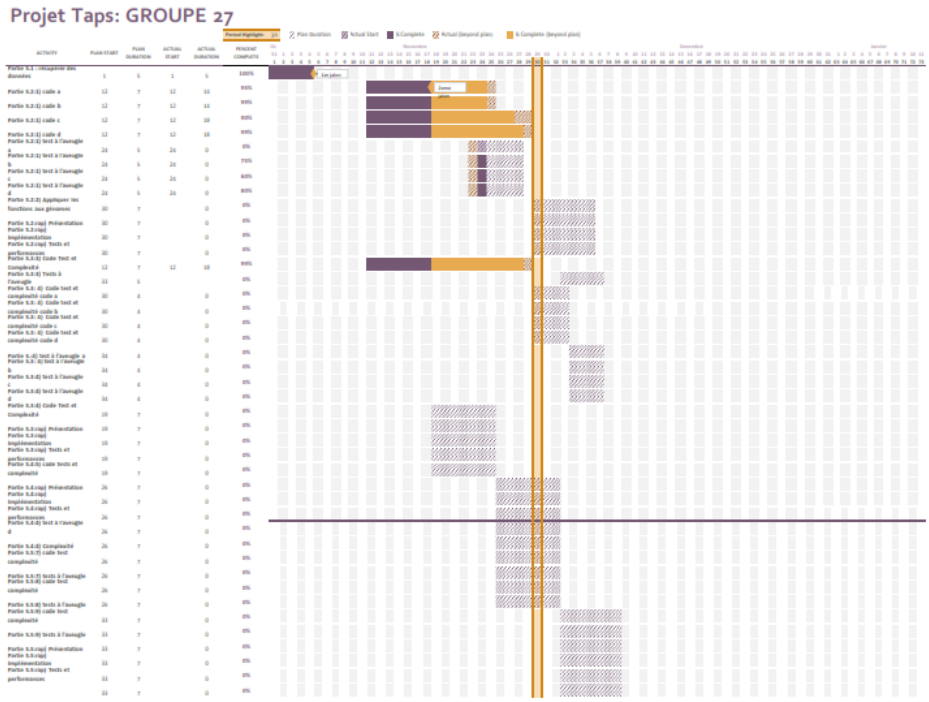


FIGURE 4.8 – Diagramme de Gantt

				Cyrielle	Hind	Gisel	Céline
Section 5.1	Récupérer données			R	R	R	A/R
Section 5.2	Q1	Code Tests Complexité	Code a	I	I	I	A/R
			Code b	I	I	A/R	I
			Code c	I	A/R	I	I
			Code d	A/R	I	I	I
		Tests à l'aveugle	Tests a	I	I	A/R	I
			Tests b	I	A/R	I	I
			Tests c	A/R	I	I	I
			Tests d	I	I	I	A/R
	Q2			R	I	I	A/R
	Rapport 5.2	Présentation 5.2		R	I	I	A/R
		Implémentation 5.2		R	I	I	A/R
		Tests et performance 5.2		R	I	I	A/R
Section 5.3	Q3	Code Tests Complexité		I	I	A/R	I
		Tests à l'aveugle		I	A/R	I	I
	Q4	Code Tests Complexité	Code a	I	A/R	I	I
			Code b	A/R	I	I	I
			Code c	I	I	I	A/R
			Code d	I	I	A/R	I
		Tests à l'aveugle	Tests a	A/R	I	I	I
			Tests b	I	I	I	A/R
			Tests c	I	I	A/R	I
			Tests d	I	A/R	I	I
	Rapport 5.3	Présentation 5.3		I	I	I	I
		Implémentation 5.3		I	I	I	I
		Tests et performance 5.3		I	I	I	I
Section 5.4	Q5	Code Tests Complexité		I	I	I	A/R
		Tests à l'aveugle		I	I	I	I
	Q6			I	I	I	I
	Rapport 5.4	Présentation 5.4		I	I	I	I
		Implémentation 5.4		I	I	I	I
		Tests et performance 5.4		I	I	I	I
Section 5.5	Q7	Code Tests Complexité		I	I	I	A/R
		Tests à l'aveugle		I	I	I	I
	Q8	Code Tests Complexité		I	I	I	A/R
		Tests à l'aveugle		I	I	I	I
	Q9	Code Tests Complexité		I	I	I	A/R
		Tests à l'aveugle		I	I	I	I
	Rapport 5.5	Présentation 5.5		I	I	I	I
		Implémentation 5.5		I	I	I	I
		Tests et performance 5.5		I	I	I	I

FIGURE 4.9 – Matrice RACI

Projet TAPS 1A - Compte-rendu n°9

Type de réunion : Réunion d'avancement	**Lieu de la réunion :** Réunion en visioconférence sur Discord
Présents : • ZHU Céline • LACRAMPE--DITER Cyrielle • LATRACHE Hind • RODRIGUEZ BAIDE Gisel **Rédaction du compte-rendu :** • LACRAMPE--DITER Cyrielle	**Date :** 29 novembre 2020 **Heure :** 16h00 **Durée :** 1h

• Ordre du jour : Fonctions question 1

○ Tests temporels

Parmi les tests que l'on applique à nos fonctions, nous allons vérifier que nos fonctions s'exécutent en un temps inférieur à un temps de référence. Il faudra calculer ce temps de référence selon la complexité de la fonction et selon la méthode du TD1 de TOP.

○ Diagramme de Gantt

Mise à jour du diagramme de Gantt et détermination de la to-do list jusqu'au 6 décembre.

• Décisions

○ Todo list

Tâche	Pilote	Echéance	Livrables
5.1	LATRACHE Hind	02/12/20	Demander de l'aide à Christophe Boutier pour

FIGURE 4.10 – Compte-rendu feuille 1

			réussir à importer Biopython
Q1 code a	ZHU Céline	02/12/20	Rédiger le rapport et faire les tests temporels des fonctions `fas2ch`, `mediane` et `premquartil`
Q1 code b	RODRIGUEZ BAIDE Gisel	02/12/20	Mettre le rapport sur le drive et faire les tests temporels des fonctions `et_n` et `var_n`
Q1 code c	LATRACHE Hind	02/12/20	Rédiger le rapport et faire les tests temporels des fonctions `et_leng`, `var_leng` et `fold_change`
Q1 code d	LACRAMPE--DITER Cyrielle	02/12/20	Tests temporels des fonctions `geog` et `date`
Q1 tests a	RODRIGUEZ BAIDE Gisel	02/12/20	Tests des fonctions `fas2ch`, `mediane` et `premquartil`
Q1 tests b	LATRACHE Hind	02/12/20	Tests temporels des fonctions `et_n` et `var_n`
Q1 tests c	LACRAMPE--DITER Cyrielle	02/12/20	Tests de la fonctions `fold_change` et tests temporels des fonctions `et_leng` et `var_leng`
Q1 tests d	ZHU Céline	02/12/20	Tests temporels des fonctions `geog` et `date`
Q2	ZHU Céline LACRAMPE--DITER Cyrielle	06/12/20	Rédaction de la Q2 pour le rapport
Rapport 5.2	ZHU Céline LACRAMPE--DITER Cyrielle	06/12/20	Rapport LaTeX de la section 5.2
Q3 tests	LATRACHE Hind	02/12/20	Tests de la fonction `codons`
Q4 code a	LATRACHE Hind	02/12/20	Code, rapport et tests de `mediane_aa` et `premquartil_aa`
Q4 code b	LACRAMPE--DITER Cyrielle	02/12/20	Code, rapport et tests de `et_aa` et `var_aa`
Q4 code c	ZHU Céline	02/12/20	Code, rapport et tests de `et_leng_aa` et `var_leng_aa`
Q4 code d	RODRIGUEZ BAIDE Gisel	02/12/20	Code, rapport et tests de `date_aa` et `geog_aa`

FIGURE 4.11 – Compte-rendu feuille 2

Q4 tests a	LACRAMPE--DITER Cyrielle	06/12/20	Tests de `mediane_aa` et `premquartil_aa`
Q4 tests b	ZHU Céline	06/12/20	Tests de `et_aa` et `var_aa`
Q4 tests c	RODRAGUEZ BAIDE Gisel	06/12/20	Tests de `et_leng_aa` et `var_leng_aa`
Q4 tests d	LATRACHE Hind	06/12/20	Tests de `date_aa` et `geog_aa`

- **Prochaines réunions**
  - **Réunion d'avancement**

Date : 2 décembre 2020
Heure : 17h00
Sujet : Avancement du travail
Participants : ZHU Céline, RODRIGUEZ BAIDE Gisel, LATRACHE Hind et LACRAMPE--DITER Cyrielle

  - **Réunion d'avancement**

Date : 6 décembre 2020
Heure : 16h00
Sujet : Avancement du travail
Participants : ZHU Céline, RODRIGUEZ BAIDE Gisel, LATRACHE Hind et LACRAMPE--DITER Cyrielle

FIGURE 4.12 – Compte-rendu feuille 3