UNIVERSITÉ DU QUÉBEC À MONTRÉAL

# UN NOUVEL ALGORITHME POUR RETROUVER LES RELATIONS PHYLOGÉNÉTIQUES ENTRE LA DISTRIBUTION GÉOGRAPHIQUE DES ESPÈCES ET LEURS COMPOSITIONS GÉNÉTIQUES

MÉMOIRE
PRÉSENTÉ
COMME EXIGENCE PARTIELLE
DE LA MAÎTRISE EN INFORMATIQUE

PAR
NADIA TAHIRI

DÉCEMBRE 2012

# REMERCIEMENTS

Au terme de cette maîtrise, je tiens à remercier vivement mon directeur de recherche, Dr. Vladimir Makarenkov, pour sa patience sans égale et ses nombreuses suggestions pour la réalisation de ce projet de recherche. Je tiens également à exprimer mes plus vifs remerciements à Dr. Pedro Peres-Neto, mon codirecteur de recherche, pour m'avoir conseillée et orientée tout au long de ma maîtrise. Qu'ils trouvent ici toute l'expression de ma gratitude et de ma profonde reconnaissance.

Je désire également remercier Dr. Abdoulaye Banire Diallo, pour ces nombreux conseils dans le domaine de la phylogénie, ainsi qu'à tous mes collègues de bio-informatique, Dunarel Badescu, Alix Boc, Alpha Boubacar Diallo, Mehdi Layeghifard, Mickael Leclercq, Étienne Lord, Alena Tsikhanovich, Matthieu Willems et tout le personnel du département d'Informatique de l'UQÀM, dont Jérôme Tremblay.

Je désirerai aussi remercier les professeurs qui liront ce mémoire. Je leur suis reconnaissante du temps qu'ils ont accordé pour lire ce travail.

J'adresse ma totale reconnaissance au comité d'accréditation des bourses FARE, Hydro-Québec et la RBC Banque Royale, ainsi qu'à la fondation de UQÀM.

À toute ma famille, mes amis et tous ceux qui m'ont soutenue durant mes deux années de maîtrise, qu'ils trouvent ici mes remerciements les plus sincères.

À mes parents, Aïcha Tahiri et Mohamed Tahiri.

TABLE DES MATIÈRES

# LISTE DES FIGURES

# LISTE DES TABLEAUX

# LISTE DES ACRONYMES

| | |
|---|---|
| AA | Acide Aminé |
| ADN | Acide DésoxyriboNucléique |
| APOB | Apolipoprotéine |
| ARN | Acide RiboNucléique |
| ASM | Alignement de Séquences Multiples |
| ATP | Adenoside TriPhosphate |
| $ATP_{ASE}$ | ATP synthase |
| BD | Base de Données |
| BDNF | Brain Derived Neurotrophic Factor |
| BRCA1 | Breast Cancer susceptibility Protein 1 |
| CO | Cytochrome Oxidase |
| GHR | Growth Hormone Receptor |
| LDL | Low-Density Lipoprotein |
| ML | Maximum Likelihood |
| NADH | Nicotinamide Adenine Dinucleotide |
| NC$\alpha$P-1 | Nicotinic Cholinergic receptor Alpha Polypeptide 1 precursor |
| NCBI | National Center for Biotechnology Information |
| NJ | Neighbor Joining |
| PERL | Pratical Extraction and Report Language |
| PHYLIP | PHYLogency Inference Package |
| PHYML | Phylogenetic estimation using Maximum Likelihood |
| PPNOC | Prepronociceptine |
| RAP-1 | Recombination Activing Protein 1 |
| RBP | Retinoid Binding Protein |
| RF | distance de Robinson et Foulds |
| SGBD | Système de Gestion de Bases de données |
| SRY | Sex determining Region Y protein |
| UPGMA | Unweighted Pair Group Method with Arithmetic mean |

# RÉSUMÉ

L'objectif de ce projet de maîtrise est de développer un nouvel algorithme permettant de retrouver les relations phylogénétiques entre un arbre de référence (par exemple, l'arbre de la distribution géographique des espèces ou des paramètres climatiques) et un arbre caractérisant un fragment de l'alignement de séquences multiples (ASM).

Pour ce faire, nous récupérerons d'abord les différents fragments d'un ASM donné. Nous les soumettrons par la suite aux différents programmes du paquet PHYLIP (Seqboot, ProtDist ou DnaDist, Neighbor et Consense) et le programme PhyML afin d'obtenir un arbre consensus avec les valeurs de bootstrap sur ses branches. À partir de chaque arbre consensus, nous calculerons son bootstrap moyen. De plus, nous comparerons topologiquement l'arbre consensus obtenu à l'arbre de référence pour connaître la distance de Robinson et Foulds (RF) normalisée entre eux. Pour chaque fragment d'un ASM, nous conserverons uniquement les données relatives à des fragments correspondant à la distance RF normalisée la plus petite (i.e., celle qui représente la plus grande similitude entre les deux arbres). Dans le cas où plusieurs fragments correspondront à la même valeur de la distance RF normalisée, l'estimation se poursuivra sur l'arbre consensus ayant le score de bootstrap le plus élevé (i.e., meilleur support de l'arbre).

Pour connaître la performance de notre algorithme, nous utiliserons un jeu de données de 52 espèces appartenant au groupe des Carnivores se localisant en Amérique du Nord. Nous récupérons aussi 21 protéines issues de la base de données GenBank. La construction des arbres de référence se fera à partir de données climatiques de l'habitat de ces espèces (i.e., température, précipitation et altitude).

Notre algorithme permettra de trouver des sous-séquences des gènes donnant une similarité topologique accrue entre l'arbre de référence et l'arbre phylogénétique obtenu à partir des séquences.

Mots clés : arbre phylogénétique, phylogéographie, distance de Robinson et Foulds, bootstrap, alignement de séquences multiples, paquet PHYLIP, GenBank.

# INTRODUCTION

La problématique développée dans ce mémoire de bio-informatique explorera les méthodes pour interpréter certains types de données environnementales en fonction du patrimoine génétique. Les préoccupations relatives aux changements climatiques provoquent de croissantes inquiétudes. C'est en prenant connaissance des milieux d'habitats des espèces et en les corrélant avec les patrimoines génétiques de ces mêmes espèces que nous pourrons retrouver des liens entre ces différents ensembles de données. Ce mémoire porte sur le développement d'un nouvel algorithme permettant de retrouver les relations entre un arbre de référence (i.e., l'arbre de distributions géographiques des espèces, les arbres : de températures, d'altitudes, de précipitations des habitats ou autres) avec leurs compositions génétiques. Ce nouvel algorithme nous permettra donc de retrouver quels gènes ou quelles sous-parties d'un gène sont sensibles ou propices à un environnement donné.

Ce mémoire se présente en quatre chapitres dont l'organisation est comme suit :

## Organisation

- Le **premier chapitre** présentera une revue de la littérature sur les différents termes employés dans le domaine de la phylogénie. Nous énumérerons dans un premier temps, les principales définitions rudimentaires sur les arbres phylogénétiques, ainsi que quelques propriétés qui s'y appliquent. Par la suite, nous exposerons les différentes techniques de reconstruction d'arbres.

- Le **deuxième chapitre** introduira l'algorithme permettant de détecter le fragment d'un alignement de séquences multiples correspondant à la plus petite distance de Robinson et Foulds (i.e., une très grande similitude topologique des

arbres comparés). Dans le cas où plusieurs fenêtres ont la même valeur de la distance RF, la sélection se poursuivra sur le bootstrap moyen qui doit être le plus élevé (i.e., meilleur support statistique de l'arbre).

- Le **troisième chapitre** introduira différents jeux de données que nous avons sélectionnés. Suite à l'énumération des espèces prises en compte dans notre étude, nous proposerons une description sommaire des différents gènes obtenus depuis la base de données GenBank. Nous énumérerons par la suite tous les arbres de référence. Cette liste nous permettra de les comparer efficacement aux arbres relatifs des fragments d'alignements de séquences multiples. Enfin, nous indiquerons la liste des localisations géographiques de l'arbre de référence $T_2$.

- Le **quatrième chapitre** mettra en application l'algorithme décrit dans le chapitre II sur les données réelles décrites dans le chapitre III. Nous inspecterons la performance de notre algorithme sur un même jeu de données en utilisant les fenêtres de tailles différentes. Puis, en résumé de ce chapitre, nous analyserons et discuterons des résultats obtenus.

- En **conclusion**, nous rappellerons les points importants de ce mémoire et indiquerons les perspectives futures qui sont actuellement envisagées pour accroître la performance de notre algorithme.

# CHAPITRE I

# REVUE DE LA LITTÉRATURE

## 1.1 Introduction

Avant d'introduire l'algorithme permettant la détection des fragments de gènes en lien avec les paramètres environnementaux, qui sera décrit dans le chapitre II, nous réaliserons une revue de la littérature. À travers ce chapitre, nous tracerons un survol historique de la discipline de la phylogénie, nous en spécifierons certains des aspects fondamentaux (e.g., arbre phylogénétique) et en proposerons une définition sommaire. Le vocabulaire et la terminologie employée seront explicités, ainsi que quelques propriétés mathématiques caractérisant les arbres phylogénétiques. C'est à l'aide de ces notions que nous pourrons alors décrire les différentes techniques permettant la reconstruction des arbres ainsi que le mode de validation de la robustesse de ces branches. Le dernier point nous permettra d'indiquer une définition sommaire de la phylogéographie. Cette définition aura pour fonction de comprendre le lien entre la phylogénie et la phylogéographie.

* * *

## 1.2 L'arbre phylogénétique

### 1.2.1 Historique du terme phylogénie

Le terme **phylogénie** est composé de plusieurs mots grecs : phylon («tribu», «race», «espèce»), génésis («genèse », «création») et gennaô («engendrer», «créer»). Ce terme a été

introduit par Ernst Haekel en 1866 pour signifier les relations entre les différentes espèces animales ou végétales. En 1874, Haeckel a publié son travail principal *"Anthropogenie oder entwickelungsgeschichte des menschen"* (Haeckel, 1874).

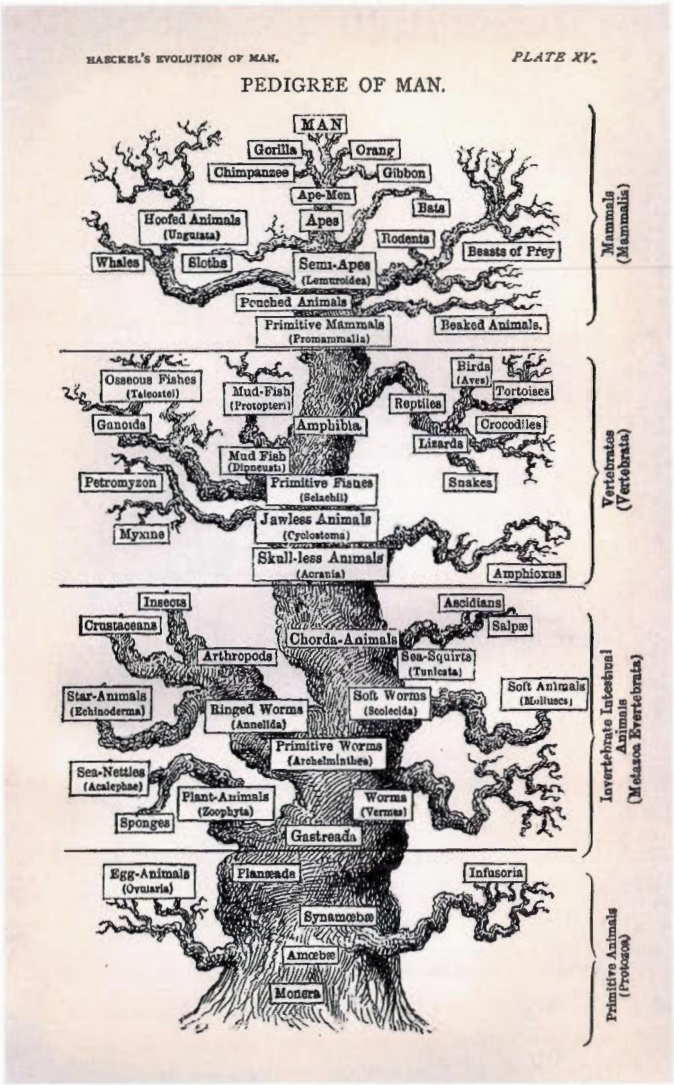


Figure 1.1: Représentation d'un arbre phylogénétique des êtres vivants (Haeckel, 1874).

Sur la figure 1.1 de 1874, Haeckel réalise des liens entre les différentes espèces. La différence majeure avec les représentations actuelles est que la division des clades (i.e., même

groupe d'espèces) se fait de façon horizontale (voir la section 1.2.3).

En 1859, Charles Darwin, dans la première édition de l'*Origine des espèces* (Darwin, 1859), définit le terme phylogénie par : "les lignes généalogiques de tous les êtres organisés". C'est la définition la plus utilisée actuellement. Cependant, la première représentation d'un arbre d'évolution des espèces de Darwin date de 1837 (Darwin, 1837), ce qui est illustrée sur la figure 1.2.

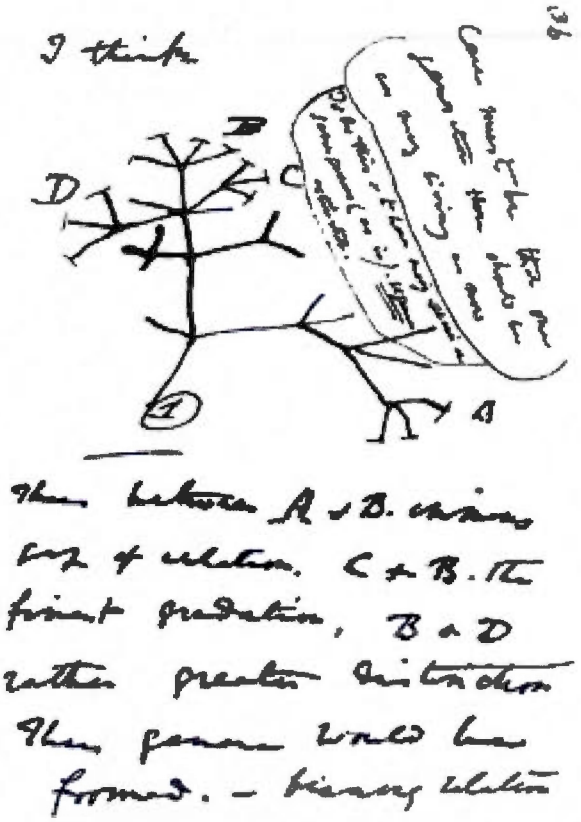

Figure 1.2: Première schématisation d'arbre phylogénétique de Darwin (Darwin, 1837).

Sur la figure 1.2, Darwin réalise la construction de cet arbre phylogénétique par le rapprochement, essentiellement morphologique, des espèces puisqu'à cette même époque, le

support de l'information génétique n'était pas encore découvert. Nous verrons plus loin que le support morphologique s'avèrera souvent peu efficace.

En 1809, Jean-Baptiste Lamarck a introduit dans son premier tome de "*Philosophie zoologique*" (Lamarck, 1809), portant sur la notion de filiation des animaux, bien que cette notion ai été illustrée plus tardivement dans le second tome en 1830 (Lamarck, 1830). Cette représentation est indiquée sur la figure 1.3 et met en évidence un ordre naturel reflétant l'ordre des liens de reproduction des espèces entre elles.

Sur la figure 1.3 de Lamarck, qui se lit de haut en bas, on peut noter la particularité d'introduire indirectement l'aspect temporel de l'arbre. Bien que cet aspect ne figure pas directement sur le tableau, il est subtilement mentionné dans le titre par le terme d'*origine*. Dans la nouvelle version de "*Philosophie zoologique*" du tome I, revue par Martins (Lamarck et Martins, 1873), ce dernier sera d'ailleurs encore plus explicite à cet égard puisqu'il le remplacera par "le temps pour la formation de ces alternatives".

Après la découverte de l'acide désoxyribonucléique (ADN), qui est le support de l'information génomique, en 1889 par Richard Altmann, plusieurs équipes de chercheurs ont tenté d'étudier la différence de composition des séquences entre les espèces pour mieux connaître l'histoire évolutive. Ces différentes études permettront alors de conclure que des changements successifs (ou mutations) de l'ADN peuvent se produire et ainsi aboutir à la formation d'une nouvelle espèce (Darwin, 1859). Ce processus est aussi nommé spéciation. Nous verrons que cette approche qui utilise le matériel génétique comme support de regroupement des espèces serait plus efficace que celle utilisant un support morphologique. Les explications et les raisons du choix de ce support de comparaison d'un lot d'espèces seront indiquées dans la partie 1.4.1.1.

En 1965, l'équipe de Willi Hennig a introduit la classification phylogénétique des espèces, c'est-à-dire définissant la notion de **clade** ou **groupe monophylétique**[1] (Hen-

1. Dans un arbre phylogénétique, un groupe est monophylétique si tous les membres de ce groupe sont dans un même sous arbre.

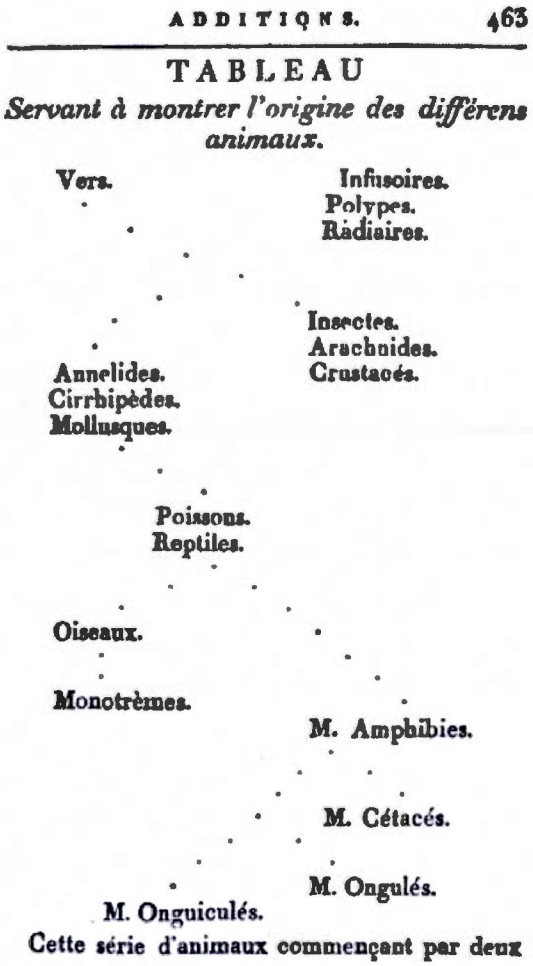


Figure 1.3: Représentation de l'origine des différents animaux (Lamarck, 1830).

nig, 1965). Cette approche permet de mettre en évidence les liens de parenté des espèces et ainsi de mieux comprendre leur histoire évolutive (Felsenstein, 2004).

### 1.2.2 Définition de l'arbre phylogénétique

À la suite de cette étude historique du terme **phylogénie**, nous pouvons donner une définition plus formelle de l'arbre phylogénétique. D'un point de vue mathématique, un arbre est une structure particulière. Il s'agit, en effet, d'un graphe acyclique. Et d'un

point de vue biologique, un arbre phylogénétique devra illustrer les liens de parenté entre les espèces au cours du temps.

Pour illustrer ce concept, prenons un petit exemple simple de trois espèces distinctes $A$, $B$ et $C$ (voir figure 1.4).

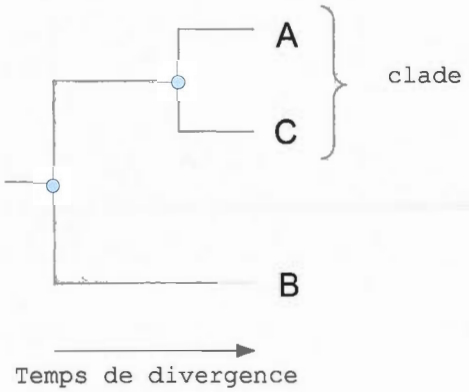


Figure 1.4: Illustration d'un arbre phylogénétique à trois espèces.

Ces espèces peuvent être actuelles ou éteintes. Supposons que l'espèce A est très proche génétiquement de l'espèce C, en remontant le temps, on devrait relier l'espèce A et l'espèce C via la même espèce ancestrale. Le temps qu'il a fallu pour unir ces deux espèces représente le degré de divergence de ces espèces ou encore définit la "somme des modifications" selon Darwin. Les espèces A et C constituent un groupe d'organismes (i.e., clade) dérivant de la même espèce ancestrale. Puis en continuant à remonter le temps, nous allons finir par réunir l'ancêtre des espèces A et C avec l'espèce B par un ancêtre commun.

### 1.2.3 Terminologie des arbres

Un arbre phylogénétique est une structure qui contient quatre principaux éléments (Felsenstein, 2004).

- Les **feuilles** ou les **nœuds externes** sont la représentation des espèces actuelles ou éteintes, voir illustration à la figure 1.5, pour lesquelles on dispose des informations

permettant la construction de l'arbre.

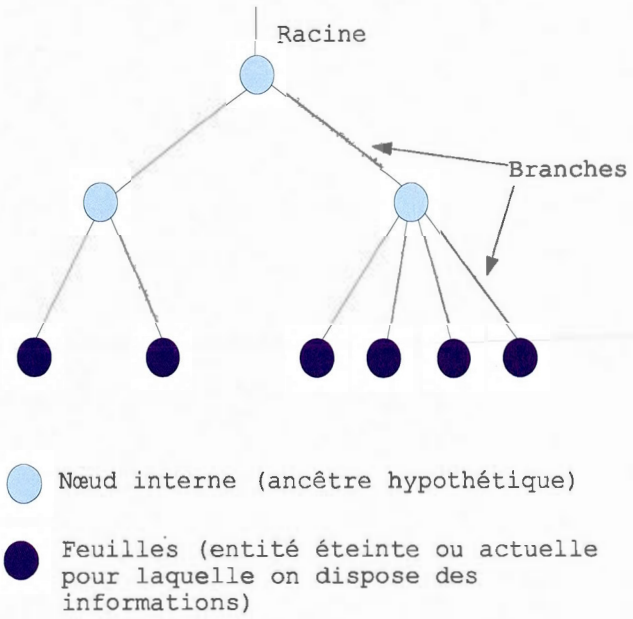


Figure 1.5: Représentation schématique d'un arbre phylogénétique basique avec annotations.

- Les **branches** ou les **segments** définissent les liens entre les espèces (i.e., taxons) en terme de descendance. Une branche peut être décrite en utilisant un ensemble de quatre critères.

  - La **mesure** constitue une distance d'évolution, c'est-à-dire le temps conduisant à la divergence des espèces avec le scénario le plus minimal que possible. Cette mesure peut se baser sur plusieurs critères tels que la quantité d'évolution ayant pour unité le temps d'évolution ou le nombre de mutations en se basant sur des données génétiques des espèces.

  - Le **poids** correspond à une pondération indiquée par une mesure. Pour les mêmes exemples illustrés dans le point précédent, il peut s'agir de la vitesse d'évolution

d'une part et du taux de mutation, ainsi que du coût des mutations d'autre part.

- L'**orientation** des branches illustre le sens de la branche. Par définition, on indiquera le premier nœud, l'*ancêtre* et le second nœud, le *descendant.*

- Le **degré d'un nœud** ou le **nombre de branches attachées à un sommet** permet d'illustrer le nombre de nœuds qui ont un lien direct avec le nœud en question. Quant aux nœuds internes, leur degré est toujours de 3 pour les arbres phylogénétiques résolus[2], et varie de 3 à $x$ avec $x \in \mathbf{N}$ (et $x > 3$) pour les arbres phylogénétiques non résolus[3].

- Les **nœuds internes** sont associés à des ancêtres hypothétiques ou virtuels.

- La **racine** représente l'ancêtre commun de toutes les espèces considérées. Elle peut être placée par la méthode de l'outgroup ou du point du médian. La première méthode consiste à ajouter aux séquences traitées, avant le calcul de l'arbre, une séquence très éloignée. En faisant cela, nous introduisons une nouvelle espèce qui n'appartiendra pas au groupe des espèces sélectionnées. Quant à la deuxième méthode, elle consiste à affecter à chaque nœud interne de l'arbre une séquence correspondant au consensus de ses fils, et choisir comme racine le nœud dont la séquence est la plus proche de la séquence consensus de toutes les feuilles.

## 1.3 Caractéristiques des arbres phylogénétiques

### 1.3.1 Variétés des arbres

Nous pouvons citer quatre variétés majeures d'arbres phylogénétiques : le *dendogramme*, le *cladogramme*, le *phénogramme* et le *phylogramme*. Rappelons brièvement les caracté-

2. Arbre phylogénétique dichotomique (bifurcation ou binaire)

3. Arbre phylogénétique polytomique (multifurcation ou n-aire)

ristiques de ces différentes variétés.

- Le **dendogramme** permet de dessiner un arbre sur la base de regroupements hiérarchiques (Phipps, 1971). Cette représentation s'apparente aux méthodes de regroupement (i.e., clustering). Cette structure met en avant les liens entre les différents taxons sous la forme d'une succession de branchements.

- Le **cladogramme** ou encore systématique phylogénétique est une structure utilisée pour retranscrire sur un arbre les espèces selon leurs liens de parenté. Cette structure est donc un dendogramme indiquant les liens phylogénétiques entre les taxons (Wheeler, 2003).

- Le **phénogramme** est un arbre qui traduit les relations de parenté entre des molécules, établi à partir d'une méthode phénétique (Colless, 1970). Il s'agit d'un dendogramme réalisé à partir d'une taxonomie numérique où les relations expriment les degrés de similitude globale.

- Le **phylogramme** est une représentation d'un arbre ayant des longueurs de branches proportionnelles à la quantité de changements des séquences. Il montre la valeur de la différence entre les taxons terminaux d'une manière quantitative. Sa représentation est celle d'un dendogramme ayant des branchements cladistiques avec un degré de divergence adaptative subséquente aux branchements.

### 1.3.2 Types d'arbres

De plus, il y a deux types d'arbres phylogénétiques : des arbres enracinés et des arbres non enracinés. Un arbre est enraciné lorsque sur l'arbre en question, un ancêtre commun a été identifié. Il s'agit d'un arbre orienté dans le sens du temps de l'évolution des taxons. Il reflète également les relations de descendances entre les nœuds. Cependant, il s'avère parfois impossible d'identifier de façon formelle l'ancêtre commun des différents taxons

(Fitch, 1971; Fitch et Margoliash, 1967). C'est pour cette raison qu'il est biologiquement plus conforme d'utiliser des arbres non enracinés (i.e., arbres sans la présence d'une racine ou d'un ancêtre commun à cet ensemble de taxons).

Pour aborder le prochain point traitant des différentes méthodes de reconstruction des arbres phylogénétiques, il faut connaître la taille de l'ensemble de toutes les possibilités d'arbres définis sur le même ensemble d'espèces. En prenant connaissance de ce nombre, on peut se demander s'il est réaliste de chercher toutes les possibilités d'arbres phylogénétiques différentes, afin de déterminer l'arbre le plus "réaliste" qui se rapproche le plus des connaissances biologiques actuelles.

Le nombre de différents arbres phylogénétiques dichotomiques enracinés définis sur un ensemble de $n$ espèces données, pour tout $n \geq 2$, est obtenu par la formule 1.1.

$$T_n{}^{enraciné} = \frac{(2n-3)!}{(2^{(n-2)})(n-2)!} \tag{1.1}$$

Le nombre de différents arbres phylogénétiques dichotomiques non enracinés définis sur un ensemble de $n$ espèces, pour tout $n \geq 3$, est obtenu par la formule 1.2.
Sachant que $T_n{}^{non\ enraciné} = T_{n-1}{}^{enraciné}$, donc :

$$T_n{}^{non\ enraciné} = \frac{(2n-5)!}{(2^{(n-3)})(n-3)!} \tag{1.2}$$

Le tableau suivant indique l'évolution des nombres d'arbres phylogénétiques possibles en fonction du nombre de taxons pour des arbres dichotomiques enracinés et des arbres dichotomiques non enracinés.

Tableau 1.1: Nombre de topologies différentes possibles pour les arbres enracinés et non enracinés en fonction du nombre de taxons.

| Nombres de taxons | Nombres d'arbres enracinés possibles | Nombres d'arbres non enracinés possibles |
|---|---|---|
| 2 | 1 | - |
| 3 | 3 | 1 |
| 4 | 15 | 3 |
| 5 | 105 | 15 |
| 6 | 945 | 105 |
| 7 | 10 395 | 945 |
| 8 | 135 135 | 10 395 |
| 9 | 2 027 025 | 135 135 |
| 10 | 34 459 425 | 2 027 025 |
| 15 | 2,134 E +14 | 7,906 E +12 |
| 20 | 8,200 E +21 | 2,216 E +20 |

Sur le tableau 1.1, nous constatons que la recherche exhaustive s'avérera très coûteuse en temps pour un nombre d'espèces aui est supérieur à 10. Or, la constitution d'un jeu de données exploitable pour la réalisation d'une étude, d'un point de vue de la pertinence de l'information, est généralement un ensemble de plus de 10 espèces (Fitch et Margoliash, 1967).

### 1.3.3 Représentations d'arbres

À travers la littérature, nous observons différents types de représentations d'arbres phylogénétiques. La figure 1.6 illustre les tracés les plus fréquemment utilisés (Barthélemy et Guénoche, 1988). Ces différentes représentations ont été obtenues à partir de la version web du logiciel T-Rex (Makarenkov, 2001; Boc, Diallo et Makarenkov, 2012). La représentation la plus utilisée est celle du tracé hiérarchique (figure 1.6 (c)).

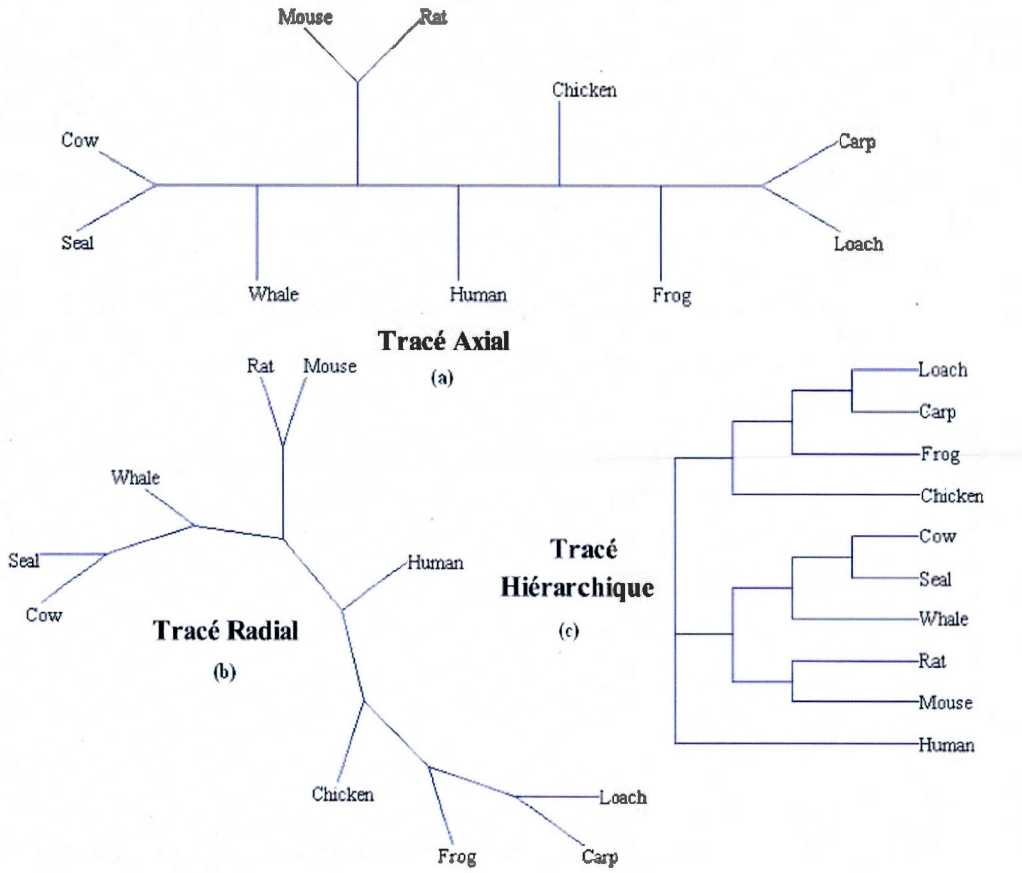


Figure 1.6: Différents types de tracés d'arbres phylogénétiques inférés par la version web du logiciel T-Rex (Makarenkov, 2001; Boc, Diallo et Makarenkov, 2012).

Il est aussi fréquent de voir, dans de récentes publications, une représentation circulaire (Chevenet et al., 2006). Cette représentation permet l'illustration de certains phénomènes biologiques, comme notamment les transferts horizontaux de gènes (Letunic et Bork, 2011).

### 1.3.4 Propriétés des arbres phylogénétiques

Il existe de nombreuses propriétés relatives aux arbres qui doivent être prises en compte pour permettre leur meilleure reconstruction. C'est la raison pour laquelle ces différentes méthodes de reconstruction tiennent compte de certaines propriétés combinatoires des

arbres (Barthélemy et Guénoche, 1988; Barthélemy et Guénoche, 1991), dont les principales sont les suivantes.

**Définition 1.** Soit $d(x, y)$ la distance entre le sommet $x$ et le sommet $y$ dans un arbre phylogénétique $T$. Cette distance est définie par la somme de toutes les longueurs des arêtes du chemin unique liant le sommet $x$ au sommet $y$.

**Définition 2.** Soit $X$ un ensemble fini de $n$ taxons et $d$ la dissimilarité sur l'ensemble $X$, qui est une fonction non négative sur $(X * X) \in \mathbf{R}+$ tel que pour tout $x, y$ appartenant à $X$ :

$$d(x, y) = d(y, x). \forall x, y \in X * X, et \tag{1.3}$$

$$d(x, x) = d(y, y) = 0 \ \forall x, y \in X, et \tag{1.4}$$

$$d(x, y) \geq d(x, x) \tag{1.5}$$

**Définition 3.** La dissimiliraté $d$ sur l'ensemble $X$ satisfait la condition des quatre points si pour tout $x$, $y$, $z$ et $w$ de $X$ :

$$d(x, y) + d(z, w) \leq Max\{d(x, y) + d(y, w); d(x, w) + d(y, z)\} \tag{1.6}$$

À la suite de la définition 3, nous présentons le théorème principal, de Zarestskii, Buneman, Patrinos, Hakimi et Dobson, sur la condition des quatre points et la représentabilité pour un arbre phylogénétique.

**Théorème 1.** *Toute dissimilarité d qui satisfait la condition des quatre points peut être représentée par un arbre phylogénétique, tel que pour tout $x, y$ appartenant à $X$, la valeur $d(x, y)$ est égale à la longueur du chemin liant les feuilles $x$ et $y$ dans $T$. Cette dissimilarité est nommée une distance d'arbre. Cet arbre sera unique.*

**Définition 4.** Pour un ensemble fini $X$, un arbre phylogénétique[4] avec une paire ordonnée ($T$,$\varphi$) consistant en un arbre $T$, avec un ensemble de sommets $V$ et un ensemble

4. i.e., un arbre additif ou un X-arbre (Barthélemy et Guénoche, 1988)

de relations $\varphi : X \to V$, ayant la propriété que, pour tout $x \in X$ avec un degré d'au moins deux, $x \in \varphi(X)$. Un arbre phylogénétique est binaire si $\varphi$ est une bijection de $X$ dans l'ensemble de feuilles de $T$ et que chaque sommet interne a un degré égal à 3.

## 1.4 Méthodes de reconstruction des arbres phylogénétiques

Il existe différentes méthodes de reconstruction d'arbres. L'enjeu principal des phylogénéticiens est de développer et de mettre en place des algorithmes efficaces permettant la construction des arbres le plus fidèlement reliés aux connaissances biologiques. Comme, nous l'avons indiqué dans la section 1.3.2, il est impossible de chercher tous les arbres possibles quand le nombre de feuilles de l'arbre est supérieur à 10 (voir tableau 1.1).

Actuellement, il existe au moins deux principales approches permettant de réaliser la reconstruction d'arbres phylogénétiques : les méthodes basées sur les distances et les méthodes basées sur les caractères (Felsenstein, 2004).

### 1.4.1 Approche basée sur les distances

#### 1.4.1.1 Support de la comparaison

La reconstruction d'arbres phylogénétiques s'appuie sur l'aspect des différentes modifications réalisées sur certains critères à travers la descendance. Ces critères étaient initialement de l'ordre morphologique. C'est à la suite de la découverte de l'ADN, puis de l'ARN, et ensuite des protéines que les phylogénéticiens ont pu réaliser la reconstruction des arbres sur ces différents supports. On constate que l'approche morphologique s'avère dès lors moins précise que celle basée sur le matériel génétique l'ADN, l'ARN ou les protéines. Il peut y avoir par exemple, quelques modifications sur les séquences qui ne sont pas visibles morphologiquement. De plus, les modifications sur les séquences sont plus précises, plus quantifiables et surtout non subjectives. C'est pour ces raisons, que nous allons, par la suite, nous baser exclusivement sur la comparaison du matériel génétique.

#### 1.4.1.2 Étape préliminaire : alignement des séquences

Le concept pour réussir la construction d'un arbre s'appuie sur l'analyse des modifications dans des séquences sur les différentes espèces étudiées. Lors de la reconstruction de l'arbre, nous avons une étape préliminaire qui consiste à mettre en correspondance des sites de séquences de manière à pouvoir comparer ce qui est comparable. Cette étape est nommée "alignement". Les séquences utilisées, comme nous l'avons mentionné, peuvent être de différents types : soit de l'ADN, soit de l'ARN, ou soit des protéines.

Tableau 1.2: Processus d'alignement.

```
>A
EDHRNVTVLSCQFRS
>B
EDHRNVTTLSCRFRS
>C
EYHRNVTFLSCQFRS
>D
EYDQNVTFLSCQFSR
>E
EYDQNVTFLSCQFRS
>F
EYDRNVTFLSCQFRS
```

(a)

⇒

```
6 15
A     EDHRNVTVLSCQFRS
B     EDHRNVTTLSCRFRS
C     EYHRNVTFLSCQFRS
D     EYDQNVTFLSCQFSR
E     EYDQNVTFLSCQFRS
F     EYDRNVTFLSCQFRS
```

(b)

Le tableau 1.2 illustre le processus d'alignement d'un ensemble de séquences, pour les espèces A, B, C, D, E et F.

Le tableau 1.2 (a) visualise le fichier d'entrée contenant des séquences de chaque espèce au format Fasta. Ces données proviennent souvent de la base de données publique GenBank de NCBI (`http://www.ncbi.nlm.nih.gov/genbank/`) (Maglott et al., 2007).

Le tableau 1.2 (b) représente l'alignement de séquences multiples (ASM) (Edgar et Batzoglou, 2006; Wallace, Blackshields et Higgins, 2005), au format Phylip. Il existe différents logiciels permettant l'alignement de séquences, par exemple, ClustalW (Li,

2003; Thompson, Gibson et Higgins, 2002), MUSCLE (Edgar, 2004) et MAFFT (Katoh et al., 2005) accessibles depuis le site web de T-Rex (`http://www.trex.uqam.ca/`).

Pour réaliser les alignements, il faut observer les différentes mutations qui se produisent sur les séquences d'un gène, pour un ensemble d'espèces, au cours de l'évolution. Les mutations se produisent en permanence sur le matériel génétique, entraînant ainsi la diversité des espèces. Il s'agit de modifications qui se produisent lors de la réplication de l'ADN. Parfois, ces modifications sont volontaires et nécessaires, c'est le cas, par exemple, du système immunitaire répondant à la production d'anticorps spécifiques.

Une fois les séquences alignées, une approche de reconstruction d'arbres phylogénétiques est utilisée afin d'obtenir un arbre qui reflète au mieux les données.

#### 1.4.1.3 De la matrice de dissimilarités à l'arbre phylogénétique

Cette approche réalise la reconstruction d'arbres à partir d'une matrice d'estimations de la distance évolutive $d$ (i.e., une matrice de dissimilarités). Cette matrice est construite en comparant les séquences deux à deux. Elle peut-être basée sur les distances observées entre toutes les paires d'espèces. Une fois ces mesures établies, les méthodes de reconstruction d'arbre par agglomération successive de lignées (e.g., Neighbor Joining (Saitou et Nei, 1987; Wang, Guo et Xing, 2012)) permettent de rechercher l'arbre phylogénétique ayant les longueurs de branches qui se rapprochent au mieux des distances mesurées. L'illustration de ces étapes est représentée ci-dessous.

Tableau 1.3: Alignement de séquences multiples au format Phylip pour 6 espèces.

```
6  15
A         EDHRNVTTLSCQFRS
B         EDHRNVTTLSCRFRS
C         EYHRNVTFLSCQFRS
D         EYDQNVTFLSCQFSR
E         EYDQNVTFLSCQFRS
F         EYDRNVTTLSCQFRS
```

À partir de ces alignements, nous pouvons calculer la matrice de dissimilarités en ap-

pliquant une transformation Séquences-Distances appropriée (e.g., Jukes Cantor (Jukes, 1969; Tuffley et al., 2012)).

Tableau 1.4: Matrice de dissimilarités pour 6 espèces.

| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| A | 0 | 2 | 3 | 8 | 8 | 7 |
| B | | 0 | 3 | 8 | 8 | 7 |
| C | | | 0 | 7 | 7 | 6 |
| D | | | | 0 | 2 | 3 |
| E | | | | | 0 | 3 |
| F | | | | | | 0 |

Cette matrice de dissimilarités de 6 espèces a été construite à partir des séquences du tableau 1.3. Indiquons que plus la dissimilarité est grande entre deux espèces, plus ces deux espèces seront éloignées dans l'arbre phylogénétique.

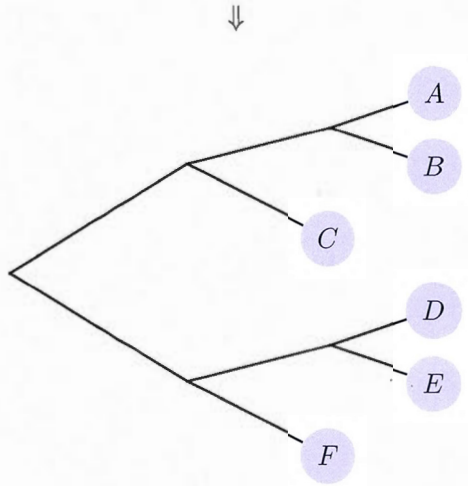


Figure 1.7: Arbre phylogénétique construit à partir de la matrice de dissimilarités du tableau 1.3.

La matrice de dissimilarités peut être calculée à partir des séquences alignées ou à partir d'autres types d'informations, en comptabilisant le nombre de substitutions qui se sont produites et en y affectant un poids pour chaque type de substitution. Soulignons un biais

essentiel relatif à cette technique. Ce calcul ne tient pas compte de la probabilité qu'un site puisse avoir plus d'une mutation au cours du temps d'évolution et également ne comptabilise pas les mutations neutres. Des mutations multiples peuvent alors survenir sans être observables. Notons que ce biais peut être éliminé par le choix stratégique du modèle d'évolution qui doit se faire en fonction des données observées.

### 1.4.2 Approche basée sur les caractères

Les approches basées sur les caractères sont généralement des méthodes plus robustes que celles basées sur les distances d'un point de vue statistique. Mais en contrepartie, ces approches sont plus lentes. Ces approches englobent les méthodes de parcimonie, de maximum de vraisemblance ainsi que les méthodes bayésiennes (Felsenstein, 2004).

- La cladistique ou encore systématique phylogénétique est l'étude de la classification des espèces permettant d'établir les relations de parenté sur des critères partagés entre les espèces étudiées. Cette méthode se base sur *l'approche de maximum de parcimonie* (Czelusniak et al., 1990). Ce dernier est le résultat de l'évolution demandant le minimum d'événements possible au cours de l'évolution (Sanderson, 2002).

- Le probabilisme ou *maximum de vraisemblance* est une méthode mise au point par Ronald Aylmer Fisher en 1922 (Fisher, 1922). Il s'agit d'une méthode statistique permettant d'inférer un arbre phylogénétique en termes de probabilités d'un échantillon donné (Aßfalg et Erdfelder, 2012). La probabilité calculée s'appuie essentiellement sur l'ordre de branchements et la longueur des arêtes d'un arbre en fonction d'un modèle évolutif. Le meilleur arbre est celui qui présente la meilleure vraisemblance, $Pb$, donnée par la formule 1.7.

$$Pb = \pi_c(\sum s[\pi br * p(br)]) \tag{1.7}$$

$\pi_c$ représente donc la probabilité de chaque caractère produit, qui est multiplié par la somme de tous les états des caractères représentés par $\sum s[\pi br * p(br)]$. Les méthodes de maximum de vraisemblance les plus connues sont PhyML (Guindon et al., 2009; Guindon et al., 2005) et RaxML (Stamatakis, Hoover et Rougemont, 2008).

- Les méthodes basées sur *les méthodes bayésiennes* (Bernardo, Smith et Berliner, 1994) sont liées aux méthodes basées sur le maximum de vraisemblance, à la différence que les premières approches utilisent une distribution *a priori*. Les méthodes bayésiennes sont basées sur le jugement d'experts. Il faut connaître une *prior* (i.e., avoir une connaissance *a priori*) et de connaître la probabilité *a postériori* (i.e., une *posterior*) (Huelsenbeck et al., 2001). La formule générale du théorème bayésien est donnée par les équations 1.8 et 1.9.

$$P(\Theta|D) = \frac{p(D|\Theta)p(\Theta)}{p(D)} \tag{1.8}$$

$$p(D) = \int p(D|\Theta)p(\Theta)d\Theta \tag{1.9}$$

Où :

$\Theta$ sont les paramètres du modèle,

$D$ sont les données,

$p(\Theta)$ est la distribution des données *a priori*,

$p(D|\Theta)$ est la vraisemblance,

$p(D)$ est la vraisemblance des observations,

$p(\Theta|D)$ est la distribution *a postériori*.

## 1.5 Validation des arbres

La validation des arbres phylogénétiques est basée sur des techniques de rééchantillonnage. Il s'agit le plus fréquemment du bootstrap ou de jackknife qui simulent des échantillons de façon indépendante (Efron, 1979). Cette mesure indiquera le niveau de robustesse de l'arbre à étudier. Si les modifications de rééchantillonnage ont une faible influence, alors cela indiquera que l'arbre estimé est robuste. Dans le cas contraire, l'arbre sera peu robuste, et de ce fait les analyses qui se baseront sur cet arbre seront peu fiables. La valeur de robustesse de l'arbre, généralement valeur du bootstrap, est particulièrement importante, car il serait peu significatif de produire des solutions sans y associer un taux de confiance.

## 1.6 La phylogéographie

La phylogéographie est l'étude des principes et des processus qui gouvernent la distribution des lignées généalogiques, spécialement celle de niveau intraspécifique. La répartition géographique des espèces est souvent corrélée aux motifs associés aux gènes des espèces (Avise, 2000). Ce terme a été introduit pour décrire la corrélation des données géographiques avec les structures génétiques au sein d'un groupe d'espèces. Cette corrélation nous permet de réaliser des liens entre l'aspect génétique des espèces et les différents milieux d'habitats (Knowles et Maddison, 2002).

Lors d'une étude phylogéographique, il faut tenir compte de trois processus majeurs (Nagylaki, 1992) qui sont les suivants :

1. La dérive génétique est le résultat d'erreurs d'échantillonnage des allèles. Ces erreurs sont dues à la transmission générationnelle des allèles et aux barrières géographiques. La dérive génétique est fonction de la taille de la population. En effet, plus la population est grande, plus la dérive génétique est faible. Ceci est dû à la capacité de maintenir la diversité des gènes dans la population initiale. Par convention, on dit qu'un allèle est fixé, si celui-ci atteint la fréquence de 100% et on dit

qu'il est perdu s'il atteint la fréquence de 0%.

2. Le flux génétique ou la migration est un processus important pour la réalisation d'une étude phylogéographique. Il s'agit de transfert d'allèles d'une population à une autre, augmentant la diversité intrapopulation et diminuant la diversité inter-population.

3. Il existe de nombreuses sélections chez toutes les espèces, indiquons les deux plus importantes lors d'une étude phylogéographique.

   (a) La sélection sexuelle est un phénomène résultant d'une caractéristique attractive entre deux espèces. Cette sélection est donc fonction de la taille de la population.

   (b) La sélection naturelle est fonction à la fois de la fertilité, de la mortalité et de l'adaptation d'une espèce vis-à-vis d'un habitat.

* * *

## 1.7 Conclusion

Ce chapitre nous a permis de mieux comprendre la structure d'un arbre phylogénétique (i.e., une phylogénie), en passant à travers les concepts fondamentaux de construction des arbres phylogénétiques. Les arbres phylogénétiques sont des structures de base utilisés par notre algorithme. Une des entrées de notre programme (qui sera décrit dans le prochain chapitre) sera le fichier qui contient des arbres au format Newick (i.e., des structures parenthésées de l'arbre phylogénétique). Nous avons terminé ce chapitre en proposant une ébauche de définition de la phylogéographie, puisque c'est un des points essentiels de ce mémoire. Nous tenterons dans le prochain chapitre, via un algorithme efficace, de trouver les corrélations qui existent entre les paramètres des habitats des espèces et leurs données génétiques.

# CHAPITRE II

# DÉVELOPPEMENT D'UN ALGORITHME POUR LA DÉTECTION DE LA SIMILITUDE ENTRE UN FRAGMENT D'UN GÈNE ET UN ARBRE DE RÉFÉRENCE

## 2.1 Introduction

Ce chapitre introduira un nouvel algorithme original permettant de déterminer les fragments d'un ensemble de gènes ayant une grande similitude d'évolution avec les arbres de références. Avant d'introduire cet algorithme, nous donnerons une description de chaque programme externe que nous avons utilisé pour mener à bien cette tâche. Par la suite, nous exposerons la méthodologie algorithmique que nous avons employée. Enfin, nous parlerons de la complexité de l'algorithme développé.

* * *

## 2.2 Les différents programmes utilisés

### 2.2.1 Paquet PHYLIP

Le paquet PHYLIP de la version 3.69 (Felsenstein, 1980) comprend de nombreux programmes écrits en langage C. Pour ce projet, nous avons utilisé 5 programmes de ce paquet (Seqboot, ProtDist, DnaDist, Neighbor et Consense). Nous avons choisi le paquet PHYLIP pour à la fiabilité éprouvée de ses logiciels qui sont d'ailleurs utilisés par plus de 15 000 utilisateurs à travers le monde (Abdennadher et Boesch, 2007). Nous

indiquerons les différentes caractéristiques de chaque programme, en les présentant dans l'ordre dans lequel ils sont employés dans notre projet.

#### 2.2.1.1 Seqboot

Seqboot est un programme permettant le bootstrapping général des alignements des séquences multiples. Ce programme a été conçu pour permettre à l'utilisateur de générer de multiples ensembles de données qui sont rééchantillonnées en fonction des paramètres d'entrée. La figure 2.1 illustre le menu principal du programme Seqboot avec les paramètres par défaut.

```
Bootstrapping algorithm, version 3.69

Settings for this run:
D     Sequence, Morph, Rest., Gene Freqs?  Molecular sequences
J Bootstrap, Jackknife, Permute, Rewrite?  Bootstrap
%   Regular or altered sampling fraction?  regular
B     Block size for block-bootstrapping?  1 (regular bootstrap)
R                    How many replicates?  100
W              Read weights of characters?  No
C                Read categories of sites?  No
S    Write out data sets or just weights?  Data sets
I             Input sequences interleaved?  Yes
0     Terminal type (IBM PC, ANSI, none)?  ANSI
1     Print out the data at start of run   No
2   Print indications of progress of run   Yes

Y to accept these or type the letter for one to change
```

Figure 2.1: Menu interactif principal du programme Seqboot.

En incorporant le programme Seqboot à notre algorithme, nous avons employé tous les paramètres par défaut. Nos jeux de données étaient composés de séquences moléculaires pour lesquels, nous souhaitions avoir 100 replicats. Chaque ensemble de replicats ne contenait qu'une seule modification. L'incorporation du programme Seqboot à notre programme nécessitait évidemment un fichier d'entrée. Ce fichier contenait l'alignement de séquences multiples (ASM) pour un ensemble d'espèces étudiées codé au format Phylip. Ce programme retourne en sortie un fichier contenant tous les replicats de l'ASM

donné. La complexité du programme Seqboot est fonction du nombre de replicats, du nombre d'espèces et de la taille des ASM.

#### 2.2.1.2 ProtDist et DnaDist

À la suite de l'obtention du ré-échantillonnage de l'alignement, nous réaliserons la construction de différentes matrices de distances. Pour exécuter cette opération, nous utiliserons deux programmes en fonction des types de données.
Le programme ProtDist sera utilisé pour des données de type protéique et le programme DnaDist sera utilisé pour des données de type nucléique.

Les deux figures suivantes représentent les deux menus des programmes ProtDist et DnaDist avec leurs paramètres par défaut.

```
Protein distance algorithm, version 3.69

Settings for this run:
P Use JTT, PMB, PAM, Kimura, categories model?  Jones-Taylor-Thornton matrix
G Gamma distribution of rates among positions?  No
C          One category of substitution rates?  Yes
W                    Use weights for positions?  No
M                  Analyze multiple data sets?  No
I                 Input sequences interleaved?  Yes
0               Terminal type (IBM PC, ANSI)?  ANSI
1          Print out the data at start of run   No
2        Print indications of progress of run   Yes

Are these settings correct? (type Y or the letter for one to change)
```

Figure 2.2: Menu interactif principal du programme ProtDist.

```
Nucleic acid sequence Distance Matrix program, version 3.69

Settings for this run:
D Distance (F84, Kimura, Jukes-Cantor, LogDet)?  F84
G         Gamma distributed rates across sites?  No
T               Transition/transversion ratio?  2.0
C          One category of substitution rates?  Yes
W                       Use weights for sites?  No
F              Use empirical base frequencies?  Yes
L                    Form of distance matrix?  Square
M                  Analyze multiple data sets?  No
I                 Input sequences interleaved?  Yes
0          Terminal type (IBM PC, ANSI, none)?  ANSI
1          Print out the data at start of run   No
2        Print indications of progress of run   Yes

Y to accept these or type the letter for one to change
```

Figure 2.3: Menu interactif principal du programme DnaDist.

Pour les deux programmes, nous avons dû spécifier qu'il s'agissait d'une analyse d'un ensemble de données multiples au nombre de 100 replicats. L'autre paramètre que nous avons dû modifier concerne le modèle d'évolution. Pour les séquences protéiques, nous avons sélectionné le modèle d'évolution Kimura-protéines (voir figure 2.2) et pour les séquences nucléiques, le modèle d'évolution Kimura-2-paramètres (voir figure 2.3). Ces deux modèles d'évolution seront décrits dans la section 2.2.1.3.

#### 2.2.1.3 Modèles d'évolution

Pour les approches basées sur les distances, nous avons choisi le modèle d'évolution Kimura-2-paramètres pour des séquences nucléiques. Le choix de ce modèle est dû au fait que nous n'avions aucune connaissance à priori de la constitution des séquences nucléiques de nos jeux de données. En effet, le modèle Kimura assume que les taux de fréquences des bases nucléiques sont égaux (Kimura, 1980). Ce type de modèle est basé sur trois hypothèses suivantes :

(a) : Tous les sites de l'alignement évoluent indépendamment.

(b) : Il existe deux taux de substitutions : un taux pour les transitions qui concerne

les changements de types $G \leftrightarrow A$, $C \leftrightarrow T$ et $C \leftrightarrow U$, et un taux de transversion, correspondant aux autres changements (voir figure 2.4).

(c) : Le processus de substitution se produit à un taux constant dans le temps.

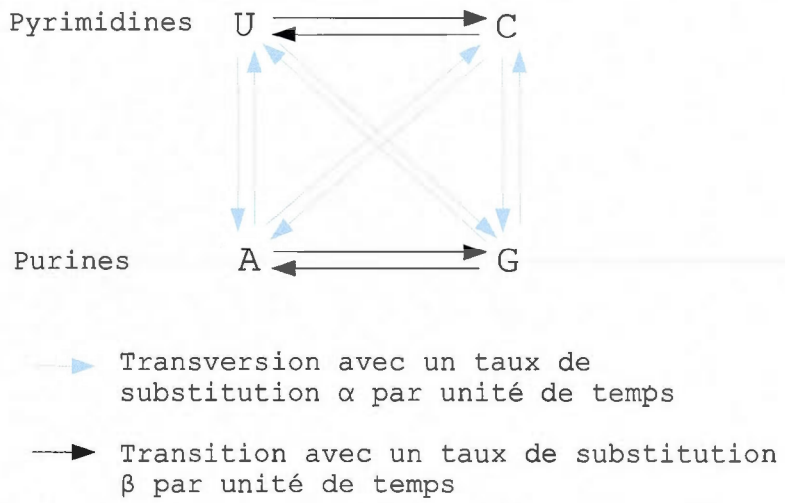


Figure 2.4: Illustration des substitutions de base au cours de l'évolution avec leurs taux par unité de temps selon le modèle Kimura-2-paramètres (Kimura, 1980).

Légende de la figure 2.4 :

G : Guanine — A : Adénine

C : Cytosine — U : Uracile pour des séquences d'ARN [1]

Une fois ces hypothèses établies, il faut quantifier la distance évolutive notée $d$. Cette distance $d$ est fonction de la fraction des différences observées ($p$ : transitions, $q$ : transversions) donnée par la formule 2.1 de Kimura (Kimura, 1980).

$$d = -\frac{1}{2} ln \left[ (1 - 2p - q)\sqrt{1 - 2q} \right] \tag{2.1}$$

1. T : Thymine pour des séquences d'ADN

Pour des séquences de types protéiques, nous avons sélectionné le modèle d'évolution Kimura-protéines, qui a également trois hypothèses de base :

(a) : Tous les sites évoluent indépendamment selon le même processus.

(b) : Chaque type de remplacement d'un acide aminé a une probabilité donnée. Les probabilités de remplacement de chaque acide aminé par un autre ont été corrélées selon différentes matrices, qui sont les suivantes : PAM, JTT et WAG (Kosiol, Goldman et Buttimore, 2004; Whelan et Goldman, 2001).

(c) : Le processus de remplacement des acides aminés se produit à taux constant dans le temps.

Le calcul de la distance évolutive $d$ se fait en fonction du nombre total de remplacements compatibles avec l'ensemble des différences observées entre deux séquences et les probabilités de chaque remplacement. Cette approximation empirique de Kimura (Kimura, 1985) se calcule comme suit :

$$d = -ln(1 - p - 0.2p^2) \tag{2.2}$$

Notons que $p$ représente la fraction des différences observées entre les deux séquences.

#### 2.2.1.4 Neighbor

Une fois les différentes matrices de vraisemblance générées, nous utiliserons le programme Neighbor (voir figure 2.5). Ce programme construit des arbres phylogénétiques par agglomérations successives de lignées. La construction de l'arbre phylogénétique se réalise en plusieurs étapes, telle qu'illustrée dans l'algorithme suivant (Saitou et Nei, 1987) :

Étape 1) Pour chaque feuille $i$, calculer la longueur de branche $u_i = \sum_{j:j\neq i}^{n} \frac{D_{ij}}{n-2}$,

Étape 2) Choisir $i$ et $j$ pour lesquels $D_{ij} - u_i - u_j$ est la plus petite,

Étape 3) Fusionner les éléments $i$ et $j$. Calculer les longueurs des branches de $i$ au nouveau nœud ($v_i$) et de $j$ au nouveau nœud ($v_j$) de la manière suivante :

$$v_i = \frac{1}{2}D_{ij} + \frac{1}{2}(u_i - u_j),$$

$$v_j = \frac{1}{2}D_{ij} + \frac{1}{2}(u_j - u_i),$$

Étape 4) Calculer toutes les distances entre le nouveau nœud ($ij$) et les feuilles restantes en utilisant la formule $D_{(ij),k} = \frac{D_{ik}+(D_{jk}-D_{ij})}{2}$,

Étape 5) Supprimer les feuilles $i$ et $j$ de la matrice de distances et les remplacer par le nouveau nœud ($ij$) qui sera considéré comme une feuille,

Étape 6) Si le nombre de nœuds restants est supérieur à deux, recommencer depuis l'étape 1. Sinon, connecter les deux nœuds restants (nœud $l$ et nœud $m$) par une branche de longueur $D_{lm}$.

Ce programme aboutit à l'obtention d'un arbre non enraciné, ce qui explique que l'arbre ainsi obtenu n'assume pas d'horloge d'évolution.

```
Neighbor-Joining/UPGMA method version 3.69

Settings for this run:
N       Neighbor-joining or UPGMA tree?  Neighbor-joining
O                      Outgroup root?  No, use as outgroup species  1
L         Lower-triangular data matrix?  No
R         Upper-triangular data matrix?  No
S                      Subreplicates?  No
J     Randomize input order of species?  No. Use input order
M           Analyze multiple data sets?  No
0   Terminal type (IBM PC, ANSI, none)?  ANSI
1   Print out the data at start of run   No
2 Print indications of progress of run   Yes
3                     Print out tree   Yes
4       Write out trees onto tree file?  Yes


Y to accept these or type the letter for one to change
```

Figure 2.5: Menu interactif principal du programme Neighbor.

La figure 2.5 illustre le menu principal du programme Neighbor avec les différents paramètres par défaut. Tout comme les programmes ProtDist et DnaDist, nous spécifierons qu'il s'agira d'un ensemble de données multiples de 100 replicats.

#### 2.2.1.5 Consense

Enfin, après l'obtention du fichier qui contiendra les 100 arbres phylogénétiques au format Newick, nous le soumettrons au programme Consense afin d'obtenir un arbre consensus (Barrett, Donoghue et Sober, 1991) avec les valeurs de bootstrap sur ses branches.

Il existe de nombreuses techniques permettant l'obtention d'arbres consensus (voir figure 2.6). Dans notre cas, nous avons utilisé la méthode de consensus absolu.

```
Consensus tree program, version 3.69

Settings for this run:
C Consensus type (MRe, strict, MR, Ml):  Majority rule (extended)
O                       Outgroup root:  No, use as outgroup species  1
R         Trees to be treated as Rooted:  No
T    Terminal type (IBM PC, ANSI, none):  ANSI
1          Print out the sets of species:  Yes
2 Print indications of progress of run:  Yes
3                       Print out tree:  Yes
4        Write out trees onto tree file:  Yes

Are these settings correct? (type Y or the letter for one to change)
```

Figure 2.6: Menu interactif principal du programme Consense.

La figure 2.6 visualise le menu principal du programme Consense pour lequel nous avons utilisé les paramètres par défaut indiqués dans le menu.

### 2.2.2 Programme PhyML

Ce programme, conçu par Guindon, S. et Gascuel, O. en 2003, a été écrit en langage C. Il s'agit d'un programme qui implémente la méthode de vraisemblance, le plus fréquemment utilisé de nos jours. Cette méthode repose sur les trois hypothèses (Guindon et Gascuel, 2003) suivantes :

(a) : Le processus de substitution suit un modèle probabiliste pour lequel on ne connaît pas les valeurs numériques, mais ayant des expressions mathématiques qui ont été établies.

(b) : L'évolution des sites se fait de façon indépendante.

(c) : Le taux de substitution peut changer entre différentes branches, mais ne doit pas varier au cours du temps sur la même branche.

### 2.2.3 Programmes pour le calcul de la distance de Robinson et Foulds (RF)

En 1981, Robinson, D.F. et Foulds, L.R. ont proposé une mesure de distance permettant de comparer topologiquement deux structures d'arbres (Robinson et Foulds, 1981). L'idée de ces chercheurs était de définir une distance topologique entre deux arbres phylogénétiques. Cette distance ne tient pas compte des longueurs des branches de l'arbre. La valeur de la distance RF représente le nombre minimum d'opérations élémentaires (i.e., contraction et expansion de nœuds) permettant la transformation d'un arbre phylogénétique en un autre. La figure 2.7 illustre un exemple soulignant différentes étapes permettant de passer de l'arbre $T_1$ à l'arbre $T_2$. Les arbres phylogénétiques $T_1$ et $T_2$ sont des arbres non enracinés. Dans ce cas, cela nécessitera deux opérations élémentaires.

Afin de comparer topologiquement deux arbres phylogénétiques, il faut que les deux arbres, $T_1$ et $T_2$, contiennent les mêmes espèces. Si ce n'est pas le cas, alors une étape de prétraitement sera nécessaire. Cette étape de prétraitement aura pour tâche de ne conserver uniquement les noms des espèces qui sont présentes dans les deux arbres phylogénétiques à la fois.

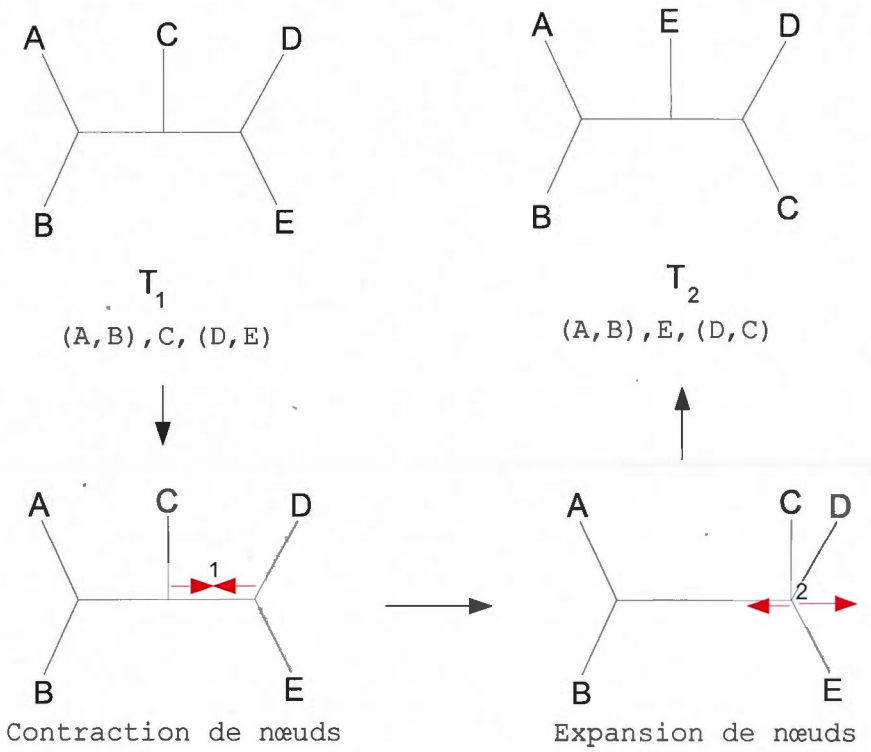


Figure 2.7: Les étapes de transformations de l'arbre phylogénétique $T_1$ en l'arbre phylogénétique $T_2$.

Légende de la figure 2.7 :

Les taxons $A$, $B$, $C$, $D$ et $E$ correspondent aux feuilles. Il peut s'agir d'espèces actuelles ou éteintes.

$((A,B),C,(D,E))$ et $((A,B),E,(D,C))$ correspondent aux structures parenthésées des arbres, qui sont respectivement $T_1$ et $T_2$ (au format Newick).

Nous constatons, en observant la figure 2.7, qu'il a suffit de deux opérations élémentaires pour transformer topologiquement l'arbre $T_1$ en l'arbre $T_2$. Le premier mouvement établit une contraction de la branche $C$ avec l'intersection des branches adjacentes à $D$ et $E$. Le deuxième mouvement établit une expansion de la branche $E$ de l'intersection des branches adjacentes à $C$, $D$ et $E$.

Nous pouvons dire que plus la distance RF est faible, plus cela signifie que les deux arbres comparés sont proches topologiquement. Inversement, plus la distance RF est élevée, plus les deux arbres comparés sont éloignés topologiquement. Comme nous l'avons vu dans l'exemple précédent (voir la figure 2.7), deux opérations élémentaires étaient nécessaires pour confondre topologiquement les arbres $T_1$ et $T_2$. Cette valeur n'est cependant pas révélatrice, car elle ne tient pas compte du nombre d'espèces. Nous proposons donc de la normaliser pour pouvoir comparer les valeurs de cette distance obtenues pour les arbres des tailles différentes. Pour normaliser la distance RF, nous appliquerons la formule suivante :

$$RF_{normalisé} = \frac{RF}{2n-6} * 100\ \%,$$

Où $n$ est le nombre d'espèces identiques dans les deux arbres.

Nous utiliserons la distance RF comme premier critère d'optimisation, le second étant le bootstrap moyen des arbres construits à partir des fragments de gènes analysés (voir l'algorithme 2 et la figure 2.10 pour le flux général de l'algorithme).

## 2.3 Développement de l'algorithme

### 2.3.1 Méthodologie

Nous avons permis une grande flexibilité de notre algorithme, en utilisant deux critères d'optimisation ainsi que de nombreux paramètres qui sont les suivants :

- Il sera permis de modifier la valeur seuil de bootstrap. Ce paramètre sera un des deux critères seuils permettant l'exécution du programme PhyML. Si le bootstrap moyen de l'arbre calculé pour une fenêtre est plus grand ou égal que le bootstrap moyen seuil, alors cette fenêtre aura satisfait un des deux critères d'optimisation permettant

la conservation de cette fenêtre comme fenêtre résultat.

- La distance RF seuil sera le deuxième critère utilisé pour déterminer si le programme PhyML doit être exécuté ou pas. Si la distance RF normalisée entre l'arbre de référence et l'arbre de la fenêtre en cours d'étude est plus petite que la distance RF seuil, alors il s'agira d'une fenêtre potentiellement significative.

- Le nombre de fenêtres de tailles différentes à examiner permet de réaliser des traitements successifs de notre algorithme sur un ensemble de données avec une complexité algorithmique voulue. À ce stade d'avancement du programme, ces traitements se réalisent séquentiellement, mais il serait pertinent par la suite de les réaliser de façon parallèle.

Le premier et le deuxième paramètre mentionnés permettront l'optimisation de notre algorithme, puisque dans un premier temps, les méthodes de reconstruction des arbres phylogénétiques basées sur les distances seront utilisées (e.g., NJ). Il s'agit de méthodes plus rapides que les méthodes basées sur la vraisemblance. Par la suite, nous réaliserons des tests selon les valeurs seuils, c'est-à-dire la validation selon la distance RF normalisée et la validation selon le bootstrap moyen. Si ces deux critères sont respectés, cela indique qu'il s'agit d'une fenêtre pertinente, pour laquelle on pourra raffiner l'analyse en faisant un traitement par PhyML qui est la méthode de reconstruction d'arbres la plus précise, mais aussi la plus lente basée sur la vraisemblance. Cette méthode est très coûteuse en temps, car la durée d'exécution est fonction du nombre de replicats, du nombre d'espèces et de la longueur totale de l'alignement.

### 2.3.2 Algorithme

L'algorithme que nous proposons permet la détection des relations entre un arbre phylogénétique relatif aux espèces et un arbre des différents paramètres environnementaux qui sont en lien avec leur distribution géographique. Cet algorithme aura six étapes ma-

jeures, qui sont les suivantes :

**Première étape** : *Validation des paramètres d'entrée (voir figure 2.8).*

- Nombre d'alignements : $nbTA \geq 1$.
- Noms des fichiers pour chacun des alignements : $file\_TA$. Ces alignements doivent être au format Phylip.
- Type de données des alignements : soit les données de types nucléiques ou soit les données de types protéiques.
- Nombre d'arbres de référence : $nbTree \geq 1$.
- Noms des fichiers pour chacun des arbres de référence : $file\_tree$. Les arbres de référence sont au format Newick.
- Valeur seuil du bootstrap moyen comprise entre 0 et 100%.
- Valeur seuil de la distance RF normalisée comprise entre 0 et 100%.
- Nombre de tailles différentes de fenêtres : $nbTF \geq 1$. L'introduction de la variable $nbTF$ permettra la réalisation des mêmes traitements, mais avec différentes tailles de fenêtres.
- Valeurs des tailles de fenêtres : $TF \geq 1$.
- Pas d'avancement de la fenêtre sur l'alignement de séquences multiples : $P \geq 1$.

Une fois que la validation des différents paramètres d'entrées est réalisée, nous préparons les fichiers contenant les alignements correspondants à la position fixe de différentes fenêtres coulissantes considérées par l'algorithme. Ces fichiers seront les fichiers d'entrées du programme Seqboot du paquet PHYLIP (voir $2^{ième}$ étape).

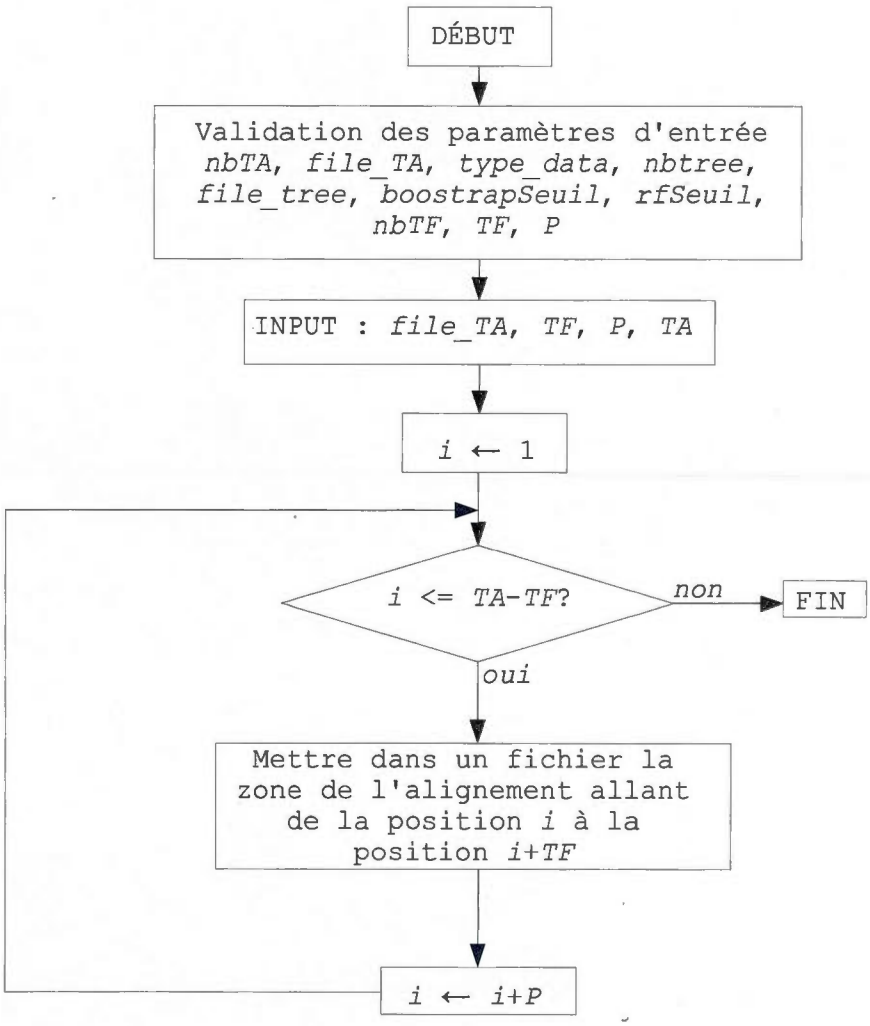


Figure 2.8: Validation des paramètres d'entrée suivie de la préparation et la récupération de la fenêtre de l'alignement étudié.

Légende de la figure 2.8 :

$TA$ : Taille de l'alignement.

$TF$ : Taille de la fenêtre coulissante.

$P$ : Pas d'avancement de la fenêtre coulissante.

$i$ : Compteur de la boucle. Cette variable indique également les positions des fenêtres coulissantes sur l'ASM.

**Deuxième étape** : *Paquet PHYLIP (voir algorithme 1 et figure 2.9).*
Cette étape permet de construire un arbre consensus avec les valeurs de bootstrap sur ces branches. Cet arbre consensus est obtenu à partir d'une fenêtre coulissante de l'ASM débutant à une position fixe. Cette fenêtre coulissante sera avancée le long de l'ASM avec un pas d'avancement P. Pour permettre la construction de l'arbre phylogénétique consensus, il faudra passer en paramètres le type des alignements ainsi que le fichier contenant cet alignement. Le processus est comme suit : pour chaque fichier d'entrée, nous générerons 100 replicats de l'alignement donné (programme Seqboot). Une fois ces alignements obtenus, nous calculerons les matrices de distances suivant un modèle d'évolution choisi. Concernant les jeux de données nucléiques, nous utiliserons le modèle d'évolution Kimura-2-paramètres (programme DnaDist) et pour les jeux de données protéiques, nous utiliserons le modèle d'évolution Kimura-protéines (Kimura, 1985; Kimura, 1980) (programme ProtDist). Ensuite, nous construirons un arbre non enraciné pour chaque matrice de distances précédemment obtenue (programme Neighbor). À partir des 100 arbres non enracinés, nous construirons un arbre consensus qui contiendra les valeurs de bootstrap sur ces branches (programme Consense).

Cette étape a donc pour objectif d'obtenir un arbre consensus, contenant les valeurs de bootstrap sur ces branches, pour les séquences contenues dans une fenêtre prédéterminée. La complexité de cette étape est la somme des complexités des différents programmes du paquet PHYLIP utilisés.

Où :
$O(Seqboot)$ : La complexité du programme Seqboot.
$O(DnaDist/ProtDist)$ : La complexité des programmes DnaDist ou ProtDist, selon le type de données.
$O(Neighbor)$ : La complexité du programme Neighbor.
$O(Consense)$ : La complexité du programme Consense.

Soit :

$$O(PHYLIP) = O(Seqboot) + \underbrace{O(DnaDist/ProtDist)}_{O(D/P)} + O(Neighbor) + O(Consense).$$

Ce qui revient à la complexité maximale des programmes utilisés.

$$O(PHYLIP) = MAX\left(O(Seqboot)\,;\, O(D/P)\,;\, O(Neighbor)\,;\, O(Consense)\right).$$

**Algorithm 1** Execute_Package_PHYLIP

**INPUT** :
*typeData* //caractère indiquant le type de données.
*FileAlignement* //fichier de l'alignement de séquences multiples.

**OUTPUT** :
*FileResultTmp* //fichier de sortie provisoire écrasé à chaque sortie.
*TreeConsense* //fichier de sortie qui contiendra l'arbre consensus au format Newick.

//Réalisation des 100 alignements avec quelques variations aléatoires.
*FileResultTmp* ← Seqboot(*FileAlignement*, *typeData*) ;

//Pour chaque alignement,
//calculer la matrice de distances
//selon le modèle d'évolution choisi.
**if** (*typeData*=="Nucléotides") **then**
    *FileResultTmp* ← DnaDist(*FileResultTmp*) ;
**else**
    *FileResultTmp* ← ProtDist(*FileResultTmp*) ;
**end if**

//Pour chaque matrice de distances,
//construire l'arbre phylogénétique non enraciné.
*FileResultTmp* ← Neighbor(*FileResultTmp*) ;

//Pour l'ensemble des 100 arbres phylogénétiques non enracinés,
//produire un arbre phylogénétique consensus.
*TreeConsense* ← Consense(*FileResultTmp*) ;

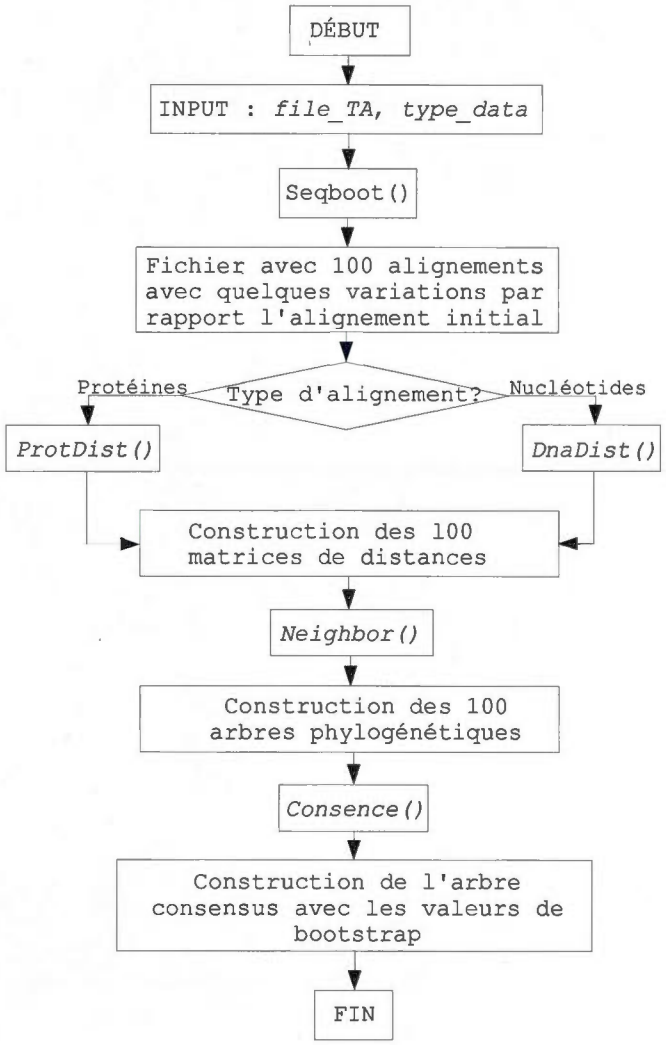


Figure 2.9: Deuxième étape : l'exécution des applications du paquet PHYLIP

Légende de la figure 2.9 :

*file_TA* : Fichier contenant les alignements des séquences multiples au format Phylip.

*type_data* : Type de données à traiter.

- les données de type protéique,
- les données de type nucléique.

La figure 2.9 illustre le processus d'entrelacement des programmes du paquet PHYLIP.

**Troisième étape** : *Validation des résultats selon la valeur moyenne du bootstrap de l'arbre (voir l'algorithme 3 et la figure 2.10).*

Cette validation aura pour entrée le fichier de l'arbre consensus issu de la deuxième étape. Nous conserverons les informations relatives aux arbres qui ont une valeur moyenne du bootstrap supérieure ou égale au bootstrap seuil (i.e., la valeur seuil du bootstrat que l'utilisateur aura auparavant indiqué en paramètre d'entrée). Dans les autres cas, les arbres obtenus ne seront pas retenus comme solution.

**Quatrième étape** : *Validation des résultats selon la valeur de la distance RF (voir l'algorithme 3 et la figure 2.10).*

Durant cette étape, nous réaliserons le test de validation suivant : toutes les informations relatives à la fenêtre de l'alignement étudiée seront conservées, si la distance RF normalisée est inférieure ou égale à la distance RF seuil (i.e., la distance RF seuil que l'utilisateur aura auparavant indiqué en paramètre d'entrée). Les autres seront supprimées.

**Cinquième étape** : *Construction de la matrice de positions contenant les valeurs de la distance RF et la valeur moyenne de bootstrap.*

Cette étape consistera à construire pour chaque arbre de référence une matrice de positions. La matrice de positions contiendra les positions indiquant la plus petite valeur de la distance RF normalisée en fonction de chaque gène et de chaque taille de fenêtre testée. Si par contre, plusieurs fenêtres avec la même taille présentent la même distance RF normalisée, alors on indiquera la position donnant le score moyen de bootstrap le plus grand. Si plusieurs positions, pour une même taille de la fenêtre, donnent la même distance RF normalisée de même que le même score moyen de bootstrap, alors nous retiendrons la première position selon l'ordre d'apparition sur l'alignement. Nous indiquerons sur la matrice de positions, la distance RF normalisée ainsi que le bootstrap moyen de l'arbre considéré.

**Sixième étape** : *L'exécution du programme PhyML (voir algorithme 3 et figure 2.10).* Une fois que les deux conditions seront remplies (i.e., validation du bootstrap moyen et de la distance RF), nous raffinerons la précision de la distance topologique entre les deux arbres en réalisant une construction d'arbre basée sur la vraisemblance. Dans notre cas il s'agira du programme PhyML.

La figure 2.10 et l'algorithme général 2, qui suivent, permettent de visualiser l'imbrication de l'ensemble des différentes étapes de l'algorithme développé. Il sera mentionné par la suite deux fonctions qui sont :

- Fonction_validation_seuil() (voir algorithme 3),
- Fonction_rfMin_bootstrapMax_Position() (voir algorithme 4).

**Algorithm 2** Pseudo-code du programme général

**INPUT** :
*nbAlignements* //nombre d'alignements à traiter.
*tabFileAlignement*[*nbAlignements*] //noms des fichiers des alignements.
*typeData* //variable indiquant le type de données.
*nbTeeRef* //nombre d'arbres de référence.
*tabFileTreeRef*[*nbTreeRef*] //noms des fichiers des arbres de référence.
*bootstrapSeuil* //valeur seuil de bootstrap.
*rfSeuil* //valeur seuil de la distance de Robinson et Foulds.
*nbTF* //nombre de tailles de fenêtres.
*tabTF*[*nbTF*] ; //tableau des différentes tailles de fenêtres.
*P* //pas d'avancement de la fenêtre coulissante.

**OUTPUT** :
FileMatriceBilan //fichier contenant le bilan (voir cinquième étape).

**VARIABLES** :
//Pour chaque treeRef, alignement et TF,
//conservation dans trois matrices les valeurs :
*rfMin*[*nbTF*][*nbAlignements*][*nbTreeRef*] //rf min.
*bootstrapMax*[*nbTF*][*nbAlignements*][*nbTreeRef*] //bootstrap max.
//index sur l'alignement ayant rf min et bootstrap max.
*indexPosition*[*nbTF*][*nbAlignements*][*nbTreeRef*].
*tabFileAlignementFenetre*[*nbAlignements*][*nb*] ;
*nb* étant le nombre de fenêtres sur l'ASM (voir section 2.3.3).

La suite du pseudo-code du programme général.

```
25: Validation_Parameter(nbAlignements, typeData, nbTeeRef, bootstrapSeuil,
    rfSeuil, nbTF, P);
26: //compteurs sur le nombre :
27: i = 0; //de tailles de fenêtres,
28: j = 0; //d'alignements,
29: y = 0; //d'arbres de référence,
30: z = 0; //de fenêtres sur l'ASM.
31: for each tabFileTreeRef[y] in tabFileTreeRef do
32:   for each tabFileAlignement[j] in tabFileAlignement do
33:     for alignementTotal do
34:       Fonction_validation_seuil(typeData, tabFileAlignement[j],
          tabFileTreeRef[y], rfSeuil, bootstrapSeuil);
35:     end for
36:     for each tabTF[i] in tabTF do
37:       tabFileAlignementFentre ← Recuperation_Zone(tabFileAlignement[j]);
38:       for each tabTF[i] in tabTF do
39:         Fonction_validation_seuil(typeData, tabFileAlignementFentre[j][z],
            tabFileTreeRef[y], rfSeuil, bootstrapSeuil);
40:         z++;
41:       end for
42:       z = 0;
43:       i++;
44:     end for
45:     j++;
46:   end for
47:   y++;
48: end for
49: //Imprimer la matrice bilan dans le fichier FileMatriceBilan.
50: FileMatriceBilan ← Impression_matrice(rfMin, bootstrapMax, indexPosition);
```

Description des fonctions utilisées sur l'algorithme 2 :

- Validation_Parameter() : valide les paramètres d'entrée (voir première étape pour plus de détail).

- Fonction_validation_seuil() : réalise la validation des deux conditions seuils (i.e., le bootstrap moyen et la distance RF, voir respectivement la troisième et la quatrième étape pour plus de détail et l'algorithme 3).

- Recuperation_Zone() : stocke les différentes fenêtres coulissantes d'un ASM d'un gène sur des fichiers.

- Impression_matrice() : imprime la matrice de position contenant la distance RF et la valeur moyenne de bootstrap (voir cinquième étape pour plus de détail).

**Algorithm 3** Fonction_validation_seuil

1: **INPUT** :
2: *typeData* //variable indiquant le type de données.
3: *treeGene* //fichier contenant l'arbre correspondant à une fenêtre du gène.
4: *treeRef* //arbre de référence.
5: $rfSeuil$ //valeur seuil de la distance RF normalisée.
6: *bootstrapSeuil* //valeur seuil du bootstrap moyen.
7:
8: Execut_Package_PHYLIP(*typeData*) ;
9: $bootstrapMoyen$ = Bootstrap_Moyen_Tree(*treeGene*) ;
10: $rf$ = RF_tree1_to_tree2(treeRef, treeGene) ;
11: Fonction_rfMin_bootstrapMax_Position($rf$, *bootsrap*) ;
12: **if** ($rf < rfSeuil$ && $bootstrapMoyenTree > bootstrapSeuil$) **then**
13: Exécuter PhyML(*tabFileAlignement*) ;
14: conserver l'alignement ;
15: conserver l'arbre ;
16: **else**
17: supprimer l'alignement ;
18: supprimer l'arbre ;
19: **end if**

Description des fonctions utilisées sur l'algorithme 3 :

- Execute_Package_PHYLIP() est un script réalisant l'exécution des applications du paquet PHYLIP (voir algorithme 1).

- Bootstrap_Moyen_Tree() est une fonction qui prend en paramètre un fichier contenant un arbre au format Newick avec les valeurs de bootstrap sur ses branches et retourne en sortie le bootstrap moyen de l'arbre.

- RF_tree1_to_tree2() est une fonction qui prend en paramètres deux fichiers d'arbres au format Newick et retourne la distance RF normalisée entre ces arbres.

- PhyML() est une fonction faisant appel au programme PhyML, prenant en entrée un fichier d'ASM et retourne un arbre consensus.

- Pour chaque arbre de référence, Fonction_rfMin_bootstrapMax_Position() retourne la plus petite distance RF d'un gène pour une même taille de fenêtre. Dans le cas où plusieurs fenêtres ont la même valeur de la distance RF, la sélection se poursuivra sur le bootstrap moyen qui doit être le plus élevé (voir algorithme 4).

**Algorithm 4** Fonction_rfMin_bootstrapMax_Position

1: **if** $(rfMin[i][j][y] < rf)$ **then**
2: $rfMin[i][j][y] \leftarrow rf$;
3: $bootstrapMax[i][j][y] \leftarrow bootstrap$;
4: $indexPosition[i][j][y] \leftarrow y * P$;
5: **else if** $(rfMin[i][j][y] == rf$ && $bootstrapMax[i][j][y] < bootstrap)$ **then**
6: $rfMin[i][j][y] \leftarrow rf$;
7: $bootstrapMax[i][j][y] \leftarrow bootstrap$;
8: $indexPosition[i][j][y] \leftarrow y * P$;
9: **end if**

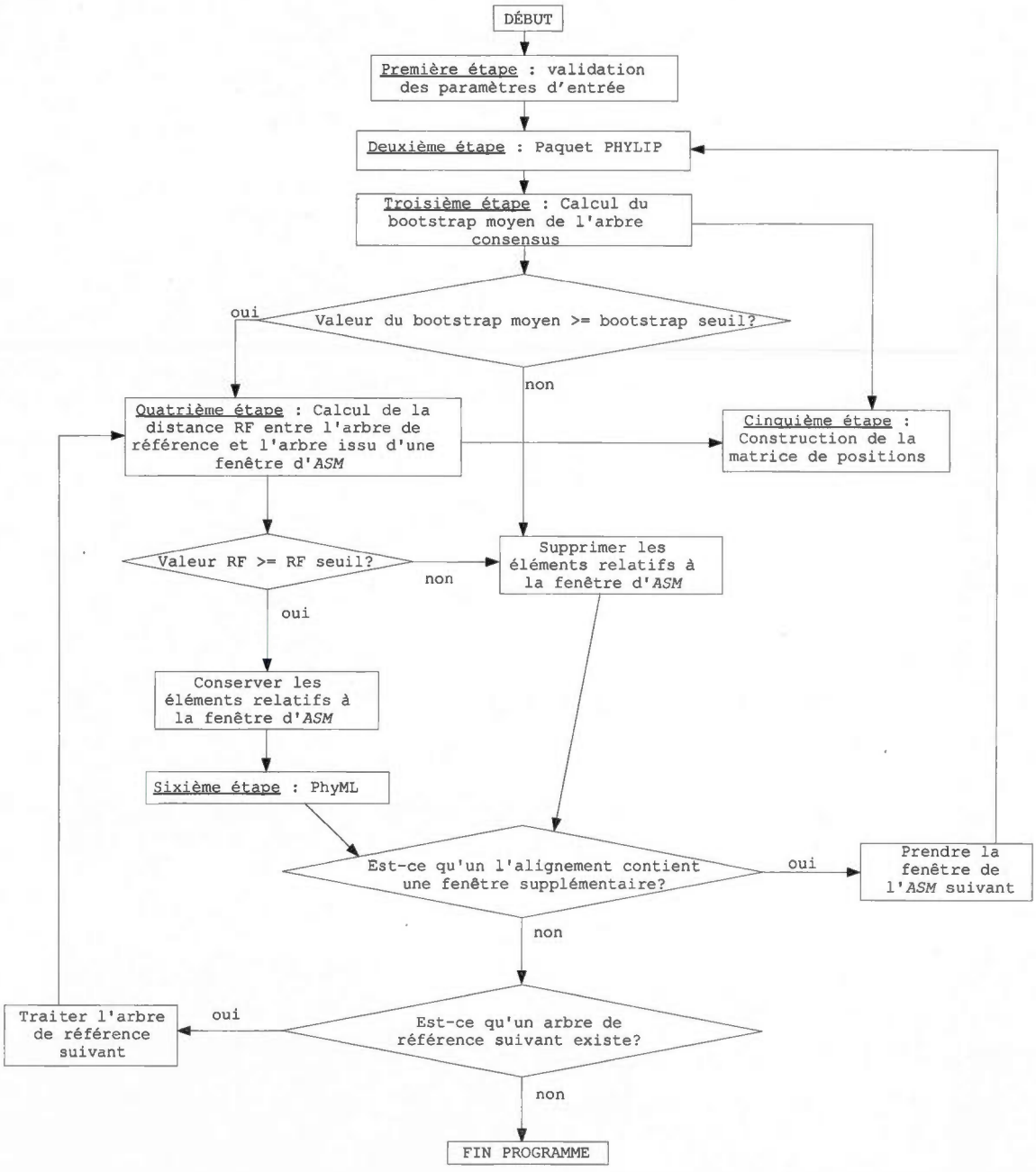


Figure 2.10: Le flux complet de l'algorithme général.

### 2.3.3 Complexité algorithmique

Pour mieux nous repérer dans les différentes notations de la complexité algorithmique, nous les avons illustrées dans le tableau 2.1 suivant. La complexité algorithmique est définie, entre autres, par le temps d'exécution ainsi que l'espace mémoire consommé par l'algorithme. Il est fréquent d'utiliser la notation "grand $O$" servant à majorer la borne supérieure asymptotiquement d'une fonction à un facteur constant près (Cormen et al., 2001) (voir tableau 2.1). La variable $n$ symbolise la taille des données.

Tableau 2.1: Notations principales sur la complexité algorithmique (Cormen et al., 2001).

| Notation | Type de complexité |
|---|---|
| $O(1)$ | Constante |
| $O(log(n))$ | Logarithmique |
| $O(\sqrt{n})$ | Racinaire |
| $O(n)$ | Linéaire |
| $O(n^2)$ | Quadratique |
| $O(n^3)$ | Cubique |
| $O(n!)$ | Factorielle |

Nous constatons que la complexité de l'algorithme décrit dans la section précédente dépend à la fois des complexités des différents programmes externes (voir tableau 2.2) ainsi que du nombre de fenêtres que l'alignement peut contenir.

Tableau 2.2: Complexités algorithmiques des différents programmes externes.

| Programmes | Complexités | Références |
|---|---|---|
| Seqboot - $O(r.n.TA)$<br>ProtDist \ DnaDist - $O(n^2)$<br>Neighbor - $O(n^3)$<br>Consense - $O(r.n^3)$ | Paquet PHYLIP<br>$O(PHYLIP) = O(r.n^3 + r.n.TA)$ | (Felsenstein, 1993) |
| PhyML | $O(PhyML) = O(e.n.TA)$ | (Guindon et Gascuel, 2003). |
| Calcul de la distance RF | $O(n^2)$ | (Makarenkov et Leclerc, 2000). |

Légende du tableau 2.2 :

- $n$ : nombre d'espèces (ou taxa),
- $r$ : nombre de réplicats,
- $TA$ : taille de l'alignement de séquences multiples,
- $e$ : nombre d'étapes de raffinages réalisées par l'algorithme PhyML.

Estimons maintenant le nombre de fenêtres que l'alignement donné peut contenir.

- Soit $TA$ la taille de l'alignement de séquences multiples (ASM).
- Soit $TF$ la taille de la fenêtre.
- Soit $P$ le pas d'avancement de la fenêtre coulissante. Sur la figure 2.11, la flèche bleue indique le sens de ce déplacement.
- Soit $nb$ le nombre de fenêtres.

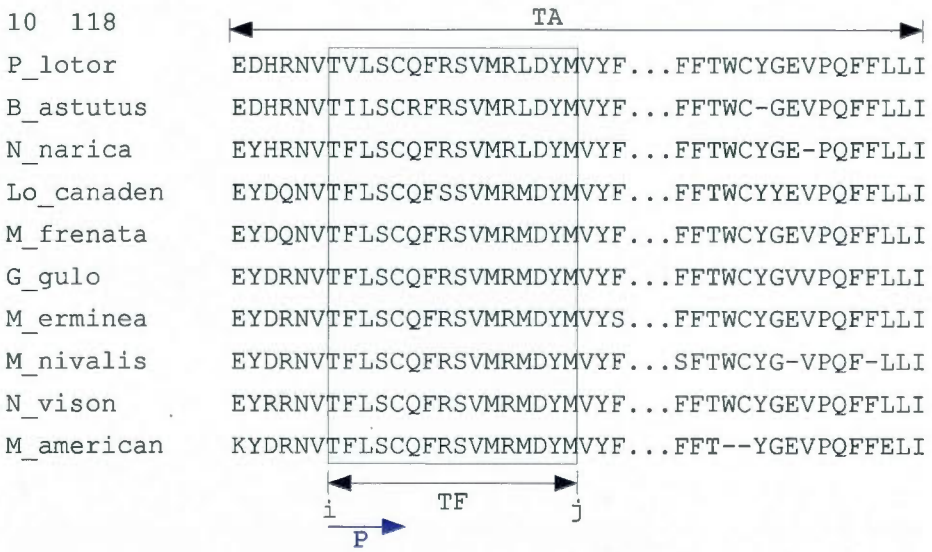


Figure 2.11: Glissement de la fenêtre coulissante sur un ASM.

**Proposition 1.**

Pour tout $TA \in N^*$ et pour tout $TF$ et $P \in N$ alors $nb = \lfloor \frac{TA-TF}{P} + 1 \rfloor$.

***Démonstration*** :

La position de la fin de toutes les fenêtres coulissantes juxtaposées vaut $TF+(nb-1)*P$, ce qui doit être égal au maximum à la taille de l'alignement $TA$ et doit être supérieur à la taille de l'alignement moins le pas d'avancement $P$, ce qui nous permet d'écrire la première inégalité.

$$\begin{aligned}
TA - P < TF + (nb-1) * P \leq TA, \\
-P < (TF - TA) + (nb-1) * P \leq 0, \\
-1 < \frac{TF-TA}{P} + nb - 1 \leq 0, \\
1 > \frac{TA-TF}{P} - (nb-1) \geq 0, \\
0 \leq \frac{TA-TF}{P} - (nb-1) < 1, \\
nb - 1 \leq \frac{TA-TF}{P} < (nb-1) + 1,
\end{aligned}$$

Par conséquent, cela nous permet d'encadrer la valeur $nb-1$ comme suit :

$$\frac{TA-TF}{P} - 1 < \underbrace{nb-1}_{entier} \leq \frac{TA-TF}{P},$$

Comme la valeur de $nb-1$ est un entier, alors on peut en déduire l'égalité suivante :

$$\begin{aligned}
nb - 1 = \left\lfloor \frac{TA-TF}{P} \right\rfloor, \ et\ donc \\
nb = \left\lfloor \frac{TA-TF}{P} + 1 \right\rfloor . \square
\end{aligned}$$

Par conséquent, pour tout $TA \in N^*$ et pour tout $TF$ et $P \in N$ alors $nb = \lfloor \frac{TA-TF}{P} + 1 \rfloor$ est vrai.

En combinant les différentes complexités de chaque programme utilisé et en connaissant la formule pour le nombre des fenêtres, nous pouvons estimer la complexité générale de notre algorithme, qui est donnée par la formule 2.3 :

$$O\Big(r * \Big(\Big\lfloor \frac{TA - TF}{P} + 1 \Big\rfloor + 1\Big) * \Big(O(rn^3 + rnTA) + O(enTA) + O(n^2)\Big)\Big), \qquad (2.3)$$

où $TF$ est la taille de la fenêtre coulissante, $P$ est le pas de progression de la fenêtre coulissante, $TA$ est la taille de l'alignement des séquences multiples, $O(enTA)$ correspond à la complexité de la méthode d'inférence d'arbres phylogénétiques de PhyML utilisée pour inférer les phylogénies à partir d'une fenêtre coulissante issue de l'ASM dont $e$ indique le nombre d'étapes de raffinages réalisées par l'algorithme PhyML, $O(rn^3 + rnTA)$ correspond à la complexité totale des différents programmes du paquet PHYLIP utilisés (Seqboot, ProtDist ou DnaDist, Neigbhor, Consense), $r$ est le nombre de réplicats et $n$ est le nombre d'espèces.

* * *

## 2.4 Conclusion

Ce chapitre nous a permis d'exposer l'algorithme qui a été développé, dans le cadre de ma maîtrise en Informatique à l'UQÀM. Nous trouverons ici l'estimation de la complexité algorithmique, ainsi que la méthodologie que nous avons employée. Il y a plusieurs points pouvant être apportés pour améliorer l'algorithme (voir le chapitre "Conclusion et perspectives"). Notons qu'un travail important a été nécessaire à faire en amont de l'utilisation de notre algorithme, consistant en la collecte du jeu de données (voir le chapitre III) et la construction des arbres de référence. Dans le prochain chapitre, nous parlerons du jeu de données réelles qui sera à la base de nos tests dont les résultats seront indiqués et analysés dans le chapitre IV.

# CHAPITRE III

# LES DONNÉES

## 3.1 Introduction

Ce chapitre introduira les données que nous avons choisies pour tester notre algorithme (voir le chapitre II). Il s'agit de la liste des espèces sélectionnées appartenant au groupe des Carnivores. La liste des gènes, que nous avons récupérés depuis la base de données GenBank de NCBI pour cet ensemble d'espèces, la liste des arbres de référence et, enfin, la liste des localisations géographiques pour l'arbre de référence $T_2$.

* * *

## 3.2 Jeux de données

### 3.2.1 La liste des espèces

Notre étude sera basée sur l'ensemble des espèces appartenant au groupe des Carnivores, se localisant essentiellement en Amérique du Nord "voir la section 3.2.4". Le groupe des Carnivores appartient à l'embranchement des mammifères (Bininda-Emonds, Gittleman et Purvis, 1999). Il se caractérise essentiellement par "de fortes canines (crocs) et de molaires tranchantes (carnassières)" (Reece et al., 2011). Cette spécificité des Carnivores se traduit essentiellement par leurs modes alimentaires variés. De plus, elle confère à ce groupe une remarquable adaptation au milieu (Van-Valkenburgh, 2007). Les Carnivores possèdent également de systèmes locomoteurs leur permettant d'explorer différents mi-

lieux (Yu et al., 2011). C'est pour toutes ces raisons que le choix de ce groupe s'avère intéressant pour notre étude. Notons que notre liste des espèces est similaire à celle utilisée par Ferguson et al. (1996) et à celle de Garland et al. (1993), ce qui nous permettra d'utiliser leurs arbres phylogénétiques comme arbres de références.

Le tableau 3.1 illustre la liste des 52 espèces qui seront considérées dans notre étude.

Tableau 3.1: Liste des 52 espèces du groupe des Carnivores considérées.

| Nom de l'espèce | Nombre d'espèces de la même famille | Nom de la famille du groupe des Carnivores |
|---|---|---|
| *Ursus maritimus*<br>*Ursus arctos*<br>*Ursus americanus* | 3 | *Ursidae* |
| *Odobenus rosmarus* | 1 | *Odobenidae* |
| *Phoca groenlandica*<br>*Phoca fasciata*<br>*Phoca largha*<br>*Phoca vitulina*<br>*Phoca hispida = Pusa hispida*<br>*Halichoerus grypus*<br>*Cystophora cristata*<br>*Mirounga angustirostris*<br>*Erignathus barbatus* | 9 | *Phocidae* |
| *Callorhinus ursinus*<br>*Eumetopias jubatus*<br>*Zalophus californianus*<br>*Arctocephalus townsendi* | 4 | *Ortaiidae* |
| *Bassariscus astutus*<br>*Nasua narica*<br>*Procyon lotor* | 3 | *Procyonidae* |

Suite du tableau à la page suivante.

Tableau 3.1: Liste des 52 espèces du groupe des Carnivores considérées (suite).

| Nom de l'espèce | Nombre d'espèces de la même famille | Nom des familles du groupe des Carnivores |
|---|---|---|
| *Martes americana* | | |
| *Martes pennanti* | | |
| *Mustela nivalis* | | |
| *Mustela erminea* | | |
| *Mustela frenata* | | |
| *Mustela vison* | | |
| *Mustela nigripes* | | |
| *Lontra canadensis = lutra canadensis* | 16 | *Mustelidae* |
| *Enhydra lutris* | | |
| *Gulo gulo* | | |
| *Taxidea taxus* | | |
| *Mephitis mephitis* | | |
| *Mephis macroura* | | |
| *Spilogale putorius* | | |
| *Spilogale pygmaea = Spilogale gracilis* | | |
| *Conepatus mesoleucus = Conepatus leuconotus* | | |
| *Canis lupus* | | |
| *Canis rufus* | | |
| *Canis latrans* | | |
| *Urocyon cinereorgenteus* | | |
| *Urocyon littoralis* | 9 | *Canidae* |
| *Vulpes vulpes* | | |
| *Alopex lagopus* | | |
| *Vulpes macrotis* | | |
| *Vulpes velox* | | |
| *Lynx canadensis* | | |
| *Lynx rufus* | 7 | *Felidae* |
| *Panthera onca* | | |

Suite du tableau à la page suivante.

Tableau 3.1: Liste des 52 espèces du groupe des Carnivores considérées (suite).

| Nom de l'espèce | Nombre d'espèces de la même famille | Nom des familles du groupe des Carnivores |
|---|---|---|
| *Puma yaguarondi = Herpailurus yaguarondi* | | |
| *Puma concolor* | | |
| *Leopardus pardalis* | | *Felidae* (suite) |
| *Leopardus weidii* | | |

À titre de référence, nous présenterons ici l'arbre de Garland et al. (1993), voir figure 3.1. Cet arbre illustre le regroupement des espèces du groupe des Carnivores en fonction des familles et des ordres. Actuellement, il s'agit du meilleur classement des espèces du groupe des Carnivores (Garland et al., 1993; Ferguson, Virgl et Lariviere, 1996). C'est la raison pour laquelle nous l'utiliserons comme point de référence.

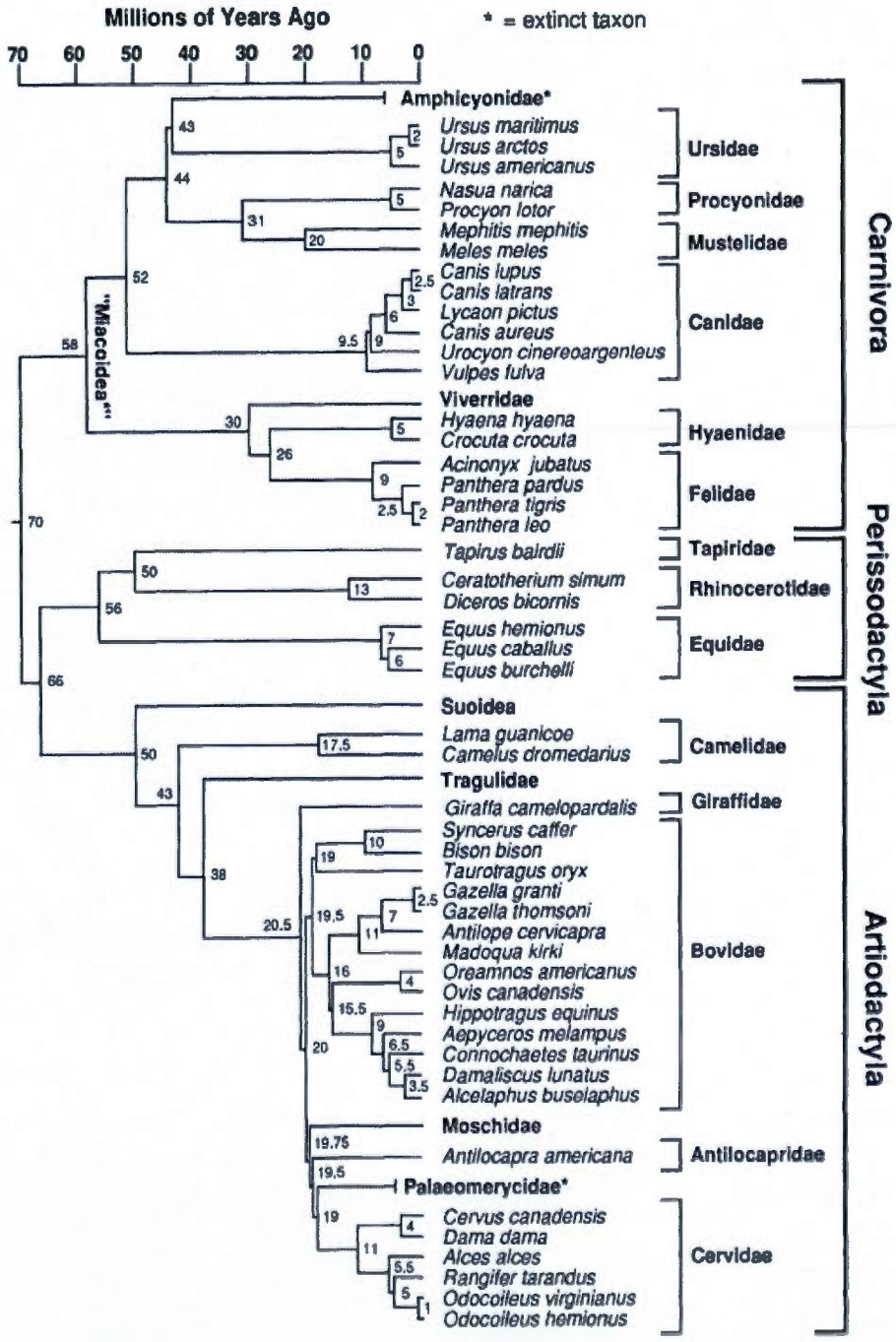


Figure 3.1: Cladogramme du meilleur compromis actuel des Carnivores en Amérique du Nord (Garland et al., 1993).

### 3.2.2 La liste des séquences génétiques

#### 3.2.2.1 Obtention des protéines

Une fois le choix des espèces établi, nous pouvons récupérer leurs différents gènes en utilisant la base de données GenBank de NCBI. Il s'agit de la plus grande base de données génétique (BD) publique (voir : `http://www.ncbi.nlm.nih.gov/genbank/`). Cependant, nous n'avons pas utilisé toutes les informations disponibles dans GenBank à cause de la redondance d'information. Pour une même espèce, GenBank peut avoir une série de séquences du même gène. Chaque séquence a été soumise par une équipe de recherche spécifique. De plus, afin d'éviter de réaliser ce travail manuellement, nous avons opté pour le développement d'un script. Ce script avait pour objectif de recenser le maximum de gènes pour un ensemble d'espèces et aussi de conserver les informations de façon unique (script permettant la construction de la matrice présence/absence voir Appendice B). Pour ce faire, nous avons conservé pour un même gène de la même espèce, les informations de la séquence ayant la plus grande longueur.

Nous avons développé un script PERL nous permettant de stocker toutes les données dans une BD locale. Le système de gestion de bases de données MySQL de version 5.1.63 a été employé. L'utilisation de cette BD se faisait à travers la librairie DBI. Nous avons également utilisé BioPerl pour récupérer les données de GenBank. Notre base de données contenait les informations relatives aux espèces, c'est-à-dire le nom de l'espèce, le nom de la séquence, la séquence, le numéro d'accession de GenBank et autres informations pertinentes. Puis, nous avons sélectionné uniquement les gènes qui sont présents pour au moins la moitié de la liste des espèces, soit 26 espèces (dans notre exemple ; voir le tableau 3.2).

#### 3.2.2.2 La liste des protéines

Dans cette étude, nous avons considéré les données protéiques. On trouve, dans le tableau 3.2, une liste de 21 protéines sélectionnées depuis la base de données GenBank de NCBI.

Tableau 3.2: Liste des 21 protéines sélectionnées pour notre étude.

| |
|---|
| Adenosine A3 receptor |
| Apolipoprotein B |
| ATP synthase F0 subunit 6 |
| ATP synthase F0 subunit 8 |
| Brain derived neurotrophic factor |
| Breast cancer susceptibility protein 1 |
| Cytochrome Oxidase Subunit I |
| Growth hormone receptor |
| NADH dehydrogenase subunit 1 |
| NADH dehydrogenase subunit 2 |
| NADH dehydrogenase subunit 4 |
| NADH dehydrogenase subunit 4L |
| NADH dehydrogenase subunit 5 |
| NADH dehydrogenase subunit 6 |
| Nicotinic cholinergic receptor alpha polypeptide 1 precursor |
| Prepronociceptin |
| Recombination activating protein 1 |
| Retinoid Binding Protein |
| Rhodopsin |
| Sex determining region Y protein |
| Von Willebrand factor |

La liste de ces protéines s'avère pertinente pour notre étude, plus particulièrement d'un point de vue de leurs fonctionnalités dont les descriptions sont décrites dans la section 3.2.2.3.

#### 3.2.2.3 Description des protéines

- Le récepteur de l'adénosine A3 est couplé aux récepteurs des protéines $G(G_i/G_q)$. Ces protéines sont impliquées dans une variété de voies de signalisations intracellulaires (Sajjadi et al., 1996) et dans diverses fonctions physiologiques (Baraldi et al., 2000). Par exemple, il y transmet une fonction soutenue de cardioprotecteur au cours de l'ischémie cardiaque (Liu et al., 1994; Tracey et al., 1997) . Ce gène intervient dans l'inhibition de la dégradation des neutrophiles (Ezeamuzie et Phi-

lips, 1999) au cours des lésions tissulaires neutrophiles dépendantes. De plus, il participe à la fois dans les effets neuroprotecteurs et les maladies neurodégénératives. Il peut également servir de médiateur, à la fois à la prolifération cellulaire et à la mort cellulaire (Fishman et al., 2001; Walker et al., 1997).

- L'apolipoprotéine B (APOB ou même ApoB) est une apolipoprotéine primaire de chylomicrons et de lipoprotéines de basses densités (LDL). Elle est responsable du passage du cholestérol aux tissus. Grâce à un mécanisme qui n'est pas entièrement connu, des niveaux élevés d'ApoB peuvent conduire à des plaques causant des maladies vasculaires (athérosclérose) et à une maladie cardiaque (Jiang et al., 2001). Il a été démontré que le niveau d'ApoB est un meilleur indicateur de risque de maladie cardiaque que le cholestérol total ou LDL (Ingelsson et al., 2007; Pischon et al., 2005; Walldius et Jungner, 2004). Cependant, le taux de cholestérol demeure la mesure des lipides primaires pour le facteur de risque de l'athérosclérose (Teng, Burant et Davidson, 1993).

Les deux protéines qui suivent (ATP-6 et ATP-8) sont deux sous-unités du complexe $F_0$ transmembranaire de type F de l'adénosine triphosphate (ATP). Les éléments de la mitochondrie vont s'avérer très pertinents pour l'étude phylogénétique, puisqu'au cours de l'évolution, les mitochondries sont provenues de l'endosymbiose d'une $\alpha$-protéobactérie il y a environ 2 milliards d'années (Yang et al., 1985). Ceci explique que la mitochondrie possède des chloroplastes ainsi que son propre ADN. De plus, une des fonctions principales des mitochondries est la production d'énergie pour les cellules. Il s'agit d'un rôle indispensable au bon fonctionnement des organismes.

- L'ATP synthase F0 subunit 6 (ATP-6) est un élément clé de la chaîne à protons. Il peut jouer un rôle direct dans la translocation de protons à travers la membrane (voir la figure 3.2 illustrant l'implication du complexe d'ATP synthase durant la phosphorilation oxidative).

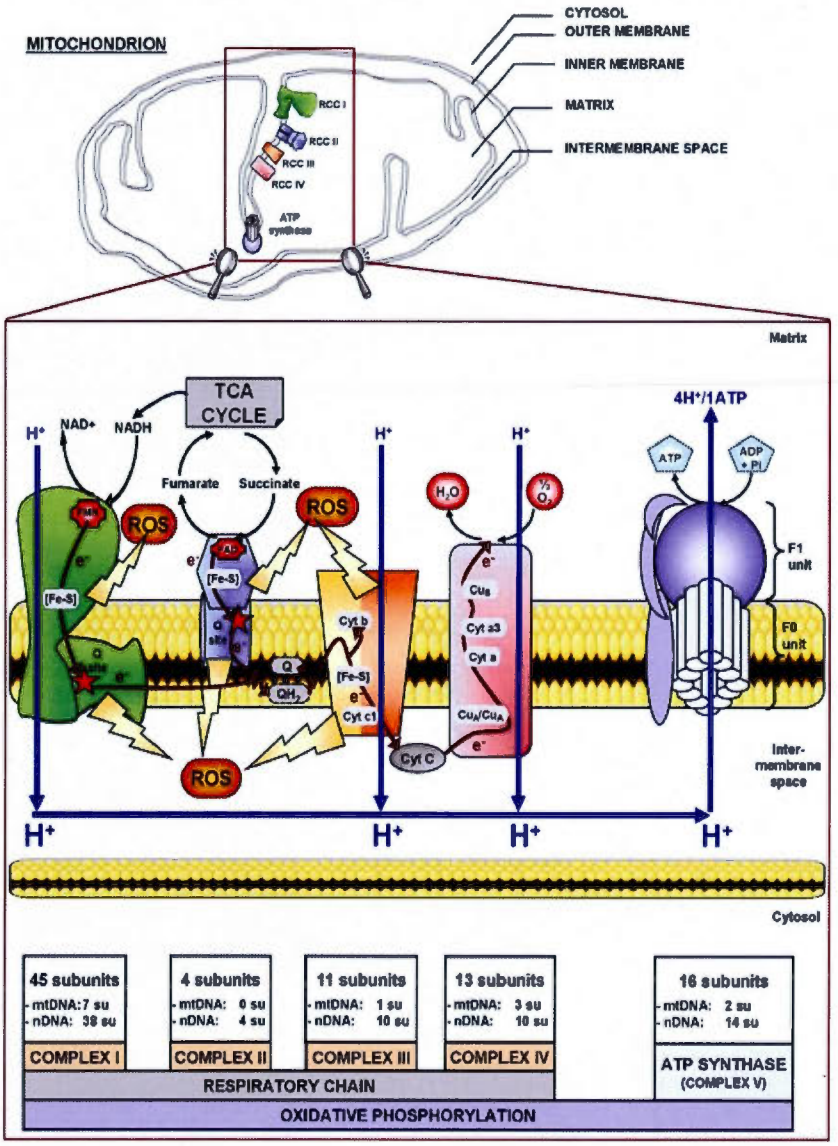


Figure 3.2: La phosphorilation oxidative (Lemarie et Grimm, 2011).

- L'ATP synthase F0 subunit 8 (ATP-8) semble être une composante intégrale de la tige dans le stator mitochondrial de la levure F-ATPases (Stephens et al., 2003). La tige du stator est ancrée dans la membrane (voir la figure 3.2). Elle agit pour empêcher la rotation inutile des sous-unités de l'ATP synthase (ATPase) par rap-

port au rotor couplé au cours de la synthèse d'ATP/l'hydrolyse.

La chaîne respiratoire, ou plus généralement la chaîne de transport des électrons, est composée d'un ensemble de complexes constitués de protéines membranaires. Elle se localise sur la membrane interne de la mitochondrie, dans les cellules eucaryotes. Ces complexes sont au nombre de cinq, les quatre premières permettent le transport d'électrons et la cinquième, dans la synthèse d'ATP. Le rôle de la chaîne respiratoire est de réoxyder les coenzymes NADH et ubiquinone (CoQ). Cette réaction produit un gradient de protons. Ce gradient de protons permettra de fabriquer de l'énergie, en produisant la molécule d'ATP. Cette étape nécessitera l'enzyme de l'ATP synthase, une protéine membranaire de la mitochondrie. Une telle réaction permettant la production d'ATP est nommée phosphorylation oxydative. Ce mécanisme a été découvert en 1978 (voir la figure 3.2).

- Le facteur du cerveau dérivé neurotrophique (brain derived neurotrophic factor, BDNF) agit sur certains neurones du système nerveux central et le système nerveux périphérique. Il permet la survie des neurones existants, d'encourager la croissance et la différenciation de nouveaux neurones et de synapses (Acheson et al., 1994; Huang et Reichardt, 2001; Leibrock et al., 1989). Dans le cerveau, il est actif dans l'hippocampe, le cortex et le cerveau antérieur. Ces zones sont impliquées dans l'apprentissage, la mémoire et la pensée (Yamada et Nabeshima, 2003; Bekinschtein et al., 2008).

- Breast cancer susceptibility protein 1 (BRCA1) est une protéine susceptible du cancer du sein. Cette protéine est responsable de la réparation de l'ADN (Bhattacharyya et al., 2000; Park et al., 2000). En 1990, le laboratoire UC Berkeley a été le premier laboratoire a avoir mis en évidence cette protéine (Hall et al., 1990). En 1994, le gène a été cloné par l'équipe du laboratoire de Myriad Genetics (Miki et al., 1994). BRCA1 est exprimé dans les cellules de cancer du sein

et d'autres tissus, où elle contribue à la réparation de l'ADN endommagé, ou détruit les cellules si l'ADN ne peut pas être réparé (Friedenson, 2007; Starita et Parvin, 2003). Cependant, si BRCA1 est altéré, l'ADN endommagé n'est pas réparé correctement, ce qui augmente les risques pour les cancers (Wang et al., 2000).

- Le cytochrome oxydase de la sous unité 1 du complexe IV (CO-1) a un rôle dans la chaîne de transport des électrons. Il s'agit d'un grand complexe d'enzymes transmembranaires trouvé dans les bactéries et les mitochondries. Son rôle est de transporter à une molécule d'oxygène les quatre électrons de chacun des quatre cytochromes C (Liu, Fiskum et Schubert, 2002).

- Le récepteur de l'hormone de croissance (en anglais growth hormone receptor ou GHR) est une protéine qui est codée par le gène GRH. Ce gène code pour une protéine qui est un récepteur transmembranaire de l'hormone de croissance. La liaison de l'hormone de croissance au récepteur conduit à la dimérisation du récepteur (Schantl et al., 2003) et l'activation d'une voie de transduction du signal intra et intercellulaire conduisant à la croissance.

Les 6 protéines suivantes (NADH1, NADH2, NADH4, NADH4L, NADH5 et NADH6) appartiennent au complexe de nicotinamide adénine dinucléotide (NADH, NAD). Le NAD est une coenzyme d'oxydoréduction, qui est présente dans toutes les cellules vivantes. Ce coenzyme peut se présenter sous deux formes : NAD+ est un agent d'oxydation et NADH un agent de réduction. Cependant, la fonction principale de NADH est le transfert des électrons. Il s'agit du complexe I intervenant dans la chaîne de transport d'électrons (voir la figure 3.3). Ce complexe se situe dans la membrane interne des mitochondries, dont le rôle est la production d'énergie. Il s'agit de la phosphorilation oxidative (voir figure 3.2). Nous allons uniquement faire une description des protéines NADH1 et NADH2, mettant en avant les conséquences majeures à la suite de mutations

sur ces protéines. Ces différentes mutations seront analysées lors de la construction des arbres phylogénétiques. Nous remarquons également que la deuxième protéine est une protéine fréquemment utilisée pour l'étude phylogénétique (Kocher et al., 1995), due à un taux de mutations élevé.

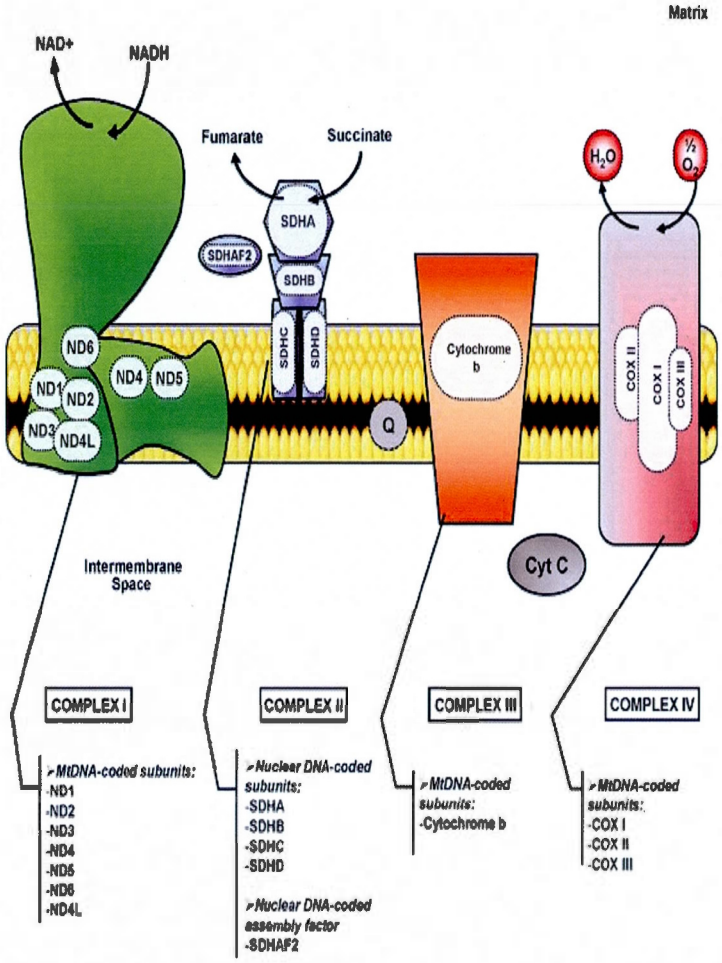


Figure 3.3: Les 4 complexes de la chaîne de transport d'électrons (Lemarie et Grimm, 2011).

- NADH dehydrogenase subunit 1 est impliqué dans la première étape de la chaîne de transport d'électrons lors de la phosphorylation oxydative. Des modifications des composants de transport d'électrons, par des mutations dans l'ADN mito-

chondrial, peuvent compromettre le flux d'électrons. Cela pourrait conduire à une augmentation de la bifurcation et la génération de radicaux superoxydases et augmenter ainsi le stress oxydatif dans les différents types de cellules cancéreuses (Yusnita, Norsiah et Rahman, 2010).

- NADH dehydrogenase subunit 2 est une protéine participant à la chaîne de transport des électrons. La longueur de séquence codant pour cette protéine est de 539 pb [1]. L'évolution du nombre de substitutions observées des nucléotides dans cette séquence est typique des gènes mitochondriaux, montrant un biais de transition (McFadden et al., 2004).

- Le nicotinic cholinergic receptor alpha polypeptide 1 precursor (NC$\alpha$P-1) empêche les dépôts de graisses dans le foie et facilite la circulation des graisses dans les cellules. Cette protéine se trouve dans le foie, les reins et le cerveau. L'étude de Kihara et al. suggère que la simulation des récepteurs cholinergiques peut entraîner la dégénérescence neurale (Kihara et al., 1997).

- La prépronociceptine (PPNOC) est un récepteur de microARN qui est exprimé dans l'épiderme. Elle est prédominante dans le système nerveux central. Ceci explique que cette protéine participe à la modulation de la douleur.

- Recombination activating protein 1 (RAP-1) est une protéine active dans le système immunitaire des vertébrés, permettant la maturation des cellules pré-B et pré-T (Liu et al., 2012). Cette protéine s'avère intéressante pour une étude phylogénétique. En effet, les systèmes immunitaires des êtres vivants doivent produire, et ceci de façon caractéristique, des anticorps pour permettre la défense d'antigènes pathogènes spécifiques. Comme les antigènes sont fonction des milieux d'habitats des espèces, cette protéine s'avère d'une importance primordiale pour

1. paire de bases.

une étude phylogéographique. En considérant ces gènes, pour le même ensemble d'espèces ayant des répartitions géographiques différentes, nous allons chercher les corrélations entre des motifs de ce gène et les paramètres environnementaux.

- Les protéines de rétinoïde de liaisons (RBP) sont une famille de protéines ayant des fonctions diverses. L'acide rétinoïque et le rétinol jouent des rôles cruciaux dans la modulation de l'expression des gènes et le développement global d'un embryon. Toutefois, le déficit ou l'excès de l'une de ces substances peut entraîner une mortalité embryonnaire précoce ou des malformations du développement (Makover et al., 1989).

- La rhodopsine, également connue sous le nom de pourpre rétinien, est un pigment biologique dans les cellules photoréceptrices de la rétine. Elle appartient à la famille des récepteurs aux protéines G couplées. Cette protéine est responsable de la perception de la lumière, c'est pour cela qu'elle est extrêmement sensible à la lumière, permettant la vision en conditions de faible luminosité.

- Sex determining region Y protein (SRY) est la protéine permettant la détermination du sexe sur le chromosome Y (mammifères placentaires et marsupiaux) (Wallis, Waters et Graves, 2008). Des mutations sur cette protéine peut entraîner des femelles XY (syndrome de Swyer) (Iliopoulos et al., 2004), ou bien une translocation d'une partie du chromosome Y contenant la protéine SRY sur le chromosome X, provoquant ainsi le syndrome XX mâle (Biason-Lauber et al., 2009)

- Facteur de Von Willebrand est une glycoprotéine du sang impliqué dans l'hémostase (Ruggeri et Zimmerman, 1987). Il est déficient ou altéré dans la maladie de Willebrand, d'où le nom de la protéine (Vincentelli et al., 2003). Il est impliqué dans un grand nombre d'autres maladies, par exemple le syndrome de Heyde, le syndrome hémolytique et urémique (Sadler, 1998).

### 3.2.3 Les arbres de référence

Pour construire, nos différents arbres de références, nous nous sommes basés sur différentes bases de données qui recensent différents critères. Pour les différents tests, nous avons sélectionné les critères suivants :

- distribution géographique des espèces sélectionnées en Amérique du Nord. Nous avons utilisé deux sources d'information :
  - la première source est une base de données de l'Université McGill ; les données ont été stockées en fonction d'un ensemble de sondes reparties à travers le monde ; dans notre étude, nous nous sommes limités à l'Amérique du Nord. Il s'agira de l'arbre de référence $T_1$.
  - quant à la deuxième source, les données proviennent du site web suivant : `http://www.atlas-mammiferes.fr/`, ce qui donnera l'arbre de référence $T_2$.
- précipitations moyennes,
- températures maximales moyennes,
- températures minimales moyennes,
- températures moyennes,
- altitudes.

Les données des cinq derniers critères provenaient d'une base de données privée de l'Université McGill.

### 3.2.4 La liste des localisations géographiques de l'arbre de référence $T_2$

En ce qui concerne la zone géographique que nous avons considérée, elle comprend le Mexique, les États-Unis et le Canada. Cette zone est intéressante, car elle exploitera la grande variabilité des paramètres, tels que la température moyenne du milieu de vie, le taux d'humidité moyen, la latitude et la longitude moyenne.

Ensuite, il a fallu délimiter des zones. Pour l'arbre de référence $T_1$, nous avons pris 10 000 points correspondant aux 10 000 stations météorologiques de la base de données de l'Université McGill à travers le globe terrestre. Pour l'arbre de référence $T_2$, il avait seulement 20 localisations.

Tableau 3.3: Liste des 20 localisations géographiques choisies pour l'arbre de référence $T_2$.

| Nom du pays | Nom de la localisation |
|---|---|
| CANADA | Yukon Territory |
| | British Columbia |
| | Northweast Territory |
| | Alberta |
| | Saskatchewan |
| | Manitoba |
| | Ontario |
| | Nunavut Territory |
| | Quebec |
| | New Brunswick et Prince Edward Island et Nova Brunswick |
| | Newfoundland et Labrador |
| ÉTATS-UNIS | Western Region |
| | Rocky Mountain Region |
| | Central Region |
| | Gulf Region |
| | Midwest Region |
| | Northeast Region |
| | Southeast Region |
| | Alaska |
| MEXIQUE | Mexique Territory |

Nous avons indiqué ici uniquement les localisations géographiques de l'arbre de référence $T_2$ ayant 20 découpements géographiques (voir le tableau 3.3).

* * *

## 3.3 Conclusion

Dans ce chapitre, nous avons présenté les jeux de données qui seront traités par notre algorithme. Les espèces choisies appartiennent au groupe des Carnivores. Ce groupe recouvre une grande surface du globe terrestre. Nous allons nous intéresser plus particulièrement à l'Amérique du Nord, pour laquelle nous avons sélectionné quelques paramètres climatiques. Enfin, nous avons récupéré un maximum de protéines pour cette liste d'espèces. Dans le prochain chapitre, nous parcourrons et analyserons les résultats issus des tests de notre algorithme sur ce jeu de données.

# CHAPITRE IV

# PRÉSENTATION DES RÉSULTATS

## 4.1 Introduction

À la suite de la récolte des données présentées au chapitre III, nous y avons testé notre algorithme décrit au chapitre II. Ce chapitre parcourra l'ensemble des résultats obtenus. Ensuite, nous tenterons de déterminer les fragments des alignements pertinents pour les différents protéines testées. Par la suite, nous effectuerons un profilage du programme écrit en Java afin de connaître l'évolution de la durée d'exécution du programme sur le même jeu de données, mais en utilisant des tailles de fenêtres différentes. Enfin, nous terminerons ce chapitre par une analyse des différents résultats obtenus.

* * *

## 4.2 Application de l'algorithme sur les données des Carnivores

### 4.2.1 Résultats

Dans cette section, nous exposerons différents tableaux qui indiquent les positions des alignements correspondants à la plus petite distance RF. Dans le cas où plusieurs segments de l'alignement auraient la même valeur de la distance RF, nous choisirons la fenêtre de la protéine qui correspond à l'arbre ayant le bootstrap moyen le plus élevé.

Légende des tableaux de 4.1 à 4.7 :

Nous utiliserons la même légende pour les différents tableaux de résultats.

- Les valeurs dans une zone grise refléteront les meilleurs résultats pour chaque alignement de séquences.

- La première valeur indiquée représente la position de la fenêtre sur l'alignement.

- Les deux valeurs indiquées qui se trouvent entre parenthèses sont respectivement la distance RF normalisée, entre l'arbre de référence et l'arbre de la fenêtre, et le bootstrap moyen de l'arbre de la fenêtre. Tous les arbres de référence ont été inférés par le logiciel T-Rex (Makarenkov, 2001; Boc, Diallo et Makarenkov, 2012).

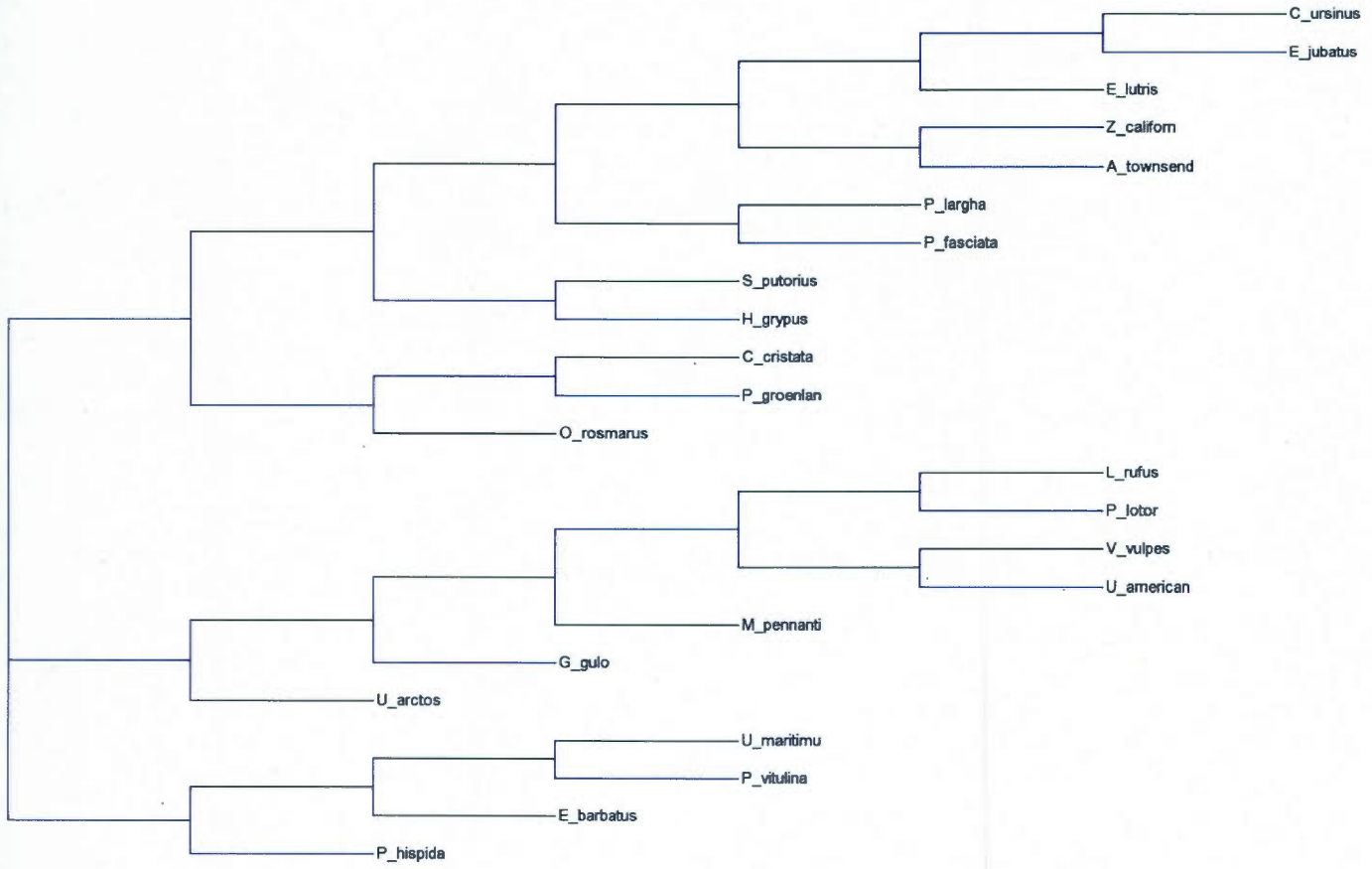


Figure 4.1: L'arbre de distribution géographique $T_1$.

Les données permettant la construction de cet arbre proviennent d'une base de données privée de l'Université McGill. Cet arbre comprend 23 espèces du groupe des Carnivores.

Tableau 4.1: Table des meilleures positions pour l'arbre de distribution géographique $T_1$.

| Noms des protéines | Tailles des fenêtres | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 10 | 20 | 30 | 40 | 50 | 60 | 70 | 80 | 90 | 100 |
| Adenosine A3 receptor | 31 (88,19) | 6 (88,27) | 1 (94,35) | 1 (94,43) | 1 (94,43) | 1 (94,45) | 1 (94,45) | 1 (94,46) | **16 (88,31)** | 1 (94,50) |
| Apolipoprotein B (ApoB) | 81 (89,33) | 86 (84,48) | 81 (84,52) | 86 (84,55) | 216 (84,58) | 221 (84,63) | 71 (84,53) | 221 (84,65) | **21 (78,59)** | 211 (84,72) |
| ATP synthase F0 subunit 6 (ATP-6) | 1 (87,38) | **116 (81,30)** | **98 (81,30)** | 86 (81,29) | **76 (81,30)** | 116 (87,40) | 56 (87,44) | 56 (87,47) | 36 (87,49) | 36 (87,49) |
| ATP synthase F0 subunit 8 (ATP-8) | 31 (88,35) | 41 (88,47) | 21 (88,55) | 11 (88,62) | 11 (94,63) | **1 (88,83)** | - - | - - | - - | - - |
| Brain derived neurotrophic factor (BDNF) | 31 (94,12) | 1 (97,20) | 1 (97,22) | 16 (94,26) | 1 (97,25) | **1 (94,28)** | 6 (94,27) | 11 (97,26) | 11 (97,27) | 11 (94,26) |
| Breast cancer susceptibility protein 1 (BRCA1) | 61 (76,40) | 251 (76,46) | 46 (76,46) | **46 (70,49)** | 46 (70,44) | 226 (76,47) | 216 (76,53) | 16 (70,42) | 206 (76,49) | 176 (76,54) |
| Cytochrome Oxidase Subunit I (CO-1) | 481 (93,47) | 391 (87,40) | 386 (87,40) | 371 (87,38) | 31 (81,24) | 361 (87,37) | 331 (81,32) | **351 (87,44)** | 321 (87,41) | 331 (87,45) |
| Growth hormone receptor (GRH) | 86 (90,23) | 26 (90,25) | 31 (90,30) | 61 (90,28) | 26 (90,31) | 11 (90,34) | 31 (90,36) | 21 (90,35) | 11 (90,36) | **1 (90,39)** |
| NADH dehydrogenase subunit 1 (NADH-1) | 261 (87,45) | 266 (87,35) | 51 (87,52) | 66 (87,58) | **266 (81,42)** | 66 (87,58) | 36 (87,59) | 66 (87,58) | 66 (87,61) | 66 (87,64) |
| NADH dehydrogenase subunit 2 (NADH-2) | 71 (90,40) | 66 (90,52) | 86 (90,62) | 61 (90,62) | 86 (90,58) | 86 (90,60) | 51 (90,67) | 21 (90,66) | **11 (90,73)** | 6 (90,71) |
| NADH dehydrogenase subunit 4 (NADH-4) | 36 (88,47) | 36 (88,53) | 406 (82,43) | 11 (82,54) | 371 (82,43) | 21 (82,59) | 331 (76,41) | **336 (76,47)** | 6 (82,60) | 316 (76,39) |
| NADH dehydrogenase subunit 4L (NADH-4L) | 46 (87,47) | 46 (87,45) | **46 (81,49)** | 16 (81,44) | 31 (81,42) | 16 (87,50) | 1 (87,49) | 11 (87,50) | 1 (87,51) | - - |
| NADH dehydrogenase subunit 5 (NADH-5) | 56 (87,46) | 491 (81,47) | 111 (81,50) | 101 (81,47) | 256 (81,49) | 456 (81,56) | 461 (81,57) | 476 (81,60) | **466 (81,63)** | 436 (81,61) |
| NADH dehydrogenase subunit 6 (NADH-6) | 111 (57,51) | 86 (71,56) | 106 (57,61) | 86 (71,67) | 111 (57,47) | **56 (57,68)** | 61 (71,72) | 36 (71,76) | 11 (57,63) | 16 (71,79) |
| Nicotinic cholinergic receptor alpha polypeptide 1 precursor (NCαP-1) | **1 (100,9)** | - - | - - | - - | - - | - - | - - | - - | - - | - - |
| Prepronociceptin | 41 (93,12) | 46 (93,29) | 46 (93,30) | 21 (90,32) | 26 (90,37) | 11 (93,37) | 1 (90,39) | **6 (90,40)** | - - | - - |
| Recombination activating protein 1 (RAP-1) | 51 (94,30) | 36 (88,34) | **26 (82,43)** | 16 (82,40) | 6 (82,38) | 21 (88,51) | 6 (88,50) | 16 (88,47) | 6 (88,48) | 11 (88,46) |
| Retinoid Binding Protein (RBP) | 336 (85,25) | **341 (80,44)** | 336 (85,45) | **341 (80,44)** | 346 (85,59) | 336 (85,61) | 326 (85,65) | 276 (85,60) | 306 (85,68) | 296 (85,67) |
| Rhodopsin | **1 (98,8)** | 36 (100,16) | 26 (100,13) | 6 (96,6) | 6 (100,14) | - - | - - | - - | - - | - - |
| Sex Determining Region Y Protein (SRY) | 36 (66,47) | 26 (66,61) | 36 (55,76) | 31 (55,72) | 136 (66,80) | 36 (55,73) | 41 (55,78) | 41 (55,78) | **41 (55,79)** | 11 (66,72) |
| Von Willebrand Factor | 1 (71,39) | 31 (64,36) | 36 (71,42) | 116 (71,43) | 26 (71,54) | 1 (64,52) | 116 (71,56) | **1 (64,56)** | 101 (71,64) | 101 (71,63) |

Les protéines SRY et NADH-6 sont significatives par leurs distances RF de 55 et 57, leurs valeurs de bootstrap moyen de 79 et 68, leurs tailles de fenêtres de 90 et 60 et leurs positions des fenêtres sur l'ASM de 41 et 56 respectivement.

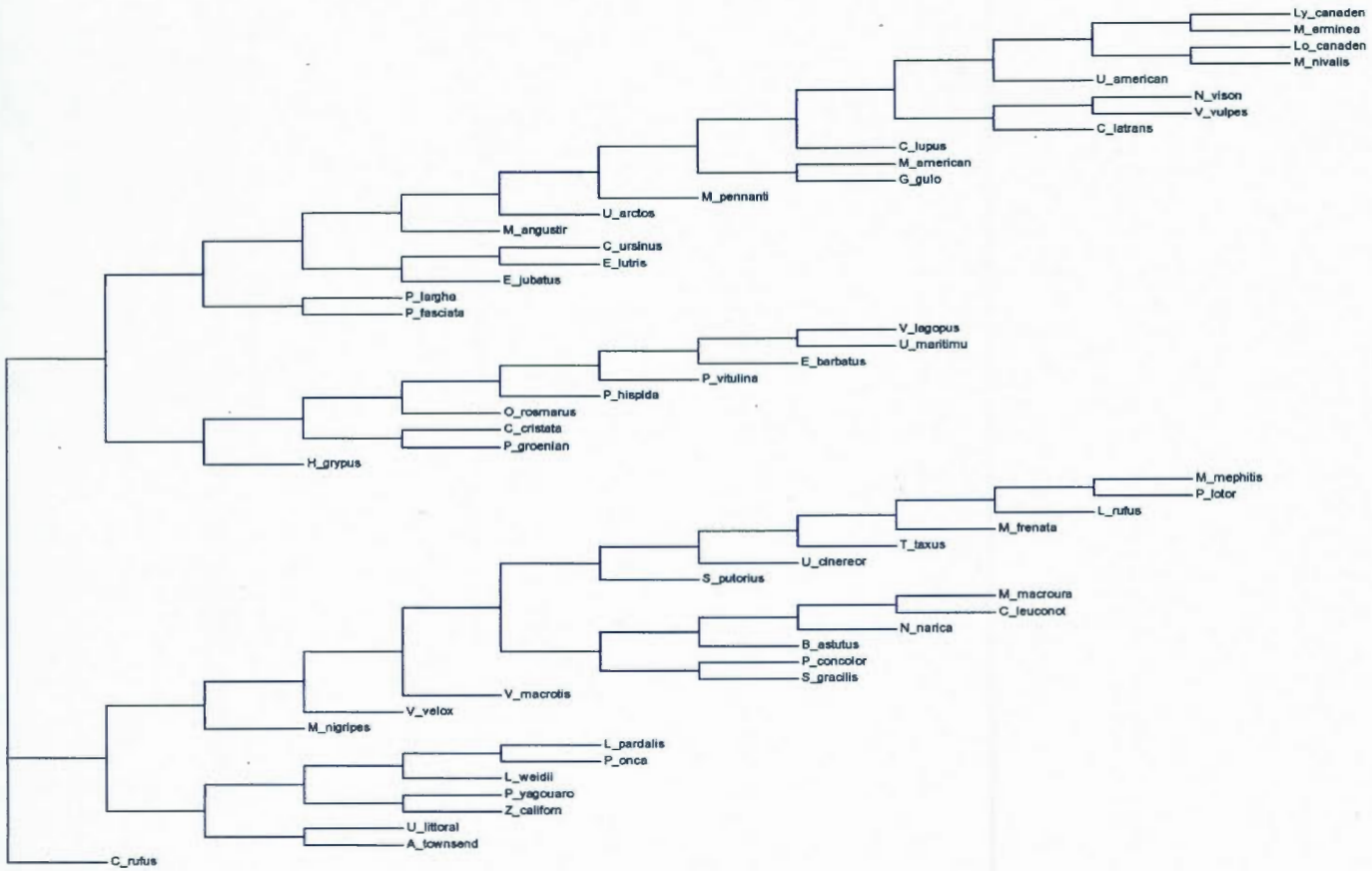


Figure 4.2: L'arbre de distribution géographique $T_2$.

Les données permettant la construction de cet arbre proviennent du site `atlas-mammiferes.fr`. Cet arbre comprend 52 espèces du groupe des Carnivores.

Tableau 4.2: Table des meilleures positions pour l'arbre de distribution géographique $T_2$.

| Noms des protéines | Tailles des fenêtres | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 10 | 20 | 30 | 40 | 50 | 60 | 70 | 80 | 90 | 100 |
| Adenosine A3 receptor | 1 (33,32) | 11 (33,27) | 1 (66,34) | 1 (33,27) | 1 (33,32) | 1 (33,39) | 1 (33,32) | **1 (33,43)** | 1 (33,40) | 1 (100,50) |
| Apolipoprotein B (ApoB) | 246 (50,19) | 246 (50,47) | 76 (50,48) | 231 (50,52) | 236 (75,59) | 231 (50,67) | 231 (50,66) | 216 (50,67) | 191 (50,72) | **191 (50,73)** |
| ATP synthase F0 subunit 6 (ATP-6) | 1 (60,35) | 6 (40,37) | 6 (40,40) | 31 (60,42) | 121 (40,31) | **21 (40,41)** | 1 (60,55) | 26 (60,45) | 11 (60,46) | 1 (60,48) |
| ATP synthase F0 subunit 8 (ATP-8) | **11 (60,45)** | 11 (80,52) | 6 (80,51) | 11 (80,62) | 11 (80,62) | 1 (80,63) | - - | - - | - - | - - |
| Brain derived neurotrophic factor (BDNF) | 66 (88,14) | 26 (88,18) | 31 (88,14) | 16 (94,26) | 16 (88,13) | **16 (88,20)** | 6 (94,27) | 16 (88,17) | 11 (94,27) | 16 (94,27) |
| Breast cancer susceptibility protein 1 (BRCA1) | **241 (0,27)** | 231 (33,42) | 126 (33,48) | 126 (33,56) | 146 (33,60) | 126 (33,57) | 126 (33,57) | 146 (33,51) | 41 (33,49) | 151 (33,53) |
| Cytochrome Oxidase Subunit I (CO-1) | 401 (60,34) | 386 (40,16) | 476 (40,45) | **368 (0,39)** | 356 (60,33) | 441 (60,44) | 371 (40,47) | 326 (40,47) | 316 (40,36) | 306 (60,56) |
| Growth hormone receptor (GRH) | 61 (40,22) | 61 (40,23) | 26 (60,27) | **31 (40,26)** | 46 (80,33) | 36 (60,32) | 31 (60,33) | 16 (80,37) | 6 (80,38) | 1 (80,39) |
| NADH dehydrogenase subunit 1 (NADH-1) | 156 (60,29) | 151 (40,20) | 51 (40,47) | 36 (60,43) | **31 (40,54)** | 251 (60,54) | 36 (60,49) | 1 (60,47) | 6 (60,55) | 66 (60,56) |
| NADH dehydrogenase subunit 2 (NADH-2) | 296 (60,35) | 286 (60,32) | 81 (60,39) | 66 (60,51) | 71 (60,51) | **58 (60,83)** | 51 (40,46) | 36 (60,53) | 11 (80,64) | 6 (80,64) |
| NADH dehydrogenase subunit 4 (NADH-4) | 251 (40,27) | 251 (40,40) | 6 (40,43) | 416 (20,23) | 236 (60,67) | **396 (40,59)** | 386 (40,43) | 186 (60,67) | 186 (60,70) | 186 (60,71) |
| NADH dehydrogenase subunit 4L (NADH-4L) | 46 (40,25) | **46 (40,26)** | 46 (60,47) | 11 (60,42) | 46 (80,46) | 16 (60,48) | 1 (80,49) | 11 (60,46) | 1 (80,49) | - - |
| NADH dehydrogenase subunit 5 (NADH-5) | 506 (40,11) | 21 (40,42) | 26 (40,50) | **251 (20,23)** | 251 (60,54) | 231 (40,48) | 216 (60,64) | 216 (60,58) | 21 (60,49) | 461 (60,57) |
| NADH dehydrogenase subunit 6 (NADH-6) | 76 (60,48) | **31 (20,35)** | 36 (40,44) | 46 (60,67) | 51 (60,67) | 41 (60,71) | 31 (60,72) | 21 (60,76) | 81 (60,76) | 71 (60,79) |
| Nicotinic cholinergic receptor alpha polypeptide 1 precursor (NCαP-1) | **1 (100,9)** | - - | - - | - - | - - | - - | - - | - - | - - | - - |
| Prepronociceptin (PPNOC) | 56 (86,12) | 21 (93,22) | 26 (93,25) | 26 (86,32) | 21 (86,29) | 11 (93,37) | **1 (73,38)** | 1 (93,41) | - - | - - |
| Recombination activating protein 1 (RAP-1) | 51 (50,30) | 31 (0,31) | 36 (0,35) | 41 (0,36) | 21 (0,38) | 11 (50,51) | 6 (0,37) | **1 (0,39)** | 31 (50,48) | 11 (50,46) |
| Retinoid Binding Protein (RBP) | 386 (25,19) | 321 (25,9) | 326 (75,53) | 326 (0,25) | 326 (75,60) | **301 (0,30)** | 326 (75,65) | 311 (75,64) | 306 (75,48) | 256 (75,51) |
| Rhodopsin | 46 (100,17) | 31 (93,6) | **26 (93,9)** | 16 (100,14) | 6 (100,14) | - - | - -) | - - | - - | - - |
| Sex Determining Region Y Protein (SRY) | 41 (0,65) | 36 (0,81) | 116 (0,81) | 126 (0,83) | 41 (0,83) | 86 (0,84) | 86 (0,85) | 56 (0,85) | **61 (0,87)** | 36 (0,86) |
| Von Willebrand Factor | 326 (33,39) | **301 (0,31)** | 11 (33,49) | 201 (33,57) | 1 (33,63) | 16 (33,51) | 1 (33,63) | 216 (66,48) | 221 (33,64) | 231 (66,56) |

Les protéines SRY, RAP1 et CO-1 sont significatives par leurs distances RF de 0, leurs valeurs de bootstrap moyen de 87 et 39 (39 pour RAP-1 et CO-1), leurs tailles de fenêtres de 90, 80 et 40 et leurs positions des fenêtres sur l'ASM de 61, 1 et 366 respectivement.

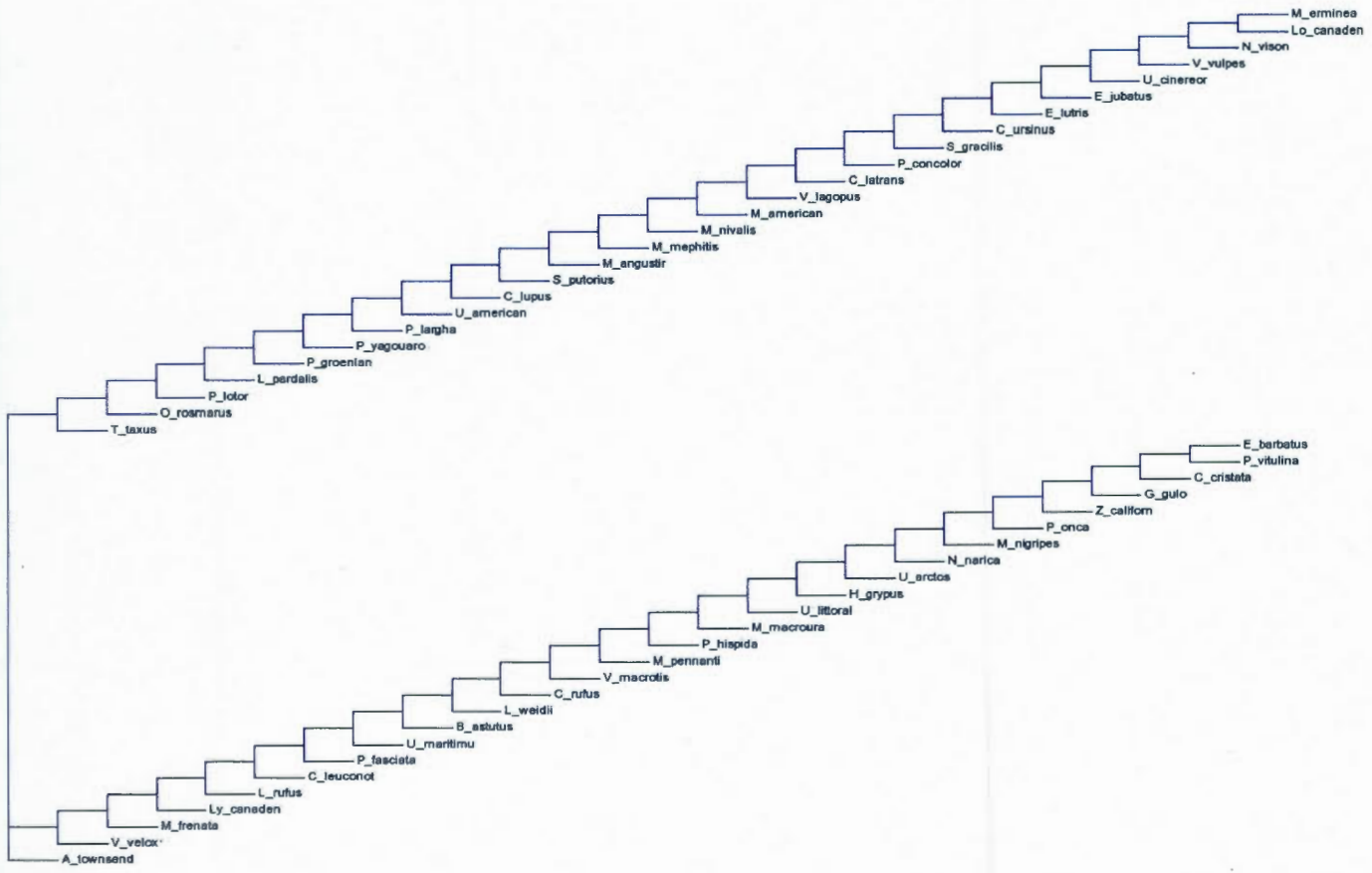


Figure 4.3: L'arbre de précipitations moyennes.

Les données permettant la construction de cet arbre proviennent d'une base de données privée de l'Université McGill. Cet arbre comprend 52 espèces du groupe des Carnivores.

Tableau 4.3: Table des meilleures positions pour l'arbre des précipitations moyennes.

| Noms des protéines | Tailles des fenêtres | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 10 | 20 | 30 | 40 | 50 | 60 | 70 | 80 | 90 | 100 |
| Adenosine A3 receptor | 1 (88,32) | 11 (88,29) | 1 (88,35) | 1 (88,43) | 1 (88,43) | 1 (88,45) | 1 (88,45) | 1 (88,46) | 1 (88,49) | **1 (88,50)** |
| Apolipoprotein B (ApoB) | 246 (89,39) | **246 (84,45)** | 76 (89,54) | 231 (89,56) | 236 (89,60) | 231 (89,67) | 231 (89,68) | 216 (89,71) | 191 (89,73) | 191 (89,76) |
| ATP synthase F0 subunit 6 (ATP-6) | 1 (87,38) | 6 (93,47) | 6 (93,47) | **31 (87,44)** | 121 (87,36) | 21 (93,55) | 1 (93,55) | 26 (93,58) | 11 (93,58) | 1 (93,59) |
| ATP synthase F0 subunit 8 (ATP-8) | 11 (94,45) | 11 (94,52) | 6 (94,56) | 11 (94,62) | **11 (94,63)** | **1 (94,63)** | - - | - - | - - | - - |
| Brain derived neurotrophic factor (BDNF) | 66 (95,18) | 26 (93,13) | 31 (93,16) | 16 (93,26) | 16 (93,25) | 16 (93,25) | **6 (93,27)** | 16 (93,26) | **11 (93,27)** | **16 (93,27)** |
| Breast cancer susceptibility protein 1 (BRCA1) | 241 (85,41) | **231 (85,45)** | 126 (91,54) | 126 (91,58) | 146 (91,60) | 126 (91,57) | 126 (91,57) | 146 (91,53) | 41 (91,54) | 151 (91,54) |
| Cytochrome Oxidase Subunit I (CO-1) | 401 (87,32) | 386 (87,20) | 476 (93,52) | 366 (87,20) | 356 (87,27) | 441 (93,49) | 371 (87,34) | 326 (87,34) | **316 (87,35)** | 306 (87,33) |
| Growth hormone receptor (GRH) | 61 (90,9) | **61 (90,16)** | 26 (95,33) | 31 (95,33) | 46 (95,33) | 36 (95,36) | 31 (95,36) | 16 (95,37) | 6 (95,38) | 1 (95,39) |
| NADH dehydrogenase subunit 1 (NADH-1) | 156 (87,17) | 151 (87,21) | 51 (87,52) | 36 (87,53) | 31 (87,51) | 251 (93,62) | 36 (93,59) | 1 (93,58) | **6 (87,54)** | 66 (93,64) |
| NADH dehydrogenase subunit 2 (NADH-2) | 296 (85,30) | **286 (85,32)** | 81 (90,64) | 66 (90,65) | 71 (90,67) | 56 (90,67) | 51 (90,67) | 36 (90,67) | 11 (90,73) | 6 (90,71) |
| NADH dehydrogenase subunit 4 (NADH-4) | 251 (88,42) | 251 (88,45) | 6 (88,46) | 416 (88,50) | 236 (88,44) | 396 (88,53) | 386 (88,51) | **186 (88,55)** | **186 (88,55)** | **186 (88,55)** |
| NADH dehydrogenase subunit 4L (NADH-4L) | 46 (93,47) | 46 (93,45) | 46 (93,49) | 11 (93,49) | 46 (93,46) | 16 (93,50) | 1 (93,49) | 11 (93,50) | **1 (93,51)** | - - |
| NADH dehydrogenase subunit 5 (NADH-5) | 506 (87,46) | 21 (87,59) | 26 (87,57) | 251 (87,41) | 251 (87,42) | 231 (87,41) | 216 (87,44) | 216 (87,42) | 21 (93,74) | **461 (87,65)** |
| NADH dehydrogenase subunit 6 (NADH-6) | 76 (57,35) | 31 (57,54) | 36 (57,58) | 46 (42,47) | 51 (57,51) | **41 (42,61)** | 31 (57,66) | 21 (57,63) | 81 (57,56) | 71 (57,60) |
| Nicotinic cholinergic receptor alpha polypeptide 1 precursor (NCαP-1) | **1 (94,9)** | - - | - - | - - | - - | - - | - - | - - | - - | - - |
| Prepronociceptin (PPNOC) | 56 (96,22) | 21 (93,22) | 26 (93,26) | 26 (93,35) | 21 (93,38) | 11 (93,37) | 1 (93,39) | **1 (93,41)** | - - | - - |
| Recombination activating protein 1 (RAP-1) | 51 (91,30) | 31 (85,35) | 36 (85,38) | 41 (85,36) | **21 (85,51)** | 11 (85,45) | 6 (91,50) | 1 (91,49) | 31 (85,48) | 11 (85,46) |
| Retinoid Binding Protein (RBP) | 386 (90,37) | 321 (90,44) | 326 (90,53) | 326 (90,59) | 326 (90,60) | 301 (90,62) | 326 (90,65) | 311 (90,63) | **308 (90,68)** | 256 (90,67) |
| Rhodopsin | **46 (93,17)** | 31 (90,12) | 26 (93,13) | 16 (93,14) | 6 (93,14) | - - | - - | - - | - - | - - |
| Sex Determining Region Y Protein (SRY) | 41 (66,65) | 36 (66,72) | 116 (77,81) | 126 (77,81) | 41 (66,75) | 86 (77,84) | **86 (66,84)** | 56 (66,69) | 61 (77,86) | 36 (66,77) |
| Von Willebrand Factor | 326 (82,17) | 301 (89,51) | **11 (75,40)** | 201 (82,52) | 1 (82,52) | 16 (82,51) | 1 (82,54) | 216 (89,71) | 221 (89,72) | 231 (89,74) |

Les protéines SRY et NADH-6 sont significatives par leurs distances RF de 66 et 42, leurs valeurs de bootstrap moyen de 84 et 61, leurs tailles de fenêtres de 70 et 60 et leurs positions des fenêtres sur l'ASM de 86 et 41 respectivement.

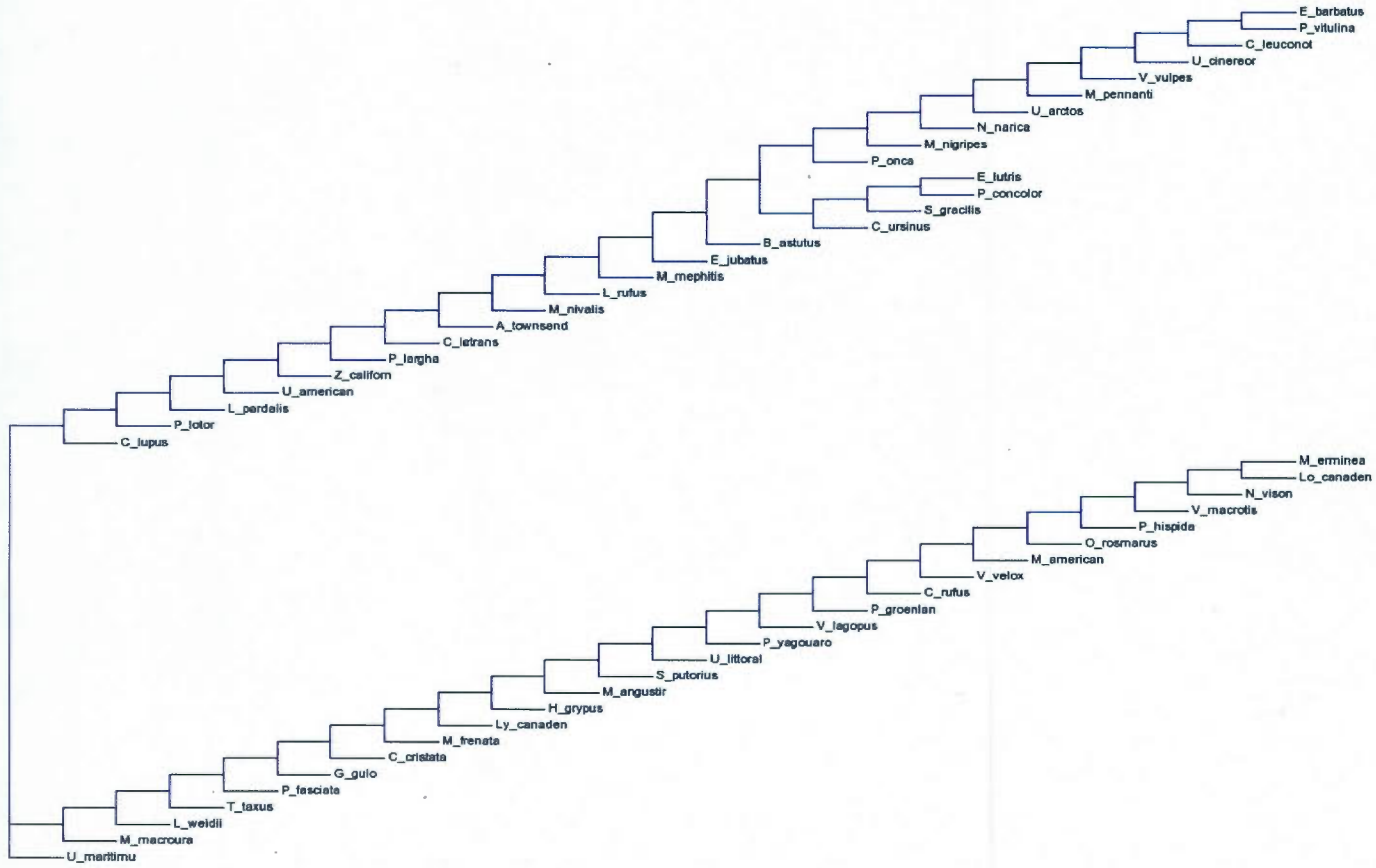


Figure 4.4: L'arbre des températures maximales moyennes.

Les données permettant la construction de cet arbre proviennent d'une base de données privée de l'Université McGill. Cet arbre comprend 52 espèces du groupe des Carnivores.

Tableau 4.4: Table des meilleures positions pour l'arbre des températures maximales moyennes.

| Noms des protéines | Tailles des fenêtres | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 10 | 20 | 30 | 40 | 50 | 60 | 70 | 80 | 90 | 100 |
| Adenosine A3 receptor | 1 (88,32) | 11 (88,29) | 1 (88,35) | 1 (88,43) | 1 (88,43) | 1 (88,45) | 1 (88,45) | 1 (88,46) | 1 (88,49) | **1 (88,50)** |
| Apolipoprotein B (ApoB) | 246 (84,39) | **246 (78,45)** | 231 (84,51) | 66 (84,52) | 216 (84,58) | 231 (89,67) | 231 (89,68) | 191 (84,68) | 186 (84,67) | 11 (84,60) |
| ATP synthase F0 subunit 6 (ATP-6) | 6 (87,39) | 6 (93,47) | 6 (93,47) | **31 (87,44)** | 121 (87,36) | 21 (93,55) | 1 (93,55) | 26 (93,58) | 11 (93,58) | 91 (90,52) |
| ATP synthase F0 subunit 8 (ATP-8) | 11 (94,45) | 11 (94,52) | 6 (94,56) | 11 (94,62) | **11 (94,63)** | **1 (94,63)** | - - | - - | - - | - - |
| Brain derived neurotrophic factor (BDNF) | 66 (95,18) | 26 (93,13) | 31 (93,16) | 16 (93,26) | 16 (93,25) | 16 (93,25) | **6 (93,27)** | 16 (93,26) | **11 (93,27)** | **16 (93,27)** |
| Breast cancer susceptibility protein 1 (BRCA1) | 241 (79,41) | 61 (79,44) | **148 (79,45)** | 151 (85,54) | 131 (85,50) | 116 (85,50) | 126 (91,57) | 101 (85,47) | 41 (91,54) | 151 (91,54) |
| Cytochrome Oxidase Subunit I (CO-1) | 391 (87,20) | 476 (93,47) | 451 (87,23) | **1 (87,29)** | 166 (87,14) | 441 (93,49) | 426 (93,47) | 406 (93,47) | 406 (93,53) | 391 (93,56) |
| Growth hormone receptor (GRH) | **26 (93,18)** | **11 (93,18)** | 26 (95,33) | 31 (95,33) | 46 (95,33) | 36 (95,36) | 31 (95,36) | 16 (95,37) | 6 (95,38) | 1 (95,39) |
| NADH dehydrogenase subunit 1 (NADH-1) | 71 (87,32) | 56 (93,58) | 51 (87,52) | 36 (87,53) | 31 (87,51) | 251 (93,62) | 36 (93,59) | 1 (93,58) | **6 (87,54)** | 66 (93,64) |
| NADH dehydrogenase subunit 2 (NADH-2) | 81 (92,46) | 116 (87,34) | 101 (87,35) | **101 (87,38)** | 71 (92,67) | 56 (92,67) | 51 (92,67) | 36 (92,67) | 11 (92,73) | 6 (92,71) |
| NADH dehydrogenase subunit 4 (NADH-4) | 186 (88,44) | 251 (88,45) | 6 (88,46) | 416 (88,50) | 236 (88,44) | 396 (88,53) | 386 (88,51) | **186 (88,55)** | **186 (88,55)** | **186 (88,55)** |
| NADH dehydrogenase subunit 4L (NADH-4L) | 46 (93,47) | 46 (93,45) | 46 (93,49) | 11 (93,49) | 46 (93,46) | 16 (93,50) | 1 (93,49) | 11 (93,50) | **1 (93,51)** | - - |
| NADH dehydrogenase subunit 5 (NADH-5) | 506 (87,46) | **61 (81,52)** | 121 (81,34) | 251 (87,41) | 251 (87,42) | 81 (87,53) | 216 (87,44) | 71 (87,54) | 21 (93,74) | 81 (87,46) |
| NADH dehydrogenase subunit 6 (NADH-6) | 26 (71,37) | 6 (85,72) | 71 (71,48) | **81 (71,51)** | 86 (85,67) | 91 (85,71) | 41 (85,73) | 36 (85,76) | 36 (85,76) | 16 (85,79) |
| Nicotinic cholinergic receptor alpha polypeptide 1 precursor (NCαP-1) | **1 (97,9)** | - - | - - | - - | - - | - - | - - | - - | - - | - - |
| Prepronociceptin (PPNOC) | 56 (96,22) | 21 (93,22) | 26 (93,26) | **16 (90,27)** | 21 (93,38) | 11 (93,37) | 1 (93,39) | 1 (93,41) | - - | - - |
| Recombination activating protein 1 (RAP-1) | 51 (88,30) | 31 (82,35) | 26 (82,43) | 41 (82,36) | 21 (82,51) | 11 (82,45) | 6 (88,50) | 1 (88,49) | **46 (78,41)** | 11 (82,46) |
| Retinoid Binding Protein (RBP) | 386 (90,37) | 341 (85,44) | 326 (85,53) | 326 (85,59) | 326 (85,60) | 301 (85,62) | 306 (85,60) | 286 (85,62) | 291 (85,65) | **256 (85,67)** |
| Rhodopsin | 46 (95,17) | **31 (92,12)** | 26 (95,13) | 16 (95,14) | 6 (95,14) | - - | - - | - - | - - | - - |
| Sex Determining Region Y Protein (SRY) | 41 (77,65) | 136 (77,74) | 116 (77,81) | 126 (77,81) | 116 (77,83) | 86 (77,84) | 76 (77,85) | 66 (77,83) | **61 (77,86)** | 66 (77,84) |
| Von Willebrand Factor | 266 (82,17) | 216 (82,41) | **206 (75,47)** | 201 (82,52) | 1 (82,52) | 16 (82,51) | 1 (82,54) | 216 (89,71) | 221 (89,72) | 231 (89,74) |

Les protéines SRY et Von Willebrand sont significatives par leurs distances RF de 75 et 71, leurs valeurs de bootstrap moyen de 47 et 51, leurs tailles de fenêtres de 30 et 40 et leurs positions des fenêtres sur l'ASM de 206 et 61 respectivement.

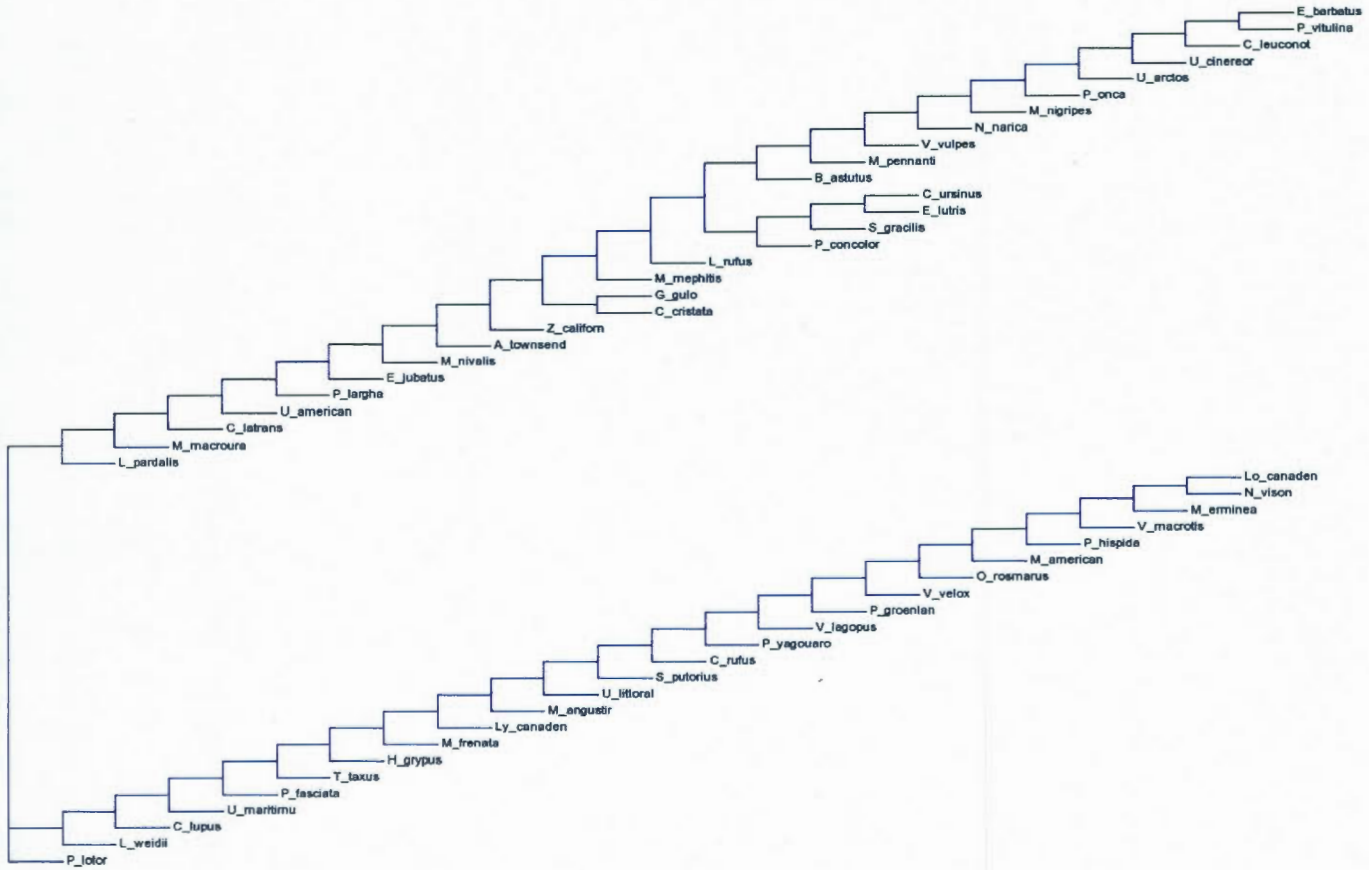


Figure 4.5: L'arbre des températures minimales moyennes.

Les données permettant la construction de cet arbre proviennent d'une base de données privée de l'Université McGill. Cet arbre comprend 52 espèces du groupe des Carnivores.

Tableau 4.5: Table des meilleures positions pour l'arbre des températures minimales moyennes.

| Noms des protéines | Tailles des fenêtres | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 10 | 20 | 30 | 40 | 50 | 60 | 70 | 80 | 90 | 100 |
| Adenosine A3 receptor | 1 (91,32) | 11 (91,29) | 1 (91,35) | 1 (91,43) | 1 (91,43) | 1 (91,45) | 1 (91,45) | 1 (91,46) | 1 (91,49) | **1 (91,50)** |
| Apolipoprotein B (ApoB) | 246 (92,39) | **248 (86,45)** | 76 (92,54) | 231 (92,56) | 236 (92,60) | 231 (92,67) | 231 (92,68) | 216 (92,71) | 191 (92,73) | 191 (92,76) |
| ATP synthase F0 subunit 6 (ATP-6) | 6 (87,39) | 6 (93,47) | 6 (93,47) | **31 (87,44)** | 121 (87,36) | 21 (93,55) | 1 (93,55) | 26 (93,58) | 11 (93,58) | 1 (93,59) |
| ATP synthase F0 subunit 8 (ATP-8) | 11 (94,45) | 11 (94,52) | 6 (94,56) | 11 (94,62) | **11 (94,63)** | **1 (94,63)** | - - | - - | - - | - - |
| Brain derived neurotrophic factor (BDNF) | 36 (95,13) | 26 (95,13) | 31 (95,16) | 16 (95,26) | 16 (95,25) | 16 (95,25) | **6 (95,27)** | 16 (95,26) | **11 (95,27)** | **16 (95,27)** |
| Breast cancer susceptibility protein 1 (BRCA1) | 126 (85,50) | 121 (85,45) | **126 (85,54)** | 136 (85,46) | 131 (85,50) | 116 (85,50) | 126 (91,57) | 101 (85,47) | 41 (91,54) | 151 (91,54) |
| Cytochrome Oxidase Subunit I (CO-1) | 391 (87,20) | 416 (87,17) | 21 (87,22) | 471 (87,39) | 461 (93,48) | 231 (87,17) | 426 (93,47) | 406 (93,47) | 416 (87,44) | **411 (87,50)** |
| Growth hormone receptor (GRH) | 86 (95,23) | 76 (95,29) | 26 (95,33) | 31 (95,33) | 6 (90,29) | **1 (90,32)** | 31 (95,36) | 16 (95,37) | 6 (95,38) | 1 (95,39) |
| NADH dehydrogenase subunit 1 (NADH-1) | 71 (87,32) | 56 (93,58) | 51 (87,52) | 36 (87,53) | 31 (87,51) | 251 (93,62) | 36 (93,59) | 1 (93,58) | **6 (87,54)** | 66 (93,64) |
| NADH dehydrogenase subunit 2 (NADH-2) | **321 (87,38)** | 321 (87,31) | 81 (92,64) | 66 (92,65) | 71 (92,67) | 56 (92,67) | 51 (92,67) | 36 (92,67) | 11 (92,73) | 6 (92,71) |
| NADH dehydrogenase subunit 4 (NADH-4) | 186 (88,44) | 251 (88,45) | 6 (88,46) | 416 (88,50) | 236 (88,44) | 396 (88,53) | 386 (88,51) | **186 (88,55)** | **186 (88,55)** | **186 (88,55)** |
| NADH dehydrogenase subunit 4L (NADH-4L) | 46 (93,47) | 46 (93,45) | 46 (93,49) | 11 (93,49) | 46 (93,46) | 16 (93,50) | 1 (93,49) | 11 (93,50) | **1 (93,51)** | - - |
| NADH dehydrogenase subunit 5 (NADH-5) | 506 (87,46) | **61 (81,52)** | 121 (81,34) | 251 (87,41) | 251 (87,42) | 81 (87,53) | 216 (87,44) | 71 (87,54) | 21 (93,74) | 81 (87,46) |
| NADH dehydrogenase subunit 6 (NADH-6) | 36 (71,60) | 116 (71,52) | 71 (57,48) | 36 (71,62) | **36 (57,61)** | 36 (71,66) | 31 (71,66) | 21 (71,63) | 36 (85,76) | 6 (71,67) |
| Nicotinic cholinergic receptor alpha polypeptide 1 precursor (NCαP-1) | **1 (97,9)** | - - | - - | - - | - - | - - | - - | - - | - - | - - |
| Prepronociceptin (PPNOC) | 56 (98,22) | 21 (95,22) | 26 (95,26) | 26 (95,35) | 21 (95,38) | 11 (95,37) | 1 (95,39) | **1 (95,41)** | - - | - - |
| Recombination activating protein 1 (RAP-1) | 51 (94,30) | 31 (88,35) | 36 (88,38) | 41 (88,36) | 21 (88,51) | 11 (88,45) | 6 (94,50) | 1 (94,49) | **46 (82,41)** | 11 (88,46) |
| Retinoid Binding Protein (RBP) | **26 (88,23)** | 321 (92,44) | 326 (92,53) | 326 (92,59) | 326 (92,60) | 301 (92,62) | 326 (92,65) | 156 (88,18) | 306 (92,68) | 256 (92,67) |
| Rhodopsin | 46 (96,17) | **31 (93,12)** | 26 (96,13) | 16 (96,14) | 6 (96,14) | - - | - - | - - | - - | - - |
| Sex Determining Region Y Protein (SRY) | 41 (77,65) | 136 (77,74) | 116 (77,81) | 126 (77,81) | 116 (77,83) | 86 (77,84) | 76 (77,85) | 66 (77,83) | **61 (77,86)** | 66 (77,84) |
| Von Willebrand Factor | 221 (82,40) | 211 (82,44) | 206 (82,47) | 201 (82,52) | 1 (82,52) | 16 (82,51) | **1 (82,54)** | 216 (89,71) | 221 (89,72) | 231 (89,74) |

Les protéines SRY et NADH-6 sont significatives par leurs distances RF de 77 et 57, leurs valeurs de bootstrap moyen de 86 et 61, leurs tailles de fenêtres de 90 et 50 et leurs positions des fenêtres sur l'ASM de 61 et 36 respectivement.

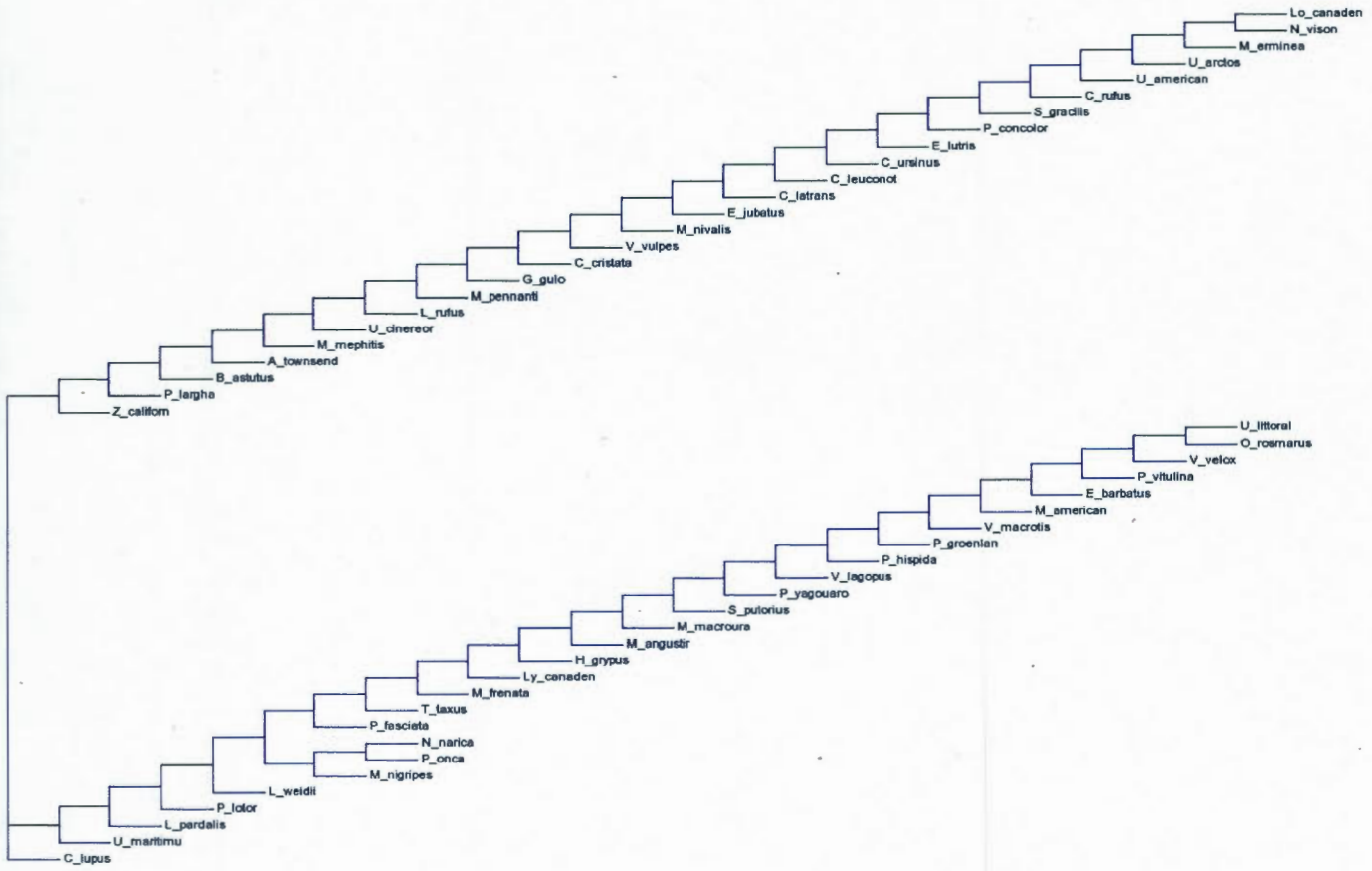


Figure 4.6: L'arbre des températures moyennes.

Les données permettant la construction de cet arbre proviennent d'une base de données privée de l'Université McGill. Cet arbre comprend 52 espèces du groupe des Carnivores.

Tableau 4.6: Table des meilleures positions pour l'arbre des températures moyennes.

| Noms des protéines | Tailles des fenêtres | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 10 | 20 | 30 | 40 | 50 | 60 | 70 | 80 | 90 | 100 |
| Adenosine A3 receptor | 1 (86,32) | 11 (86,29) | 1 (86,35) | 1 (86,43) | 1 (86,43) | 1 (86,45) | 1 (86,45) | 1 (86,46) | 1 (86,49) | **1 (86,50)** |
| Apolipoprotein B (ApoB) | 246 (86,39) | **246 (81,45)** | 76 (86,54) | 231 (86,56) | 236 (86,60) | 231 (86,67) | 231 (86,68) | 216 (86,71) | 191 (86,73) | 191 (86,76) |
| ATP synthase F0 subunit 6 (ATP-6) | 1 (87,38) | 6 (93,47) | 106 (87,34) | **31 (87,44)** | 121 (87,36) | 21 (93,55) | 1 (93,55) | 26 (93,58) | 11 (93,58) | 1 (93,59) |
| ATP synthase F0 subunit 8 (ATP-8) | 11 (94,45) | 11 (94,52) | 6 (94,56) | 11 (94,62) | 11 (94,63) | **1 (94,63)** | - - | - - | - - | - - |
| Brain derived neurotrophic factor (BDNF) | 66 (94,18) | 1 (94,20) | 16 (94,25) | 16 (91,26) | 16 (91,25) | 16 (91,25) | **6 (91,27)** | 16 (91,26) | **11 (91,27)** | 11 (91,26) |
| Breast cancer susceptibility protein 1 (BRCA1) | 241 (85,41) | **231 (85,45)** | 126 (91,54) | 126 (91,58) | 146 (91,60) | 126 (91,57) | 126 (91,57) | 146 (91,53) | 41 (91,54) | 151 (91,54) |
| Cytochrome Oxidase Subunit I (CO-1) | 481 (93,47) | 476 (93,47) | 476 (93,52) | **106 (87,10)** | 461 (93,48) | 441 (93,49) | 426 (93,47) | 406 (93,47) | 406 (93,53) | 391 (93,56) |
| Growth hormone receptor (GRH) | 51 (95,23) | 76 (95,29) | 26 (95,33) | 31 (95,33) | **6 (90,29)** | 36 (95,36) | 31 (95,36) | 16 (95,37) | 6 (95,38) | 1 (95,39) |
| NADH dehydrogenase subunit 1 (NADH-1) | 66 (93,58) | 56 (93,58) | 51 (87,52) | 36 (87,53) | 31 (87,51) | 86 (87,33) | 36 (93,59) | 1 (93,58) | **6 (87,54)** | 66 (93,64) |
| NADH dehydrogenase subunit 2 (NADH-2) | 216 (82,31) | 216 (82,42) | 221 (82,48) | 211 (82,47) | **216 (82,51)** | 56 (87,67) | 51 (87,67) | 36 (87,67) | 11 (87,73) | 6 (87,71) |
| NADH dehydrogenase subunit 4 | 251 (88,42) | 416 (82,39) | 411 (82,44) | **416 (82,50)** | 406 (88,48) | 391 (82,42) | 386 (88,51) | 186 (88,55) | 186 (88,55) | 186 (88,55) |
| NADH dehydrogenase subunit 4L (NADH-4L) | 46 (93,47) | 46 (93,45) | 46 (93,49) | 11 (93,49) | 46 (93,46) | 16 (93,50) | 1 (93,49) | 11 (93,50) | **1 (93,51)** | - - |
| NADH dehydrogenase subunit 5 (NADH-5) | 506 (87,46) | **21 (87,59)** | 26 (87,57) | 251 (87,41) | 251 (87,42) | 316 (87,48) | 216 (87,44) | 216 (87,42) | 21 (93,74) | 11 (93,74) |
| NADH dehydrogenase subunit 6 (NADH-6) | 11 (71,62) | 101 (57,54) | 31 (57,59) | **101 (57,61)** | 71 (71,63) | 101 (57,54) | 101 (57,51) | 81 (57,55) | 86 (71,57) | 6 (71,67) |
| Nicotinic cholinergic receptor alpha polypeptide 1 precursor (NCαP-1) | **1 (94,9)** | - - | - - | - - | - - | - - | - - | - - | - - | - - |
| Prepronociceptin (PPNOC) | 56 (95,22) | 21 (91,22) | 26 (91,26) | 26 (91,35) | 21 (91,38) | 11 (91,37) | 1 (91,39) | **1 (91,41)** | - - | - - |
| Recombination activating protein 1 (RAP-1) | 51 (88,30) | 31 (82,35) | 36 (82,38) | 41 (82,36) | **21 (82,51)** | 11 (82,45) | 6 (88,50) | 1 (88,49) | 31 (82,48) | 11 (82,46) |
| Retinoid Binding Protein (RBP) | 386 (88,37) | 321 (88,44) | **256 (85,34)** | 326 (88,59) | 326 (88,60) | 301 (88,62) | 326 (88,65) | 311 (88,63) | 306 (88,68) | 256 (88,67) |
| Rhodopsin | **46 (95,17)** | 36 (95,16) | 26 (95,13) | 16 (95,14) | 6 (95,14) | - - | - - | - - | - - | - - |
| Sex Determining Region Y Protein (SRY) | 46 (66,64) | **46 (55,66)** | 101 (66,60) | 181 (77,83) | 71 (55,57) | 71 (66,62) | 71 (66,68) | 66 (77,83) | 61 (77,86) | 66 (77,84) |
| Von Willebrand Factor | 41 (82,39) | 41 (82,40) | 206 (82,47) | 36 (75,38) | **1 (87,52)** | 16 (75,51) | 1 (75,54) | 1 (82,56) | 1 (82,59) | 1 (82,58) |

Les protéines adénosine A3 et NADH-6 sont significatives par leurs distances RF de 55 et 57, leurs valeurs de bootstrap moyen de 66 et 61, leurs tailles de fenêtres de 20 et 40 et leurs positions des fenêtres sur l'ASM de 46 et 101 respectivement.

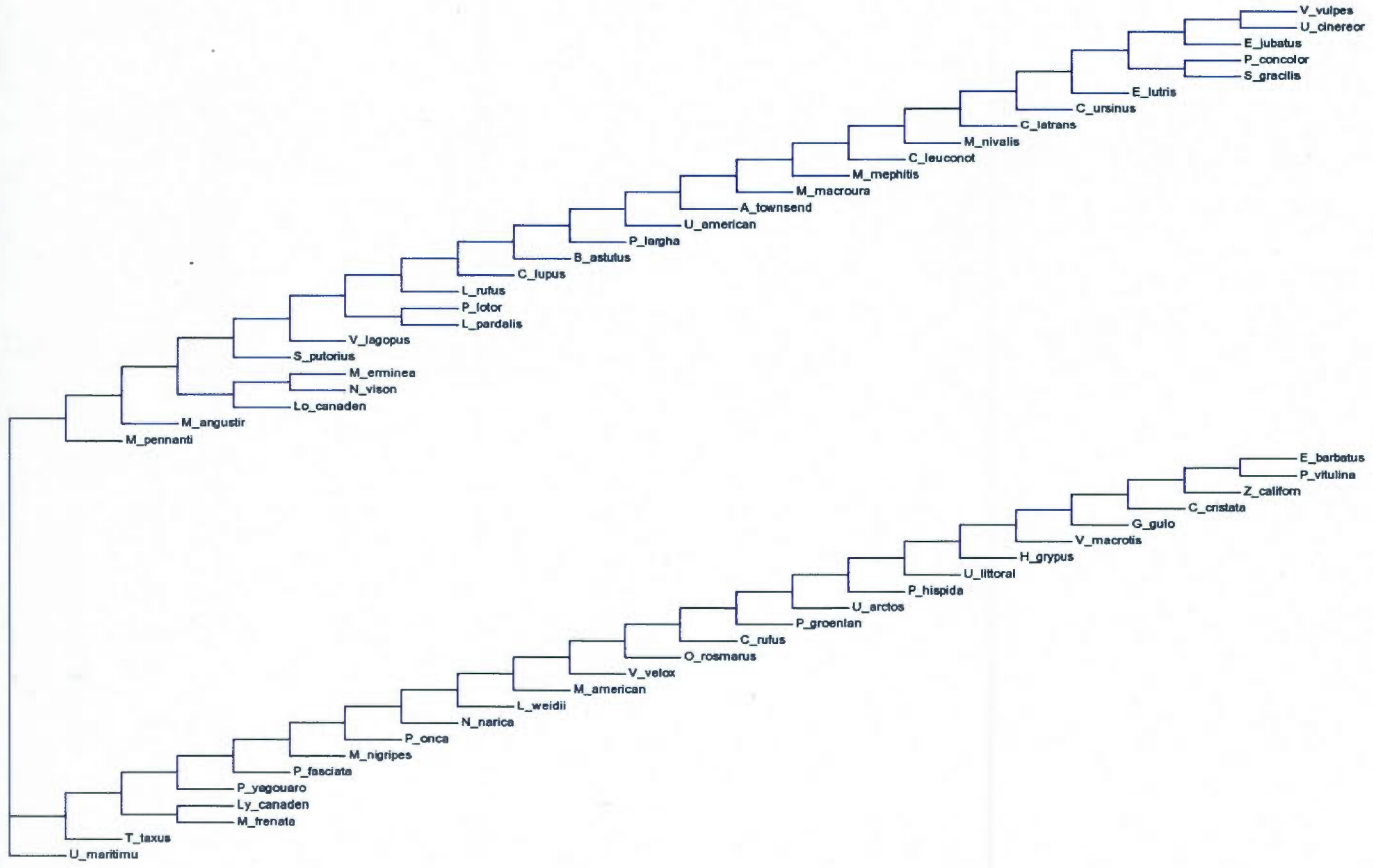


Figure 4.7: L'arbre des altitudes.

Les données permettant la construction de cet arbre proviennent d'une base de données privée de l'Université McGill. Cet arbre comprend 52 espèces du groupe des Carnivores.

Tableau 4.7: Table des meilleures positions pour l'arbre des altitudes.

| Noms des protéines | Tailles des fenêtres | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 10 | 20 | 30 | 40 | 50 | 60 | 70 | 80 | 90 | 100 |
| Adenosine A3 receptor | 1 (83,32) | 11 (83,29) | **46 (77,17)** | 1 (83,43) | 1 (83,43) | 1 (83,45) | 1 (83,45) | 1 (83,46) | 1 (83,49) | 1 (83,50) |
| Apolipoprotein B (ApoB) | 246 (84,39) | 251 (78,28) | 96 (78,42) | **86 (78,55)** | 61 (78,52) | 231 (84,67) | 231 (84,68) | 216 (84,71) | 191 (84,73) | 191 (84,76) |
| ATP synthase F0 subunit 6 (ATP-6) | 186 (90,42) | 6 (90,47) | 6 (90,47) | 21 (90,52) | 21 (90,56) | 21 (90,55) | 1 (90,55) | 26 (90,58) | 11 (90,58) | **1 (90,59)** |
| ATP synthase F0 subunit 8 (ATP-8) | 11 (91,45) | 11 (91,52) | 6 (91,56) | 11 (91,62) | **11 (91,63)** | **1 (91,63)** | - - | - - | - - | - - |
| Brain derived neurotrophic factor (BDNF) | 66 (94,18) | 26 (91,13) | 31 (91,16) | 16 (91,26) | 16 (91,25) | 16 (91,25) | **6 (91,27)** | 16 (91,26) | **11 (91,27)** | **16 (91,27)** |
| Breast cancer susceptibility protein 1 (BRCA1) | 246 (82,42) | 231 (82,45) | 231 (82,48) | 221 (82,48) | 216 (82,51) | 196 (82,49) | 196 (82,49) | **176 (82,51)** | 41 (88,54) | 151 (88,54) |
| Cytochrome Oxidase Subunit I (CO-1) | 481 (87,47) | 41 (87,25) | 46 (87,27) | 46 (87,29) | 36 (87,32) | 1 (81,31) | 26 (87,29) | **1 (81,34)** | 1 (87,36) | 31 (87,31) |
| Growth hormone receptor (GRH) | 46 (90,22) | 31 (90,25) | 31 (90,30) | 61 (90,28) | 46 (95,33) | 36 (95,36) | 6 (90,29) | **6 (90,31)** | 6 (95,38) | 1 (95,39) |
| NADH dehydrogenase subunit 1 (NADH-1) | 66 (90,58) | 151 (84,21) | 81 (84,27) | 66 (90,58) | 56 (90,57) | 251 (90,62) | **91 (84,31)** | 1 (90,58) | 66 (90,61) | 66 (90,64) |
| NADH dehydrogenase subunit 2 (NADH-2) | 311 (82,34) | 216 (82,42) | 211 (82,49) | 221 (82,52) | 221 (82,49) | 166 (82,51) | 51 (87,67) | 146 (82,55) | **151 (82,59)** | 146 (82,55) |
| NADH dehydrogenase subunit 4 (NADH-4) | 96 (85,37) | 91 (85,35) | 76 (85,34) | **191 (85,38)** | 161 (91,67) | 156 (91,68) | 126 (91,70) | 121 (91,71) | 121 (91,70) | 121 (91,71) |
| NADH dehydrogenase subunit 4L (NADH-4L) | 46 (90,47) | 46 (90,45) | 46 (90,49) | 11 (90,49) | 46 (90,46) | 16 (90,50) | 1 (90,49) | 11 (90,50) | **1 (90,51)** | - - |
| NADH dehydrogenase subunit 5 (NADH-5) | 586 (84,24) | 51 (90,62) | 51 (90,69) | 41 (90,68) | **216 (84,38)** | 11 (90,72) | 1 (90,72) | 1 (90,74) | 21 (90,74) | 11 (90,74) |
| NADH dehydrogenase subunit 6 (NADH-6) | 36 (71,60) | 36 (71,59) | 106 (71,61) | 36 (71,62) | 36 (71,61) | **41 (57,61)** | 31 (71,66) | 21 (71,63) | 36 (85,76) | 16 (85,79) |
| Nicotinic cholinergic receptor alpha polypeptide 1 precursor (NCαP-1) | **1 (94,9)** | - - | - - | - - | - - | - - | - - | - - | - - | - - |
| Prepronociceptin (PPNOC) | 56 (95,22) | 21 (91,22) | 26 (91,26) | 26 (91,35) | 21 (91,38) | 11 (91,37) | 1 (91,39) | **1 (91,41)** | - - | - - |
| Recombination activating protein 1 (RAP-1) | 51 (82,30) | 31 (82,35) | 26 (82,43) | 26 (82,49) | **21 (82,51)** | **21 (82,51)** | 6 (82,50) | 1 (82,49) | 6 (82,48) | 11 (82,46) |
| Retinoid Binding Protein (RBP) | 351 (80,33) | 351 (80,34) | 56 (80,24) | **351 (80,49)** | 66 (80,39) | 61 (80,41) | 56 (80,42) | 41 (80,41) | 71 (80,47) | **61 (80,49)** |
| Rhodopsin | 46 (95,17) | **31 (92,12)** | 26 (95,13) | 16 (95,14) | 6 (95,14) | - - | - - | - - | - - | - - |
| Sex Determining Region Y Protein (SRY) | 41 (77,65) | 136 (77,74) | 116 (77,81) | 126 (77,81) | 116 (77,83) | 86 (77,84) | 76 (77,85) | 66 (77,83) | **61 (77,86)** | 66 (77,84) |
| Von Willebrand Factor | 81 (78,42) | 81 (78,43) | 216 (78,54) | **206 (78,55)** | 1 (78,52) | 11 (78,54) | 1 (78,54) | 11 (78,54) | **11 (78,55)** | **11 (78,55)** |

Les protéines adénosine A3 et NADH-6 sont significatives par leurs distances RF de 77 et 57, leurs valeurs de bootstrap moyen de 17 et 61, leurs tailles de fenêtres de 30 et 60 et leurs positions des fenêtres sur l'ASM de 46 et 41 respectivement.

### 4.2.2 Le temps d'exécution du programme

Nous avons réalisé des tests d'exécution de notre programme pour étudier l'évolution de la durée d'exécution en variant uniquement la taille de la fenêtre, mais en prenant à chaque fois le même jeu de données et les mêmes autres paramètres d'entrée.

Pour toutes ces exécutions, nous avons utilisé les paramètres d'entrée suivants :

- le nombre d'alignements à traiter (21) voir le chapitre III,
- le type de données des alignements (protéines),
- le nombre d'arbres de référence (7),
- le pas d'avancement de 5,
- la taille des fenêtres coulissantes variant de 10 à 100 avec un intervalle de 10 pour chaque exécution.

À la suite de nos tests, nous avons construit un graphique représentant la durée d'exécution en fonction de la taille des fenêtres coulissantes (voir la figure 4.8).

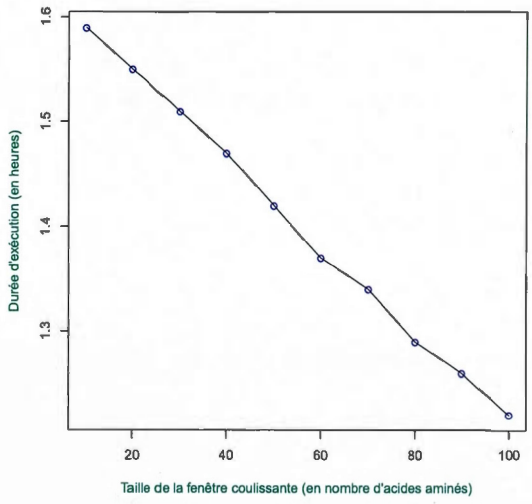


Figure 4.8: Évolution de la durée d'exécution du programme Java en fonction de la taille de la fenêtre coulissante.

Les résultats de la simulation sur la figure 4.8 montrent une diminution de la durée d'exécution quand la taille de la fenêtre coulissante augmente. Nous observons une équation approchant la droite suivante $y = -0.0042x + 1.632$ , avec le coefficient directeur $-0.0042$ (le signe $-$ indique la diminution de la droite) et d'ordonnée à l'origine de 1.632. Cette diminution correspond bien à notre attente, puisqu'en augmentant la taille de la fenêtre coulissante, pour un même jeu de données, il y aura moins de fenêtres à traiter au total. Bien que les fenêtres soient de plus grandes tailles, elles auront de ce fait un temps d'exécution des programmes externes légèrement plus long (voir le chapitre II concernant les différents programmes externes : paquet PHYLIP, PhyML en autres), leur plus petit nombre sera l'élément crucial pour la diminution de la complexité algorithmique.

### 4.2.3 Analyse des résultats

Nous constatons que sur l'ensemble des différents tableaux des résultats, la plus petite valeur de la distance RF normalisée est de 42, mis à part les résultats obtenus à partir de l'arbre de référence $T_2$. En comparant tous les arbres issus des fragments des protéines de nos jeux de données vis-à-vis des arbres de référence, nous constatons donc qu'il s'agit des topologies d'arbres relativement éloignées.

Soulignons que plusieurs valeurs de la distance RF pour les résultats correspondant à l'arbre de référence $T_2$ valent zéro. Cette différence significative des résultats obtenus pour $T_2$ avec les résultats obtenus pour les autres arbres de référence est probablement due à la construction de la matrice présence/absence. Rappelons que les données permettant la construction de l'arbre de référence $T_2$ proviennent du site web : `atlas-mammiferes.fr`. Précisons par ailleurs que la construction de la matrice présence/absence des espèces vis-à-vis d'un habitat a été réalisée manuellement. Il s'agit d'une longue tâche qui pourrait mener à quelques erreurs.

Concernant les valeurs du bootstrap moyen, obtenues pour l'ensemble des arbres, elles étaient majoritairement supérieures à 50%. Ceci suggère que les arbres construits à partir

des fragments des protéines avaient généralement un bon support statistique.

Les résultats obtenus mettent en avant les deux protéines suivantes :

- La protéine SRY, qui est la protéine de détermination du sexe sur la région Y. Le gène SRY est un gène dit « architecte ». Ce gène se localise sur le chromosome Y sur son bras court à la position Yp11.31. La traduction du gène SRY aboutit à la synthèse de la protéine TDF (Testis Determining Factor). Cette protéine activera d'autres gènes, entrainant ainsi la différenciation des gonades.

  Notons que sans la présence du gène SRY, l'évolution des gonades en testicules est un processus impossible. Ce qui est pertinent pour ce gène, est le fait qu'il se présente chez tous les mammifères, et que son activation a lieu tôt dans l'évolution embryonnaire. À ce stade de développement, les paramètres climatiques sont cruciaux. En observant nos résultats, nous pouvons constater l'influence des critères météorologiques (précipitation, altitudes, températures) et aussi des distributions géographiques des espèces en Amérique du Nord sur l'évolution du gène SRY (voir les tableaux de 4.1 à 4.5).

- NADH-6 est une protéine appartenant au complexe I se localisant dans la membrane interne des mitochondries. Cette protéine participe activement à la chaîne de transport d'électrons. Lors de ce processus, trois enzymes y participent, il s'agit de NADH déshydrogénase (complexe I), la coenzyme Q - Cytochrome c réductase (complexe III), et le cytochrome c oxydase (complexe IV).

  Le complexe I est le plus grand des deux autres et l'enzyme NADH déshydrogénase est la plus active. NADH prendra l'ion hydrogène se trouvant dans la matrice de la mitochondrie (liquide à l'intérieur de la mitochondrie), produisant ainsi le potentiel électrochimique permettant la production de l'ATP.

  Le complexe I peut aussi participer au déclenchement de l'apoptose (mort cellulaire programmée). Il a été démontré qu'au cours du développement des embryons soma-

tiques, il existe une corrélation entre les activités mitochondriales et l'apoptose (Petrussa et al., 2009).

En analysant nos résultats, nous pouvons supposer que l'évolution des gènes codant pour les protéines SRY et NADH-6 des espèces du groupe des Carnivores d'Amérique du Nord peut dépendre de critères environnementaux. Cette évolution coïncide partiellement avec les changements climatiques du milieu. De plus, nous constatons que les résultats sont encore plus significatifs pour des fragments se situant vers la position 180 aa pour l'alignement de la protéine SRY et de 60 aa pour l'alignement de la protéine NADH-6, ce qui laisse à penser qu'il s'agit des zones de haute pression sélective. Pour aller plus loin dans ce sens, nous aurions pu construire une représentation tridimensionnelle des protéines, par exemple à travers l'utilisation du logiciel PyMOL (DeLano, 2002). Avec une meilleure connaissance des structures protéiques, nous aurions pu trouver des liens entre cette analyse structurale et les positions les plus pertinentes des fenêtres ayant été sélectionnées par notre programme.

* * *

## 4.3 Conclusion

Nous avons testé notre algorithme à travers un jeu de données réelles provenant d'un ensemble d'espèces du groupe des Carnivores d'Amérique du Nord. La durée d'exécution de notre programme était raisonnable pour ce jeu de données. Par exemple, pour une taille de fenêtres de 10 aa[1], avec 21 protéines à traiter, dont les tailles des alignements s'étendaient de 14 aa à 609 aa, 7 arbres de référence à comparer aux arbres des fragments, un pas d'avancement de 5, la durée d'exécution était de 1h 36.

En plus, les résultats obtenus sur les différents jeux de données ont mis en avant deux

1. acide aminé

protéines suivantes : la protéine SRY et la protéine NADH-6, avec respectivement les longueurs des alignements de 396 aa et 175 aa.

Ces différentes informations peuvent nous permettre d'émettre la nouvelle hypothèse suivante : les fragments des ASM des protéines SRY et NADH-6, pour l'ensemble des 52 espèces du groupe des Carnivores, dépendent de certains paramètres climatiques (la température, l'altitude et les précipitations) en Amérique du Nord. Il serait pertinent par la suite de chercher la correspondance structurale de ces fragments en les corroborant avec les différentes conditions climatiques.

# CONCLUSION ET PERSPECTIVES

Dans ce mémoire de maîtrise en Informatique, nous avons décrit l'algorithme permettant de retrouver les relations existantes entre un arbre de référence et un fragment d'un gène. Nous avons implémenté cet algorithme en langage Java (voir Appendice A). Par la suite, nous avons réalisé la collecte de données réelles. Pour cette étude, nous avons considéré un ensemble de 52 espèces du groupe des Carnivores se localisant en Amérique du Nord. Nous avons sélectionné quelques paramètres environnementaux qui sont les suivants : les précipitations moyennes, les températures maximales moyennes, les températures minimales moyennes, les températures moyennes et les altitudes, mesurées pour différentes zones géographiques de l'Amérique du Nord. Pour cet ensemble d'espèces, nous avons recensé le maximum de protéines possibles, qui étaient présentes pour au moins la moitié des espèces de notre liste. Les données génétiques utilisées étaient issues de la base de données GenBank. Dans notre cas, cela a abouti à une collecte d'un ensemble de 21 protéines. Enfin, nous avons présenté l'application de notre algorithme sur ce jeu de données réelles, et avons examiné parallèlement les performances de notre programme.

Nos résultats permettent de mettre en évidence deux protéines, SRY et NADH-6. La protéine SRY, d'une longueur de 396 aa, contrôle la détermination du sexe sur la région Y, alors que la protéine NADH-6, d'une longueur de 175, participe activement à la production de l'énergie dans les cellules. Ces protéines, comme nous l'avons indiqué au chapitre IV, sont pertinentes d'un point de vue de l'influence de l'environnement sur l'activation de leurs gènes respectifs. En effet, nous constatons que nos résultats coïncident avec les changements climatiques du milieu.

Nous indiquerons ici quelques pistes pour les développements futurs.

- Concernant la performance et l'efficacité de l'algorithme :

  1. Notre algorithme ne stocke que la première meilleure position de l'alignement à la fois. Si plusieurs cas d'égalité du meilleur score se présentent, alors nous sélectionnons la position correspondant en plus au meilleur bootstrap moyen. Dans l'éventualité où plusieurs fragments d'alignement fournissaient le meilleur score, notre algorithme ne prenait en compte que la première position rencontrée. Il serait donc intéressant de conserver tous les meilleurs résultats dans une structure de tableau. En connaissant toutes les positions pertinentes pour la même taille de la fenêtre coulissante, nous pourrions alors calculer la fréquence de chaque meilleure position pour l'ensemble des tailles de fenêtres considérées.

  2. Notre algorithme pourrait être amélioré de sorte qu'il prendrait en compte tous les paramètres environnementaux disponibles. Actuellement notre algorithme n'a pas cette option. En réalisant notre étude avec plusieurs paramètres simultanés, nous pourrions prendre en compte différents paramètres environnementaux lors d'une même exécution. On peut effectuer cette analyse en examinant par exemple toutes les bipartitions majoritaires dans les arbres de référence. Cependant, nous pouvons déplacer le problème au stade de la construction des arbres de référence. En effet, si lors de la reconstruction des arbres de référence, nous parvenons à intégrer ces différents paramètres à l'aide d'un arbre consensus, alors la complexité du programme restera la même.

  3. Notons que plusieurs programmes du paquet PHYLIP ont été parallélisés par différentes équipes de chercheurs (Ropelewski, Nicholas et Mendez, 2010). Dans ce mémoire de maîtrise, nous n'avons pas utilisé les programmes parallèles, car l'optimisation de la vitesse d'exécution n'était pas notre premier objectif. Dans le futur, il serait intéressant de paralléliser notre algorithme.

La parallélisation rendrait notre algorithme applicable à des données génomiques de grandes tailles.

- Concernant l'analyse des résultats :

  1. Pour permettre une analyse plus approfondie de ce sujet, il serait intéressant de mettre les résultats obtenus, notamment les valeurs obtenues des meilleures positions des alignements de séquences multiples, en lien avec la structure dimensionnelle des protéines, ou bien en lien avec la carte de la pression sélective exercée sur les fragments d'alignement indiqués.

  2. Nous pouvons envisager une étude qui consisterait à ne sélectionner que différents phénotypes d'une seule espèce, par exemple *Homo Sapiens*, à travers différents lieux géographiques. Dans ce cas, il faudrait considérer un plus grand territoire géographique afin d'augmenter significativement la variation des paramètres climatiques sélectionnés. Une telle étude consisterait à observer l'évolution des gènes de l'espèce sélectionnée en fonction de différents paramètres climatiques.

# APPENDICE A

# PROGRAMME JAVA

Cet appendice contient les principales fonctions du programme Java permettant de détecter le fragment d'un alignement de séquences multiples correspondant à la plus petite distance de Robinson et Foulds (RF) (i.e., la similitude topologique entre les arbres comparés). Dans le cas où plusieurs fenêtres donnent la même valeur de la distance RF, la sélection se poursuit sur le bootstrap moyen qui doit être le plus élevé possible (i.e., meilleur support statistique de l'arbre).

```
/*******************************************************
* Établissement : Université du Québec a Montréal *
* @author : Nadia Tahiri *
* @version : 2012 *
*******************************************************/
```

## A.1 La classe des paramètres des programmes du paquet PHYLIP

```
1 import java.io.File;
2 import java.io.FileWriter;
3 import java.io.IOException;
4 import java.util.Scanner;
5
6
7 public class FilesInputPhylip {
8
```

```
9   /*************************************************
10   * VARIABLES                                     *
11   *************************************************/
12   FileWriter fileInputSB = new FileWriter("executable/Inputs/inputSB");
13   FileWriter fileInputProtD = new FileWriter("executable/Inputs/
         inputProtD");
14   FileWriter fileInputNJ = new FileWriter("executable/Inputs/inputNJ");
15   FileWriter fileInputCs = new FileWriter("executable/Inputs/inputCs");
16
17  /*************************************************
18   * Cette fonction permet de préparer le fichier  *
19   * contenant les paramètres d'entrée de Seqboot  *
20   *************************************************/
21  public void InputSB (String nomFolder, int nb_trees){
22      try{
23          fileInputSB.write(nomFolder);
24          fileInputSB.write("\nY");
25          fileInputSB.write("\n73");
26          fileInputSB.close();
27      }catch(Exception e){
28          System.out.println("Erreur fichier d'entrée de SeqBoot.");
29      }
30  }
31
32  /*************************************************
33   * Cette fonction permet de préparer le fichier  *
34   * contenant les paramètres d'entrée de ProtDist *
35   *************************************************/
36  public void InputProtD (String nomFolder, String pasWin, int nb_trees)
        {
37      try{
38          fileInputProtD.write("Result/Phylip/"+nomFolder+"/"+pasWin+"/
              outfileSB");
39          fileInputProtD.write("\nP");
40          fileInputProtD.write("\nP");
41          fileInputProtD.write("\nP");
```

```
42              fileInputProtD.write("\nM");
43              fileInputProtD.write("\nD");
44              fileInputProtD.write("\n100");
45              fileInputProtD.write("\nY");
46              fileInputProtD.close();
47          }catch(Exception e){
48              System.out.println("Erreur fichier d'entrée de Protdist.");
49          }
50      }
51
52      /*************************************************
53       * Cette fonction permet de préparer le fichier    *
54       * contenant les paramètres d'entrée de NJ         *
55       *************************************************/
56      public void InputNJ (String nomFolder, String pasWin, int nb_trees){
57          try{
58              fileInputNJ.write("Result/Phylip/"+nomFolder+"/"+pasWin+"/
                    outfileProtD");
59              fileInputNJ.write("\nM");
60              fileInputNJ.write("\n100");
61              fileInputNJ.write("\n73");
62              fileInputNJ.write("\nY");
63              fileInputNJ.close();
64          }catch(Exception e){
65              System.out.println("Erreur fichier d'entrée de Neighbor.");
66          }
67      }
68
69      /*************************************************
70       * Cette fonction permet de préparer le fichier,   *
71       * contenant les paramètres d'entrée de Consense   *
72       *************************************************/
73      public void InputCs (String nomFolder, String pasWin, int nb_trees){
74          try{
75              fileInputCs.write("Result/Phylip/"+nomFolder+"/"+pasWin+"/
                    outtreeNJ");
```

```
76              fileInputCs.write("\nY");
77              fileInputCs.close();
78          }catch(Exception e){
79              System.out.println("Erreur fichier d'entrée de Consence.");
80          }
81      }
82  }
```

## A.2 La classe des programmes du paquet PHYLIP

```
1  import java.io.BufferedReader;
2  import java.io.BufferedWriter;
3  import java.io.File;
4  import java.io.FileInputStream;
5  import java.io.FileWriter;
6  import java.io.IOException;
7  import java.io.InputStream;
8  import java.io.InputStreamReader;
9  import java.io.OutputStreamWriter;
10 import java.io.Writer;
11 import java.util.Scanner;
12
13
14 public class Phylip {
15
16     public static String current=(new File("").getAbsolutePath());
17     public  File repertoire=new File(current);
18
19
20      /*************************************************
21      * Copie le fichier source dans le fichier résultat *
22      * retourne vrai si cela réussit                    *
23      *************************************************/
24     public static boolean Copy(boolean modeAjout, String nomDossier,String
           source, String dest, String pasWin){
25         try{
```

```
26              FileWriter destFile = new FileWriter(current+"/Result/Phylip/"
                    +nomDossier+"/"+pasWin+"/"+dest,modeAjout);
27              InputStream ips=new FileInputStream(source);
28              InputStreamReader ipsr=new InputStreamReader(ips);
29              BufferedReader br=new BufferedReader(ipsr);
30              String ligne;
31
32              while ((ligne=br.readLine())!=null){
33                  destFile.write(ligne);
34                  destFile.write("\n");
35              }
36
37              destFile.close();
38              br.close();
39          }catch (Exception e){
40              return false; // Erreur
41          }
42          return true; // Résultat OK
43      }
44
45
46      public boolean createCommandFile(String filename, String commandeLine)
            {
47          try{
48              BufferedWriter command_file=new BufferedWriter(new FileWriter(
                    new File(filename)));
49              String command=commandeLine;
50              command_file.append(command);
51              command_file.flush();
52              command_file.close();
53          }catch(Exception e){
54              return false;
55          }
56          return true;
57      }
58
```

```
59      /*************************************************
60       * Cette fonction permet d'exécuter Seqboot      *
61       *************************************************/
62      public void SeqBoot(String NameGene, String pasWin, int nb_trees){
63          try{
64              StringBuilder st=new StringBuilder();
65              Runtime runtime = Runtime.getRuntime();
66              File fi=new File("");
67              createCommandFile("executeSeqboot", "#!/bin/sh\nexecutable/
                    seqboot <"+fi.getAbsolutePath()+"/executable/Inputs/inputSB
                     >sortieSB\n");
68              runtime.exec("chmod +x executeSeqboot");
69              String[] command=new String[5];
70
71              for (int i=0; i<command.length;i++) command[i]="";
72
73              //Execute the Makefile_test.command.sh
74              File f=new File("");
75              command[0]=f.getAbsolutePath()+"/executeSeqboot";
76              Process p = runtime.exec(command);
77              InputStream stdoutput = p.getInputStream();
78              InputStream stderr = p.getErrorStream();
79              InputStreamReader isr = new InputStreamReader(stdoutput);
80              InputStreamReader isr2 = new InputStreamReader(stderr);
81              BufferedReader br = new BufferedReader(isr);
82              BufferedReader br2 = new BufferedReader(isr2);
83              String line = null;
84
85              while ( (line = br.readLine()) != null){
86                  st.append(line+"\n");
87              }
88
89              p.destroy();
90          }catch (Exception ex){
91              System.out.println(" error unable to execute the test... Error
                     message follow:");
```

```
92          }
93
94          Writer out = new BufferedWriter(new OutputStreamWriter(System.out)
                ;
95          File fwFile = new File(current+"/Result/Phylip");
96          fwFile.mkdir();
97          File fwFileT = new File(current+"/Result/Phylip/"+NameGene);
98          fwFileT.mkdir();
99          File fwFileCompt = new File(current+"/Result/Phylip/"+NameGene+"/"
                +pasWin);
100         fwFileCompt.mkdir();
101         Copy(false, NameGene, "outfile", "outfileSB", pasWin);
102         File file = new File("outfile");
103         file.delete();
104         File fileSortie = new File("sortieSB");
105         fileSortie.delete();
106     }
107
108
109     /*****************************************************
110      * Remplace les nan dans les matrices par 0,000005   *
111      * et retourne vrai si cela réussit                  *
112      *****************************************************/
113     public void NanToSeuil (String nomDossier, String pasWin){
114         File fi=new File("");
115         try{
116             StringBuilder st=new StringBuilder();
117             Runtime runtime = Runtime.getRuntime();
118
119             createCommandFile("executeSed", "#!/bin/sh\nsed \"s/        nan
                    /0.000005/g\" "+fi.getAbsolutePath()+"/Result/Phylip/"+
                    nomDossier+"/"+pasWin+"/outfileProtD >  "+fi.
                    getAbsolutePath()+"/Result/Phylip/"+nomDossier+"/"+pasWin+"
                    /resuls.txt\n");
120             runtime.exec("chmod +x executeSed");
121             String[] command=new String[5];
```

```
122
123             for (int i=0; i<command.length;i++) command[i]="";
124
125             //Exécute the Makefile_test.command.sh
126             File f=new File("");
127             command[0]=f.getAbsolutePath()+"/executeSed";
128             Process p = runtime.exec(command);
129             InputStream stdoutput = p.getInputStream();
130             InputStream stderr = p.getErrorStream();
131             InputStreamReader isr = new InputStreamReader(stdoutput);
132             InputStreamReader isr2 = new InputStreamReader(stderr);
133             BufferedReader br = new BufferedReader(isr);
134             BufferedReader br2 = new BufferedReader(isr2);
135             String line = null;
136
137             while ( (line = br.readLine()) != null){
138                 st.append(line+"\n");
139             }
140
141             p.destroy();
142         }catch (Exception ex){
143             System.out.println(" error unable to execute the test... Error
                    message follow:");
144         }
145         Writer out = new BufferedWriter(new OutputStreamWriter(System.out)
                );
146         Copy(false, nomDossier, fi.getAbsolutePath()+"/Result/Phylip/"+
                nomDossier+"/"+pasWin+"/resuls.txt", "outfileProtD", pasWin);
147         File file = new File(fi.getAbsolutePath()+"/Result/Phylip/"+
                nomDossier+"/"+pasWin+"/resuls.txt");
148         file.delete();
149     }
150
151 }
```

## A.3 La classe de gestion des nœud des arbres

```
1 import java.io.File;
2 import java.io.FileReader;
3 import java.io.FileWriter;
4 import java.io.IOException;
5
6
7 public class NodeTree {
8
9     /*****************************************
10     * Récupère tous les noms des espèces     *
11     * à partir d'un arbre au format Newick.  *
12     *****************************************/
13    public int RecupererNomSpecies(String nomFileTree, int fileTree){
14        int nb_spe=0;
15        try{
16            /**
17            * Pour vider le contenu du fichier Output.txt au préalable.
18            */
19            FileWriter outFile = new FileWriter("tmp/Output_"+fileTree);
20            outFile.write("");
21            outFile.close();
22            File fichier= new File(nomFileTree);
23            outFile = new FileWriter("tmp/Output_"+fileTree,true);
24            FileReader flotLecture = new FileReader(fichier);
25            long longueurFichier= fichier.length();
26            int dejaLu = 0;
27            char car=0;
28            String nomSpecies="";
29
30            while (dejaLu < longueurFichier){
31                car= (char)flotLecture.read();
32                dejaLu = dejaLu + 1;
33                if(!(car=='(' || car==')' || car==',' || car==':' || car==
                      ';'|| car=='.'|| car=='0' || car=='1' || car=='2' ||
                      car=='3' || car=='4' || car=='5' || car=='6' || car=='7
                      ' || car=='8' || car=='9')){
```

```
34                      if (car=='_'){
35                          nb_spe++;
36                      }
37                      nomSpecies+=car;
38                  }else{
39                      if((car==':')){
40                          if(!nomSpecies.equalsIgnoreCase("")){
41                              outFile.write(nomSpecies);
42                              outFile.write("\n");
43                              nomSpecies="";
44                          }
45                      }
46                  }
47              }
48              outFile.close();
49              flotLecture.close();
50              return nb_spe;
51          }catch (Exception e){
52              System.out.println("Erreur lors de la lecture de l'arbre.");
53          }
54          return nb_spe;
55      }
56
57  }
```

## A.4 La classe permettant le calcul de la distance de Robinson et Foulds

```
1  import java.io.BufferedReader;
2  import java.io.IOException;
3  import java.io.InputStream;
4  import java.io.InputStreamReader;
5  import java.io.File;
6  import java.io.FileWriter;
7  import java.io.InputStream;
8  import java.io.FileInputStream;
9  import java.util.Scanner;
10
```

```
11
12 public class HGT {
13
14     /*****************************************
15      * Copie le fichier source dans          *
16      * le fichier résultat                   *
17      * Retourne vrai si cela réussit         *
18      *****************************************/
19     public static boolean Copy(boolean modeAjout, String source, String
           dest){
20         try{
21             File fichierConsense = new File ("");
22             FileWriter destFile = new FileWriter(fichierConsense.
                   getAbsolutePath()+"/Result/Trees/"+dest, modeAjout);
23
24             InputStream ips=new FileInputStream(source);
25             InputStreamReader ipsr=new InputStreamReader(ips);
26             BufferedReader br=new BufferedReader(ipsr);
27             String ligne;
28
29             while ((ligne=br.readLine())!=null){
30                 destFile.write(ligne);
31                 destFile.write("\n");
32             }
33
34             destFile.close();
35             br.close();
36
37         }catch (Exception e){
38             return false; // Erreur
39         }
40         return true; // Resultat OK
41     }
42
43     /*************************************************
44      * Cette fonction permet de calculer la distance RF *
```

```
45      ************************************************/
46      public boolean CriterionOrHGT(String fichier, String criterionOrHgt,
            String fileSortie){
47          boolean error = false ;
48          try{
49              String line;
50              StringBuilder st=new StringBuilder();
51              Runtime runtime = Runtime.getRuntime();
52              Process process = runtime.exec("executable/"+criterionOrHgt+"
                    -inputfile=Result/Trees/"+fichier.substring(0,fichier.
                    length()-4)+"/"+fichier+" -outputfile=Result/Trees/"+
                    fichier.substring(0,fichier.length()-4)+"/"+fileSortie);
53
54              InputStream stdoutput = process.getInputStream();
55              InputStream stderr = process.getErrorStream();
56              InputStreamReader isr = new InputStreamReader(stdoutput);
57              InputStreamReader isr2 = new InputStreamReader(stderr);
58              BufferedReader br = new BufferedReader(isr);
59              BufferedReader br2 = new BufferedReader(isr2);
60
61              while ( (line = br.readLine()) != null){
62                  if (line.equalsIgnoreCase("Use valgrind or gdb to fix the
                        problem")){
63                      error = true;
64                  }
65                  st.append(line+"\n");
66              }
67
68              int exitVal = process.waitFor();
69              //--Error System
70              if (exitVal!=0){
71                  System.out.println("Criterion error "+exitVal);
72                  while ( (line = br2.readLine()) != null){
73                      st.append(line+"\n");
74                  }
75              }
```

```
76              process.destroy();
77          }catch (Exception ex){
78              System.out.println(" error unable to execute the test... Error
                    message follow:");
79              ex.printStackTrace();
80          }
81
82          if (criterionOrHgt.equalsIgnoreCase("hgt")){
83              File fichierConsense=new File("");
84              StringBuilder st=new StringBuilder();
85              Runtime runtime = Runtime.getRuntime();
86              File fi=new File("");
87              Copy (false,"results.txt", fichier.substring(0,fichier.length
                    ()-4)+"/outputConsence.txt");
88              runtime.exec("chmod +x "+fichierConsense.getAbsolutePath()+"/
                    Result/Trees/"+fichier.substring(0,fichier.length()-4)+"/
                    outputConsence.txt");
89              File file = new File(fichierConsense.getAbsolutePath()+"
                    results.txt");
90              file.delete();
91          }
92
93          return error;
94      }
95  }
```

## A.5 La classe Main

```
1   /**
2    * Inport de librairies
3    */
4  import java.io.BufferedReader;
5  import java.io.File;
6  import java.io.FileInputStream;
7  import java.io.FileNotFoundException;
8  import java.io.FileWriter;
9  import java.io.IOException;
```

```
10 import java.io.InputStream;
11 import java.io.InputStreamReader;
12 import java.util.ArrayList;
13 import java.util.NoSuchElementException;
14 import java.util.Scanner;
15 import java.util.Date;
16 import java.io.BufferedWriter;
17
18 public class Main {
19
20     /************************
21      * Variables globales   *
22      ************************/
23     public static String [] listefichiers;
24     public static String [] listeFiles;
25     public static int nbFiles=0;
26     public static int nbEspeces=0;
27     public static int longueurSequence=0;
28     public static int longueurMaxNomEspece=0;
29     public static ArrayList<String> noms_Especes=new ArrayList<String>();
30     public static String current=(new File("").getAbsolutePath());
31     public  File repertoire=new File(current);
32     public  File repertoireData=new File(current+"\\data_alignement");
33     public static float moyBoostrapUtilisateur=0;
34     public static float valSeuilRF=0;
35     public static int tab_tailleWindows [];
36     public static int tailleWindows = 0;
37     public static int pasWindows=0;
38     public static String [] nomFileTree;
39     public static int nb_SpeOutput =0;
40     public static int nb_Spe [];
41     public static boolean tab_booleanAfichage = false;
42     public static boolean tab_booleanAfichageConsence = false;
43     public static int indexValeurRef [][][][];
44     public static double RFmin = Float.MAX_VALUE;
45     public static double boostrapMax = Float.MIN_VALUE;
```

```
46      public static int numero_Windows_enCours = 0;
47      public static String [] nomFiles;
48      public static int longueurNomGenePlusLong=0;
49
50      /*********************************************************************
51       * Présence d'une espèce dans un tableau                             *
52       * @params input: un élément String + un tableau de String           *
53       * @params output : true si l'élément cherché est trouve sinon false.*
54       *********************************************************************/
55      public static boolean AppartenanceTab(String nomEspeces,String []
            tabNomEspeces){
56          boolean presence = false;
57
58          for (int parc=0;parc<tabNomEspeces.length;parc++){
59              if(tabNomEspeces[parc].equalsIgnoreCase(nomEspeces)){
60                  presence=true;
61                  //Conditon de sortie
62                  parc=tabNomEspeces.length+1;
63              }
64          }
65          return presence;
66      }
67
68
69      /*********************************************************************
70       * Permet de récupérer le nombre d'espèces                           *
71       * et la longueur des alignements de séquences.                      *
72       * (détermine la taille du nom de l'espèce le plus long)             *
73       * @params input : fichier d'ASM.                                    *
74       *********************************************************************/
75      public static void Comptage(File fichierEntree)throws IOException{
76          try{
77              Scanner sc = new Scanner (fichierEntree);
78              nbEspeces=sc.nextInt();
79              longueurSequence=sc.nextInt();
80              int []longeurMaxEspece=new int[nbEspeces];
```

```
81              int compteur=-1;
82              longueurMaxNomEspece=0;
83
84              while (sc.hasNextLine()){
85                  compteur++;
86                  String ligne1=sc.next();
87                  String ligne2=sc.next();
88
89                   longeurMaxEspece[compteur]=ligne1.length();
90                   if(longueurMaxNomEspece<longeurMaxEspece[compteur]){
91                       longueurMaxNomEspece=longeurMaxEspece[compteur];
92                   }
93               }
94               sc.close();
95          }catch (Exception exception){
96                System.out.println("ERROR");
97          }
98
99      }
100
101
102     /*****************************************************************
103     * Pré traitement des ASM de chaque fenêtre                       *
104     * @params output ensemble des fichiers ASM de chaque fenêtre     *
105     * au format Phylip                                              *
106     *****************************************************************/
107     public static ArrayList<String> WindowsTrees(String nomFile,String
            fichierEntree, int tailleWindows,int pasWindows) throws IOException
            {
108         ArrayList<String> noms_fichiers=new ArrayList<String>();
109         try{
110             File Input=new File (fichierEntree);
111             Comptage(Input);
112             boolean [] especesGap = new boolean [nbEspeces];
113             boolean auMoinsUnChar;
114             int comptageNbEspeces;
```

```
115             int nbEspecesReevalue = nbEspeces;
116             //Création des répertoires Result et Tmp
117             File fbR = new File(current+"/Result");
118             fbR.mkdir();
119             File fbt = new File(current+"/tmp");
120             fbt.mkdir();
121             File fbF = new File(current+"/Result/"+nomFile);
122             fbF.mkdir();
123
124             //Préparation des fichiers qui contiendront
125             //les alignements des différentes fenêtres.
126             for (int i=longueurMaxNomEspece+1;i<=longueurSequence+
                    longueurMaxNomEspece+1-tailleWindows;i+=pasWindows){
127                 //Remettre de compteur a zéro pour chaque fenêtre
128                 comptageNbEspeces = 0;
129                 //Création des dossiers pour chaque gène
130                 File fwFile = new File(current+"/Result/"+nomFile+"/"+
                        tailleWindows+"_"+(i-longueurMaxNomEspece));
131                 fwFile.mkdir();
132                 noms_fichiers.add("Result/"+nomFile+"/"+tailleWindows+"_"
                        +(i-longueurMaxNomEspece)+"/"+nomFile+"_"+tailleWindows
                        +"_"+(i-longueurMaxNomEspece));
133                 FileWriter fastaFile = new FileWriter(current+"/Result/"+
                        nomFile+"/"+tailleWindows+"_"+(i-longueurMaxNomEspece)+
                        "/"+nomFile+"_"+tailleWindows+"_"+(i-
                        longueurMaxNomEspece));
134                 //Création du fichier temporaire
135                 FileWriter tempory = new FileWriter(current+"/Result/
                        tempory");
136
137
138                 //Permet d'indiquer la bonne taille de l'alignement
139                 if ((tailleWindows+i)<(longueurSequence+
                        longueurMaxNomEspece+1)){
140                     fastaFile.write(nbEspeces+"\t"+tailleWindows);
141                     tempory.write(nbEspeces+"\t"+tailleWindows);
```

```
142                 }else{
143                     fastaFile.write(nbEspeces+"\t"+(longueurSequence+
                            longueurMaxNomEspece+1-i));
144                     tempory.write(nbEspeces+"\t"+(longueurSequence+
                            longueurMaxNomEspece+1-i));
145                 }
146
147                 //lecture du fichier texte
148                 InputStream ips=new FileInputStream(fichierEntree);
149                 InputStreamReader ipsr=new InputStreamReader(ips);
150                 BufferedReader br=new BufferedReader(ipsr);
151                 String ligne;
152                 ligne=br.readLine();
153
154                 while ((ligne=br.readLine())!=null){
155                     //Initialisation de la variable a false
156                     //pour chaque espece!
157                     auMoinsUnChar = false;
158                     fastaFile.write("\n");
159                     tempory.write("\n");
160                     //Copie le nom des espèces
161                     for (int deb=0;deb<longueurMaxNomEspece;deb++){
162                         fastaFile.write(ligne.charAt(deb));
163                         tempory.write(ligne.charAt(deb));
164                     }
165                     fastaFile.write(" ");
166                     tempory.write(" ");
167                     //Copie la séquence
168                     if ((tailleWindows+i)<=(longueurSequence+
                            longueurMaxNomEspece+1)){
169                         for (int j=i;j<i+tailleWindows;j++){
170                             fastaFile.write(ligne.charAt(j));
171                             tempory.write(ligne.charAt(j));
172
173                             if(ligne.charAt(j)!='-'){
174                                 auMoinsUnChar = true;
```

```
175                         }
176                     }
177                     if(!auMoinsUnChar){
178                         nbEspecesReevalue-=1;
179                         especesGap[comptageNbEspeces]=true;
180                     }
181                 }else{
182                     int ind=i;
183                     while (ind<(longueurSequence+longueurMaxNomEspece
                            +1)){
184                         fastaFile.write(ligne.charAt(ind));
185                         tempory.write(ligne.charAt(ind));
186                         ind++;
187                     }
188                 }
189                 comptageNbEspeces++;
190             }
191         br.close();
192         tempory.close();
193         fastaFile.close();
194     }
195
196   }catch (Exception exception){}
197         return noms_fichiers;
198 }
199
200
201
202   /**********************************************************************
203    * Copie le fichier source dans le dossier Result.                    *
204    * Retourne vrai si cela réussit.                                     *
205    **********************************************************************/
206 public static boolean copyFile(boolean modeAjout,String source, String
        dest){
207     try{
```

```
208             File fwFileT = new File(current+"/Result/Trees/"+dest.
                    substring(0,dest.length()-4));
209             fwFileT.mkdir();
210             FileWriter destFile = new FileWriter(current+"/Result/Trees/"+
                    dest.substring(0,dest.length()-4)+"/"+dest,modeAjout);
211             InputStream ips=new FileInputStream(source);
212             InputStreamReader ipsr=new InputStreamReader(ips);
213             BufferedReader br=new BufferedReader(ipsr);
214             String ligne;
215
216             while ((ligne=br.readLine())!=null){
217                 destFile.write(ligne);
218                 destFile.write("\n");
219             }
220
221             destFile.close();
222             br.close();
223         }catch (Exception e){
224             return false; // Erreur
225         }
226       return true; // Resultat OK
227
228   }
229
230
231    /**********************************************************************
232    * Permet de récupérer la distance RF du fichier output               *
233    * du programme calculant la distance RF.                             *
234    * @params input : nb d'espèces identiques entre les deux arbres      *
235    *               : fichier output du programme RF                     *
236    * @params output: distance RF normalisée.                            *
237    **********************************************************************/
238    public static double RecuperationRF (String signe, String nomFile,
           int nbEspecesIdentiques) throws FileNotFoundException, IOException
           {
239        double RFnormalise = -1;
```

```
240         File fi = new File ("");
241         String outFile=fi.getAbsolutePath()+"/Result/Trees/"+nomFile;
242         File out = new File (outFile);
243         Scanner scanna= new Scanner(out);
244         String ligResult=scanna.next();
245         String []tabLigResult=new String[3];
246         tabLigResult=ligResult.split(signe);
247         double RF = Double.parseDouble(tabLigResult[0]);
248 
249         if(nbEspecesIdentiques!=3){
250             RFnormalise = (RF/(2*nbEspecesIdentiques-6))*100;
251         }else{
252             System.out.print("Erreur: Division par 0!\n");
253         }
254 
255         scanna.close();
256         return RFnormalise;
257     }
258 
259 
260      /*********************************************************************
261       * Permet de retourner le nombre d'espèces en commun                 *
262       * entre l'arbre de référence et les alignements.                    *
263       * @params output: n indiquant le nombre d'espèces identiques        *
264       *********************************************************************/
265     public static int NbEpseceEnCommun ( int nb_SpeAlignement, int de)
            throws FileNotFoundException{
266         int n = 0;  //nb d'espèces identiques qux deux arbres
267         //Remplissage des noms d'espèces de l'arbre étudié.
268         File f1 = new File ("tmp/Output_0");
269         Scanner sca = new Scanner (f1);
270         String []nomEspecesTree = new String [nb_SpeAlignement];
271         int cmp =0;
272 
273         while (sca.hasNext()){
274             nomEspecesTree[cmp]=sca.next();
```

```
275             cmp ++;
276         }
277
278         //Fermeture des fichiers
279         sca.close();
280
281         /*Ouverture du fichier contenant
282           la liste des noms d'espèces de l'arbre géographiques.*/
283         File f2 = new File ("tmp/Output_"+(de+1));
284         Scanner scannage = new Scanner (f2);
285
286         while (scannage.hasNext()){
287             if(AppartenanceTab(scannage.next(), nomEspecesTree)){
288                 n++;
289             }
290         }
291         return n;
292     }
293
294
295      /*****************************************************************
296      * Construction des arbres pour chaque fenêtre.                   *
297      * Puis, des calculs des différentes distances RF entre l'arbre des *
298      * fragments des ASM et les arbres de référence.                  *
299      *****************************************************************/
300     public static void Run_Process (int numeroGene, String nomFile,
            boolean windowsComplet) throws IOException, InterruptedException{
301         File fi = new File("");
302         Scanner kb = new Scanner(System.in);
303         boolean auMoinsUnArbre = false;
304         boolean auMoinsUnArbre_G = false;
305         //Conservation des indices que nous allons conserver.
306         ArrayList<Integer> indicesConservees = new ArrayList<Integer>();
307         ArrayList<Integer> indices = new ArrayList<Integer>();
308         ArrayList<String> noms_fichiers = new ArrayList<String>();
309         HGT hgt = new HGT ();
```

```
310         Boostrap boostrap = new Boostrap();
311         NodeTree node = new NodeTree ();
312         int nb_SpeAlignement = 0;
313         String TabIndex [];
314         int lenght=0;
315         int n[] = new int[nomFileTree.length];
316         //stocke le nombre des fenêtres a traiter.
317         int nb_windows=0;
318         float [] moyBoostrapArbre=new float[nb_windows+1];
319         float [] moyBoostrapArbreConsence=new float[nb_windows+1];
320         boolean [] hgtOrCriterion = new boolean [nb_windows+1];
321         double rfConsence [][] = new double [nb_windows+1][nomFileTree.
                length+1];
322         double rf [][] = new double [nb_windows+1][nomFileTree.length+1];
323
324         //Crée un dossier Trees sous le répertoire Result.
325         File fwFile = new File(current+"/Result/Trees");
326         fwFile.mkdir();
327         String pasWin="";
328         boolean trouve2 = true;
329         boolean trouve = true;
330
331         /*
332         * Récupère différents fichiers contenant
333         * les fenêtres des alignements au format Phylip
334         * dans un tableau nom_fichiers.
335         */
336
337         if (windowsComplet){
338             noms_fichiers = WindowsCompletTrees(nomFile,"data_alignement/"
                    +nomFile,tailleWindows,pasWindows);
339             nb_windows=1;
340         }else{
341             noms_fichiers = WindowsTrees(nomFile,"data_alignement/"+-
                    nomFile,tailleWindows,pasWindows);
342
```

```
343             //La taille de la nouvelle sous séquence
344             int longueurSequenceMoinsPas=longueurSequence;
345             while(tailleWindows<=longueurSequenceMoinsPas){
346                 nb_windows++;
347                 longueurSequenceMoinsPas-=pasWindows;
348             }
349         }
350
351         for (int fileTree=0; fileTree<nomFileTree.length; fileTree++){
352             RFmin = Float.MAX_VALUE;
353             boostrapMax = Float.MIN_VALUE;
354             //Traitement pour l'ensemble des fenêtres
355             // et plus un pour l'alignement au complet.
356             for (int de=0;de<(nb_windows);de++){
357                 if (trouve2){
358                     indices.add(de+1);
359                 }
360                 auMoinsUnArbre_G = true;
361
362                 //Récupère l'index a partir du nom du fichier
363                 TabIndex = noms_fichiers.get(de).split("_");
364                 lenght = TabIndex.length;
365
366                 /*****************************************
367                 *   DÉBUT DES PROGRAMMES DU PAQUET PHYLIP   *
368                 *****************************************/
369                 pasWin=""+(de+1);
370                 //création des input pour les exécutions du paquet PHYLIP.
371
372                 FilesInputPhylip fileInput = new FilesInputPhylip();
373                 fileInput.InputSB(noms_fichiers.get(de), nomFileTree.
                        length);
374                 fileInput.InputProtD(nomFile, pasWin, nomFileTree.length);
375                 fileInput.InputNJ(nomFile, pasWin, nomFileTree.length);
376                 fileInput.InputCs(nomFile, pasWin, nomFileTree.length);
377
```

```
378                 //Exécute des executables du paquet PHYLIP
379                 Phylip phylip = new Phylip();
380                 phylip.SeqBoot(nomFile, pasWin, nomFileTree.length);
381                 phylip.ProtDist(nomFile, pasWin, nomFileTree.length);
382                 phylip.Neighbor(nomFile, pasWin, nomFileTree.length);
383                 phylip.Consence(nomFile, pasWin, nomFileTree.length);
384
385                 //Formater correctement la sortie du Tree Consence
386                 fileInput.FileCs(nomFile, pasWin, nomFileTree.length);
387
388                 /*Copie du premier arbre Tree du paramètre d'entrée,
389                   afin de préparer le fichier d'entrée de la
390                   fonction Criterion*/
391                 if (copyFile(false,"data_alignement/"+nomFileTree[fileTree
                        ], nomFile+"Cs_"+(de+1)+"_"+(fileTree+1)+".txt")){
392                     System.out.print("La copie de votre fichier "+
                            nomFileTree[fileTree]+" vers "+nomFile+"Cs_"+(de+1)
                            +"_"+(fileTree+1)+".txt"+" a été correctement
                            effectuee\n");
393                 }else{
394                     System.out.print("Une erreur c'est produite lors de la
                             retranscription de votre fichier "+nomFileTree[
                            fileTree]+"\n");
395                 }
396
397                 /* Copie du deuxième arbre issu de l'exécution de NJ,
398                    afin de préparer le fichier d'entrée de
399                    la fonction Criterion*/
400                 nb_SpeAlignement = node.RecupererNomSpecies("Result/Phylip
                        /"+nomFile+"/"+pasWin+"/outtreeConsences", 0);
401                 if (copyFile(true,"Result/Phylip/"+nomFile+"/"+pasWin+"/
                        outtreeConsences", nomFile+"Cs_"+(de+1)+"_"+(fileTree
                        +1)+".txt")){
402                     System.out.print("La copie de votre fichier "+"Result/
                            Phylip/"+nomFile+"/"+pasWin+"/outtreeConsences vers
                             "+nomFile+"Cs_"+(de+1)+"_"+(fileTree+1)+".txt"+" a
```

```
                été correctement effectuée\n");
403         }else{
404             System.out.print("Une erreur c'est produite lors de la
                    retranscription de votre fichier Result/Phylip/"+
                nomFile+"/"+pasWin+"/outtreeConsences\n");
405         }
406
407         /*appel de la fonction bootstrap afin de calculer le
408           bootstrap moyen de l'arbre issu de Consences*/
409         moyBoostrapArbreConsence[de]=boostrap.
                RecupererMoyenneBoostrapConsence("Result/Phylip/"+
                nomFile+"/"+pasWin+"/outtreeConsences");
410         //Realise l'execution de RF ou HGT
411         boolean error = hgt.CriterionOrHGT(nomFile+"Cs_"+(de+1)+"_
                "+(fileTree+1)+".txt", "hgt", "outputConsence.txt");
412
413         /* Appel de la fonction NbEpseceEnCommun(),
414            retournant le nombre d'especes en commun
415            des 2 arbres (variable n) + 1 pour le root qui est
416            commun entre les deux arbres*/
417            n [fileTree]= NbEpseceEnCommun(nb_SpeAlignement,
                   fileTree)+1;
418         if (error){
419         boolean error2 = hgt.CriterionOrHGT (nomFile+"Cs_"+(de+1)+
                "_"+(fileTree+1)+".txt", "criterion", "outputConsence.
                txt");
420         rfConsence [de][fileTree] = RecuperationRF ("<>", nomFile+
                "Cs_"+(de+1)+"_"+(fileTree+1)+"/outputConsence.txt", n[
                fileTree]);
421         if (rfConsence [de][fileTree]<RFmin){
422             RFmin=rfConsence [de][fileTree];
423             boostrapMax = moyBoostrapArbreConsence[de];
424             indexValeurRef [numeroGene][numero_Windows_enCours][
                       fileTree] = Integer.parseInt(TabIndex[(lenght-1)]);
425         }else if(rfConsence [de][fileTree]==RFmin &&
                moyBoostrapArbreConsence[de]>boostrapMax){
```

```
426                     boostrapMax = moyBoostrapArbreConsence[de];
427                     indexValeurRef [numeroGene][numero_Windows_enCours][
                            fileTree] = Integer.parseInt(TabIndex[(lenght-1)]);
428                 }
429                 System.out.println("ERROR FOR RF'S CALCUL "+rfConsence[de
                        ][fileTree]);
430                 }
431
432             }
433             trouve2 = false;
434         }
435         //Pour chaque fenêtre, réalisation de PhyMl
436         //Après une pause de 1 seconde
437         Thread.sleep(1000);
438
439
440          /*****************************************
441           *     FIN DES PROGRAMMES DU PAQUET PHYLIP     *
442           *****************************************/
443
444         //Initialisation a true la variable trouve pour chaque fenêtre
445         for (int fileTree=0; fileTree<nomFileTree.length; fileTree++){
446             for (int de=0;de<(nb_windows);de++){
447                 // Appel de la fonction qui calcul la moyenne
448                 // de bootstrap de l'arbre
449                 moyBoostrapArbre[de]=boostrap.RecupererMoyenneBoostrap(
                        noms_fichiers.get(de)+"_phyml_tree.txt")*10;
450
451                 if (moyBoostrapArbreConsence[de]>=moyBoostrapUtilisateur
                        && rfConsence[de][fileTree]<valSeuilRF){
452                     if (moyBoostrapArbre[de]>=moyBoostrapUtilisateur){
453                         /*Évite de mettre plusieurs fois les mêmes
                                postions
454                           dans le tableau indicesConservees.*/
455                         if (trouve){
456                             indicesConservees.add(de+1);
```

```
457                     trouve = false;
458                 }
459
460             /*****************************************
461             *          DÉBUT DU PROGRAMME PhyML          *
462             *****************************************/
463                 //Pour chaque fenêtre, réalisation de phyMl
464                 PhyMl(noms_fichiers.get(de));
465
466                 auMoinsUnArbre=true;
467                 /*Copie du premier arbre Tree du paramètre
468                   d'entrée, afin de préparer le fichier
469                   d'entrée de la fonction Criterion*/
470                 if (copyFile(false,"data_alignement/"+nomFileTree[
                        fileTree], nomFile+"_"+(de+1)+"_"+(fileTree+1)+
                        ".txt")){
471                     System.out.print("La copie de votre fichier "+
                            nomFileTree[fileTree]+" vers "+nomFile+"_"
                            +(de+1)+"_"+(fileTree+1)+".txt"+" a été
                            correctement effectuée\n");
472                 }else{
473                     System.out.print("Une erreur c'est produite
                            lors de la retranscription de votre fichier
                             "+nomFileTree[fileTree]+"\n");
474                 }
475
476                 /*Copie du deuxième arbre issu de l'exécution
477                   de PhyMl, afin de préparer le fichier d'entrée
478                   de la fonction Criterion*/
479                 if (copyFile(true,noms_fichiers.get(de)+"
                        _phyml_tree.txt", nomFile+"_"+(de+1)+"_"+(
                        fileTree+1)+".txt")){
480                     System.out.print("La copie de votre fichier "+
                            noms_fichiers.get(de)+"_phyml_tree.txt"+"
                            vers "+nomFile+"_"+(de+1)+"_"+(fileTree+1)+
                            ".txt"+" a été correctement effectuée\n");
```

```
481                 }else{
482                     System.out.print("Une erreur c'est produite
                            lors de la retranscription de votre fichier
                             "+noms_fichiers.get(de)+"_phyml_tree.txt\n
                            ");
483                 }
484
485                 /*appel de la fonction bootstrap afin de calculer
                        le
486                 bootstrap moyen de l'arbre issu de Consences*/
487                 moyBoostrapArbreConsence[de]=boostrap.
                        RecupererMoyenneBoostrapConsence("Result/Phylip
                        /"+nomFile+"/"+pasWin+"/outtreeConsences");
488
489                 //Exécute de RF
490                 boolean error3 = hgt.CriterionOrHGT(nomFile+"_"+(
                        de+1)+"_"+(fileTree+1)+".txt", "criterion", "
                        output.txt");
491
492                 if (error3){
493                     boolean error4 = hgt.CriterionOrHGT (nomFile+"
                            _"+(de+1)+"_"+(fileTree+1)+".txt", "hgt", "
                            output.txt");
494                     rf[de][fileTree] = RecuperationRF (",",
                            nomFile+"_"+(de+1)+"_"+(fileTree+1)+"/
                            output.txt", n[fileTree]);
495                     System.out.println("ERROR FOR RF'S CALCUL "+
                            rfConsence[de][fileTree]);
496                 }else{
497                     rf [de][fileTree] = RecuperationRF ("<>",
                            nomFile+"_"+(de+1)+"_"+(fileTree+1)+"/
                            output.txt", n[fileTree]);
498                     System.out.println("NO ERROR FOR RF'S CALCUL
                            "+rfConsence[de][fileTree]);
499                     System.out.println("rfConsence "+rfConsence [
                            de][fileTree]);
```

```
500                         }
501                     }
502                 }
503
504             }
505
506         }
507
508         if(auMoinsUnArbre){
509                 //Synthèse des résultats valides
510             if (!tab_booleanAfichageConsence){
511                 AffichageResult("PhyML", rf, n, nomFile, tailleWindows,
                        pasWindows, nb_windows, indicesConservees,
                        moyBoostrapArbre, "output.txt");
512                 tab_booleanAfichageConsence = true;
513             }
514             //Affichage des tableaux résultats pour chaque taille de fenê
                    tre.
515             if(!windowsComplet){//si tableau existe
516                 AffichageTableau("PhyML", rf, n, nomFile, tailleWindows,
                        pasWindows, nb_windows, indicesConservees,
                        moyBoostrapArbre, "output.txt");
517             }
518         }
519
520         if(auMoinsUnArbre_G){
521             //Synthèse de tous les résultats
522             if (!tab_booleanAfichage){
523                 AffichageResult("NJ", rfConsence, n, nomFilé+"Cs",
                        tailleWindows, pasWindows, nb_windows, indices,
                        moyBoostrapArbreConsence, "outputConsence.txt");
524                 tab_booleanAfichage = true;
525             }
526             //Affichage des tableaux résultats pour chaque taille de fenê
                    tre.
527             if(!windowsComplet){//si tableau existe
```

```
528                 AffichageTableau("NJ", rfConsence, n, nomFile+"Cs",
                        tailleWindows, pasWindows, nb_windows, indices,
                        moyBoostrapArbreConsence, "outputConsence.txt");
529             }
530         }
531
532     }
533
534     public static void main(String[] args) throws IOException,
        InterruptedException {
535         try{
536             //Lecture des paramètres d'entrée, depuis un fichier.
537             File parametres = new File(args[0]);
538             Scanner scan = new Scanner(parametres);
539             boolean windowsComplet = true;
540
541             //Data of the File in input
542             int nb_alignemnts=scan.nextInt();
543
544             nomFiles=new String[nb_alignemnts];
545             for (int file=0; file<nomFiles.length;file++){
546                 do{
547                     nomFiles [file]= scan.next();
548                     if (!fichierExiste("data_alignement/"+nomFiles[file]))
                            {
549                         System.out.println("Fichier introuvable!\n");
550                     }
551
552                 }while(!fichierExiste("data_alignement/"+nomFiles[file]));
553                 if(longueurNomGenePlusLong<nomFiles[file].length()){
554                     longueurNomGenePlusLong=nomFiles[file].length();
555                 }
556             }
557             int nb_trees = scan.nextInt();
558             nomFileTree=new String[nb_trees];
559             for (int fileT=0; fileT<nomFileTree.length;fileT++){
```

```
560                 do{
561                     nomFileTree [fileT]= scan.next();
562                     if (!fichierExiste("data_alignement/"+nomFileTree[
                            fileT])){
563                         System.out.println("Fichier introuvable!\n");
564                     }
565
566                 }while(!fichierExiste("data_alignement/"+nomFileTree[fileT
                        ]));
567             }
568
569             moyBoostrapUtilisateur = scan.nextFloat();
570             valSeuilRF = scan.nextFloat();
571
572             int nb_TailleWindows = scan.nextInt();
573
574             tab_tailleWindows = new int [nb_TailleWindows];
575             for (int fileW=0; fileW<tab_tailleWindows.length;fileW++){
576                 do{
577                     tab_tailleWindows [fileW]= scan.nextInt();
578                     if (tab_tailleWindows [fileW]<0){
579                         System.out.println("Taille de la fenetre doit-etre
                                positive!\n");
580                     }
581
582                 }while(tab_tailleWindows [fileW]<0);
583             }
584
585             pasWindows = scan.nextInt();
586
587             //Fixe les dimensions de la matrice indexValeurRef
588             indexValeurRef = new int [nb_alignemnts][nb_TailleWindows][
                    nb_trees];
589
590             if (windowsComplet){
```

```
591                 Impression(nb_trees, nomFileTree, moyBoostrapUtilisateur,
                        valSeuilRF, tailleWindows, pasWindows);
592                 for (int file=0;file<nb_alignemnts;file++){
593                     Run_Process (file, nomFiles[file], windowsComplet);
594                     tab_booleanAfichageConsence=false;
595                     tab_booleanAfichage=false;
596                 }
597             }
598             windowsComplet=false;
599
600             if(!windowsComplet && tab_tailleWindows.length>=0 ){
601                 for (int nb_tailleWindows=0; nb_tailleWindows<
                        tab_tailleWindows.length; nb_tailleWindows++){
602                     tailleWindows = tab_tailleWindows [nb_tailleWindows];
603                     Impression(nb_trees, nomFileTree,
                            moyBoostrapUtilisateur, valSeuilRF, tailleWindows,
                            pasWindows);
604                     for (int file=0;file<nb_alignemnts;file++){
605                         Run_Process (file, nomFiles[file], windowsComplet)
                                ;
606                     }
607                     numero_Windows_enCours++;
608                 }
609             }
610
611             //Affichage de la matrice des index
612             AffichageBilan (indexValeurRef, nomFiles, tab_tailleWindows,
                    nomFileTree);
613
614             Date maDateFin = new Date();
615
616             executeRM();
617             System.out.print("FIN DU PROGRAMME.\n");
618             scan.close();
619
620         }catch(Exception e){
```

```
621                 System.out.println("\n"+e);
622             }
623         }
624
625
626 }
```

# APPENDICE B

## SCRIPT PERL

Cet appendice contient le script Perl permettant la création d'une matrice de présences/absences d'un gène vis à vis d'une espèce particulière.

/**********************************************************

* Établissement : Université du Québec a Montréal *

* @author : Nadia Tahiri *

* @version : 2012 *

**********************************************************/

```
#! /usr/bin/perl -w
use Bio::DB::EUtilities;
use Bio::DB::GenBank;

my $gb = new Bio::DB::GenBank;

#Fichier d'entree contenant la liste des espèces.
open (IN , 'Input.txt') || die "Erreur de lecture du fichier d'entrée.";
chomp(my @inputText = <IN>);
close IN;
my $compteur=0;
my $nb_supp=0;

#Compte le nombre d'espèces dans le fichier d'entrée.
foreach $_(@inputText){
```

```
16      $compteur++;
17  }
18
19  my @factory;
20  my @liste_mots=();
21  my $nb_occ=0;
22
23  #Pour chaque espèce, récupérez la liste des numéros d'accession
24  for (my $j=0; $j<$compteur; $j++){
25      my $file= "Output.csv";
26      open (OUT , '>'.$file)|| die "Erreur d'initialisation du fichier de
            sortie.";
27
28      $factory[$j] = Bio::DB::EUtilities->new(-eutil => 'esearch',
29                              -db => 'protein',
30                              -term => $inputText[$j]."[Organism]",
31                              -retmax => 200000);
32
33      print STDOUT "Query translation : ",$factory[$j]->get_query_translation
            ,"\n";
34
35      # query hits
36      close OUT;
37  }
38
39  my $file1="Output_res.txt";
40  open (OUT1 , '>'.$file1)|| die "Erreur d'initialisation du fichier de
        sortie.";
41  close OUT1;
42  for (my $i=0; $i<$compteur; $i++) {# pour chaque espèce
43      $nb_seq=$factory[$i]->get_count;
44          my %values=();
45      my $file= "Output.csv";
46      open (OUT , '>>'.$file)|| die "Erreur d'initialisation du fichier de
            sortie.";
47      open (OUT1 , '>>'.$file1)|| die "Erreur d'initialisation du fichier de
```

```
        sortie : $!";
48
49      for (my $k=0; $k<$nb_seq; $k++){#pour chaque gène
50
51          eval{
52              # Variables
53              my @ids = $factory[$i]->get_ids;
54              my $seq = $gb->get_Seq_by_gi($ids[$k]);
55
56              # Conservation des gènes pour chaque espèce
57              for my $feat_object ($seq->get_SeqFeatures) {
58                  for my $tag ($feat_object->get_all_tags) {
59                      if ($tag eq "product"){
60                          for my $value ($feat_object->get_tag_values($tag)) {
61                              $values{$value} = $value;
62                          }
63                      }
64                  }
65              }
66          };
67
68          #gestion des exceptions levées
69          if (@!) {
70              print STDOUT "\nPas de données dans GenBanck.\n";
71              print OUT "\nPas de données dans GenBanck.\n";
72          }
73      }
74      print STDOUT "Les produits sont\n";
75      foreach $value(sort keys %values){
76          print STDOUT $inputText[$i].";".$value."\n";
77          print OUT $inputText[$i].";".$value."\n";
78          print OUT1 $value."\n";
79      }
80      close OUT;
81      close OUT1;
82  }
```

```
83
84
85 open (IN1 , 'Output_res.txt') || die "Erreur de lecture du fichier d'entré
       e.";
86 chomp(my @liste_mots = <IN1>);
87 close IN1;
88
89 my $file2= "Output_res1.txt";
90 open (OUT2 , '>'.$file2)|| die "Erreur d'initialisation du fichier de
       sortie.";
91 close OUT2;
92
93 open (OUT2 , '>>'.$file2)|| die "Erreur d'initialisation du fichier de
       sortie.";
94
95 #Affichage de la liste des espèces :
96 print OUT2 "La liste des espèces.\n\n";
97 foreach $_(@inputText){
98     print OUT2 $_."\n";
99 }
100
101 my $compteurs=0;
102 foreach $_(@liste_mots){
103     print STDOUT $_."\n";
104     $compteurs++;
105 }
106
107 my %liste_nbOcc=();
108 my $val;
109 for (my $i=0; $i<$compteurs; $i++){
110     $val=$liste_mots[$i];
111     if (exists $liste_nbOcc{$val}){
112         $liste_nbOcc{$val}=$liste_nbOcc{$val}+1;
113     }else{
114         $liste_nbOcc{$val}=1;
115     }
```

```
116
117 }
118 #Affichage des noms des gènes=>nombre d'espèces pour lequel ce gène est pr
      ésent.
119 print OUT2 "\nnom gène => nombre espèce dans lequel ce gène est présent\n\
      n";
120 while( my ($k, $v) = each %liste_nbOcc ) {
121     if ($v>($compteur/2)){
122         print STDOUT "$k => $v\n";
123         print OUT2 "$k => $v\n";
124     }
125 }
126 close OUT2;
```

# GLOSSAIRE

**ADN (acide désoxyribonucléique).** Macromolécule constituée de deux chaînes enroulées en double hélice. Ses deux brins sont assemblés à partir de nucléotides. Chaque nucléotide comprend un sucre, le désoxyribose, un phosphate et une des quatre bases azotées (adénine, guanine, cytosine et thymine). L'ADN est le support de l'information génétique des organismes vivants.

**Alignement de séquences.** Opération qui consiste à disposer les unes en dessous des autres des portions de séquences similaires en minimisant leurs différences (on peut aligner entre eux des gènes d'une même famille multigénique ou des gènes d'espèces différentes). Si ces gènes sont homologues, les différences d'aminoacides ou d'acides nucléiques entre les séquences actuelles sont le témoignage de mutations qui ont eu lieu dans le passé.

**Aminoacide (acide aminé).** Unité constitutive des protéines. Il existe 20 acides aminés communs : alanine, arginine, asparagine, aspartate, cystéine, glutamine, glycine, histidine, isoleucine, leucine, lysine, méthionine, phénylalanine, proline, glutamate, sérine, thréonine, tryptophane, tyrosine et valine.

**Arbre enraciné.** Graphe connexe non cyclique ayant une origine ou appelé souvent racine ou ancêtre, ce qui explique que les liens sont orientés.

**Arbre non enraciné.** Graphe connexe non cyclique. Ce qui veut dire qu'il y a absence de boucle, c'est-à-dire, qu'il existe un seul et unique chemin permettant de passer d'un sommet à un autre.

**ARN (acide ribonucléique).** Polymère linéaire dont la sous-unité de base, un ribonucléotide, contient le sucre ribose.

**Carnivores.** Le groupe des Carnivores appartient à l'embranchement des mammifères. Il se caractérise essentiellement par "de fortes canines (crocs) et de molaires

tranchantes (carnassières)". Cette spécificité se traduit essentiellement par leurs modes alimentaires variés.

**Clade.** Vient du grec clados qui signifie arête. Taxon strictement monophylétique, c'est-à-dire contenant un ancêtre et tous ses descendants.

**Extragroupe (outgroup).** On dit aussi groupe extérieur ou encore "outgroup" tiré de l'anglais. Groupe que l'on sait a priori placé en dehors d'un ensemble de taxons dont on cherche les relations de parenté.

**Horloge moléculaire (hypothèse).** L'hypothèse selon laquelle les molécules d'une même classe fonctionnelle évoluent régulièrement dans le temps à un rythme égal dans différentes lignées. Ainsi la quantité des différences moléculaires constatées de nos jours dans des séquences homologues d'espèces distinctes peut être utilisée pour estimer le temps écoulé depuis le dernier ancêtre commun à ces espèces (ou le temps de divergence).

**Racine.** Le segment de l'arête en amont du nœud du rang le plus important, définissant le groupe extérieur (voir Extragroupe). En d'autres termes, c'est la position dans l'arbre du groupe extérieur. La racine peut être considérée comme un point de référence pour l'interprétation des caractères : les états de caractères de l'extragroupe (outgroup) sont des états plésiomorphes, les états qui en diffèrent sont apomorphes. Remarque : pour pouvoir comparer aisément deux arbres, il faut les enraciner chacun avec la même espèce ou avec le même taxon.

**Taxon.** Ensemble des organismes reconnus et définis dans chacune des catégories de la classification biologique hiérarchisée. En d'autres termes : contenu concret d'une catégorie. Exemple : *Canis lupus*, le loup, est un taxon de rang spécifique (catégorie : espèce) ; les canidés (Chien, Loup, Renard) constituent un taxon de rang familial (catégorie : famille).

# RÉFÉRENCES


Abdennadher, N., et R. Boesch. 2007. « Deploying phylip phylogenetic package on a large scale distributed system ». In *Cluster Computing and the Grid, 2007. CCGRID 2007. Seventh IEEE International Symposium on*, p. 673–678. IEEE.

Acheson, A., J. Conover, J. Fandl, T. DeChiara, M. Russell, A. Thadani, S. Squinto, G. Yancopoulos et R. Lindsay. 1994. « A bdnf autocrine loop in adult sensory neurons prevents cell death ». *Nature*, vol. 374, no. 6521, p. 450–453.

Aßfalg, A., et E. Erdfelder. 2012. « Caml—maximum likelihood consensus analysis ». *Behavior research methods*, vol. 44, no. 1, p. 189–201.

Avise, J. 2000. *Phylogeography : the history and formation of species*. Harvard University Press.

Baraldi, P., B. Cacciari, R. Romagnoli, S. Merighi, K. Varani, P. Borea et G. Spalluto. 2000. « A3 adenosine receptor ligands : history and perspectives ». *Medicinal research reviews*, vol. 20, no. 2, p. 103–128.

Barrett, M., M. Donoghue et E. Sober. 1991. « Against consensus ». *Systematic Zoology*, vol. 40, no. 4, p. 486–493.

Barthélemy, J.-P., et A. Guénoche. 1988. *Les arbres et les représentations des proximités*. Masson, Paris.

———. 1991. *Trees and proximity representations*. Wiley, New York.

Bekinschtein, P., M. Cammarota, C. Katche, L. Slipczuk, J. Rossato, A. Goldin, I. Izquierdo et J. Medina. 2008. « Bdnf is essential to promote persistence of long-term memory storage ». *Proceedings of the National Academy of Sciences*, vol. 105, no. 7, p. 2711–2716.

Bernardo, J., A. Smith et M. Berliner. 1994. *Bayesian theory*. T. 62. Wiley New York.

Bhattacharyya, A., U. Ear, B. Koller, R. Weichselbaum et D. Bishop. 2000. « The breast cancer susceptibility gene brca1 is required for subnuclear assembly of rad51 and survival following treatment with the dna cross-linking agent cisplatin ». *Journal of Biological Chemistry*, vol. 275, no. 31, p. 23899–23903.

Biason-Lauber, A., D. Konrad, M. Meyer, C. DeBeaufort et E. Schoenle. 2009. « Ovaries and female phenotype in a girl with 46, xy karyotype and mutations in the *CBX2* gene ». *The American Journal of Human Genetics*, vol. 84, no. 5, p. 658–663.

Bininda-Emonds, O. R., J. L. Gittleman et A. Purvis. 1999. « Building large trees by combining phylogenetic information : a complete phylogeny of the extant carnivora (mammalia) ». *Biological Reviews*, vol. 74, no. 2, p. 143–175.

Boc, A., A. B. Diallo et V. Makarenkov. 2012. « T-rex : a web server for inferring, validating and visualizing phylogenetic trees and networks ». *Nucleic Acids Research*, vol. 40, no. W1, p. W573–W579.

Chevenet, F., C. Brun, A. Bañuls, B. Jacq et R. Christen. 2006. « Treedyn : towards dynamic graphics and annotations for analyses of trees ». *BMC bioinformatics*, vol. 7, no. 1, p. 439.

Colless, D. 1970. « The phenogram as an estimate of phylogeny ». *Systematic Biology*, vol. 19, no. 4, p. 352–362.

Cormen, T. H., C. E. Leiserson, R. L. Rivest et C. Stein. 2001. *Introduction to algorithms*. The MIT press, 2 édition.

Czelusniak, J., M. Goodman, N. Moncrief et S. Kehoe. 1990. « Maximum parsimony approach to construction of evolutionary trees from aligned homologous sequences ». *Methods in enzymology*, vol. 183, p. 601–615.

Darwin, C. 1837. « Notebook b :[transmutation of species (1837-1838)] ». *CULDAR121, transcribed by Kees Rookmaaker*, vol. 1.

Darwin, C. 1859. *On the Origin of Species by Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life*. Murray, J., London.

DeLano, W. 2002. « The pymol molecular graphics system », vol. 382.

Edgar, R. 2004. « Muscle : multiple sequence alignment with high accuracy and high throughput ». *Nucleic acids research*, vol. 32, no. 5, p. 1792–1797.

Edgar, R., et S. Batzoglou. 2006. « Multiple sequence alignment ». *Current opinion in structural biology*, vol. 16, no. 3, p. 368–373.

Efron, B. 1979. « Bootstrap methods : another look at the jackknife ». *The annals of Statistics*, vol. 7, no. 1, p. 1–26.

Ezeamuzie, C., et E. Philips. 1999. « Adenosine a3 receptors on human eosinophils mediate inhibition of degranulation and superoxide anion release ». *British journal of pharmacology*, vol. 127, no. 1, p. 188–194.

Felsenstein, J. 1980. *PHYLIP : Phylogeny Inference Package*. En ligne. <`http://evolution.genetics.washington.edu/phylip.html`>. Consulté le 20 Août 2012.

Felsenstein, J. 1993. « Phylip : phylogenetic inference package, version 3.5 c. ». *Department of Genetics, University of Washington, Seattle*.

Felsenstein, J. 2004. *Inferring Phylogenies*. Sunderland, Massachusetts.

Ferguson, S., J. Virgl et S. Lariviere. 1996. « Evolution of delayed implantation and associated grade shifts in life history traits of north american carnivores ». *Ecoscience. Sainte-Foy*, vol. 3, no. 1, p. 7–17.

Fisher, R. 1922. « On the interpretation of $\chi$ 2 from contingency tables, and the calculation of p ». *Journal of the Royal Statistical Society*, vol. 85, no. 1, p. 87–94.

Fishman, P., S. Bar-Yehuda, F. Barer, L. Madi, A. Multani et S. Pathak. 2001. « The a3 adenosine receptor as a new target for cancer therapy and chemoprotection ». *Experimental cell research*, vol. 269, no. 2, p. 230–236.

Fitch, W., et E. Margoliash. 1967. « Construction of phylogenetic trees ». *Science*, vol. 155, no. 3760, p. 279–284.

Fitch, W.-M. 1971. « Toward defining the course of evolution : Minimum change for a specific tree topology. ». *Systematic Zoology*, vol. 20, p. 406–416.

Friedenson, B. 2007. « The brca1/2 pathway prevents hematologic cancers in addition to breast and ovarian cancers ». *BMC cancer*, vol. 7, no. 1, p. 152–163.

Garland, T. J., A. W. Dickerman, C. M. Janis et J. A. Jones. 1993. « Phylogenetic analysis of covariance by computer simulation ». *Systematic Biology*, vol. 42, no. 3, p. 265–292.

Guindon, S., F. Delsuc, J.-F. Dufayard et O. Gascuel. 2009. « Estimating maximum likelihood phylogenies with phyml », vol. 537, p. 113–137.

Guindon, S., et O. Gascuel. 2003. « A simple, fast, and accurate algorithm to estimate large phylogenies by maximum likelihood ». *Systematic biology*, vol. 52, no. 5, p. 696–704.

Guindon, S., F. Lethiec, P. Duroux et O. Gascuel. 2005. « Phyml online—a web server for fast maximum likelihood-based phylogenetic inference ». *Nucleic acids research*, vol. 33, no. suppl 2, p. W557–W559.

Haeckel, E. 1874. *Anthropogenie oder Entwickelungsgeschichte des Menschen : gemeinverständliche wissenschaftliche Vorträge über die Grundzüge der menschlichen Keimes-und Stammes-Geschichte*. Wilhelm Engelmann.

Hall, J., M. Lee, B. Newman, J. Morrow, L. Anderson, B. Huey et M. King. 1990. « Linkage of early-onset familial breast cancer to chromosome 17q21 ». *Science*, vol. 250, no. 4988, p. 1684–1689.

Hennig, W. 1965. « Phylogenetic systematics ». *Annual Review of Entomology*, vol. 10, no. 1, p. 97–116.

Huang, E., et L. Reichardt. 2001. « Neurotrophins : roles in neuronal development and

function ». *Annual review of neuroscience*, vol. 24, no. 1, p. 677–736.

Huelsenbeck, J., F. Ronquist, R. Nielsen et J. Bollback. 2001. « Bayesian inference of phylogeny and its impact on evolutionary biology ». *science*, vol. 294, no. 5550, p. 2310–2314.

Iliopoulos, D., N. Volakakis, A. Tsiga, I. Rousso et N. Voyiatzis. 2004. « Description and molecular analysis of sry and ar genes in a patient with 46, xy pure gonadal dysgenesis (swyer syndrome) », vol. 47, no. 2, p. 185–190.

Ingelsson, E., E. Schaefer, J. Contois, J. McNamara, L. Sullivan, M. Keyes, M. Pencina, C. Schoonmaker, P. Wilson, R. D'Agostino et S. V. Ramachandran. 2007. « Clinical utility of different lipid measures for prediction of coronary heart disease in men and women ». *JAMA : the journal of the American Medical Association*, vol. 298, no. 7, p. 776–785.

Jiang, X., S. Qin, C. Qiao, K. Kawano, M. Lin, A. Skold, X. Xiao et A. Tall. 2001. « Apolipoprotein b secretion and atherosclerosis are decreased in mice with phospholipid-transfer protein deficiency ». *Nature medicine*, vol. 7, no. 7, p. 847–852.

Jukes, T. 1969. « Evolution of protein molecules. ». *Manmmalian Protein Metabolism*, p. 21–132.

Katoh, K., K. Kuma, H. Toh et T. Miyata. 2005. « Mafft version 5 : improvement in accuracy of multiple sequence alignment ». *Nucleic acids research*, vol. 33, no. 2, p. 511–518.

Kihara, T., S. Shimohama, H. Sawada, J. Kimura, T. Kume, H. Kochiyama, T. Maeda et A. Akaike. 1997. « Nicotinic receptor stimulation protects neurons against $\beta$-amyloid toxicity ». *Annals of neurology*, vol. 42, no. 2, p. 159–163.

Kimura, M. 1980. « Simple model for estimating evolutionary rates of base substitutions through comparative studies of nucleotide sequences ». *Journal of molecular evolution*, vol. 16, no. 2, p. 111–120.

———. 1985. *The neutral theory of molecular evolution*. Cambridge University Press.

Knowles, L., et W. Maddison. 2002. « Statistical phylogeography ». *Molecular Ecology*, vol. 11, no. 12, p. 2623–2635.

Kocher, T., J. Conroy, K. McKaye, J. Stauffer et S. Lockwood. 1995. « Evolution of nadh dehydrogenase subunit 2 in east african cichlid fish ». *Molecular Phylogenetics and Evolution*, vol. 4, no. 4, p. 420–432.

Kosiol, C., N. Goldman et N. Buttimore. 2004. « A new criterion and method for amino acid classification ». *Journal of theoretical biology*, vol. 228, no. 1, p. 97–106.

Lamarck, J. 1809. *Philosophie zoologique, ou Exposition des considerations relatives à l'histoire naturelle des animaux*. T. 1. Verdière, Paris.

———. 1830. *Philosophie zoologique, ou Exposition des considerations relatives à l'histoire naturelle des animaux.* T. 2. Baillière, J.-B., Paris.

Lamarck, J., et C. Martins. 1873. *Philosophie zoologique, ou Exposition des considérations relatives à l'histoire naturelle des animaux.* T. 1. Savy, F., Paris.

Leibrock, J., F. Lottspeich, A. Hohn, M. Hofer, B. Hengerer, P. Masiakowski, H. Thoenen et Y. Barde. 1989. « Molecular cloning and expression of brain-derived neurotrophic factor ». *Nature*, vol. 341, no. 6238, p. 149–152.

Lemarie, A., et S. Grimm. 2011. « Mitochondrial respiratory chain complexes : apoptosis sensors mutated in cancer ? ». *Oncogene*, vol. 30, no. 38, p. 3985–4003.

Letunic, I., et P. Bork. 2011. « Interactive tree of life v2 : online annotation and display of phylogenetic trees made easy ». *Nucleic acids research*, vol. 39, no. suppl 2, p. W475–W478.

Li, K. 2003. « Clustalw-mpi : Clustalw analysis using distributed and parallel computing ». *Bioinformatics*, vol. 19, no. 12, p. 1585–1586.

Liu, E., C. Li, M. Govindasamy, H. Neo, T. Lee, C. Low et S. Tachibana. 2012. « Elevated prepronociceptin, nociceptin/orphanin fq and nocistatin concentrations in rat chronic constriction nerve injury and diabetic neuropathic pain models ». *Neuroscience Letters*, vol. 506, no. 1, p. 104–106.

Liu, G., S. Richards, R. Olsson, K. Mullane, R. Walsh et J. Downey. 1994. « Evidence that the adenosine a3 receptor may mediate the protection afforded by preconditioning in the isolated rabbit heart ». *Cardiovascular research*, vol. 28, no. 7, p. 1057–1061.

Liu, Y., G. Fiskum et D. Schubert. 2002. « Generation of reactive oxygen species by the mitochondrial electron transport chain ». *Journal of neurochemistry*, vol. 80, no. 5, p. 780–787.

Maglott, D., J. Ostell, K. Pruitt et T. Tatusova. 2007. « Entrez gene : gene-centered information at ncbi ». *Nucleic acids research*, vol. 35, no. suppl 1, p. D26–D31.

Makarenkov, V. 2001. « T-rex : reconstructing and visualizing phylogenetic trees and reticulation networks ». *Bioinformatics*, vol. 17, no. 7, p. 664–668.

Makarenkov, V., et B. Leclerc. 2000. « Comparison of additive trees using circular orders ». *Journal of Computational Biology*, vol. 7, no. 5, p. 731–744.

Makover, A., D. Soprano, M. Wyatt et D. Goodman. 1989. « An in situ-hybridization study of the localization of retinol-binding protein and transthyretin messenger rnas during fetal development in the rat ». *Differentiation*, vol. 40, no. 1, p. 17–25.

McFadden, C., I. Tullis, M. Breton Hutchinson, K. Winner et J. Sohm. 2004. « Variation in coding (nadh dehydrogenase subunits 2, 3, and 6) and noncoding intergenic

spacer regions of the mitochondrial genome in octocorallia (cnidaria : Anthozoa) ». *Marine Biotechnology*, vol. 6, no. 6, p. 516–526.

Miki, Y., J. Swensen, D. Shattuck-Eidens, P. Futreal, K. Harshman, S. Tavtigian, Q. Liu, C. Cochran, L. Bennett, W. Ding *et al.* 1994. « A strong candidate for the breast and ovarian cancer susceptibility gene brca1 ». *Science*, vol. 266, no. 5182, p. 66–71.

Nagylaki, T. 1992. *Introduction to theoretical population genetics*. T. 21. Springer.

Park, J., R. Irvine, G. Buchanan, S. Koh, J. Park, W. Tilley, M. Stallcup, M. Press et G. Coetzee. 2000. « Breast cancer susceptibility gene 1 (brca1) is a coactivator of the androgen receptor ». *Cancer research*, vol. 60, no. 21, p. 5946–5949.

Petrussa, E., A. Bertolini, V. Casolo, J. Krajňáková, F. Macrì et A. Vianello. 2009. « Mitochondrial bioenergetics linked to the manifestation of programmed cell death during somatic embryogenesis of abies alba ». *Planta*, vol. 231, no. 1, p. 93–107.

Phipps, J. 1971. « Dendrogram topology ». *Systematic Biology*, vol. 20, no. 3, p. 306–308.

Pischon, T., C. Girman, F. Sacks, N. Rifai, M. Stampfer et E. Rimm. 2005. « Non–high-density lipoprotein cholesterol and apolipoprotein b in the prediction of coronary heart disease in men ». *Circulation*, vol. 112, no. 22, p. 3375–3383.

Reece, J., L. Urry, M. Cain, S. Wasserman, P. Minorsky et R. Jackson. 2011. *Campbell Biology*. T. 1. Pearson USA, 9 édition.

Robinson, D., et L. Foulds. 1981. « Comparison of phylogenetic trees ». *Mathematical Biosciences*, vol. 53, no. 1-2, p. 131–147.

Ropelewski, A., H. Nicholas et R. Mendez. 2010. « Mpi-phylip : Parallelizing computationally intensive phylogenetic analysis routines for the analysis of large protein families ». *PloS one*, vol. 5, no. 11, p. e13999.

Ruggeri, Z., et T. Zimmerman. 1987. « von willebrand factor and von willebrand disease ». *Blood*, vol. 70, no. 4, p. 895–904.

Sadler, J. 1998. « Biochemistry and genetics of von willebrand factor ». *Annual review of biochemistry*, vol. 67, no. 1, p. 395–424.

Saitou, N., et M. Nei. 1987. « The neighbor-joining method : a new method for reconstructing phylogenetic trees. ». *Molecular biology and evolution*, vol. 4, no. 4, p. 406–425.

Sajjadi, F., K. Takabayashi, A. Foster, R. Domingo et G. Firestein. 1996. « Inhibition of tnf-alpha expression by adenosine : role of a3 adenosine receptors ». *The Journal of Immunology*, vol. 156, no. 9, p. 3435–3442.

Sanderson, M. 2002. « Estimating absolute rates of molecular evolution and diver-

gence times : a penalized likelihood approach ». *Molecular Biology and Evolution*, vol. 19, no. 1, p. 101–109.

Schantl, J., M. Roza, A. De Jong et G. Strous. 2003. « Small glutamine-rich tetratricopeptide repeat-containing protein (sgt) interacts with the ubiquitin-dependent endocytosis (ube) motif of the growth hormone receptor ». *Biochemical Journal*, vol. 373, no. Pt 3, p. 855–863.

Stamatakis, A., P. Hoover et J. Rougemont. 2008. « A rapid bootstrap algorithm for the raxml web servers ». *Systematic biology*, vol. 57, no. 5, p. 758–771.

Starita, L., et J. Parvin. 2003. « The multiple nuclear functions of brca1 : transcription, ubiquitination and dna repair ». *Current opinion in cell biology*, vol. 15, no. 3, p. 345–350.

Stephens, A., M. Khan, X. Roucou, P. Nagley et R. Devenish. 2003. « The molecular neighborhood of subunit 8 of yeast mitochondrial f1f0-atp synthase probed by cysteine scanning mutagenesis and chemical modification ». *Journal of Biological Chemistry*, vol. 278, no. 20, p. 17867–17875.

Teng, B., C. Burant et N. Davidson. 1993. « Molecular cloning of an apolipoprotein b messenger rna editing protein ». *Science*, vol. 260, no. 5115, p. 1816–1819.

Thompson, J., T. Gibson et D. Higgins. 2002. « Multiple sequence alignment using clustalw and clustalx ». *Current protocols in bioinformatics.*

Tracey, W., W. Magee, H. Masamune, S. Kennedy, D. Knight, R. Buchholz et R. Hill. 1997. « Selective adenosine a3 receptor stimulation reduces ischemic myocardial injury in the rabbit heart ». *Cardiovascular research*, vol. 33, no. 2, p. 410–415.

Tuffley, C., W. White, M. Hendy et D. Penny. 2012. « Correcting the apparent mutation rate acceleration at shorter time scales under a jukes-cantor model ». *Molecular Biology and Evolution.*

Van-Valkenburgh, B. 2007. « Déjà vu : the evolution of feeding morphologies in the carnivora ». *Society for Integrative and Comparative Biology*, vol. 47, no. 1, p. 147–163.

Vincentelli, A., S. Susen, T. Le Tourneau, I. Six, O. Fabre, F. Juthier, A. Bauters, C. Decoene, J. Goudemand, A. Prat et B. Jude. 2003. « Acquired von willebrand syndrome in aortic stenosis ». *New England Journal of Medicine*, vol. 349, no. 4, p. 343–349.

Walker, B., M. Jacobson, D. Knight, C. Salvatore, T. Weir, D. Zhou et T. Bai. 1997. « Adenosine a3 receptor expression and function in eosinophils ». *American journal of respiratory cell and molecular biology*, vol. 16, no. 5, p. 531–537.

Wallace, I., G. Blackshields et D. Higgins. 2005. « Multiple sequence alignments ». *Current opinion in structural biology*, vol. 15, no. 3, p. 261–266.

Walldius, G., et I. Jungner. 2004. « Apolipoprotein b and apolipoprotein a-i : risk indicators of coronary heart disease and targets for lipid-modifying therapy ». *Journal of Internal Medicine*, vol. 255, no. 2, p. 188–205.

Wallis, M., P. Waters et J. Graves. 2008. « Sex determination in mammals—before and after the evolution of sry ». *Cellular and molecular life sciences*, vol. 65, no. 20, p. 3182–3195.

Wang, J., M. Guo et L. Xing. 2012. « Fastjoin, an improved neighbor-joining algorithm. ». *Genetics and molecular research : GMR*, vol. 11, no. 3, p. 1909.

Wang, Y., D. Cortez, P. Yazdi, N. Neff, S. Elledge et J. Qin. 2000. « Basc, a super complex of brca1-associated proteins involved in the recognition and repair of aberrant dna structures ». *Genes & development*, vol. 14, no. 8, p. 927–939.

Wheeler, W. 2003. « Implied alignment : a synapomorphy-based multiple-sequence alignment method and its use in cladogram search ». *Cladistics*, vol. 19, no. 3, p. 261–268.

Whelan, S., et N. Goldman. 2001. « A general empirical model of protein evolution derived from multiple protein families using a maximum-likelihood approach ». *Molecular biology and evolution*, vol. 18, no. 5, p. 691–699.

Yamada, K., et T. Nabeshima. 2003. « Brain-derived neurotrophic factor/trkb signaling in memory processes ». *Journal of pharmacological sciences*, vol. 91, no. 4, p. 267–270.

Yang, D., Y. Oyaizu, H. Oyaizu, G. Olsen et C. Woese. 1985. « Mitochondrial origins ». *Proceedings of the National Academy of Sciences*, vol. 82, no. 13, p. 4443–4447.

Yu, L., P. Luan, W. Jin, O. Ryder, L. Chemnick, H. Davis et Y. Zhang. 2011. « Phylogenetic utility of nuclear introns in interfamilial relationships of caniformia (order carnivora) ». *Systematic Biology*, vol. 60, no. 2, p. 175–187.

Yusnita, Y., M. Norsiah et A. Rahman. 2010. « Mutations in mitochondrial nadh dehydrogenase subunit 1 (mtnd1) gene in colorectal carcinoma ». *The Malaysian journal of pathology*, vol. 32, no. 2, p. 103–110.