

UNIVERSITÉ DU QUÉBEC À MONTRÉAL

INF4500

---

## Devoir 1

---

*Par :*

Guillaume Lahaie  
LAHG04077707

*Remis à :*

Abdoulaye Baniré Diallo

*Date de remise :*

Le 1<sup>er</sup> novembre 2013



# Table des matières

<b>1</b>	<b>Question 1</b>	<b>3</b>
a	Description du gène ALS2 . . . . .	3
b	Loci voisins du gène ALS2 . . . . .	3
c	Annotations du gène ALS2 . . . . .	4
d	Fichier genbank du gène ALS2 . . . . .	5
e	Homologues du gène ALS2 . . . . .	5
<b>2</b>	<b>Question 2</b>	<b>6</b>
a	Qualité des contigs . . . . .	6
b	Blast des contigs . . . . .	7
c	Fonctions prédites des contigs . . . . .	7
d	Géome des séquences fortement similaires . . . . .	7
<b>3</b>	<b>Question 3</b>	<b>8</b>
a	Identification du vecteur pANNE . . . . .	8
<b>4</b>	<b>Question 4</b>	<b>9</b>
a	Alignement multiple . . . . .	10
b	Facilité et rapidité des programmes . . . . .	10
c	Qualité des alignements . . . . .	11
d	Meilleur alignement . . . . .	11
<b>5</b>	<b>Annexes</b>	<b>12</b>
1	Image du locus du gène ALS2 chez l'homo sapien du UCSC genome browser . . . . .	12
2	Image du locus du gène ALS2 chez l'homo sapien de NCBI . . . . .	12
3	Script biopython de calcul des fréquences nucléotidiques . . . . .	12
4	Vue de MapViewer du locus du gène ALS2 chez l'humain, 202,555 à 202,656. . . . .	13

5	Vue de MapViewer du locus du gène ALS2 chez l'humain, 202M à 203M. . . . .	14
6	Vue de SequenceViewer du locus du gène ALS2. . . . .	14
7	Script biopython Pour choisir 5 contigs au hasard à partir du résultat de CAP3, et d'effectuer un blast sur ces contigs . . . . .	14
8	Information à propos de la qualité du contig 7 . . . . .	16
9	Information à propos de la qualité du contig 132 . . . . .	17
10	Information à propos de la qualité du contig 195 . . . . .	18
11	Information à propos de la qualité du contig 209 . . . . .	20
12	Information à propos de la qualité du contig 214 . . . . .	21
13	Résultat graphique du blast des 5 contigs. . . . .	23
14	Script Biopython pour obtenir les fichiers d'accèsion des 10 premiers résultats du blast pour un contig donné. . . . .	27
15	Résultat graphique du blast du gène FOXP4 de l'Homo sapiens. . . . .	29
16	Information du système qui a fait les alignements multiples. . . . .	30
17	Log de l'exécution de clustalw2. . . . .	31
18	Log du travail de RepeatMasker sur le fichier foxp4_ortho.fa. . . . .	32
19	Log de l'exécution de Mavid sur foxp4_ortho.fa. . . . .	33
20	Arbre phylogénique créé par ClustalW . . . . .	35
21	Arbre phylogénique créé par Dialign . . . . .	35
22	Arbre phylogénique créé par Mavid . . . . .	36
23	Script biopython pour identifier la composition du vecteur pANNE . . . . .	36
24	Temps d'exécution des alignements multiples . . . . .	38
25	Fichiers genbank utilisés pour ce rapport . . . . .	38

# 1 Question 1

## a Description sommaire du locus du gène ALS2 chez l'*Homo sapiens*.

J'ai consulté deux sources pour obtenir des informations sur la location du gène ALS2 (*amyotrophic lateral sclerosis 2 (juvenile)*) de l'humain. La première source, afin d'identifier la location du locus, était le *UCSC Genome Browser* [1]. La recherche a été effectuée sur la version GRCh37/hg19 du génome humain. Comme on peut le voir sur l'image en annexe 1, le gène ALS2 est sur le brin q, c'est-à-dire qu'il est sur le long bras du chromosome.

Le locus du gène est situé sur le chromosome, à la position 2q33.1. La position précise du gène est la séquence 202 564 986 à 202 645 895 du chromosome 2. Sa taille est donc 80 910 paires de nucléotides. J'ai effectué la même recherche ensuite dans la base de données du *National Center for Biotechnology Information* (NCBI). J'ai effectué une recherche dans la base de données des gènes, j'insère en annexe 2 une image de la région du chromosome 2 où se situe le gène. On retrouve les mêmes informations à propos du gène ALS2 : la location, la taille et sa location précise sont les mêmes que UCSC. Afin de trouver la distribution des nucléotides dans le gène, j'ai écrit un script biopython (annexe 3) prenant en entrée le fichier genbank de la séquence du gène et retournant la distribution. Le résultat de cette analyse est détaillé à la table 1.

NCBI donne un sommaire du rôle du gène ALS2. Il encode une protéine, et son nom est donné dans le fichier genbank de la séquence nucléotidique : l'alsin. Afin d'avoir une meilleure idée de la fonction du gène ALS2, j'ai effectué une recherche sur *Genetics Home Reference* (GHR) [2]. La protéine semble avoir plusieurs rôles, entre autres l'activation de protéines de la famille des GTPases, qui joue un rôle important dans la division cellulaire, la différenciation. Ce gène semble surtout connu pour son lien à la Sclérose latérale amyotrophique, aussi connue sous le nom de la maladie de Lou Gherig. Toutefois, seulement 5 à 10% des cas de cette maladie seraient de cause génétique [3].

## b Comparaison des loci voisins au gène ALS2.

J'ai tout d'abord suivi le lien *Map Viewer* de la description du gène ALS2 pour trouver la location du gène ALS2 (annexe 4). Cette vue permet d'identifier deux loci voisins immédiats, les loci des gènes MPP4 et RPS2P16. J'ai ensuite changé les valeurs des nucléotides de début et de fin afin de voir la séquence 202M - 203M du chromosome 2 de l'humain (annexe 5). Cette vue permet d'observer plusieurs autres loci de gènes. J'ai choisi de décrire les loci des gènes MPP4, RPS2P16, CDK15, TMEM237 et ENO1P4.

Nucléotides	Compte	Pourcentage
A	25414	28.91%
C	16471	18.74%
G	17296	19.67%
T	28729	32.68%
Total	87910	100%

TABLE 1 : Distribution des nucléotides dans le locus du gène ALS2

On peut séparer ces gènes en deux catégories. La première catégorie sont des pseudogènes. En effet, Selon la description des gènes RPS2P16 et ENO1P4, il s'agit de deux pseudogènes, c'est-à-dire des gènes qui ne sont plus exprimés chez l'humain [5]. Ils ont donc un rôle différent du gène ALS2. Ils ont toutefois une relation avec ce gène, car ils sont situés à la même location du chromosome 2. Le gène RPS2P16 (*ribosomal protein S2 pseudogene 16*) est situé à la location 2q33.1 du chromosome 2, et il est orienté 5' → 3'. Plus précisément, il occupe la séquence 202627437 - 202628333 du chromosome 2, et a une taille de 896 bp. Le gène ENO1P4 (*enolase 1, (alpha) pseudogene 4*) est situé à la location 2q33.1, plus précisément la séquence 202486369 à 202488153 de la séquence de référence du chromosome 2 de l'humain. Il a donc une taille de 1784bp. Il a une orientation 3' → 5'.

La deuxième catégorie de gènes est plus similaires à ALS2. En effet, les gènes MPP4 (*membrane protein, palmitoylated 4 (MAGUK p55 subfamily 4)*), CDK15 (*cyclin-dependant kinase 15*) et TMEM237 (*transmembrane protein 237*) sont tous des gènes qui encodent une protéine. On remarque aussi que dans les noms synonymes, ils ont un nom similaire à ALS2. Le nom ALS2CR5 pour MPP4, ALS2CR7 pour CDK15 et ALS2CR4 pour TMEM237, alors que le gène ALS2 est aussi connu sous le nom ALS2CR6. Toutefois une recherche pour identifier ce lien n'a pas permis de découvrir un lien précis, cela semble lié à la région du chromosome.

Les fonctions possibles de ces trois gènes semblent différentes de la fonction de ALS2. Le gène MPP4, situé à la position 2q33.2, il est donc lui aussi sur le long bras du chromosome 2. Le gène a une taille de 53 821bp. Il produit une protéine membre de la famille MAGUK, qui se retrouve surtout dans la rétine. Le gène CDK15 est situé à la location 2q33.2, plus précisément à la séquence 202655177 - 202760273 du chromosome 2, il a donc une taille de 105 096 bp. Il est situé sur le long bras du chromosome. Il serait impliqué dans le contrôle du cycle "eucaryotique" des cellules. Finalement, le gène TMEM237 est situé à la location 2q33.2, donc sur le long bras du chromosome, plus précisément à la séquence 202484907 - 202508252 du chromosome 2. Il a donc une taille de 23345 bp. La protéine encodée par ce gène semble être impliquée dans "WNT signaling" (refseq 2012).

## c Description des annotations du locus du gène ALS2.

Afin de décrire les annotations du locus du gène ALS2, Je prendrai la vue de MapViewer en annexe 3. La première annotation, donnant des informations du gène, a déjà été décrite en a. La prochaine annotation, celle des OMIM, nous propose quatre résultats. Trois des résultats sont des maladies génétiques reliées au gène ALS2. La première est *Primal Lateral Sclerosis Juvenile* (PLSJ) [7], une maladie se rapprochant de ALS, mais qui est généralement moins sévère. La seconde maladie est la sclérose amyotrophique latérale [8], que nous avons déjà décrite, et la dernière maladie est *Spastic paralysis, infantile-onset, ascending* (IAHSP) [11]. Ces trois maladies ont un lien avec une mutation du gène ALS2. Le dernier résultat de la recherche OMIM présente des informations à propos de l'alsin [?], la protéine encodée par le gène ALS2. Selon Yang et al. [12] et Hadano et al. [13], ce gène comporte 34 exons dans une région de 83kb.

La prochaine annotation envoie vers HGNC [2], un annuaire de liens utiles à propos de ALS2. L'annotation suivante (sv) est un lien vers sequence viewer, une représentation graphique de la séquence nucléotidique du gène. On peut ainsi voir graphiquement la position des features du fichier, par exemple les STS, les CDS. Ensuite, l'annotation pr montre les résultats de la recherche dans la banque de données des protéines de la protéine encodée par ce gène. On y retrouve 19 résultats, la majorité reliée à la protéine alsin, qui est encodée par ALS2.

L'annotation dl permet d'obtenir un fichier contenant la séquence nucléotidique exacte du gène. Il est possible d'obtenir le fichier en format fasta ou genbank. Selon la prochaine annotation (ev), on peut retrouver 37 exons et 2 gènes dans la séquence de 81 710bp du gène ALS2, Ce résultat provient de l'alignement de 12 modèles. La

prochaine annotation (hm) nous donne l'ensemble des gènes homologues au gène ALS2, nous reviendrons sur cette annotation au point e. Finalement, l'annotation sts nous permet de trouver les neuf STS contenus dans le locus du gène ALS2. On remarque ici qu'on obtient un nombre différent de STS que dans l'annotation sv.

## d Analyse du fichier genbank du locus du gène ALS2.

J'ai utilisé le fichier genbank NG\_008775.1 comme représentation de la séquence des nucléotides du gène ALS2. J'ai trouvé cette séquence dans Sequence Viewer, c'était une des options offertes pour le gène ALS2. J'ai choisi ce fichier car il contient plus de features que le fichier de référence du chromosome 2.

Regardons tout d'abord les informations générales du fichier. La première ligne nous donne quelques informations de base : le numéro d'accession, la taille de la séquence (87 910 bp), qu'il s'agit d'une séquence linéaire d'ADN, et que la dernière modification date du 20 août 2013. On voit ensuite que cette séquence appartient au génome de l'*Homo sapiens*, et qu'il s'agit de la première version de cette séquence.

Regardons maintenant les features. Regardons tout d'abord le feature gene. On y retrouve 2 features gene, le premier celui du gène ALS2, qui se situe à la sous-séquence 5001..85910. On dénote ensuite un feature pour l'ARNm de ce gène, qui est un suivi de join. Ce fichier contient aussi les exons du gène, ce qui semble être une annotation plus récente. On dénote 34 exons pour le gène ALS2. Le feature gene comprend aussi le feature CDS, indiquant la région codante du gène. Encore une fois, il s'agit d'une série de join. Ce feature donne la translation du CDS en séquence d'acide aminé, et donne le nom de la protéine produite : *alsin isoform 1*, ainsi que le numéro d'accession et d'autres liens à propos de cette protéine.

Le second feature gene nous informe qu'une partie de la séquence fait partie du gène MPP4, toutefois seulement 441bp. On n'a donc très peu d'information sur ce gène dans ce fichier, seulement le feature mRNA, indiquant que la fin de la séquence (87479..>87586) transcrit en ARNm. On a donc dans les notes de ce feature le numéro d'accession de la séquence en question. Finalement, on retrouve la position de 9 STS dans cette séquence, ce qui correspond à l'annotation sts de *Map Viewer*.

## e Homologues du gène ALS2.

Afin de vérifier si ce gène a des homologues, j'ai consulté l'annotation hm de *Map Viewer* pour le gène ALS2. (Il aurait été aussi possible de faire une recherche blast pour trouver le résultat.) On retrouve un homologue de ALS2 chez les mammifères suivants : le chimpanzé (*Pan troglodytes*), le macaque rhésus (*Macaca mulatta*), le chien (*Canis lupus familiaris*), le boeuf (*Bos taurus*), la souris (*Mus musculus*) et le rat de Norvège (*Rattus norvegicus*).

J'ai consulté le fichier la fiche du gène pour toutes ces espèces. Le gène ALS2 joue le même rôle chez tous ces mammifères, en effet il encode une protéine. Ces pages ne contiennent toutefois pas d'information sur la fonction possible de cette protéine chez ces espèces. On peut toutefois remarquer que la protéine obtenue pour chaque espèce est l'alsin. De plus, on peut aussi remarquer que le nombre d'acides aminés des protéines obtenues est très similaire : il varie de 1649 à 1681 acides aminés. Il semble donc que ces protéines jouent des rôles similaires chez ces espèces, mais nous n'avons pas assez d'information disponible pour conclure du rôle précis de ce gène chez ces mammifères.

## 2 Question 2

Utilisez le programme CAP3 pour assembler les reads du SARS. Vous obtiendrez comme résultat plusieurs contigs. Concentrez votre analyse sur cinq contigs que vous choisirez.

### a Que pouvez-vous dire de la qualité des contigs ?

Afin de créer les contigs à partir des reads, j'ai utilisé la version de CAP3 [14] disponible sur le serveur Moby@Pasteur. L'application a produite 216 contigs à partir des données. J'ai ensuite écrit un script biopython (annexe 7) permettant de choisir au hasard 5 contigs et d'effectuer un blast [15] sur ces contigs. Le script a choisi au hasard (à l'aide d'un générateur de nombres aléatoires) les contigs 7, 132, 195, 209 et 214.

Regardons tout d'abord la qualité de ces contigs. J'ai évalué la qualité en analysant le fichier seqs.data.cap.contigs.qual. Pour le contig 7 (résultat annexe 8), la grande majorité des valeurs de qualité est 25, avec quelques endroits ayant une valeur de 10, et à certains endroits des 5 ou des 0. Ces valeurs sont basées sur l'échelle Phred [?], qui est un score ayant une relation logarithmique. Avec une valeur de 25, la probabilité d'une identification incorrecte se situe à plus de 99%, alors qu'une valeur de 10 indique qu'il y a 90% de chance d'un bon choix. Il semble donc que le contig est de bonne qualité, avec quelques endroits où la qualité est plus basse. Ces endroits se situent surtout au début et à fin du contig, ce à quoi on s'attend.

Le résultat de qualité du contig 132 (annexe 9) est plus bas que le précédent. La majorité du contig a une valeur de qualité de 20, et il y a une séquence d'environ 400 nucléotides à la fin du contig ayant une qualité variant de 0 à 15, ainsi qu'une séquence d'environ 80 nucléotides au début ayant une qualité de 10 ou 15. Le résultat est quand même généralement acceptable, avec une moyenne de 99% de précision, sauf aux extrémités, où le contig a environ 90% de précision.

Le contig 195 (résultat de qualité annexe 10) a une qualité assez faible. La première moitié du contig a un score de qualité de 15 en moyenne, alors que la seconde moitié a un score de 10, avec une très petite séquence au milieu du contig ayant un résultat de 20 ou plus. Ce contig a donc un résultat de précision beaucoup plus que les deux autres contigs choisis.

Le contig 209 (résultat de qualité annexe 11) a un résultat de qualité similaire au contig 7. Le score des bases est majoritairement 25, avec une séquence à la fin ayant un score plus faible, entre 10 et 15. La précision pour ce contig est donc très haute.

Finalement, le contig 214 (résultat de qualité annexe 12) a une très grande variation de qualité. On retrouve au milieu de contig une qualité très élevée, avec certaines séquences ayant un score de qualité de 30 ou même 35. Toutefois, le début et la fin du contig souffre d'une qualité beaucoup plus basse, avec une séquence ayant un score de 0, et une séquence ayant un score de 10 à la fin du contig.

### b Blastez vos contigs

Les cinq contigs choisis ont été blastés avec le script biopython en annexe 7, qui enregistre les résultats dans des fichiers xml. Afin d'obtenir une interprétation graphique des résultats, j'ai aussi effectué le blast à partir



de l'interface web de NCBI. Pour les deux séries de blast, j'ai choisi comme option de blaster contre la base de données nr, et d'utiliser blastn, c'est-à-dire de rechercher des séquences assez similaires.

Le résultat graphique du blast des cinq contigs est disponible à l'annexe 13. Au premier regard, on remarque que tous les résultats sont à peu près pareils. Un regard rapide aux fichiers xml confirme ce fait. Tous les résultats ont le même total score, environ la même longueur de hit, et ont tous une e-value de 0.

### c Quelles sont les fonctions prédites par les différents contigs choisis ?

Pour essayer de prédire les fonctions, je vais regarder les informations des fichiers genbank des onze premiers résultats blast de chaque contig choisi. Pour ce faire, j'utilise le script en annexe 14.

Quatre des contigs choisis semblent avoir une fonction commune. En effet, pour les contigs 7, 195, 209 et 214, on retrouve des annotations semblables dans les fichiers genbank obtenus des résultats des blast. On y retrouve des annotations pour deux gènes et deux CDS encodant les protéines suivantes : *orf1ab polyprotein* ou *polyprotein orf1ab* (PP1ab), ou encore *orf1a polyprotein* ou *polyprotein orf1a* (PP1a). Les fichiers genbank commençant par JX ont tous les annotations de gène et de CDS, tandis que les fichiers commençant par d'autres lettres ont seulement les annotations CDS. De plus, comme les résultats des blasts sont toutes différentes formes du virus, chaque annotation des protéines encodées fait référence à un fichier genbank différent dans la base de données des protéines.

Les fichiers dénotent aussi que les résultats du blast ne représentent qu'une partie de ces deux formes du gène. En effet, la séquence couverte par les résultats du blast du contig 7 fait 799bp (*query cover* de 89%), celle du contig 195 fait 1368bp (*query cover* de 89%), les résultats du contig 209 font 762bp (*query cover* de 89%) et finalement les résultats du contig 214 font 1042bp (*query cover* de 86%), alors que la taille de la protéine, selon les fichiers consultés, est de 7013 acides aminés, donc beaucoup plus grand. Aucune information précise n'est disponible concernant le rôle possible de ces deux protéines.

Les annotations des résultats du blast du contig 132 font aussi référence à un gène, ayant comme nom "S" et encodant la protéine *spike glycoprotein precursor [SARS coronavirus]*. Toutefois, un des dix fichiers consultés (AY559090.gb) n'a aucune annotation concernant ce gène ou cette région codante. Encore une fois, le contig ne représenterait qu'une partie de la région codante ou du gène, car la protéine encodée a une taille de 1255 acides aminés. On peut toutefois prédire la fonction de ce contig, qui ferait donc partie d'un gène encodant cette protéine. On ne peut affirmer quel rôle cette protéine peut jouer toutefois.

### d Quel(s) sont le(s) génome(s) dans lesquels vous trouvez des séquences fortement similaires ? Que pouvez-vous en déduire ?

En consultant les fichiers genbank des cinq contigs choisis, j'ai remarqué que certains génomes du Virus du SARS apparaissent dans les résultats de blasts différents. Entre autres, deux résultats de blast sont présents dans les cinq contigs, quatre de cinq contigs ont six résultats en commun.

Les génomes qui sont communs à tous sont le *SARS coronavirus HKU-39849 isolate recSARS-CoV HKU-39849, complete genome* et le *SARS coronavirus HKU-39849 isolate UOB, complete genome*, alors que les génomes de

*SARS coronavirus isolate Tor2/FP1-10851, FP1-10895, FP1-10912, FP1-10851* font parties des résultats de blast des contigs 7, 132, 195 et 214.

Il serait donc intéressant de poursuivre une recherche plus approfondie des contigs avec ces génomes, car considérant les scores obtenus pour les blasts, et le fait qu'on retrouve ces résultats dans plusieurs contigs, il y a de fortes chances que les contigs proviennent de ce génome.

### 3 Question 3

#### a Identificateur du vecteur pANNE.

Afin d'identifier toutes les régions du vecteur donné, j'ai écrit un script qui fait un blast, retire le premier hit du blast du vecteur, et qui refait un autre blast, jusqu'à ce que la séquence soit trop petite pour obtenir un résultat intéressant (23). J'ai utilisé l'option megablast pour obtenir des séquences fortement similaire. J'ai aussi modifié le script pour ne pas concaténer les sous-séquences restantes, pour vérifier si cela affectait le résultat. J'ai obtenu les mêmes résultats pour la série de blast, il était toutefois plus facile de reconstruire une représentation graphique du résultat en ne concaténant pas les parties restantes du vecteur.

Le vecteur semble donc être la composition de trois séquences différentes. La figure ?? représente en image ce résultat. Les couleurs des séquences correspondent à leur équivalent dans la séquence du résultat de blast. La flèche indique l'orientation de la séquence. Par exemple, dans le vecteur pHT2, on peut remarquer qu'un des hits s'est fait de façon plus/minus, donc en prenant la séquence complémentaire. De plus, on peut y voir un chevauchement pour certains résultats des hits.

Le vecteur pANNE semble provenir en grande partie de l'*Expression vector pHT2, complete sequence*, avec deux petites séquences provenant de *PgeneClip hMGFP Vector, complete sequence* et de *Cloning Vector EN.Cherry, complete sequence*. J'ai créé cette représentation à l'aide de la librairie GenomeDiagram, de biopython. Le script est disponible en annexe ??.

### 4 Question 4

#### a Aligner ces CDS en utilisant ClustalW, dialign et Mavid

Afin de trouver huit orthologues du gène FOXP4, j'ai tout d'abord cherché ce gène dans la base de données des gènes de NCBI. Une série de résultat était disponible pour ce gène, chez différentes espèces. J'ai choisi de trouver des orthologues du gène chez l'humain. J'ai donc choisi la page du gène FOXP4 chez l'*Homo sapiens*, et j'ai ensuite chargé le fichier genbank de la séquence référence de ce gène.

À partir du fichier genbank, j'ai demandé à NCBI de blaster la région sur la base de données nr. J'ai tout d'abord tenté d'utiliser blastn, toutefois après cinq minutes NCBI a arrêté la recherche car elle prenait trop de mémoire. J'ai alors utilisé l'option megablast pour avoir des séquences hautement similaires (graphique du résultat en annexe 15). J'ai alors analysé les résultats pour obtenir les CDS de huit orthologues. J'ai donc laissé de côté tous

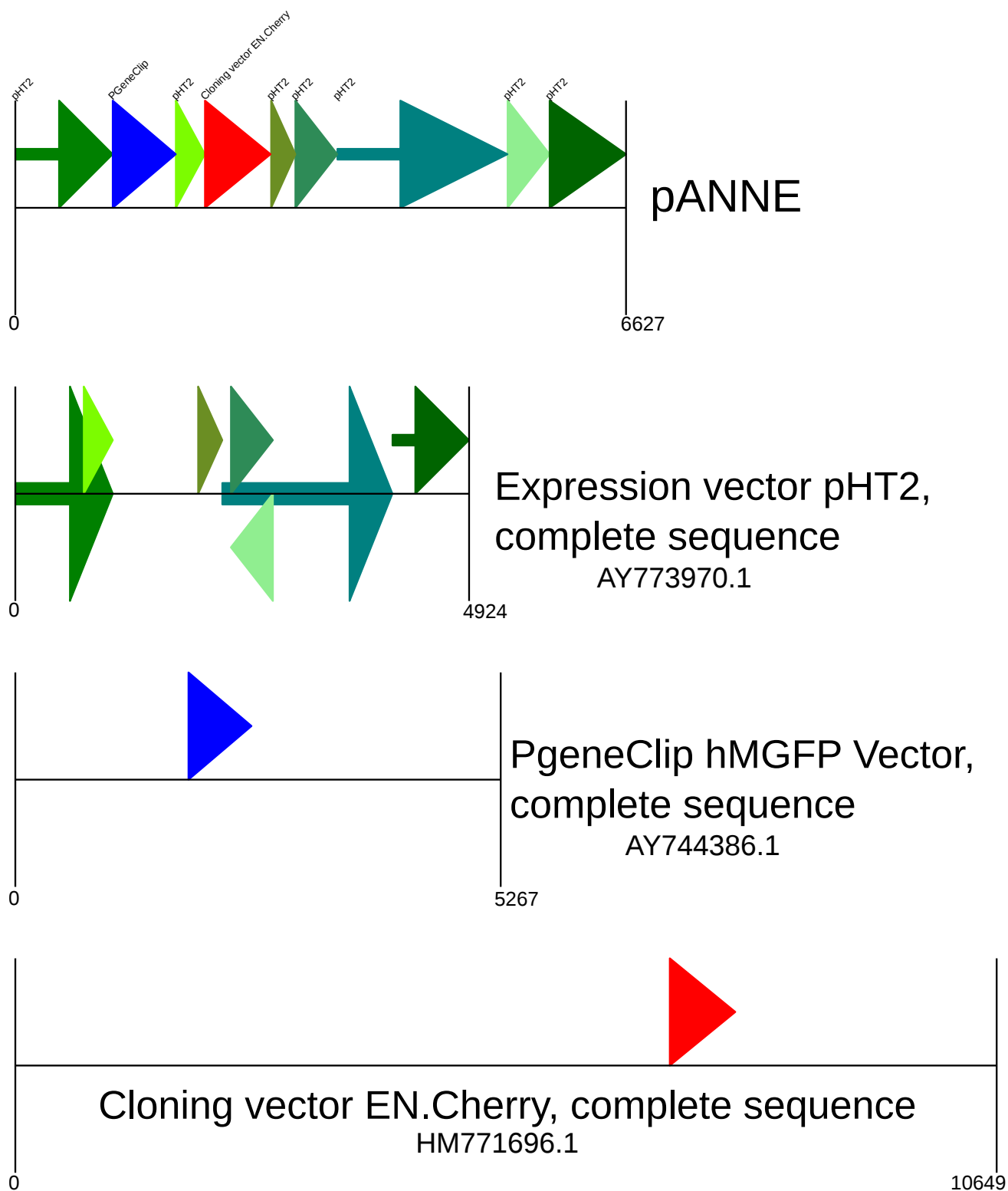


FIGURE 1 : Représentation graphique de la composition du vecteur pANNE

les résultats chez l'humain, et aussi tous les résultats dont les fichiers genbank ne contenait pas d'annotation pour des CDS. Pour les séquences choisies, j'ai aussi ignoré les courtes séquences, et je me suis concentré sur le résultat principal.

Les huit séquences choisies proviennent des espèces suivantes : *Pan troglodytes*, *Pan paniscus*, *Gorilla gorilla gorilla*, *Pongo abelii*, *Nomascus Leucogenys*, *Macaca fascicularis*, *Macaca mulatta* et *Saimiri boliviensis*. Pour chacune des espèces, j'ai vérifié les annotations du fichier genbank pour m'assurer que le CDS du fichier correspondait au gène FOXP4. J'ai ensuite téléchargé les CDS en format fasta, tel qu'offert par l'interface de NCBI. Finalement, j'ai créé un fichier multiple fasta pour ces huit séquences en copiant/collant dans un nouveau fichier le contenu des huit fichiers.

J'ai ensuite installé ClustalW, Dialign et Mavid sur mon ordinateur (information sur le système en annexe 16).

## **b Discutez de la performance en terme de facilité d'utilisation et de rapidité des différents programmes.**

J'ai d'abord testé ClustalW [16]. J'ai utilisé la commande suivante pour lancer l'alignement : `time clustalw2 foxp4.ortho.fa > clustalw.log` (log de l'exécution en annexe 17). Ensuite, j'ai fait le même processus avec Dialign [17]. J'ai utilisé la commande suivante pour lancer l'alignement : `time dialign -n foxp4.ortho.fa`. Ici, le programme ne produit pas de log de son exécution. J'ai utilisé l'option -n car je tentais d'aligner des séquences nucléotidiques, et non des séquences de protéine, ce qui est le défaut de Dialign.

Finalement, j'ai effectué l'alignement avec Mavid [?]. Le processus fut beaucoup plus difficile. Pour lancer Mavid, j'ai dû modifier le script pour créer l'arbre guide auparavant. Ensuite Mavid demandait un fichier de masquage, et j'ai passé plusieurs heures à tenter d'installer localement RepeatMasker [18] pour effectuer le masquage, lorsque cela n'a pas fonctionné de faire le masquage à l'aide d'une application en ligne, et finalement lorsque le résultat de RepeatMasker était qu'il n'y avait pas de séquence répétitive (annexe 18). Après plusieurs recherches, j'ai créé un fichier de masquage vide pour les séquences, c'est-à-dire une copie du fichier multifasta avec l'extension .masked, et j'ai ensuite lancé Mavid avec la commande suivante : `time ./mavid.pl foxp4.ortho.fa` (log de l'exécution en annexe 19).

Le temps d'exécution des applications est disponible en annexe 24. Mavid semble être le programme le plus rapide pour créer l'alignement, toutefois son manque de documentation rend son utilisation très difficile. De plus, il requiert une étape de plus que les autres applications. Dialign est plus lent, il pourrait aussi donner plus d'information à l'utilisateur sur son fonctionnement. ClustalW semble être relativement rapide, tout en étant facile à utiliser.

## **c Analysez et discutez de la qualité de l'alignement donné par chaque méthode.**

Regardons d'abord le résultat obtenu par ClustalW. À première vue, l'alignement semble être de qualité, on y remarque très peu de gap et de mismatch. On peut s'attendre à ce résultat selon l'arbre guide créé par ClustalW. En général, ClustalW donne un résultat de qualité si on utilise des séquences partageant un ancêtre commun. Comme nous avons choisi des orthologues d'un gène, on peut donc s'attendre à un résultat de qualité. Comme

on peut le voir en 20, la distance dans l'arbre guide est assez petite ici, on peut donc s'attendre à un résultat de qualité.

Les repères pour la qualité de Dialign sont plus difficiles à évaluer. Le programme crée lui aussi un arbre guide (21), qui est plus balancé que celui de ClustalW. En regardant le fichier de sortie, on peut remarquer des scores d'alignement assez élevé pour la majorité du fichier, on retrouve à quelques endroits une série de gap, mais en général le résultat semble très similaire à celui de ClustalW.

Finalement, le fichier d'alignement créé par Mavid ne contient aucune information à propos de l'alignement. Mavid produit un fichier mfa, donc multiple fasta. On peut y remarquer des gaps assez long à certains endroits. Pour plus d'information, Mavid produit aussi un fichier phylip plus facile à analyser. On y remarque certaines régions de gap, mais il est très difficile de remarquer les mismatches. L'arbre guide produit (22) ressemble à celui de Dialign, mais les distances entre les espèces ne sont pas aussi grandes.

## **d Quel est votre meilleur alignement ? Justifiez votre choix.**

Le choix d'un alignement dépend de ce qu'on veut en faire. Pour en faire une analyse plus poussée, il est généralement nécessaire d'éditer l'alignement multiple, selon les connaissances des séquences qui ont été alignées. Un choix d'alignement pourrait donc être basé sur la minimisation du travail d'édition à faire sur le résultat.

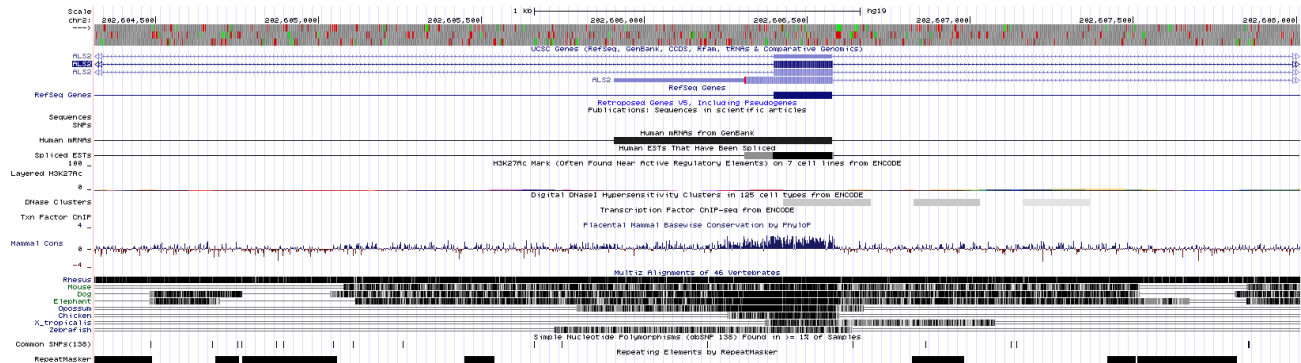
On peut donc éliminer l'alignement de Mavid, car le format mfa qu'il produit semble nécessiter plus de travail, ne serait-ce que pour avoir une représentation montrant clairement où sont les mismatches.

Il est plus difficile de départager entre les alignements de ClustalW et Dialign. Selon Mower et Annaiah [19], Dialign excelle lorsque les séquences ont une distance évolutive très élevée. Ce n'est pas le cas ici, donc il serait préférable de prendre l'alignement multiple produit par ClustalW. De plus, ClustalW produit un résultat plus rapidement que Dialign. Ce n'est pas un critère très important ici, car les séquences alignées sont assez courtes, mais pour des séquences plus longues, ce serait un point à considérer.

Je crois donc que le meilleur alignement est celui que ClustalW a produit.

## 5 Annexes

### 1 Image du locus du gène ALS2 chez l'*Homo sapiens* du UCSC genome browser



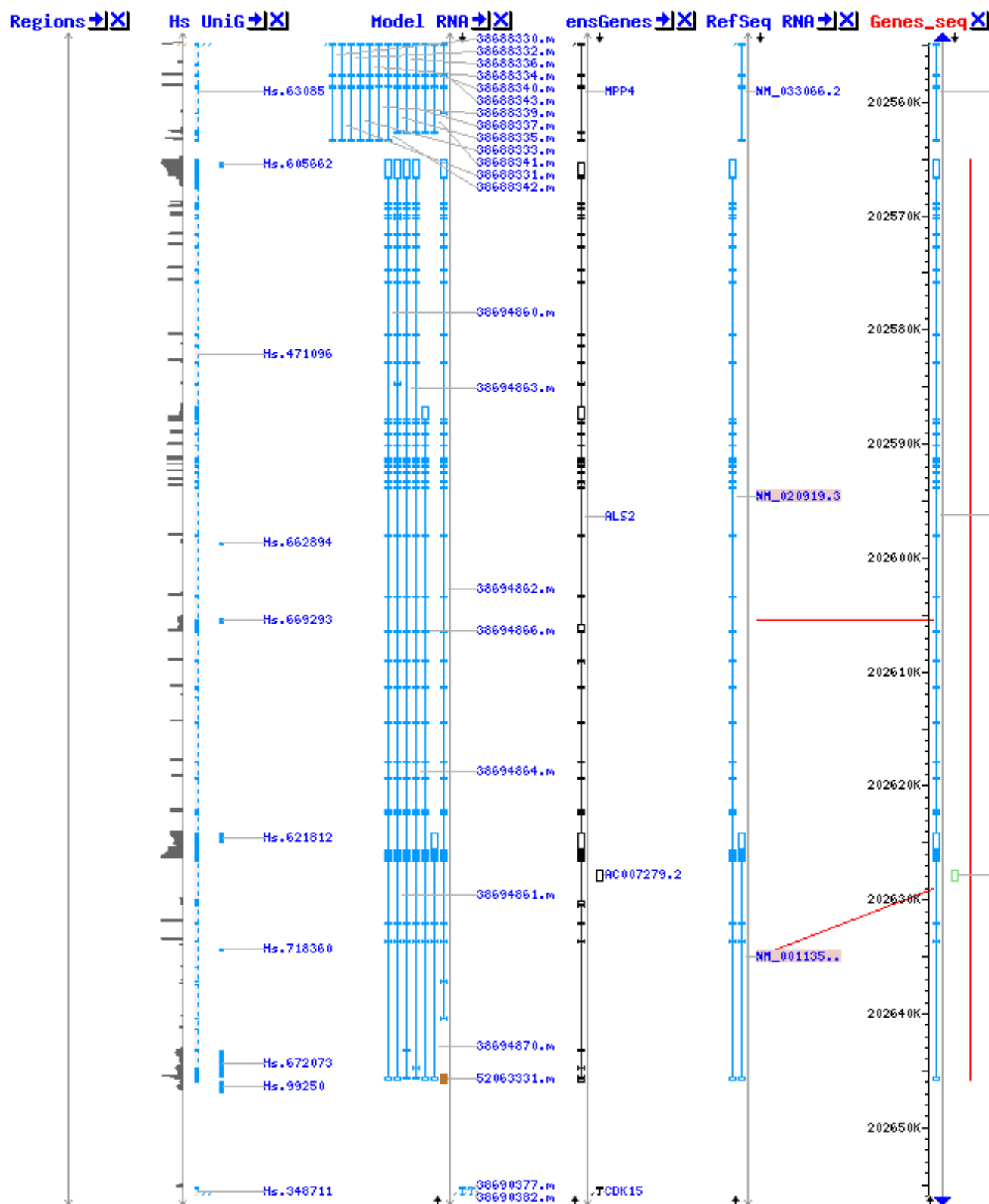
### 2 Image du locus du gène ALS2 chez l'*Homo sapiens* de NCBI



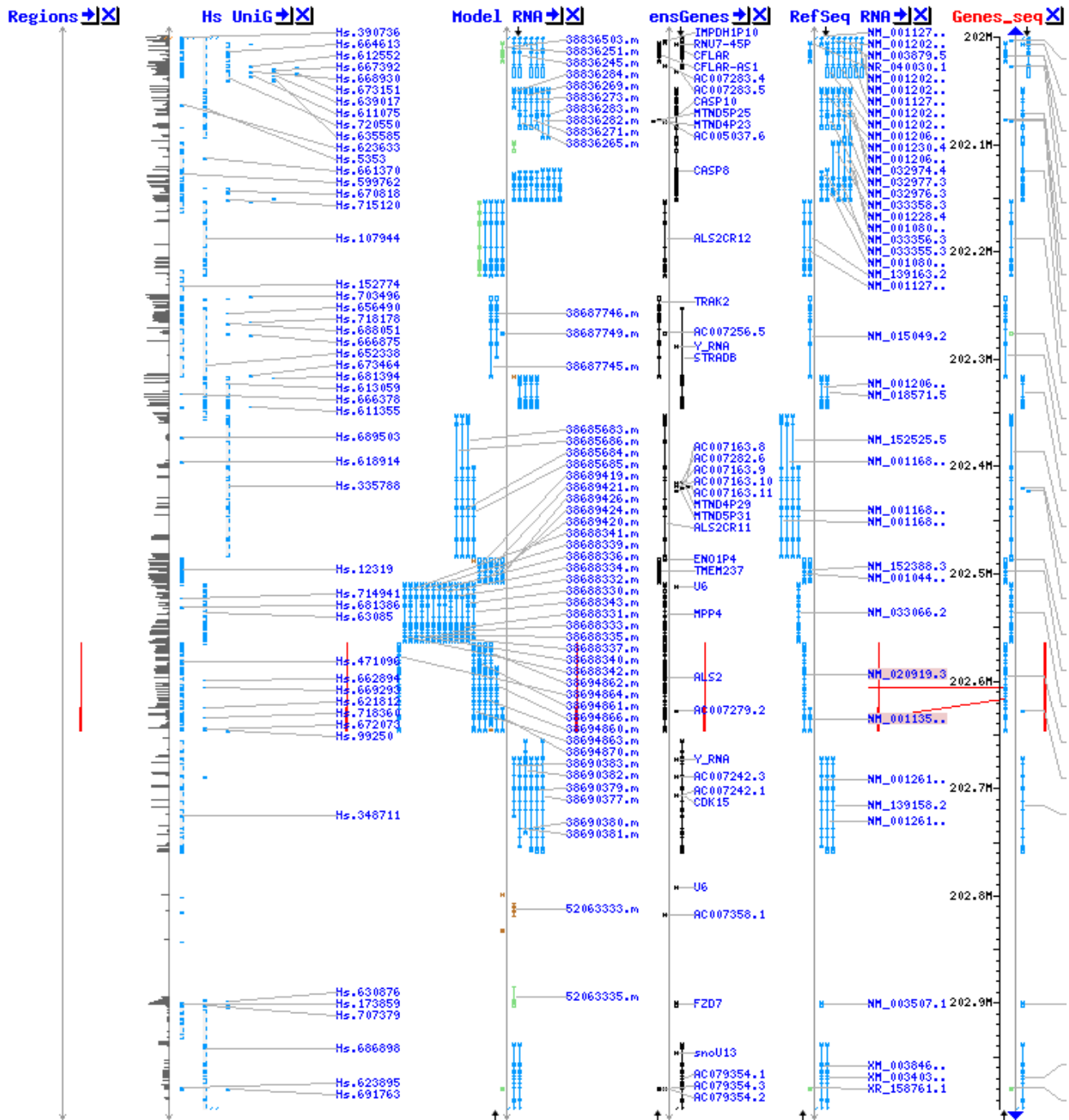
### 3 Script biopython de calcul des fréquences nucléotidiques

```
1 # -*- coding: utf-8 -*-#
2 from Bio import SeqIO
3 from Bio.SeqRecord import SeqRecord
4 handle = open("NC_000002.202564986-202645895.gb", "r")
5 seq_record = SeqIO.parse(handle, 'gb')
6 for seq in seq_record:
7     dist_a = seq.seq.count("A")
8     dist_c = seq.seq.count("C")
9     dist_g = seq.seq.count("G")
10    dist_t = seq.seq.count("T")
11    print "A: count: " + str(dist_a) + " %=" + \
12          str(float(dist_a)/len(seq)*100)
13    print "C: count: " + str(dist_c) + " %=" + \
14          str(float(dist_c)/len(seq)*100)
15    print "G: count: " + str(dist_g) + " %=" + \
16          str(float(dist_g)/len(seq)*100)
17    print "T: count: " + str(dist_t) + " %=" + \
18          str(float(dist_t)/len(seq)*100)
19    print "total=" + str(dist_a+dist_c+dist_g+dist_t)
```

#### 4 Vue de MapViewer du locus du gène ALS2 chez l'humain, 202,555 à 202,656.



## 5 Vue de MapViewer du locus du gène ALS2 chez l'humain, 202M à 203M.



## 6 Vue de SequenceViewer du locus du gène ALS2.

## 7 Script biopython Pour choisir 5 contigs au hasard à partir du résultat de CAP3, et d'effectuer un blast sur ces contigs

```
1 # -*- coding: utf-8 -*- #
```



```

2 import random
3 from Bio.Blast import NCBIWWW
4
5 contigs = {}
6 contig_no = None
7 contig_seq = ""
8 contig_size = 0
9
10 with open("seq.data.cap.contigs", "r") as f:
11     for line in f:
12         # on regarde d'abord si c'est un contig ou non
13         if line[0] == '>':
14             if contig_no == None:
15                 contig_no = int(line[7:])
16             if contig_seq != "":
17                 contigs.update({contig_no:contig_seq})
18                 contig_seq = ""
19                 contig_no = int(line[7:])
20                 contig_size+= 1
21             else :
22                 contig_seq = contig_seq + line.replace("\n","")
23             contigs.update({contig_no:contig_seq})
24             contig_size +=1
25
26 # Maintenant, on a nos contigs, on en choisit 5 au hasard
27 random_contig = []
28
29 #Je m'assure ici de ne pas avoir de doublon
30 for i in range(5):
31     random_c = random.randint(1, contig_size)
32     while random_c in random_contig:
33         random_c = random.randint(1,contig_size)
34     random_contig.append(random_c)
35
36 #On blast maintenant les contigs choisis:
37 for i in random_contig:
38     result_handle = NCBIWWW.qblast("blastn", "nr", contigs[i])
39
40     #on enregistre le r sultat
41     nom_fichier = "blast_contig-" + str(i) + ".xml"
42     save_file = open(nom_fichier, "w")
43     save_file.write(result_handle.read())
44     save_file.close()
45     result_handle.close()
46
47 print "5_contigs_cherch s"

```

## 8 Information à propos de la qualité du contig 7

```
>Contig7
```

[illegible]

[illegible]

## 9 Information à propos de la qualité du contig 132

```
>Contig132
```

[illegible]

20 5 20 20 20 20 15 15 15 15 0 15 15 15 15 15 15 15 15 15  
 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15  
 0 15 15 15 15 15 15 15 0 0 15 15 15 0 15 15 15 0 15 15  
 15 15 15 15 15 15 15 0 0 0 0 15 15 15 15 15 15 15 15  
 15 15 0 15 15 15 15 15 15 15 0 15 15 0 15 15 15 15 0  
 15 15 15 0 15 15 15 15 15 15 15 0 15 15 15 15 0 15 15  
 15 0 15 0 15 15 15 15 0 15 15 15 15 15 15 15 15 15 15  
 15 0 15 15 15 15 15 15 15 15 15 15 15 15 15 0 0 15 15  
 15 15 15 15 15 15 15 15 15 15 15 15 15 0 15 0 15 15 15  
 0 15 15 15 15 15 15 10 10 10 10 10 10 10 10 10 10 10 10  
 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10  
 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10  
 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10  
 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10  
 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10  
 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10  
 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10  
 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10  
 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10

## 10 Information à propos de la qualité du contig 195

>Contig195

15 0 15 15 15 15 0 15 15 15 15 15 15 15 15 15 0 15 15  
 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15  
 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15  
 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15  
 15 15 15 15 15 15 15 15 15 0 0 0 0 0 0 0 0 0  
 0 0 0 0 0 15 15 15 15 15 15 15 15 15 15 15 15 15 15  
 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15  
 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15  
 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15  
 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15  
 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15  
 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15  
 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15  
 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15  
 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15

[illegible]

[illegible]

## 11 Information à propos de la qualité du contig 209

```
>Contig209
```

[illegible]

25	25	25	25	25	25	25	25	25	25	25	25	25	25	25	25	25	10	25	25
25	10	25	25	25	25	25	25	25	25	25	25	25	25	25	10	25	25	25	25
25	25	25	25	25	25	25	25	25	25	10	25	25	25	25	25	25	25	25	25
25	25	25	25	25	25	25	10	25	25	25	25	25	25	25	25	25	25	25	25
25	25	25	25	10	25	25	25	25	25	25	25	25	25	25	25	25	25	25	25
25	25	25	25	25	25	10	25	25	25	25	25	10	25	25	25	25	10	25	25
25	25	25	25	25	10	25	25	25	25	25	25	25	10	25	25	25	25	10	10
25	25	10	10	10	25	25	25	25	0	0	25	10	10	10	25	25	25	10	25
25	25	25	25	25	25	0	10	25	25	25	25	25	25	25	10	25	25	25	25
25	25	25	25	25	25	25	25	25	25	25	25	10	25	25	20	20	20	20	20
20	20	20	20	20	20	20	5	20	5	20	20	20	5	20	15	15	15	15	15
15	15	15	15	15	15	15	15	15	15	15	15	15	15	15	10	10	10	10	10
10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10
10	10	10	10	10	10	10	10	10	10	10									

## 12 Information à propos de la qualité du contig 214

```
>Contig214
```

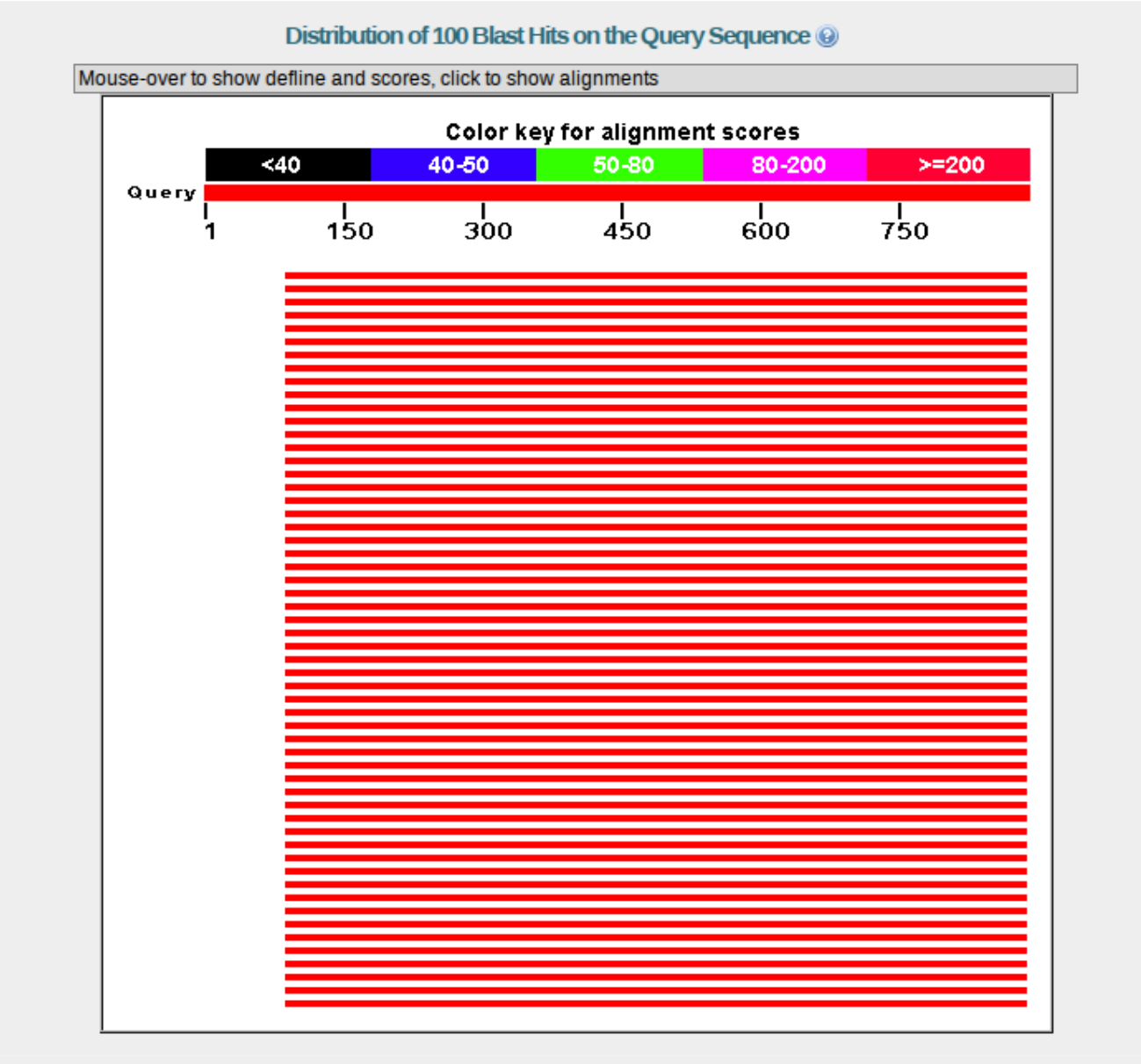
[illegible]

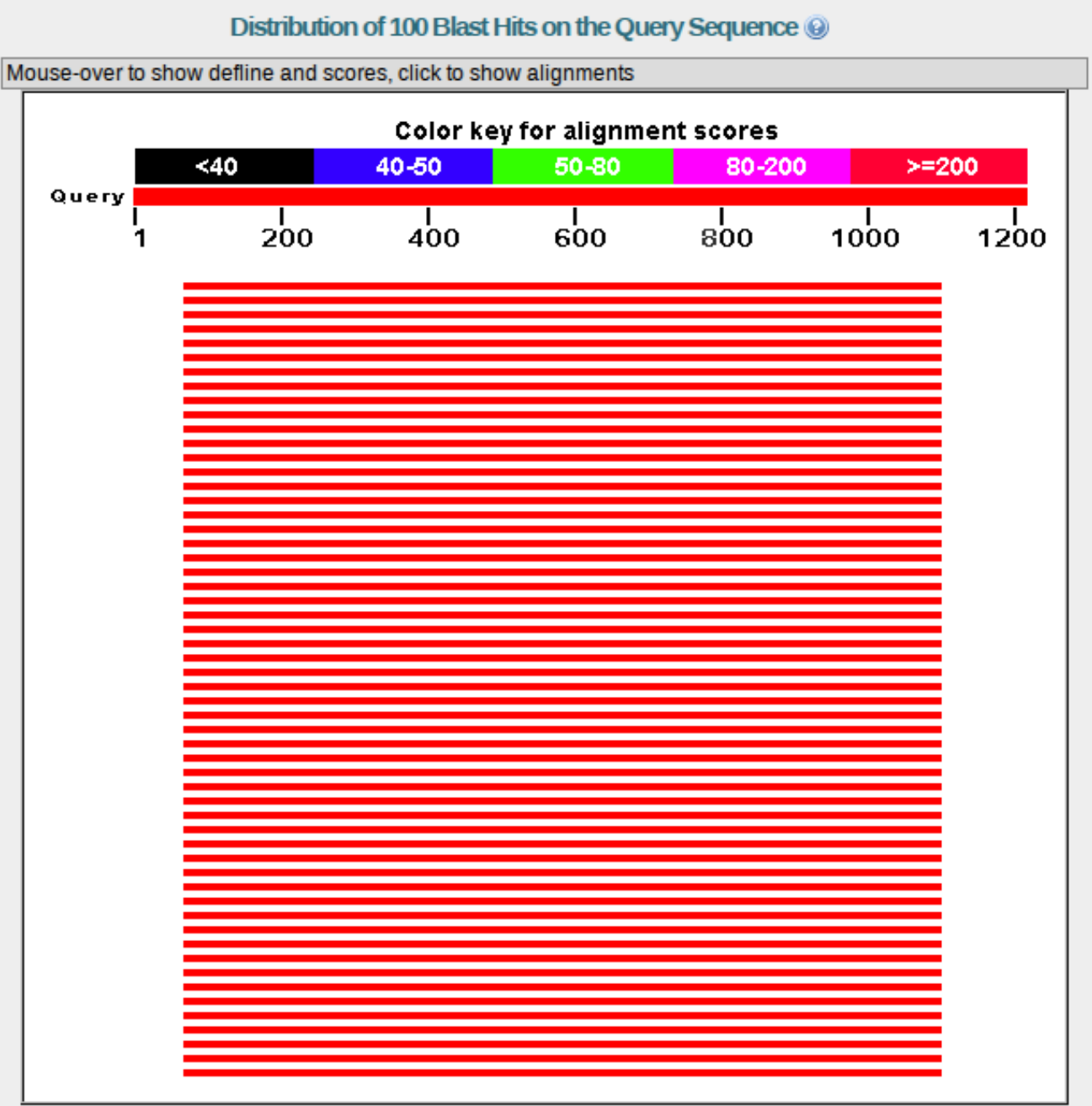
[illegible]

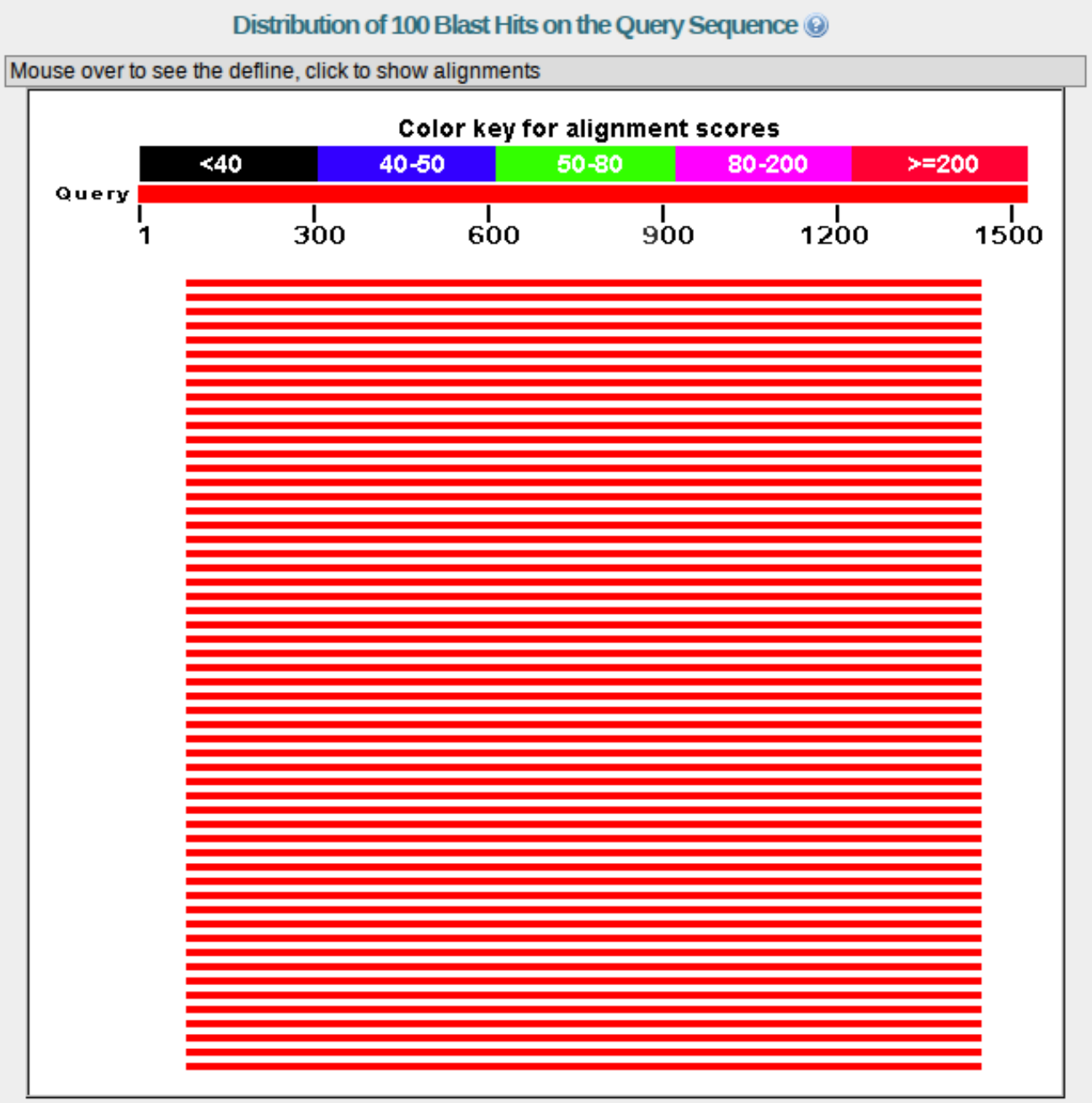


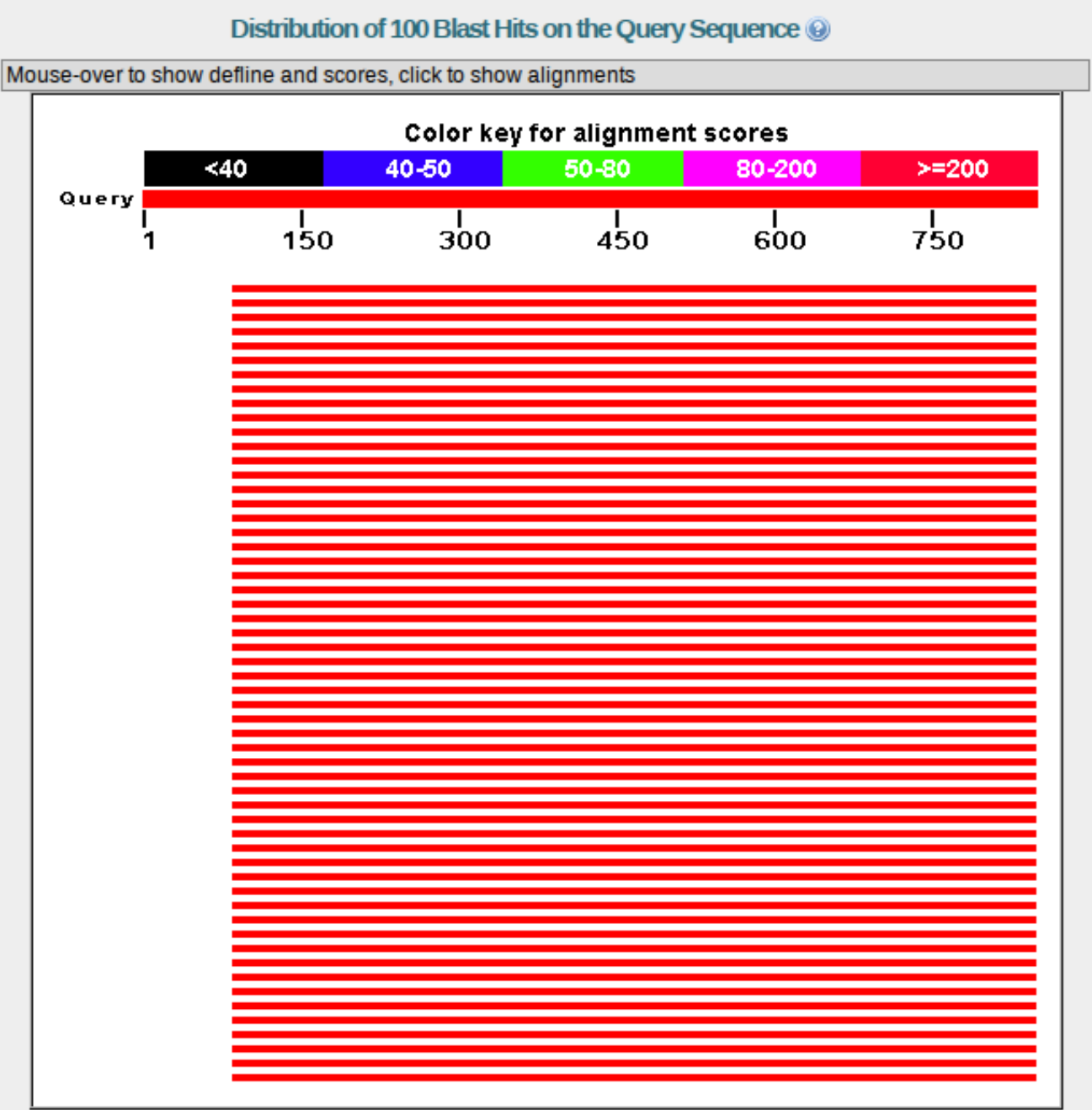
13    Résultat graphique bu blast des 5 contigs.

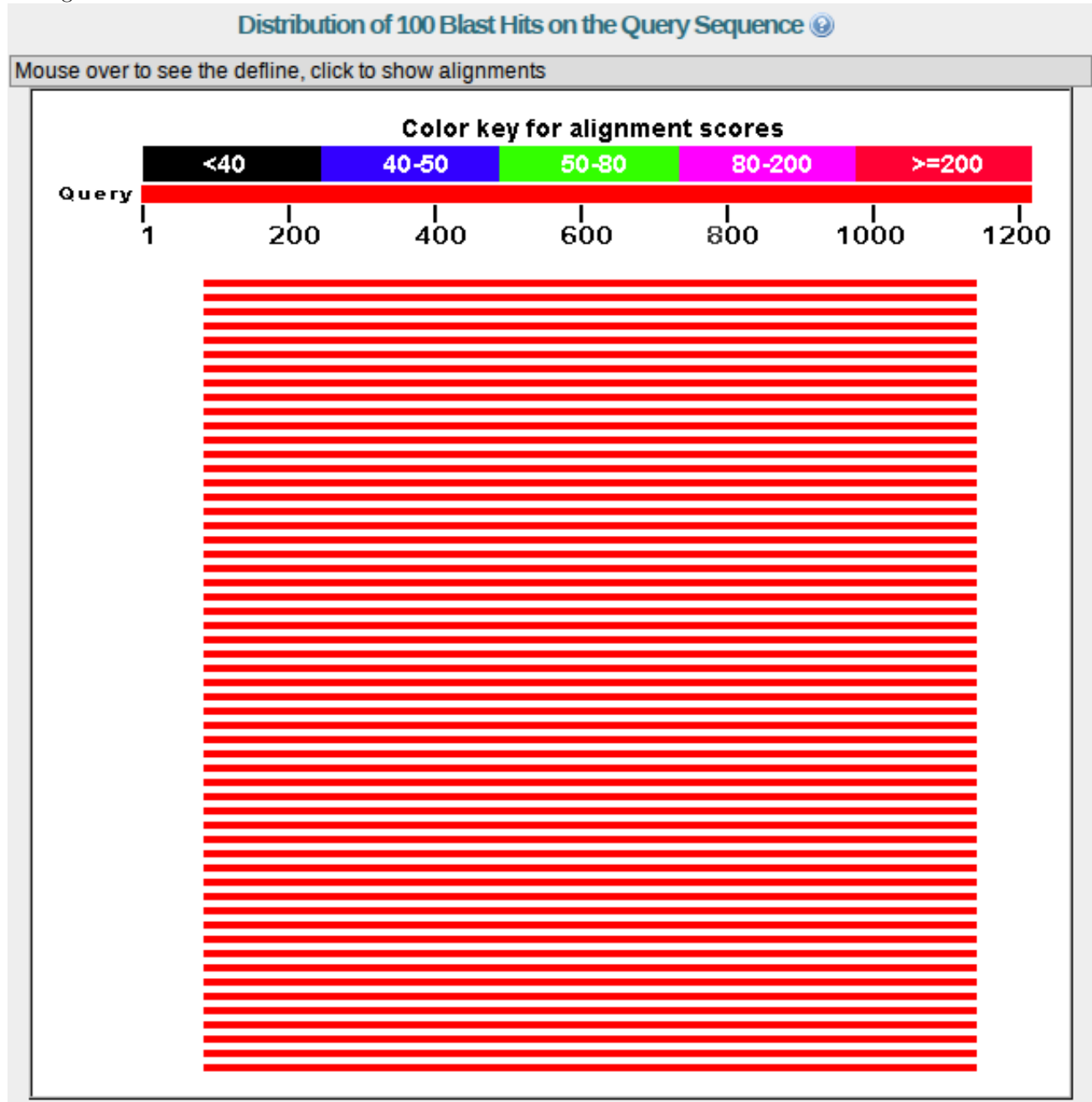
Contig 7











#### 14 Script Biopython pour obtenir les fichiers d'accension des 10 premiers résultats du blast pour un contig donné.

```

1 # -*- coding:utf-8 -*- #
2
3 #Parser pour un fichier XML de resultat blast
4 #Specifique a la question 2 du devoir 1. Je sais
5 #ici que chaque hit a seulement un hsp, donc en
6 #specifiant le # d'accension et le sbjct_start et end,

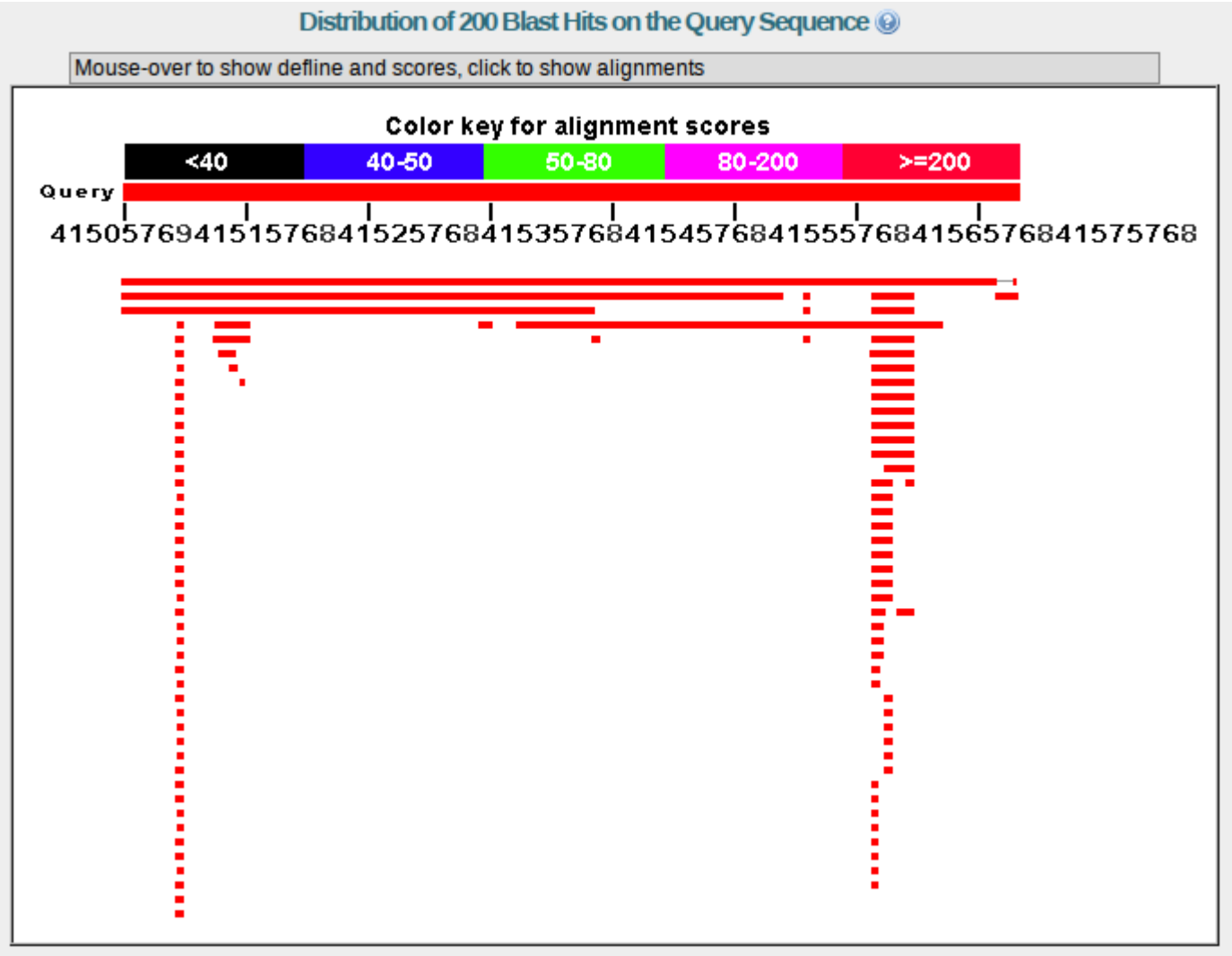
```

```

7  #j'obtiens ce que je cherche
8
9  import sys
10 import os
11 from Bio.Blast import NCBIXML
12 from Bio import SeqIO
13 from Bio.SeqRecord import SeqRecord
14 from Bio import Entrez
15
16 #On choisit une E-VALUE
17 E_VALUE_THRESH = 0.04
18 Entrez.email = "glahaie@gmail.com"
19 path = "annexes/question_2/"
20
21 path_fichier = path + "blast_contig-"+sys.argv[1] + ".xml"
22
23 with open(path_fichier) as fichier:
24     blast_record = NCBIXML.read(fichier)
25     i = 0
26     path_result = path + "contig-"+sys.argv[1]+"/"
27     if not os.path.exists(path_result):
28         os.makedirs(path_result)
29     for alignment in blast_record.alignments:
30         for hsp in alignment.hsps:
31             if hsp.expect < E_VALUE_THRESH:
32 #On obtient alors le fichier genbank
33                 handle = Entrez.efetch(db="nucleotide", rettype="gb",
34                     retmode="text", id=alignment.accession,
35                     seq_start=hsp.sbjct_start, seq_stop=hsp.sbjct_end)
36                 seq_record= SeqIO.read(handle, "gb")
37                 handle.close()
38                 nom_fichier = path_result + alignment.accession + ".gb"
39                 SeqIO.write(seq_record, nom_fichier, "gb")
40
41             i += 1
42         if i > 10:
43             break

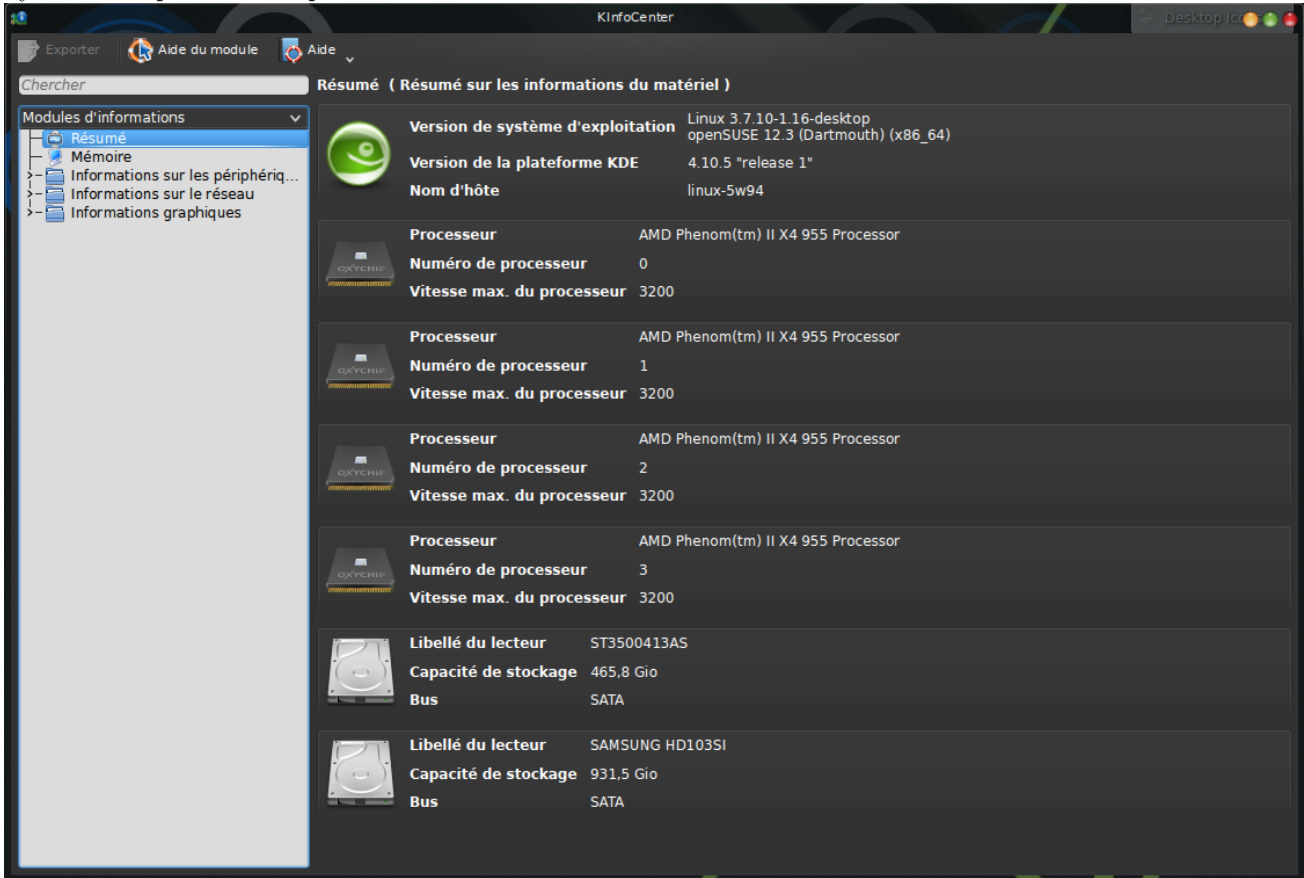
```

15    Résultat graphique du blast du gène FOXP4 de l'*Homo sapiens*.











## 16 Information du système qui a fait les alignements multiples.

Système d'exploitation et processeurs

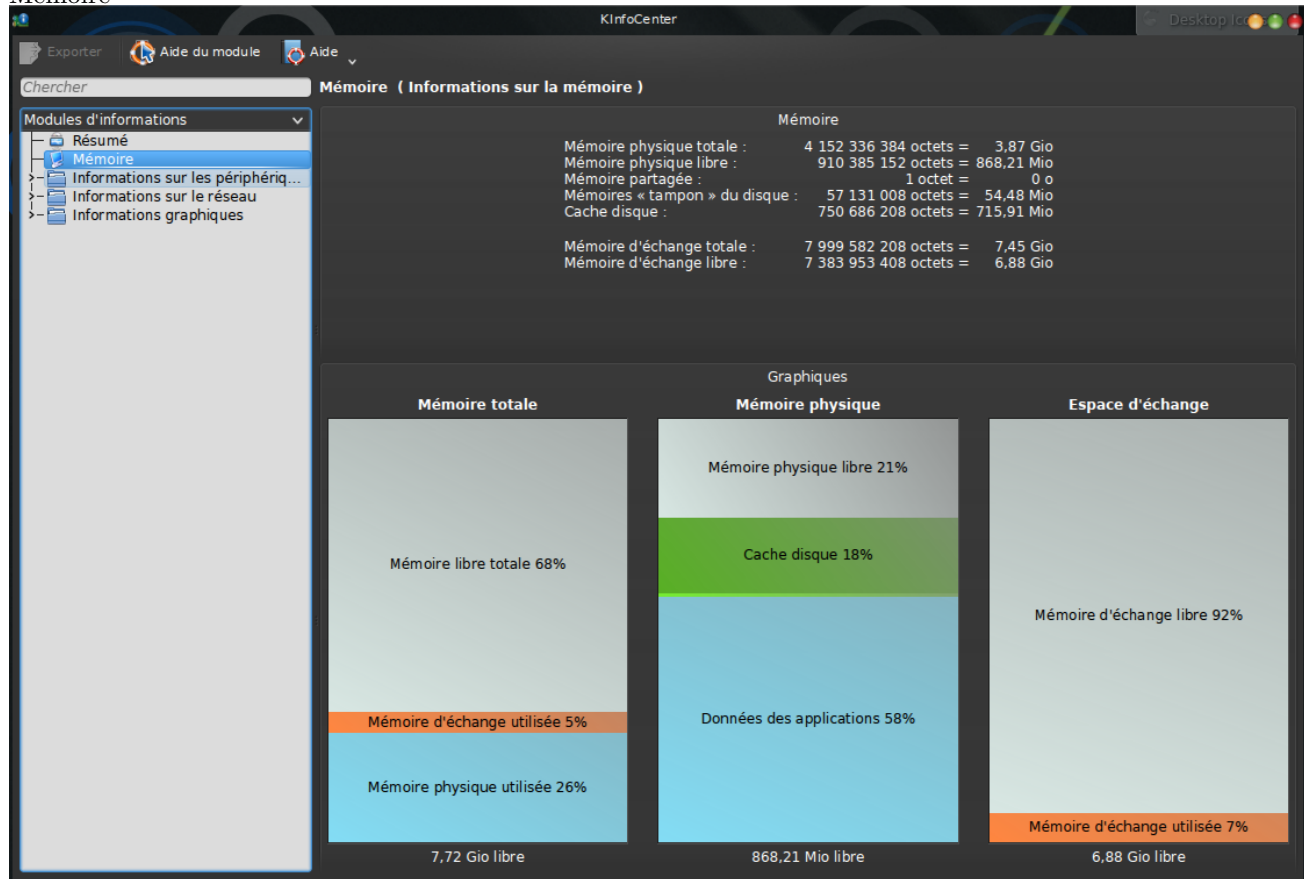


The screenshot shows the KInfoCenter application window. The left sidebar contains a tree view under 'Modules d'informations' with the following items: 'Résumé' (selected), 'Mémoire', 'Informations sur les périphériq...', 'Informations sur le réseau', and 'Informations graphiques'. The main area displays the 'Résumé ( Résumé sur les informations du matériel )' section. It lists system details such as the operating system version (Linux 3.7.10-1.16-desktop, openSUSE 12.3 (Dartmouth) (x86\_64)), KDE platform version (4.10.5 "release 1"), and host name (linux-5w94). Below this, it lists four processors, all identified as 'AMD Phenom(tm) II X4 955 Processor', with processor numbers 0, 1, 2, and 3, and a maximum speed of 3200. At the bottom, it lists two storage devices: a 'ST3500413AS' drive with a capacity of 465,8 Gio and a 'SAMSUNG HD103SI' drive with a capacity of 931,5 Gio, both using SATA buses.

Résumé ( Résumé sur les informations du matériel )	
	<b>Version de système d'exploitation</b> Linux 3.7.10-1.16-desktop openSUSE 12.3 (Dartmouth) (x86_64)
	<b>Version de la plateforme KDE</b> 4.10.5 "release 1"
	<b>Nom d'hôte</b> linux-5w94
	<b>Processeur</b> AMD Phenom(tm) II X4 955 Processor
	<b>Numéro de processeur</b> 0
	<b>Vitesse max. du processeur</b> 3200
	<b>Processeur</b> AMD Phenom(tm) II X4 955 Processor
	<b>Numéro de processeur</b> 1
	<b>Vitesse max. du processeur</b> 3200
	<b>Processeur</b> AMD Phenom(tm) II X4 955 Processor
	<b>Numéro de processeur</b> 2
	<b>Vitesse max. du processeur</b> 3200
	<b>Processeur</b> AMD Phenom(tm) II X4 955 Processor
	<b>Numéro de processeur</b> 3
	<b>Vitesse max. du processeur</b> 3200
	<b>Libellé du lecteur</b> ST3500413AS
	<b>Capacité de stockage</b> 465,8 Gio
	<b>Bus</b> SATA
	<b>Libellé du lecteur</b> SAMSUNG HD103SI
	<b>Capacité de stockage</b> 931,5 Gio
	<b>Bus</b> SATA



## Mémoire



## 17 Log de l'exécution de clustalw2.

### CLUSTAL 2.1 Multiple Sequence Alignments

Sequence format is Pearson

```
Sequence 1: lcl|XM_518463.3_cdsid_XP_518463.2      2058 bp
Sequence 2: lcl|XM_003833312.1_cdsid_XP_003833360.1  2004 bp
Sequence 3: lcl|XM_004043991.1_cdsid_XP_004044039.1  2004 bp
Sequence 4: lcl|XM_002816867.2_cdsid_XP_002816913.1  2043 bp
Sequence 5: lcl|XM_003266293.1_cdsid_XP_003266341.1  2004 bp
Sequence 6: lcl|XM_005553053.1_cdsid_XP_005553110.1  2004 bp
Sequence 7: lcl|NM_001266091.1_cdsid_NP_001253020.1  2043 bp
Sequence 8: lcl|XM_003922988.1_cdsid_XP_003923037.1  2004 bp
```

Start of Pairwise alignments

Aligning...

Sequences (1:2) Aligned. Score: 98

Sequences (1:3) Aligned. Score: 97

Sequences (1:4) Aligned. Score: 97

```

Sequences (1:5) Aligned. Score: 97
Sequences (1:6) Aligned. Score: 96
Sequences (1:7) Aligned. Score: 96
Sequences (1:8) Aligned. Score: 96
Sequences (2:3) Aligned. Score: 99
Sequences (2:4) Aligned. Score: 98
Sequences (2:5) Aligned. Score: 98
Sequences (2:6) Aligned. Score: 97
Sequences (2:7) Aligned. Score: 97
Sequences (2:8) Aligned. Score: 97
Sequences (3:4) Aligned. Score: 98
Sequences (3:5) Aligned. Score: 98
Sequences (3:6) Aligned. Score: 98
Sequences (3:7) Aligned. Score: 97
Sequences (3:8) Aligned. Score: 96
Sequences (4:5) Aligned. Score: 98
Sequences (4:6) Aligned. Score: 98
Sequences (4:7) Aligned. Score: 98
Sequences (4:8) Aligned. Score: 97
Sequences (5:6) Aligned. Score: 98
Sequences (5:7) Aligned. Score: 98
Sequences (5:8) Aligned. Score: 96
Sequences (6:7) Aligned. Score: 99
Sequences (6:8) Aligned. Score: 96
Sequences (7:8) Aligned. Score: 96
Guide tree file created: [foxp4_ortho.dnd]

```

There are 7 groups  
Start of Multiple Alignment

```

Aligning...
Group 1: Sequences: 2      Score:37737
Group 2: Sequences: 2      Score:37091
Group 3: Sequences: 3      Score:37148
Group 4: Sequences: 4      Score:37047
Group 5: Sequences: 5      Score:37481
Group 6: Sequences: 7      Score:37220
Group 7: Sequences: 8      Score:36779
Alignment Score 440506

```

CLUSTAL-Alignment file created [foxp4\_ortho.aln]

## 18 Log du travail de RepeatMasker sur le fichier foxp4\_ortho.fa.

There were no repetitive sequences detected in /usr/local/rmsrver/tmp/RM2\_foxp4\_ortho.fa\_1383262617

## 19 Log de l'exécution de Mavid sur foxp4\_ortho.fa.

```
./utils/randtree/randtree foxp4_ortho.fa
./mavid ./mavid.ph foxp4_ortho.fa
```

```
*****
*
*           Welcome to MAVID.           *
*           (version 2.0, build 4)       *
*
*****
```

```
Aligning 1 versus 1
Aligning [0,2003] to [0,2057]
Aligning 1 versus 2
Aligning [0,2003] to [0,2058]
Aligning 1 versus 1
Aligning [0,2003] to [0,2042]
Aligning 1 versus 1
Aligning [0,2003] to [0,2003]
Aligning 1 versus 2
Aligning [0,2042] to [0,2003]
Aligning 2 versus 3
Aligning [0,2042] to [0,2042]
Aligning 3 versus 5
Aligning [0,2058] to [0,2042]
MAVID worked!
```

```
clustalw2 ./mavid.mfa -tree
```

### CLUSTAL 2.1 Multiple Sequence Alignments

Sequence format is Pearson

```
Sequence 1: lcl|XM_004043991.1_cdsid_XP_004044039.1 2059 bp
Sequence 2: lcl|XM_005553053.1_cdsid_XP_005553110.1 2059 bp
Sequence 3: lcl|XM_518463.3_cdsid_XP_518463.2      2059 bp
Sequence 4: lcl|XM_003266293.1_cdsid_XP_003266341.1 2059 bp
Sequence 5: lcl|NM_001266091.1_cdsid_NP_001253020.1 2059 bp
Sequence 6: lcl|XM_002816867.2_cdsid_XP_002816913.1 2059 bp
Sequence 7: lcl|XM_003833312.1_cdsid_XP_003833360.1 2059 bp
Sequence 8: lcl|XM_003922988.1_cdsid_XP_003923037.1 2059 bp
```

Phylogenetic tree file created: [./mavid.ph]

```
../utils/root_tree/root_tree ./mavid.ph
./mavid ./mavid.ph foxp4_ortho.fa
```

```
*****
*                                     *
*           Welcome to MAVID.         *
*           (version 2.0, build 4)    *
*                                     *
*****
```

```
Aligning 1 versus 1
Aligning [0,2003] to [0,2042]
Aligning 1 versus 1
Aligning [0,2057] to [0,2003]
Aligning 1 versus 2
Aligning [0,2003] to [0,2003]
Aligning 3 versus 1
Aligning [0,2003] to [0,2042]
Aligning 1 versus 4
Aligning [0,2003] to [0,2042]
Aligning 2 versus 5
Aligning [0,2042] to [0,2042]
Aligning 1 versus 7
Aligning [0,2003] to [0,2042]
MAVID worked!
```

```
clustalw2 ./mavid.mfa -tree
```

## CLUSTAL 2.1 Multiple Sequence Alignments

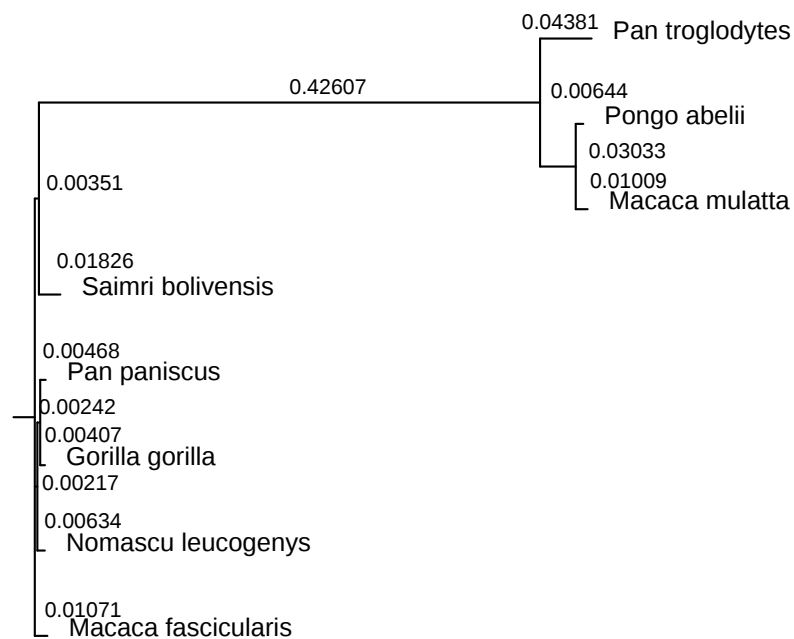
Sequence format is Pearson

```
Sequence 1: lcl|XM_003922988.1_cdsid_XP_003923037.1 2098 bp
Sequence 2: lcl|XM_005553053.1_cdsid_XP_005553110.1 2098 bp
Sequence 3: lcl|NM_001266091.1_cdsid_NP_001253020.1 2098 bp
Sequence 4: lcl|XM_003266293.1_cdsid_XP_003266341.1 2098 bp
Sequence 5: lcl|XM_004043991.1_cdsid_XP_004044039.1 2098 bp
Sequence 6: lcl|XM_518463.3_cdsid_XP_518463.2 2098 bp
Sequence 7: lcl|XM_003833312.1_cdsid_XP_003833360.1 2098 bp
Sequence 8: lcl|XM_002816867.2_cdsid_XP_002816913.1 2098 bp
```

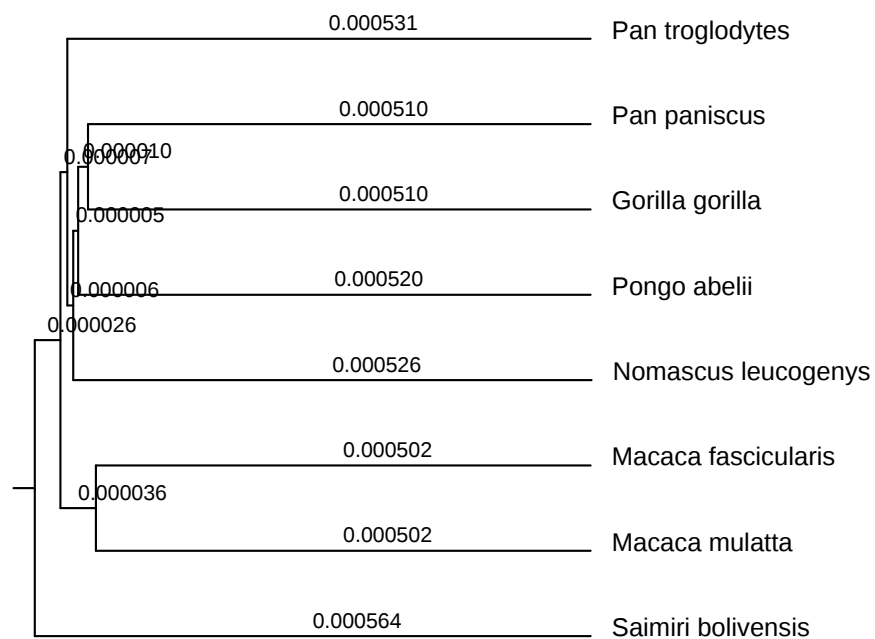
Phylogenetic tree file created: [./mavid.ph]

../utils/root\_tree/root\_tree ./mavid.ph

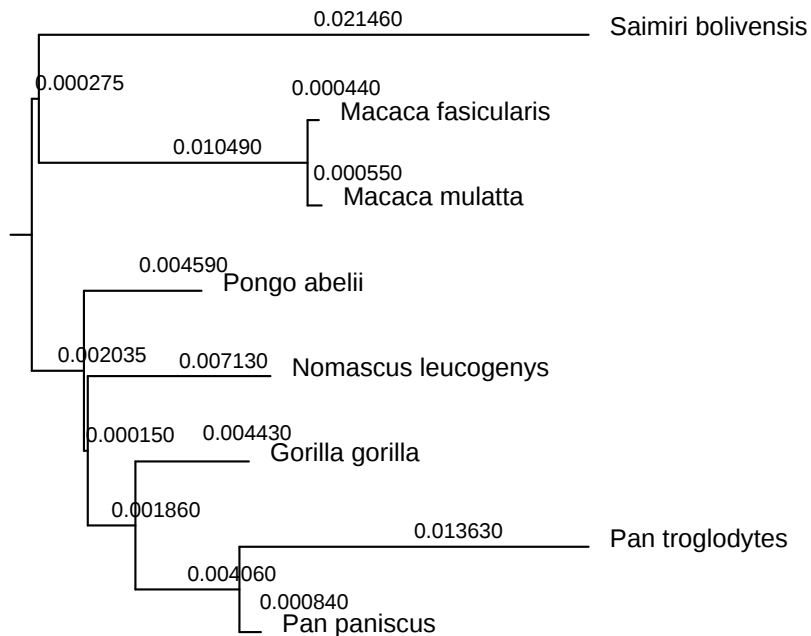
## 20 Arbre guide créé par ClustalW



## 21 Arbre guide créé par Dialign



## 22 Arbre guide créé par Mavid



## 23 Script biopython pour identifier la composition du vecteur pANNE

```
1 # *- coding:utf-8 -* #
2
3 # Script pour le num ro 3 du devoir 1: Cette partie ne fait
4 #qu'envoyer la requ te blast au serveur du NCBI, et ensuite
5 #enregistre le r sultat dans un fichier.
6
7 from Bio.Blast import NCBIWWW
8 from Bio.Blast import NCBIXML
9
10 path_fichier = "annexes/question_3/"
11 nom_resultat = "blast_fichier"
12 LEN_THRESH = 100
13 E_THRESH = 1e-50
14 # Tout d'abord on ouvre le fichier
15
16 sequence = ""
17 with open(path_fichier+"pANNE.txt", 'r') as f:
18     for line in f:
19         sequence = sequence + line.strip()
20
21 #On enl ve les retour de chariot du fichier
22 i = 1
23 while len(sequence) > LEN_THRESH:
24
```

```

25 #Maintenant, on fait le blast
26     print "i=" + str(i)
27     print "on fait un blast sur la s quence de longueur "
28         + str(len(sequence))
29     result_handle = NCBIWWW.qblast("blastn", "nr",
30         sequence, megablast=True)
31
32 #on enregistre le r sultat
33     save_file = open(path_fichier+nom_resultat+str(i)+".xml", "w")
34     save_file.write(result_handle.read())
35     save_file.close()
36     result_handle.close()
37
38     list_start = []
39     list_end = []
40     sequences = []
41 #Maintenant on enl ve de la s quence les zones identifi es
42     with open(path_fichier+nom_resultat+str(i)+".xml", "r") as result:
43         blast_record = NCBIXML.read(result)
44         alignment = blast_record.alignments[0]
45         for alignment in blast_record.alignments:
46             for hsp in alignment.hsps:
47 #On met jour la s quence pour enlever ce r sultat
48                 if hsp.expect < E_THRESH:
49                     list_start.append(hsp.query_start)
50                     list_end.append(hsp.query_end)
51
52             break
53 #On a les points enlever
54 #sort sur les listes
55         list_start.sort()
56         list_end.sort()
57         start = -1
58         end = -1
59         for s_start, s_end in zip(list_start, list_end):
60             if end < 0:
61                 sequences.append(sequence[: s_start - 1])
62                 end = s_start - 1
63             else:
64                 end = s_start - 1
65                 sequences.append(sequence[start: end])
66                 start = s_end - 1
67             sequences.append(sequence[start:])
68             sequence = ""
69         for fragment in sequences:
70             sequence += fragment

```

```

71 #Pour vérifier les résultats, j'enregistre la nouvelle
72 #séquence dans un fichier
73     print "On a écrit le reste de la séquence avec i=" + str(i)
74     with open(path_fichier+"pANNE"+str(i)+".txt", "w") as f:
75         f.write(sequence)
76     i +=1

```

## 24 Temps d'exécution des alignements multiples

Programme	Temps d'exécution
ClustalW	real 0m4.840s
dialign	real 0m18.424s
Mavid	real 0m0.296s

## 25 Script Biopython pour obtenir les fichiers d'accèsion des 10 premiers résultats du blast pour un contig donné.

```

1 #!/ coding: utf-8 -#
2
3 from reportlab.lib import colors
4 from reportlab.lib.units import cm
5 from Bio.Graphics import GenomeDiagram
6 from Bio.Graphics.GenomeDiagram import CrossLink
7 from reportlab.lib import colors
8
9 gd_diagram = GenomeDiagram.Diagram("Composition du vecteur pANNE.txt")
10 gd_track_for_features = gd_diagram.new_track(1, name="Annotated_Features",
11     start=0, end=6627)
12 gd_feature_set = gd_track_for_features.new_set()
13 from Bio.SeqFeature import SeqFeature, FeatureLocation
14
15 colors = [colors.green, colors.lightgreen, colors.teal, colors.darkgreen,
16     colors.seagreen, colors.lawngreen, colors.olivedrab]
17
18 #Essai la main, la lecture des fichiers XML
19 #blast #1: pHT2
20 feature = SeqFeature(FeatureLocation(1, 1056), strand = +1)
21 gd_feature_set.add_feature(feature, name="pHT2", label=True, color=colors[0],
22     sigil="ARROW", arrowhead_length=0.5, arrowshaft_height=0.1)
23 feature = SeqFeature(FeatureLocation(5342, 5798), strand = +1)
24 gd_feature_set.add_feature(feature, name="pHT2", label=True, color=colors[1],
25     sigil="ARROW", arrowhead_length=1, arrowshaft_height=0.1)
26 feature = SeqFeature(FeatureLocation(3495, 5341), strand = +1)
27 gd_feature_set.add_feature(feature, name="pHT2", label=True, color=colors[2],

```



```

28     sigil="ARROW", arrowhead.length=1,arrowshaft_height=0.1)
29 feature = SeqFeature(FeatureLocation(5798, 6627), strand = +1)
30 gd_feature_set.add_feature(feature, name="pHT2", label=True, color=colors[3],
31     sigil="ARROW", arrowhead.length=1,arrowshaft_height=0.1)
32 feature = SeqFeature(FeatureLocation(3039, 3494), strand = +1)
33 gd_feature_set.add_feature(feature, name="pHT2", label=True, color=colors[4],
34     sigil="ARROW", arrowhead.length=1,arrowshaft_height=0.1)
35 feature = SeqFeature(FeatureLocation(1742, 2057), strand = +1)
36 gd_feature_set.add_feature(feature, name="pHT2", label=True, color=colors[5],
37     sigil="ARROW", arrowhead.length=1,arrowshaft_height=0.1)
38 feature = SeqFeature(FeatureLocation(2776, 3039), strand = +1)
39 gd_feature_set.add_feature(feature, name="pHT2", label=True, color=colors[6],
40     sigil="ARROW", arrowhead.length=1,arrowshaft_height=0.1)
41
42 #blast 2:
43 feature = SeqFeature(FeatureLocation(1057, 1741), strand = +1)
44 gd_feature_set.add_feature(feature, name="PGeneClip", label=True, color="blue",
45     sigil="ARROW", arrowhead.length=1)
46
47 #blast 3
48 feature = SeqFeature(FeatureLocation(2058, 2770), strand = +1)
49 gd_feature_set.add_feature(feature, name="Cloning_vector_EN.Cherry", label=True,
50     color="red", sigil="ARROW",arrowhead.length=1)
51
52 #On essayi d'ajouter d'autres track pour repr senter la position des blasts
53 gd_track_for_features = gd_diagram.new_track(1, name="pHT2", start=0, end=4924)
54 gd_feature_set = gd_track_for_features.new_set()
55 feature = SeqFeature(FeatureLocation(2246, 4092), strand = None)
56 gd_feature_set.add_feature(feature, name="pHT2", label=False, color=colors[2],
57     sigil="ARROW", arrowhead.length=0.2,arrowshaft_height=0.1)
58 feature = SeqFeature(FeatureLocation(1, 1058), strand = None)
59 gd_feature_set.add_feature(feature, name="pHT2", label=False, color=colors[0],
60     sigil="ARROW", arrowhead.length=0.2,arrowshaft_height=0.1)
61 feature = SeqFeature(FeatureLocation(4093, 4922), strand = +1)
62 gd_feature_set.add_feature(feature, name="pHT2", label=False, color=colors[3],
63     sigil="ARROW", arrowshaft_height=0.1)
64 feature = SeqFeature(FeatureLocation(2795, 2339), strand = -1)
65 gd_feature_set.add_feature(feature, name="pHT2", label=False, color=colors[1],
66     sigil="ARROW", arrowshaft_height=0.1)
67 feature = SeqFeature(FeatureLocation(2340, 2795), strand = +1)
68 gd_feature_set.add_feature(feature, name="pHT2", label=False, color=colors[4],
69     sigil="ARROW", arrowshaft_height=0.1)
70 feature = SeqFeature(FeatureLocation(743, 1058), strand = +1)
71 gd_feature_set.add_feature(feature, name="pHT2", label=False, color=colors[5],
72     sigil="ARROW", arrowshaft_height=0.1)
73 feature = SeqFeature(FeatureLocation(1984, 2246), strand = +1)

```

```

74 gd_feature_set.add_feature(feature, name="pHT2", label=False,
75     color=colors[6], sigil="ARROW", arrowshaft_height=0.1)
76
77 #On essaie d'ajouter d'autres track pour repr senter la position des blasts
78 gd_track_for_features = gd_diagram.new_track(1, name="PGeneClip", start=0,
79     end=5267)
80 gd_feature_set = gd_track_for_features.new_set()
81 feature = SeqFeature(FeatureLocation(1879, 2563), strand = +1)
82 gd_feature_set.add_feature(feature, name="PGeneClip", label=False, color="blue",
83     sigil="ARROW", arrowhead_length=1, arrowshaft_height=0.1)
84
85 gd_track_for_features = gd_diagram.new_track(1,
86     name="Cloning_vector_EN.Cherry", start=0, end=10649)
87 gd_feature_set = gd_track_for_features.new_set()
88 feature = SeqFeature(FeatureLocation(7102, 7813), strand = +1)
89 gd_feature_set.add_feature(feature, name="Cloning_Vector_EN.Cherry", label=False,
90     color="red", sigil="ARROW", arrowhead_length=1, arrowshaft_height=0.1)
91
92 gd_diagram.draw(format='linear', pagesize="LETTER", orientation="portrait",
93     fragments=1, start=0, end=10649)
94 gd_diagram.write("GD_labels_default.eps", "eps")

```

## 26 Fichiers genbank utilisés pour ce rapport

Nom	Numéro d'accension	Nom	Numéro d'accension
Cloning Vector EN.Cherry, complete sequence	HM771696.1	PGeneClip hMGFP Vector, complete sequence	AY744386.1
Expression vector pHT2, complete sequence	AY773970.1	Homo sapiens chromosome 6, GRCh37.p13 Primary Assembly	NC_000006.11
SARS coronavirus MA15 ExoN1 isolate d3om5, complete genome	JF292906.1	SARS coronavirus MA15 isolate d2ym4, complete genome	JF292909.1
SARS coronavirus MA15 isolate d4ym5, complete genome	JF292915.1	SARS coronavirus HKU-39849 isolate recSARS-CoV HKU-39849, complete genome	JN854286.1
SARS coronavirus HKU-39849 isolate UOB, complete genome	JQ316196.1	SARS coronavirus isolate Tor2/FP1-10912, complete genome	JX163923.1
SARS coronavirus isolate Tor2/FP1-10851, complete genome	JX163924.1	SARS coronavirus isolate Tor2/FP1-10895, complete genome	JX163925.1
SARS coronavirus isolate Tor2/FP1-10912, complete genome	JX163926.1	SARS coronavirus isolate Tor2/FP1-10851, complete genome	JX163927.1
SARS coronavirus isolate Tor2/FP1-10895, complete genome	JX163928.1	SARS coronavirus SinP3, complete genome	AY559090.1
SARS coronavirus HKU-39849 isolate TCVSP-HARROD-00001, complete genome	GU553363.1	SARS coronavirus HKU-39849 isolate recSARS-CoV HKU-39849, complete genome	JN854286.1

Nom	Numéro d'accension	Nom	Numéro d'accension
SARS coronavirus HKU-39849 isolate TCVSP-HARROD-00002, complete genome	GU553364.1	SARS coronavirus HKU-39849 isolate TCVSP-HARROD-00003, complete genome	GU553365.1
SARS coronavirus Sin850, complete genome	AY559096.1	SARS coronavirus MA15 isolate P3pp3, complete genome	FJ882948.1
SARS coronavirus MA15 ExoN1 isolate P3pp3, complete genome	FJ882951.1	SARS coronavirus MA15 isolate P3pp4, complete genome	FJ882952.1
SARS coronavirus MA15, complete genome	FJ882957.1	SARS coronavirus MA15 isolate P3pp7, complete genome	FJ882958.1
SARS coronavirus MA15 ExoN1 isolate P3pp6, complete genome	FJ882959.1	SARS coronavirus MA15 isolate P3pp5, complete genome	FJ882961.1
SARS coronavirus ExoN1 isolate c5P1, complete genome	JF292922.1	SARS coronavirus ExoN1 isolate c5P10, complete genome	JX162087.1
SARS coronavirus ExoN1 strain	KF514407.1	PREDICTED : Pan troglodytes forkhead box P4, transcript variant 2 (FOXP4), mRNA	XM_518463.3
PREDICTED : Pan paniscus forkhead box P4, transcript variant 2 (FOXP4), mRNA.	XM_003833312.1	PPREDICTED : Gorilla gorilla gorilla forkhead box P4, transcript variant 2 (FOXP4), mRNA.	XM_004043991.1
PREDICTED : Pongo abelii forkhead box P4, transcript variant 1 (FOXP4), mRNA.	XM_002816867.2	PREDICTED : Nomascus leucogenys forkhead box P4, transcript variant 2 (FOXP4), mRNA.	XM_003266293.1
PREDICTED : Macaca fascicularis forkhead box P4 (FOXP4), transcript variant X3, mRNA.	XM_005553053.1	Macaca mulatta forkhead box P4 (FOXP4), mRNA.	NM_001266091.1
PREDICTED : Saimiri boliviensis boliviensis forkhead box P4, transcript variant 2 (FOXP4), mRNA	XM_003922988.1	Homo sapiens chromosome 2, GRCh37.p13 Primary Assembly	NC_000002.11
Homo sapiens amyotrophic lateral sclerosis 2 (juvenile) (ALS2), transcript variant 1, mRNA	NM_020919.3	Homo sapiens amyotrophic lateral sclerosis 2 (juvenile) (ALS2), transcript variant 2, mRNA	NM_001135745.1
Pan troglodytes chromosome 2B, Pan.troglodytes-2.1.4	NC_006470.3	Macaca mulatta chromosome 12, Mmul.051212, whole genome shotgun sequence	NC_007869.1
Canis lupus familiaris breed boxer chromosome 37, CanFam3.1, whole genome shotgun sequence	NC_006619.3	Bos taurus breed Hereford chromosome 2, Bos.taurus.UMD.3.1, whole genome shotgun sequence	AC_000159.1
Mus musculus strain C57BL/6J chromosome 1, GRCm38.p1 C57BL/6J	NC_000067.6	Rattus norvegicus strain BN/SsNHsdMCW chromosome 9, Rnor_5.0	NC_005108.3
Gallus gallus isolate #256 breed Red Jungle fowl, inbred line UCD001 chromosome 7, Gallus.gallus-4.0, whole genome shotgun sequence	NC_006094.3	Danio rerio strain Tuebingen chromosome 6, Zv9	NC_007117.5
Homo sapiens chromosome 2 genomic contig, GRCh37.p13 Primary Assembly	NT_005403.17	alsin isoform 1 [Homo sapiens]	NP_065970.2
alsin [Pan troglodytes]	NP_001073389.1	forkhead box protein P4 isoform 1 [Homo sapiens]	NP_001012426.1

## 27 Fichiers de gène de NCBI utilisé pour ce rapport

Nom	Gene ID	Nom	Gene ID
ALS2 amyotrophic lateral sclerosis 2 (juvenile) [ Homo sapiens (human) ]	57679	ALS2 amyotrophic lateral sclerosis 2 (juvenile) [ Pan troglodytes (chimpanzee) ]	470613
ALS2 amyotrophic lateral sclerosis 2 (juvenile) [ Macaca mulatta (Rhesus monkey) ]	703263	ALS2 amyotrophic lateral sclerosis 2 (juvenile) [ Canis lupus familiaris (dog) ]	100856109
ALS2 amyotrophic lateral sclerosis 2 (juvenile) [ Bos taurus (cattle) ]	535750	Als2 amyotrophic lateral sclerosis 2 (juvenile) [ Mus musculus (house mouse) ]	74018
Als2 amyotrophic lateral sclerosis 2 (juvenile) [ Rattus norvegicus (Norway rat) ]	363235	FOXP4 forkhead box P4 [ Homo sapiens (human) ]	116113

## Références

- [1] Kent WJ, Sugnet CW, Furey TS, Roskin KM, Pringle TH, Zahler AM, Haussler D. The human genome browser at UCSC. *Genome Res.* 2002 Jun;12(6):996-1006.
- [2] National Library of Medicine (US). Genetics Home Reference [Internet]. Bethesda (MD) : The Library ; 2013 Oct 26. ALS2 ; [reviewed 2012 Aug ; cited 2013 Oct 26]. Available from : <http://ghr.nlm.nih.gov/gene/ALS2>
- [3] Sclérose latérale amyotrophique. In Wikipedia. Retrieved October 26, 2013, from [http://fr.wikipedia.org/wiki/Sclérose\\_latérale\\_amyotrophique](http://fr.wikipedia.org/wiki/Sclérose_latérale_amyotrophique)
- [4] Stohr, H., Weber, B. H. F. Cloning and characterization of the human retina-specific gene MPP4, a novel member of the p55 subfamily of MAGUK proteins. *Genomics* 74 : 377-384, 2001. [PubMed : 11414766] [Full Text : Elsevier Science]
- [5] Pseudogène. In Wikipedia. Retrieved October 26, 2013, from <http://fr.wikipedia.org/wiki/Pseudogène>
- [6] Huang, L., Szymanska, K., Jensen, V. L., Janecke, A. R., Innes, A. M., Davis, E. E., Frosk, P., Li, C., Willer, J. R., Chodirker, B. N., Greenberg, C. R., McLeod, D. R., and 31 others. TMEM237 is mutated in individuals with a Joubert syndrome related disorder and expands the role of the TMEM family at the ciliary transition zone. *Am. J. Hum. Genet.* 89 : 713-730, 2011. [PubMed : 22152675] [Full Text : Elsevier Science]
- [7] Online Mendelian Inheritance in Man, OMIM®. Johns Hopkins University, Baltimore, MD. MIM Number : 606353 : 03/26/2009 : . World Wide Web URL : <http://omim.org/>
- [8] Online Mendelian Inheritance in Man, OMIM®. Johns Hopkins University, Baltimore, MD. MIM Number : 205100 : 07/19/2012 : . World Wide Web URL : <http://omim.org/>
- [9] Online Mendelian Inheritance in Man, OMIM®. Johns Hopkins University, Baltimore, MD. MIM Number : 607225 : 09/17/2013 : . World Wide Web URL : <http://omim.org/>
- [10] Eymard-Pierre, E., Lesca, G., Dollet, S., Santorelli, F. M., di Capua, M., Bertini, E., Boespflug-Tanguy, O. Infantile-onset ascending hereditary spastic paralysis is associated with mutations in the alsin gene. *Am. J. Hum. Genet.* 71 : 518-527, 2002. [PubMed : 12145748, images] [Full Text : Elsevier Science]
- [11] Online Mendelian Inheritance in Man, OMIM®. Johns Hopkins University, Baltimore, MD. MIM Number : 606352 : 09/16/2013 : . World Wide Web URL : <http://omim.org/>
- [12] Yang, Y., Hentati, A., Deng, H.-X., Dabbagh, O., Sasaki, T., Hirano, M., Hung, W.-Y., Ouahchi, K., Yan, J., Azim, A. C., Cole, N., Gascon, G., Yagmour, A., Ben-Hamida, M., Pericak-Vance, M., Hentati, F., Siddique, T. The gene encoding alsin, a protein with three guanine-nucleotide exchange factor domains, is mutated in a form of recessive

- amyotrophic lateral sclerosis. *Nature Genet.* 29 : 160-165, 2001. Note : Erratum : *Nature Genet.* 29 : 352 only, 2001. [PubMed : 11586297] [Full Text : Nature Publishing Group]
- [13] Hadano, S., Hand, C. K., Osuga, H., Yanagisawa, Y., Otomo, A., Devon, R. S., Miyamoto, N., Showguchi-Miyata, J., Okada, Y., Singaraja, R., Figlewicz, D. A., Kwiatkowski, T., and 9 others. A gene encoding a putative GTPase regulator is mutated in familial amyotrophic lateral sclerosis 2. *Nature Genet.* 29 : 166-173, 2001. Note : Erratum : *Nature Genet.* 29 : 352 only, 2001. [PubMed : 11586298] [Full Text : Nature Publishing Group]
- [14] Huang, X. and Madan, A. (1999) CAP3 : A DNA Sequence Assembly Program. *Genome Research*, 9 : 868-877.
- [15] Basic Local Alignment Search Tool (Altschul et al., *J Mol Biol* 215 :403-410 ; 1990).
- [16] ClustalW and ClustalX version 2 (2007) Larkin MA, Blackshields G, Brown NP, Chenna R, McGettigan PA, McWilliam H, Valentin F, Wallace IM, Wilm A, Lopez R, Thompson JD, Gibson TJ and Higgins DG *Bioinformatics* 2007 23(21) : 2947-2948. doi :10.1093/bioinformatics/btm404
- [17] A.R. Subramanian, M. Kaufmann, B. Morgenstern (2008) DIALIGN-TX : greedy and progressive approaches for segment-based multiple sequence alignment. *Algorithms for Molecular Biology* 3,6.
- [18] A.F.A. Smit, R. Hubley & P. Green, unpublished data. Current Version : open-4.0.3 ( RMLib : 20130422 & Dfam : 1.2 )
- [19] Mower, Jeff, Annaiah, Kiran. Multiple Sequence Alignment. A survey of the various programs available and application of MSA in addressing certain biological problems. PowerPoint Presentation. School of Informatics, Indiana University. Downloaded on 2013-10-31. Available at [bio.informatics.indiana.edu/L529/papers/kiran-jeff.ppt](http://bio.informatics.indiana.edu/L529/papers/kiran-jeff.ppt).
- [20] Score de qualité phred. In Wikipedia. Retrieved November 1, 2013, from [http://fr.wikipedia.org/wiki/Score\\_de\\_qualite\\_phred](http://fr.wikipedia.org/wiki/Score_de_qualite_phred)
- [21] N. Bray and L. Pachter, MAVID : Constrained ancestral alignment of multiple sequences,, *Genome Research*, 14 :693-699 (2004)