Travail Encadré de Recherche : Analyse de données fonctionnelles appliquée à la classification de séquences d'ADN basées sur l'étude de musaraignes

Sujet de Sophie Dabo
traité par Florent Maret et Zakariya Olabi

2020

Table des matières

1 Sujet d'étude

Ces dernières années, il y a eu une croissance rapide en analyse de données génétiques des espèces. Les objets des données génétiques sont souvent considérés comme des fonctions et sont corrélés dû aux relations phylogénétiques (cf. [9]). De plus, dans ce domaine, les échantillons concernent souvent des données de dimensions élevées (cf. [12]) et sont de petites tailles.
L'analyse de données fonctionnelles (FDA) est conçue pour des données ayant une structure fonctionnelle (cf. [12]). Elle comprend un ensemble de techniques statistiques en considérant les données comme des fonctions, des formes, des objets, vus comme des réalisations de procédés stochastiques (cf. [13]), à valeurs dans des espaces de fonctions, de dimension éventuellement infinie.
En phylogénétique, il est naturel d'étudier des séquences d'ADN de dimension très grande (au moins 1000), à l'aide de la statistique multivariée. Revell (cf. [11]) a fourni un aperçu des procédures statistiques adéquates pour la correction de la taille phylogénétique et l'analyse en composantes principales (ACP). Leng et Müller (cf. [10]) ont proposé une méthode utilisant un outil de régression binaire fonctionnelle basé sur l'analyse en composantes principales fonctionnelles (ACPF) pour classifier des gènes. La méthode proposée par Leng et Müller (cf. [10]) permet d'identifier les gènes qui peuvent être mal classés par les méthodes de classification biologique traditionnelles. Les données concernant les espèces peuvent dépendre de l'histoire de leurs ancêtres. On peut comparer les séquences d'un groupe ciblé avec l'échantillon obtenu par la GenBank (banque de séquences ADN en libre accès créée au Centre national pour l'information biotechnologique).
Bien que ces échantillons peuvent être étudiés avec l'ACP et l'ACPF, ces méthodes ne parviennent pas toujours à déterminer les statuts génétiques des espèces. La statistique joue donc un rôle crucial dans la classification d'espèces à l'aide de données génétiques, et il est essentiel dans une classification de telles données, de prendre en considération une distance adaptée à la nature des données phylogénétiques et les avis des experts en biologie.
Biologistes et statisticiens se doivent de travailler ensemble, par exemple, dans le cas des musaraignes en Malaisie, malgré les avancées en biologie/écologie, beaucoup de questions demeurent sur l'origine de leur habitat. Par conséquent, il est intéressant d'étudier si la FDA utilisée sur les séquences ADN permet d'identifier l'origine de certaines espèces.
Pour ce projet, nous nous intéressons à la FDA appliquée aux séquences ADN de musaraignes (en Malaisie) pour essayer d'identifier leur habitat et déterminer des groupes d'individus.

2 Introduction à l'analyse de données fonctionnelles

L'analyse de données fonctionnelles (abrégée FDA en anglais) est une branche des statistiques (apparue il y a un peu plus de 20 ans) qui analyse les données de grande dimension et qui apporte des informations sur des courbes, des surfaces, des fonctions. Ce type de données provient de presque toutes les branches de la science, de l'ingénierie à la géologie, en passant par la biologie, la médecine et la chimie.
Dans ce travail, nous disposerons de données dans de grands espaces, par exemple $\mathbb{R}^d$ ($d \geq 1000$), chaque donnée de $\mathbb{R}^d$ sera considérée comme un seul objet fonctionnel (une courbe).
Ensuite, comme ces objets fonctionels sont à valeurs dans des espaces de fonctions nous allons en pratique réduire la dimension de l'espace fonctionnel (qui est de dimension habituellement infinie). C'est là le principe de l'analyse de données fonctionnelles : passer du multivarié à un espace de dimension infinie, puis travailler dans un sous-espace de dimension finie (adéquatement choisie) de l'espace fonctionnel considéré.

Commençons d'abord par définir quelques notions de base de la FDA (cf. [3], Chapitre 3).

Définition 1 (Fonction de carré intégrable) *Soit X une fonction. On dit que X est une fonction de carré intégrable ($X \in \mathcal{L}^2$) si :*

$$\int X^2(t)dt < \infty$$

Remarque 1 *Les fonctions sont en général des objets de dimension infinie. On va considérer que les fonctions étudiées sont dans l'espace des fonctions de carré intégrable $\mathcal{L}^2$. Cet espace est idéal pour l'étude de nos fonctions car on a le produit scalaire suivant (pour 2 fonctions f, et g de $\mathcal{L}^2$) :*

$$< f, g >= \int f(t)g(t)dt,$$

qui est analogue au produit scalaire en dimension finie.

2.1 Position du problème

Définition 2 (Variable aléatoire fonctionnelle) *Soit $(\Omega, \mathcal{A}, \mathbb{P})$ un espace probabilisé. X est une variable aléatoire si $\forall \omega \in \Omega, X(\omega)$ est une fonction déterministe. On admettra que $\forall \omega \in \Omega, X(\omega) \in \mathcal{L}^2$*

Quand on fait de l'analyse de données, on observe en général n réalisations de variables aléatoires X_i réelles. En analyse de données fonctionnelles, on veut que les X_i soient des fonctions régulières. On analyse donc un échantillon de courbes dont on connait les valeurs en certains points et l'on souhaite connaître les fonctions X_i pour tous les points sur le domaine considéré (ici un intervalle de $\mathbb{R}$), et cette analyse peut être grandement facilitée avec l'utilisation de packages sur R.
Lorsque l'on peut dire que nos observations sont sans erreur on utilisera le procédé d'interpolation, sinon on devra utiliser des techniques de lissage.
La forme la plus courante en analyse de données fonctionnelles est la suivante :

$$x_{n,t_{j,n}} \in \mathbb{R}, t_{j,n} \in [T_1, T_2], n = 1, ...N, j = 1, ..., J_n$$

où les $x_{n,t_{j,n}}$ sont les N observations (i.i.d) discrétisées du processus considéré, $T_1, T_2 \in \mathbb{R}$ et $N, J_n \in \mathbb{N}$

2.2 Lissage des données discrètes $x_{n,t_{j,n}}$ en données fonctionnelles X_n

En pratique nous avons des données discrètes, pour appliquer des techniques d'analyse de données fonctionnelles, il faut d'abord rendre les données discrètes en type fonctionnel, donc les lisser. L'idée principale est que les objets étudiés en FDA sont des fonctions généralement lisses. Ainsi, la première chose à faire est d'exprimer les données discrètes comme une combinaison linéaire de fonctions de bases de l'espace fonctionnel considéré.

Définition 3 (Fonction de base) *Une fonction de base est une fonction apparaissant dans une base fixée d'un espace fonctionnel. Par exemple, on retrouve les B-splines, les fonctions cosinus et sinus, ainsi que les transformées de Fourier.*

Pour des données périodiques, on utilisera la base de Fourier et pour les non périodiques ce sera souvent la base des B-splines

On obtient alors :

$$X_n(t) = \sum_{m=1}^{\infty} c_{nm} B_m(t), 1 \leq n \leq N$$

où les B_m sont des fonctions de bases (splines, sinus, cosinus..).

2.2.1 Exemple des B-splines

Définition 4 (B-splines) *Etant donnés ... $< t_0 < t_1 < t_2 <$ ... , on définit récursivement :*

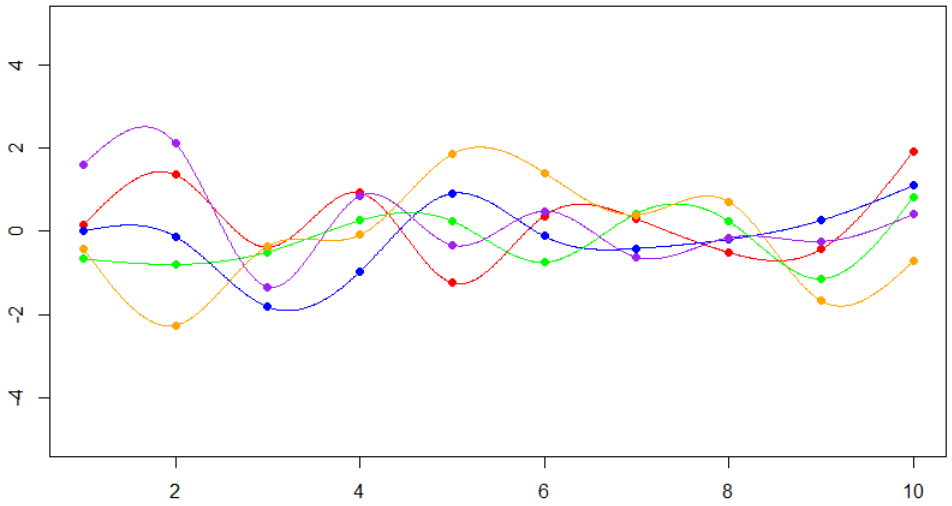
$$B_j^0(x) = \begin{cases} 1 & si\ x \in [t_j, t_j + 1] \\ 0 & sinon \end{cases}$$

et pour $k \geq 1$

$$B_j^k(x) = \frac{x - t_j}{t_{j+k} - t_j} B_j^{k-1}(x) + \frac{t_{j+k+1} - x}{t_{j+k+1} - tj + 1} B_{j+1}^{k-1}(x)$$

A l'aide de cette base, on va pouvoir lisser des données discrètes non périodiques en les exprimant en fonction de celle-ci. La théorie nous assure alors une stablité numérique et que si l'on perturbe une seule de nos données, la fonction obtenue par la combinaison linéaire ne sera perturbée que dans un petit intervalle car c'est une base de fonctions localisées (non nulle que sur un petit nombre de sous-intervalles de la partition).

Exemple 1 *Lissage par les B-splines*
On se donne 5 observations de 10 points $x_i = (i, z_i)$ *où* z_i *est une observation d'une loi normale* $\mathcal{N}(0, 1)$.
On parvient à trouver une estimation de chacune des 5 fonctions sur l'intervalle $[0, 1]$ *grâce à l'interpolation en utilisant les B-splines.*



2.2.2 Exemple de la base de Fourier

Pour les fonctions périodiques de période T, il est préférable d'utiliser la base de Fourier définie de la façon suivante :

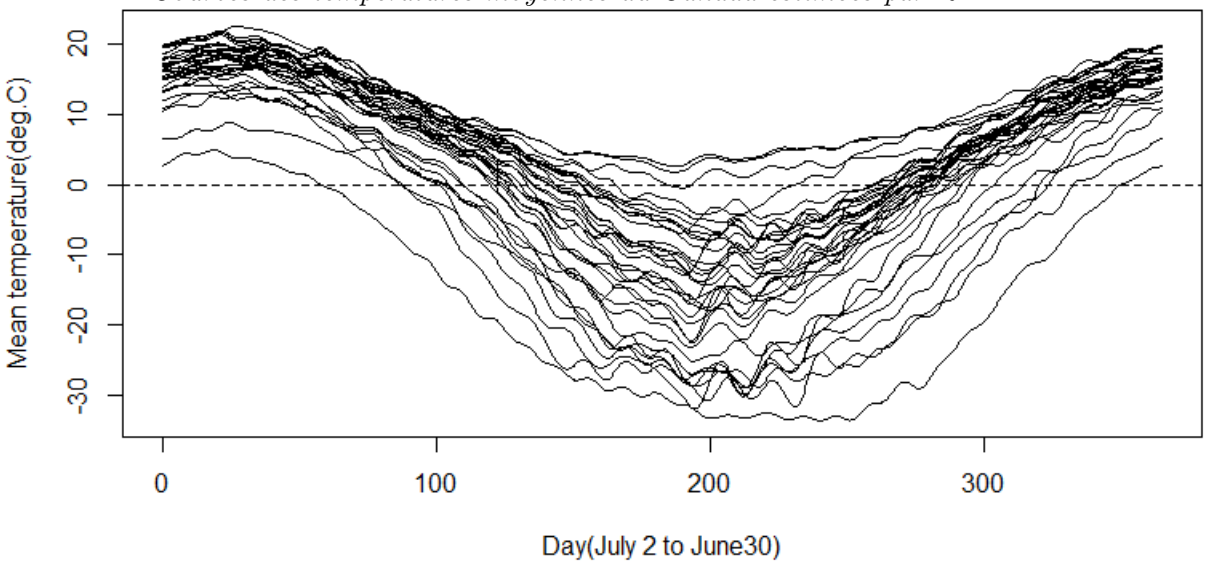
Définition 5 (Base de Fourier) *(cf. [2],Chapitre 3)*
Soit ω définie par $\omega = \frac{2\pi}{T}$ alors on a :

$$\begin{aligned} B_1(t) &= 1 \\ B_2(t) &= sin(\omega t) \\ B_3(t) &= cos(\omega t) \\ B_4(t) &= sin(2\omega t) \\ B_5(t) &= cos(2\omega t) \\ &\vdots \end{aligned}$$

Cette base est très utile pour les données périodiques car seulement deux informations sont nécessaires à sa création : le nombre de fonctions K voulues dans la base et la période T.

Exemple 2 *Lissage par la base de Fourier (cf. [2],Chapitre 4)*
Nous allons maintenant, en guise d'exemple, afficher 35 courbes représentant les températures moyennes au Canada pendant un an. Ces courbes sont obtenues en lissant les données provenant des stations météorologiques canadiennes à l'aide de la base de Fourier.

Courbes des températures moyennes au Canada estimées par R.



2.3 Moyenne, variance et covariance empirique de variables fonctionnelles

Maintenant que nous avons défini X_i comme une fonction de carré intégrable, nous allons vouloir l'étudier (on rappelle que les X_i sont i.i.d). Pour cela on va définir la moyenne empirique, la variance empirique et la covariance empirique de l'échantillon, X_i, $i \in [1, N]$.

Remarque 2 *On travaille ici avec des fonctions aléatoires. On pourra donc définir de la même manière que pour des variables aléatoires réelles, la moyenne empirique, la covariance empirique et la variance empirique utiles en analyse de données.*

On définit la moyenne empirique $\hat{\mu}$ de la manière suivante :

$$\hat{\mu}(t) = \frac{1}{N} \sum_{i=1}^{N} X_i(t).$$

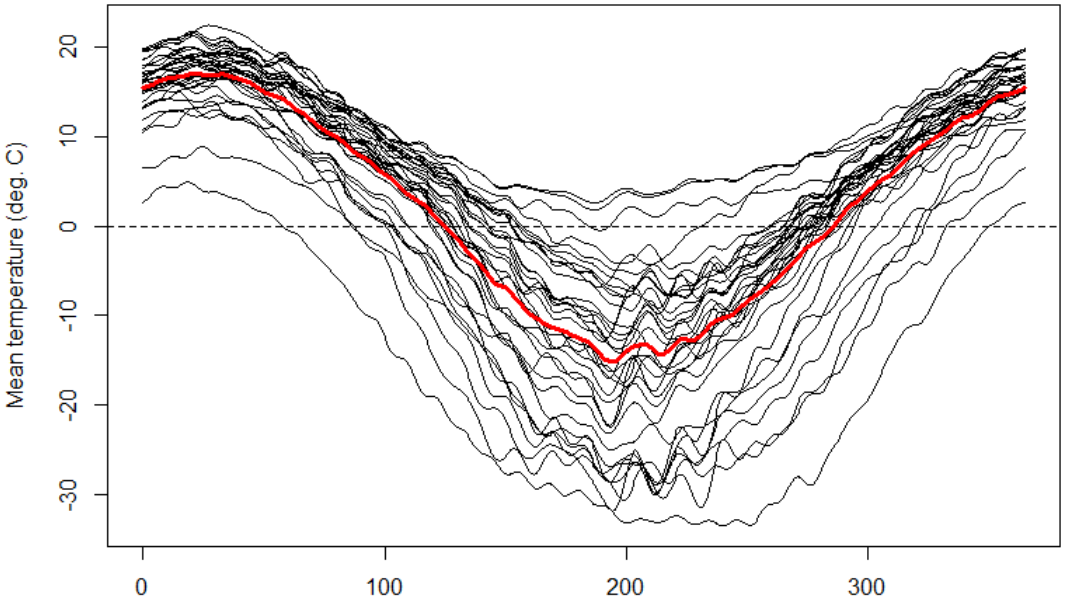
La covariance empirique est :

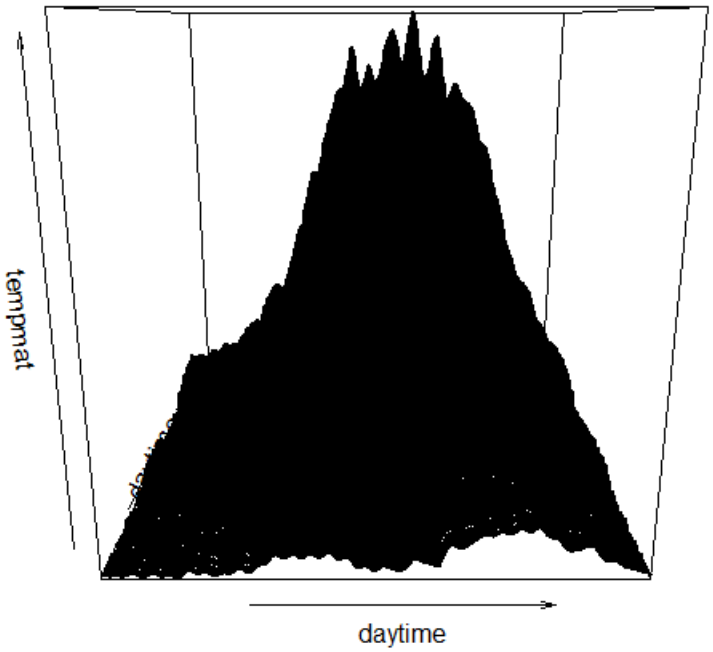
$$\hat{c}(t, s) = \frac{1}{N} \sum_{i=1}^{N} (X_i(t) - \hat{\mu}(t))(X_i(s) - \hat{\mu}(s)).$$

Et la variance empirique est :

$$\hat{s}^2(t) = \frac{1}{N} \sum_{i=1}^{N} (X_i(t) - \hat{\mu}(t))^2).$$

On va illustrer ces notions à l'aide d'un exemple sur les températures moyennes au Canada pour plusieurs stations (disponibles au niveau du package fda de R). Nous obtenons la moyenne (en rouge) et la covariance empirique (graphe en 3D) ci-dessous.





2.4 Réduction de la dimension infinie de l'espace $\mathcal{L}^2([T_1, T_2])$

A ce stade, nous avons une fonction, dans un espace de dimension infinie, et nous ne pouvons donc pas travailler avec en pratique.

C'est pourquoi nous allons nous ramener à une approximation

$$X_n(t) \approx \sum_{m=1}^{M} c_{nm} B_m(t), 1 \leq n \leq N$$

En utilisant la projection dans le sous-espace engendré par les M premières fonctions d'une base de fonctions (bases des B-splines, de Fourier, de l'ACP fonctionelle) permettant d'approcher de manière appropriée les fonctions voulues (ici les données fonctionnelles étudiées).

3 Analyse en composantes principales fonctionnelles

Dans cette partie, nous travaillons toujours avec les fonctions aléatoires X_i précédentes. Pour réaliser une analyse en composantes principales fonctionnelles (ACPF), on va travailler avec des données centrées (cf. [3], Chapitre 1). Ainsi nous pouvons noter dans la suite $\widetilde{X}(t) = X(t) - \hat{\mu}(t)$. La motivation de l'ACPF est de chercher une fonction ξ qui révèle les variations les plus importantes dans le jeu de données de base. Cette recherche utilise l'opérateur de covariance empirique ci-dessus.

3.1 Définitions

Définition 6 (Score des composantes principales) *On définit le score $p_\xi(X_i)$ des composantes principales par*

$$p_\xi(X_i) = \int \xi(t) X_i(t) dt$$

Ainsi la motivation précédente revient à trouver ξ qui maximise $\sum_i p_\xi^2(X_i)$ et telle que $\int \xi^2(t)dt = 1$.

Définition 7 (Fonction propre) *La fonction propre f d'un opérateur linéaire $\mathcal{A}$ sur un espace fonctionnel est un vecteur propre de l'opérateur linéaire. On a donc*

$$\mathcal{A}f = \lambda f,$$

avec λ la valeur propre associée à f.

Similairement à l'ACP, $max(\sum_i p_\xi^2(X_i))$ et ξ sont les plus grandes valeurs propres et fonctions propres (vecteur propre en ACP classique) de la fonction de variance-covariance estimée, notée $\hat{c}(t, s)$.
On peut trouver ces valeurs propres pas à pas, et à l'étape l, on a :

$$\int \xi_j(t)\xi_l(t)dt = 0, j = 1, ..., l-1 \; et \int \xi_l^2(t)dt = 1$$

La fonction propre ξ_j est trouvée en résolvant l'équation fonctionnelle propre suivante (avec λ_j la j^{eme} plus grande valeur propre) :

$$\int \hat{c}(s,t)\xi_j(t)dt = \lambda_j \xi_j(s)$$

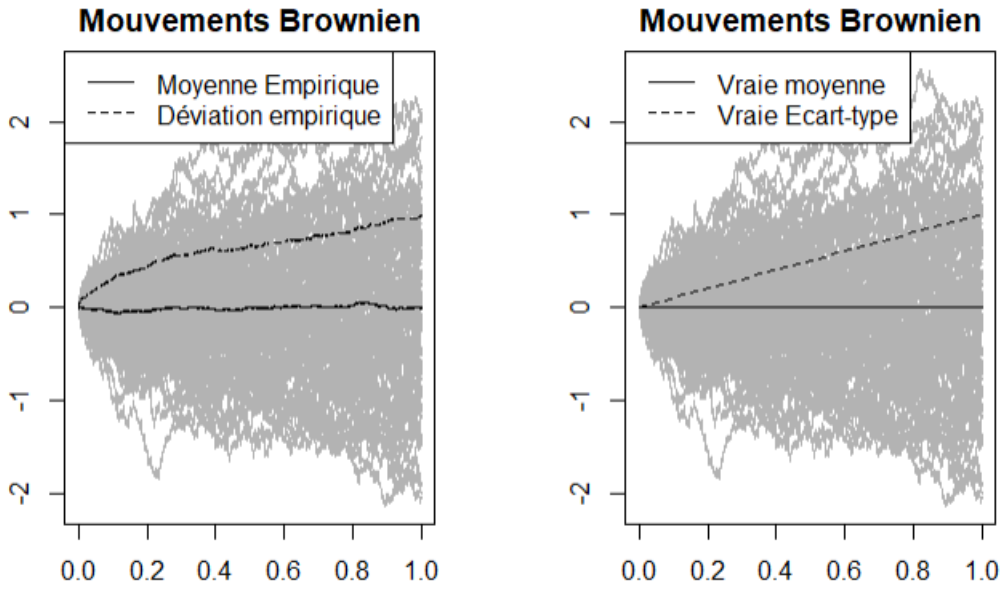
Une différence entre l'ACP et l'ACPF est qu'en ACP, le nombre de variables est généralement plus petit que le nombre d'observations, alors qu'en ACPF, le nombre de valeurs observées pour les fonctions est généralement plus grand que le nombre de fonctions observées.

3.2 Application de l'ACP fonctionnelle à des données simulées

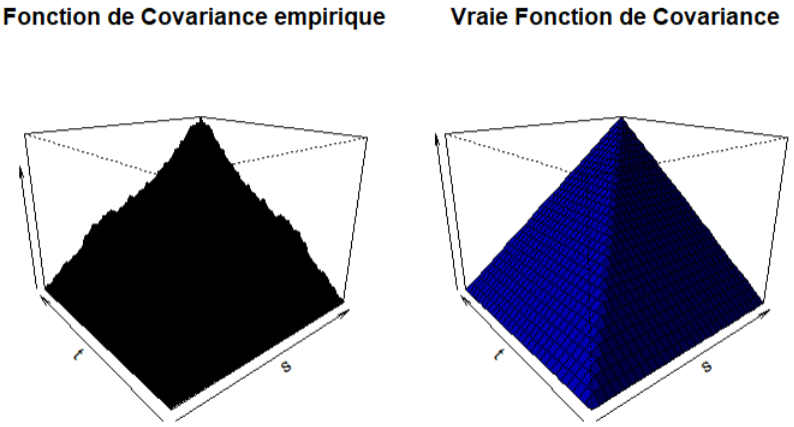
Le but de cette partie est d'illustrer à l'aide d'un exemple l'ACP fonctionnelle.
Nous avons choisi un exemple temporel : le mouvement Brownien.
Le but est d'observer les différentes informations telles que la moyenne, variance, covariance empiriques et les résultats de l'ACP sur ces données.

On a simulé $N = 100$ courbes suivant un processus Brownien discrétisées en 10000 points sur $[0, 1]$. A la place de considérer ces données comme 100 séries temporelles observées en 10000 points discrets, nous allons considérer chacune des séries comme une donnée fonctionnelle. On a ainsi 100 données fonctionnelles après un lissage.
Pour un mouvement Brownien $X(t)$,
$E(X(t)) = 0$, $Var(X(t)) = t$ et $Cov(X(t), X(s)) = min(t, s)$.

Sur les graphes qui suivent, on a représenté à gauche la moyenne et l'écart-type empiriques en noir des 100 courbes en gris, et à droite la moyenne et l'écart-type théoriques.

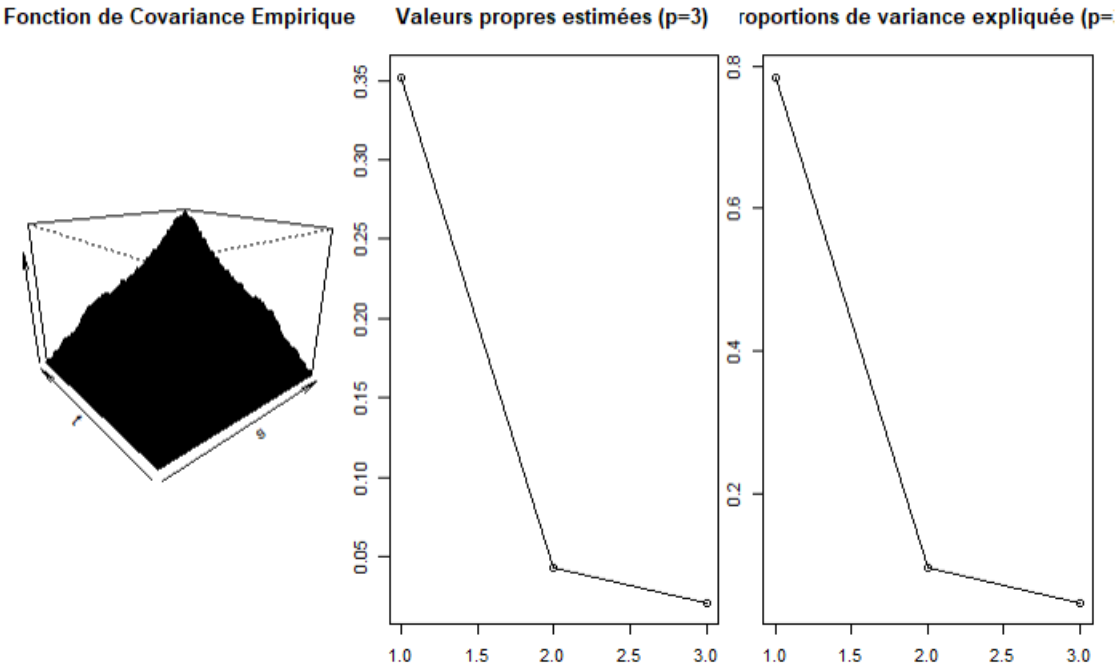


Les graphes ci-dessous représentent à gauche la covariance empirique et à droite la covariance exacte de $X(t)$. L'estimation de la fonction de covariance est satisfaisante, même si l'on remarque qu'au niveau des bords elle est moins bonne ; elle peut être améliorée en augmentant le nombre de courbes (ici 100).

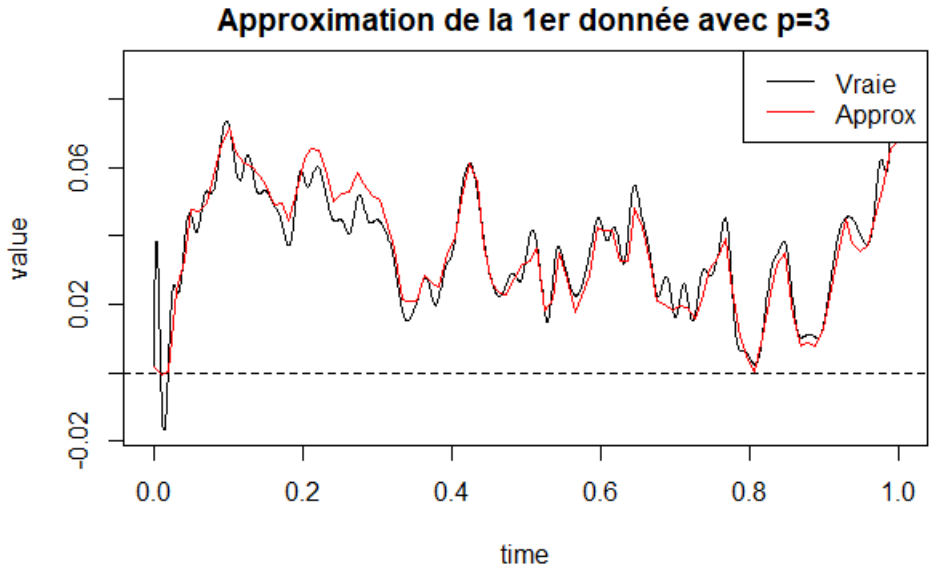


Nous allons maintenant réaliser une ACP fonctionnelle sur nos données simulées. La première étape consiste à lisser les données pour les rendre fonctionelles par exemple

à l'aide de la base des splines, le nombre de fonctions splines utilisées est optimisé par la validation croisée. Nous obtenons alors des objets fonctionnels dans un espace de dimension infinie, il faut donc résumer l'information en réduisant la dimension et ainsi représenter les X_i avec les p premières composantes $\hat{v}_j$ de l'ACP :
$X_i(t) \approx \hat{\mu}(t) + \sum_{j=1}^{p} \hat{\xi}_{ij}\hat{v}_j(t)$.
On obtient les valeurs propres suivantes, et nous choisissons de réduire la dimension à 3 (le sous-espace engendré par les 3 premières fonctions propres de l'ACP) car les 3 premières valeurs propres représentent presque toute l'information (l'inertie).



Pour illustrer la qualité de l'approximation des données via les 3 premières fonctions propres de l'ACP, nous avons tracé la première courbe de l'échantillon de mouvements browniens en noir, et en rouge la courbe approximée obtenue.

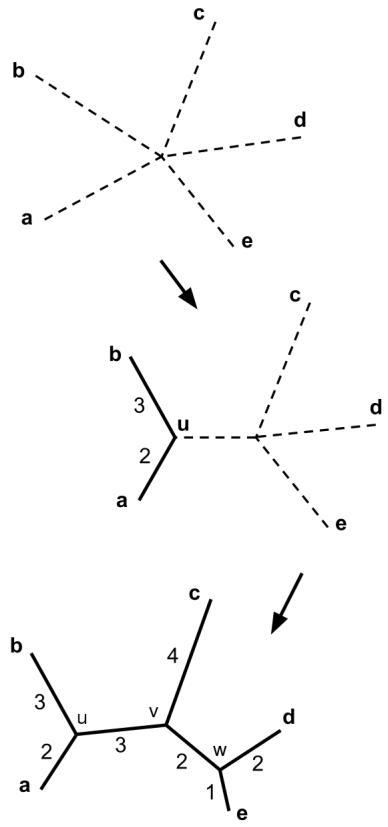


4 Applications aux données réelles

Maintenant que nous avons vu la théorie, nous allons passer à l'exemple qui nous intéresse : celui des musaraignes (nom commun donné à plusieurs espèces de petits mammifères insectivores).
Le but est d'étudier les musaraignes à partir de leur ADN, et de les regrouper en différentes classes afin de déterminer les différentes espèces.
Tout d'abord nous allons voir différentes méthodes de classification génétique : la méthode classique NJ, les méthodes multivariées basées sur des transformations numériques des séquences ADN présentant chacune des avantages et des inconvénients, puis enfin la classification fonctionnelle utilisant la FDA des sections 2 et 3.
Ensuite, nous appliquerons ces différentes méthodes à nos deux bases de données de séquences ADN de musaraignes (crocinew et chimseq).

Définition 8 (ADN) *L'Acide désoxyribonucléique (ou ADN) contient toute l'information génétique des individus. Il est composé d'une multitude de bases nucléiques (ou base azotée). Il y a 4 bases nucléiques : Adénine, Thymine, Cytosine et Guanine qui sont représentées par A, T, C, G.*

4.1 Classification multivariée de séquences ADN par la méthode NJ (Neighbour Joining)



En bio-informatique, le neighbour joining (souvent abrégé NJ) est une méthode de reconstruction d'arbres phylogénétiques. Il faut d'abord définir une fonction Q comme ci-dessous.

$$Q(i,j) = (n-2)d(i,j) - \sum_{k=1}^{n} d(i,k) - \sum_{k=1}^{n} d(j,k)$$

où $d(i,j)$ est une distance quelconque calculée entre les individus i et j, par exemple la distance euclidienne.

Cela consiste à calculer les longueurs de branches de l'arbre et choisir l'arbre qui minimise la longueur totale.
A chaque étape on regroupe 2 individus, on doit donc calculer les distances entre les individus (avec la distance Q), puis choisir celle qui est la plus petite. Ensuite on relit les deux individus les plus proches et on crée un noeud qui est la moyenne de ces individus.
On réitère ce procédé jusqu'à ce que tous les individus soient regroupés. C'est une classification type hiérarchique comme la CAH. Ci-contre se trouve un exemple de classification NJ pour 5 individus a, b, c, d, e (cf. [8]).

4.2 Classifications multivariées de séquences ADN transformées numériquement : différentes modélisations des bases nucléiques

Nos données sont des échantillons de A, T, C, G : on ne peut donc pas appliquer ce que nous avons vu précédemment (moyenne, variance, etc.), c'est pourquoi nous les convertissons d'abord en numérique.

Le tableau suivant représente différentes méthodes de conversion numérique des bases nucléiques (cf. [5]).
Nous avons choisi de tester les modélisations EIIP et Real. Chaque méthode présente des avantages et des inconvénients, elles ont toutes un sens mathématique mais pas forcément génétique. Par exemple, celle des Integer est simple, mais elle introduit des propriétés mathématiques qui ne sont pas présentes en génétique (G≥T). Celle qui est la plus intéressante pour notre analyse est la EIIP parce qu'elle reflète des propriétés physico-chimiques de l'ADN (cf. [6]).

Table 1. Selected DNA numerical representations.

	Name	Numeric representation	Example for sequence X = [*AACTGT*]
1	Integer	$\hat{\mathbf{X}}(i) = \begin{cases} 3 & \text{if } X(i) = \mathsf{G} \\ 2 & \text{if } X(i) = \mathsf{A} \\ 1 & \text{if } X(i) = \mathsf{C} \\ 0 & \text{if } X(i) = \mathsf{T} \end{cases}$	$\hat{\mathbf{X}} = [2, 2, 1, 0, 3, 0]$
2	Real	$\hat{\mathbf{X}}(i) = \begin{cases} -0.5 & \text{if } X(i) = \mathsf{G} \\ -1.5 & \text{if } X(i) = \mathsf{A} \\ 0.5 & \text{if } X(i) = \mathsf{C} \\ 1.5 & \text{if } X(i) = \mathsf{T} \end{cases}$	$\hat{\mathbf{X}} = [-1.5, -1.5, 0.5, 1.5, -0.5, 1.5]$
3	EIIP	$\hat{\mathbf{X}}(i) = \begin{cases} 0.0806 & \text{if } X(i) = \mathsf{G} \\ 0.1260 & \text{if } X(i) = \mathsf{A} \\ 0.1340 & \text{if } X(i) = \mathsf{C} \\ 0.1335 & \text{if } X(i) = \mathsf{T} \end{cases}$	$\hat{\mathbf{X}} = [0.1260, 0.1260, 0.1340, 0.1335, 0.0806, 0.1335]$
4	Atomic Number	$\hat{\mathbf{X}}(i) = \begin{cases} 78 & \text{if } X(i) = \mathsf{G} \\ 70 & \text{if } X(i) = \mathsf{A} \\ 58 & \text{if } X(i) = \mathsf{C} \\ 66 & \text{if } X(i) = \mathsf{T} \end{cases}$	$\hat{\mathbf{X}} = [70, 70, 58, 66, 78, 66]$
5	Paired Numeric	$\hat{\mathbf{X}}(i) = \begin{cases} 1 & \text{if } X(i) = A \vee T \\ -1 & \text{otherwise} \end{cases}$	$\hat{\mathbf{X}} = [1, 1, -1, 1, -1, 1]$
6	Voss	$\hat{\mathbf{X}}_1(i) = \begin{cases} 1 & \text{if } X(i) = A \\ 0 & \text{otherwise} \end{cases}$ $\hat{\mathbf{X}}_2(i) = \begin{cases} 1 & \text{if } X(i) = G \\ 0 & \text{otherwise} \end{cases}$ $\hat{\mathbf{X}}_3(i) = \begin{cases} 1 & \text{if } X(i) = C \\ 0 & \text{otherwise} \end{cases}$ $\hat{\mathbf{X}}_4(i) = \begin{cases} 1 & \text{if } X(i) = T \\ 0 & \text{otherwise} \end{cases}$	$\hat{\mathbf{X}}_1 = [1, 1, 0, 0, 0, 0]$ $\hat{\mathbf{X}}_2 = [0, 0, 0, 0, 1, 0]$ $\hat{\mathbf{X}}_3 = [0, 0, 1, 0, 0, 0]$ $\hat{\mathbf{X}}_4 = [0, 0, 0, 1, 0, 1]$
7	Tetrahedron	$\hat{\mathbf{X}}_1(i) = \begin{cases} \frac{2\sqrt{2}}{3} & \text{if } X(i) = T \\ -\frac{\sqrt{2}}{3} & \text{if } X(i) = C \vee G \\ 0 & \text{otherwise} \end{cases}$ $\hat{\mathbf{X}}_2(i) = \begin{cases} \frac{\sqrt{6}}{3} & \text{if } X(i) = C \\ -\frac{\sqrt{6}}{3} & \text{if } X(i) = G \\ 0 & \text{otherwise} \end{cases}$ $\hat{\mathbf{X}}_3(i) = \begin{cases} 1 & \text{if } X(i) = A \\ -\frac{1}{3} & \text{otherwise} \end{cases}$	$\hat{\mathbf{X}}_1 = \left[0, 0, -\frac{\sqrt{2}}{3}, \frac{2\sqrt{2}}{3}, -\frac{\sqrt{2}}{3}, 2\frac{1}{3}\right]$ $\hat{\mathbf{X}}_2 = \left[0, 0, \frac{\sqrt{6}}{3}, 0, -\frac{\sqrt{6}}{3}, 0\right]$ $\hat{\mathbf{X}}_3 = \left[1, 1, -\frac{1}{3}, -\frac{1}{3}, -\frac{1}{3}, -\frac{1}{3}\right]$

(*Continued*)

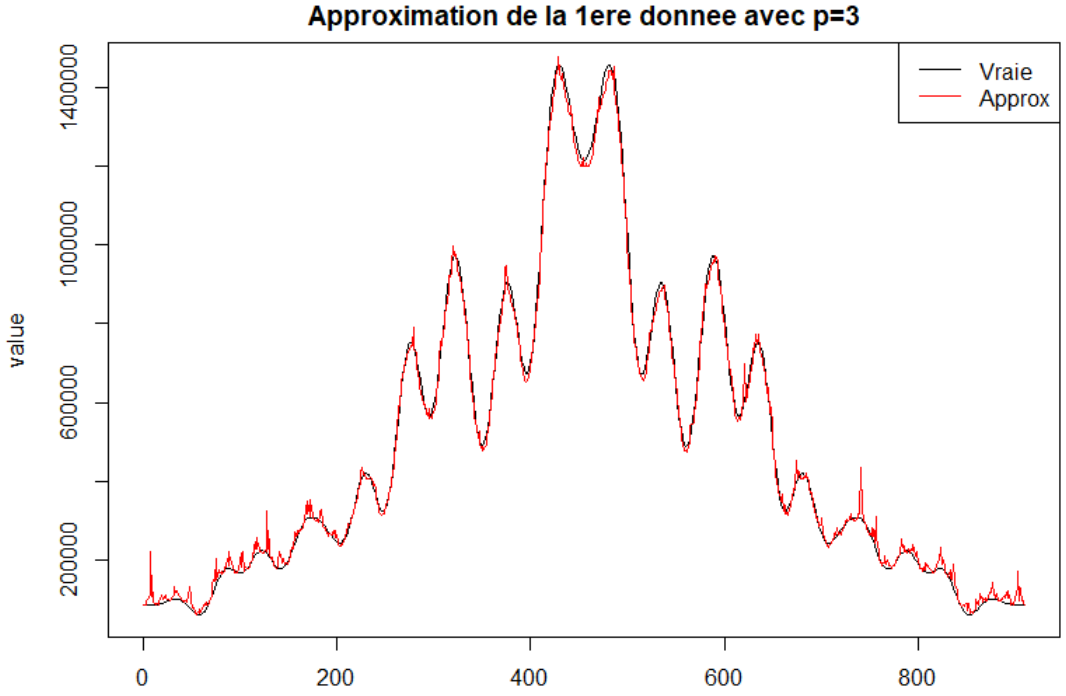
Table 1. (*Continued*)

	Name	Numeric representation	Example for sequence X = [*AACTGT*]
8	Z-Curve	$\hat{\mathbf{X}}_1(i) = \begin{cases} \mathbf{X}(i-1)+1 & \text{if } X(i) = T \vee G \\ \mathbf{X}(i-1)+(-1) & \text{otherwise} \end{cases}$ $\hat{\mathbf{X}}_2(i) = \begin{cases} \mathbf{X}(i-1)+1 & \text{if } X(i) = A \vee C \\ \mathbf{X}(i-1)+(-1) & \text{otherwise} \end{cases}$ $\hat{\mathbf{X}}_3(i) = \begin{cases} \mathbf{X}(i-1)+1 & \text{if } X(i) = A \vee T \\ \mathbf{X}(i-1)+(-1) & \text{otherwise} \end{cases}$	$\hat{\mathbf{X}}_1 = [-1,-2,-3,-2,-1,0]$ $\hat{\mathbf{X}}_2 = [1,2,3,2,1,0]$ $\hat{\mathbf{X}}_3 = [1,2,1,2,1,2]$
9	DNA walk	$\hat{\mathbf{X}}(i) = \begin{cases} \mathbf{X}(i-1)+1 & \text{if } X(i) = C \vee T \\ \mathbf{X}(i-1)+(-1) & \text{otherwise} \end{cases}$	$\hat{\mathbf{X}} = [-1,-2,-1,0,-1,0]$

https://doi.org/10.1371/journal.pone.0173288.t001

4.3 Classification fonctionnelle

Dans cette partie, nous allons appliquer l'analyse de données fonctionnelles à savoir la transformation de séquences génétiques en objets fonctionnels (Section 2) et l'ACP fonctionnelle (Section 3) pour réaliser la classification des données utilisées dans les sections 5.4-5.5.
Tout d'abord, nous allons prendre comme tableau de données, le tableau disjonctif complet de la base de données choisie en remplaçant les valeurs N par les moyennes.
Ces données étant très irrégulières (beaucoup de sauts de valeurs), on ne va pas les lisser directement : on passe par la transformée de Fourier (voir la courbe en noir de la figure ci-dessous représentant l'observation du premier individu de la base chimseq).
Ensuite, on réalise l'ACP fonctionnelle sur ces nouvelles données, maintenant vues comme des fonctions.
Enfin, après avoir choisi de représenter les données sur les 3 premiers axes factoriels (courbe en rouge), nous avons utilisé les scores de l'ACP pour réaliser la nouvelle classification.



4.4 Application à la première base de données : crocinew

Nous avons donc pour but de classifier des musaraignes en Malaisie. Les données de cette section comportent 54 individus, et pour chaque individu nous observons une suite de A, T, C, G de longueur 1139. On sait que, pour cette base de données, l'individu "Suncus Murinus" est un ancêtre commun aux autres individus, et cela influera sur la classification. Des chercheurs en Malaisie ont effectué une classification de ces musaraignes mais elle n'a pas encore été validée par des experts. Nous allons donc essayer de réaliser plusieurs classifications afin de déterminer quelle méthode se rapproche le plus de celle des chercheurs.

Pour nous permettre de réaliser ces classifications, nous avons une base de données pour laquelle les séquences ADN ont toutes la même longueur : pour ce faire, il a donc fallu à certains endroits remplacer A, T, C, G par un N, qui signifie que nous ne sommes pas sûrs de quelle base nucléique il s'agit.

Ci-dessous se trouve une représentation des séquences ADN des musaraignes.

La partition témoin avec laquelle nous devons comparer nos résultats est celle qui suit. Elle a été obtenue par les chercheurs malaisiens en biologie (qui nous ont procuré les données) à l'aide d'un logiciel (Mega, logiciel type clic-bouton) qui utilise la classification NJ. Pour retrouver une partition similaire, nous allons effectuer plusieurs classifications et les comparer avec celle-ci pour voir celle qui correspond le plus.

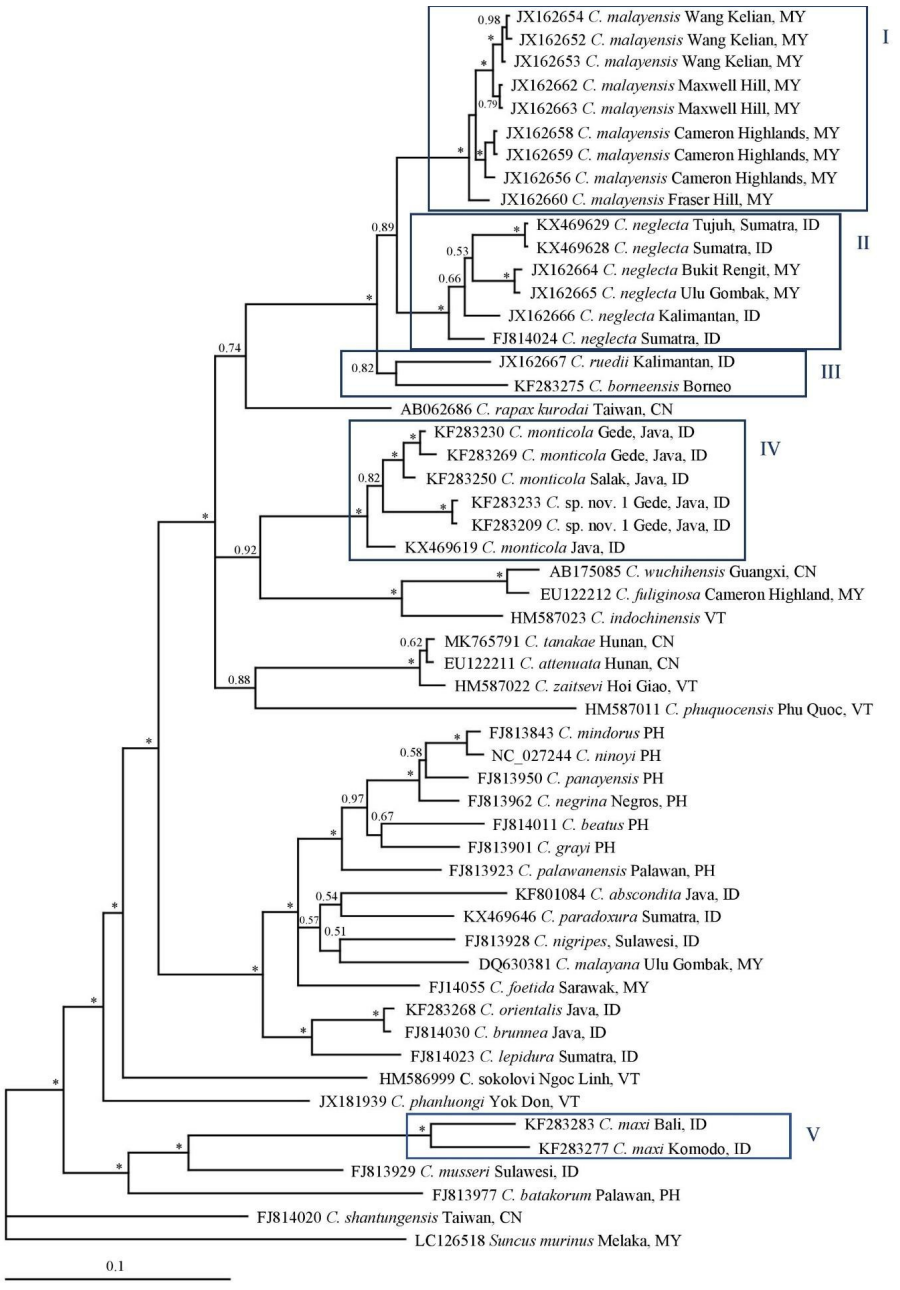
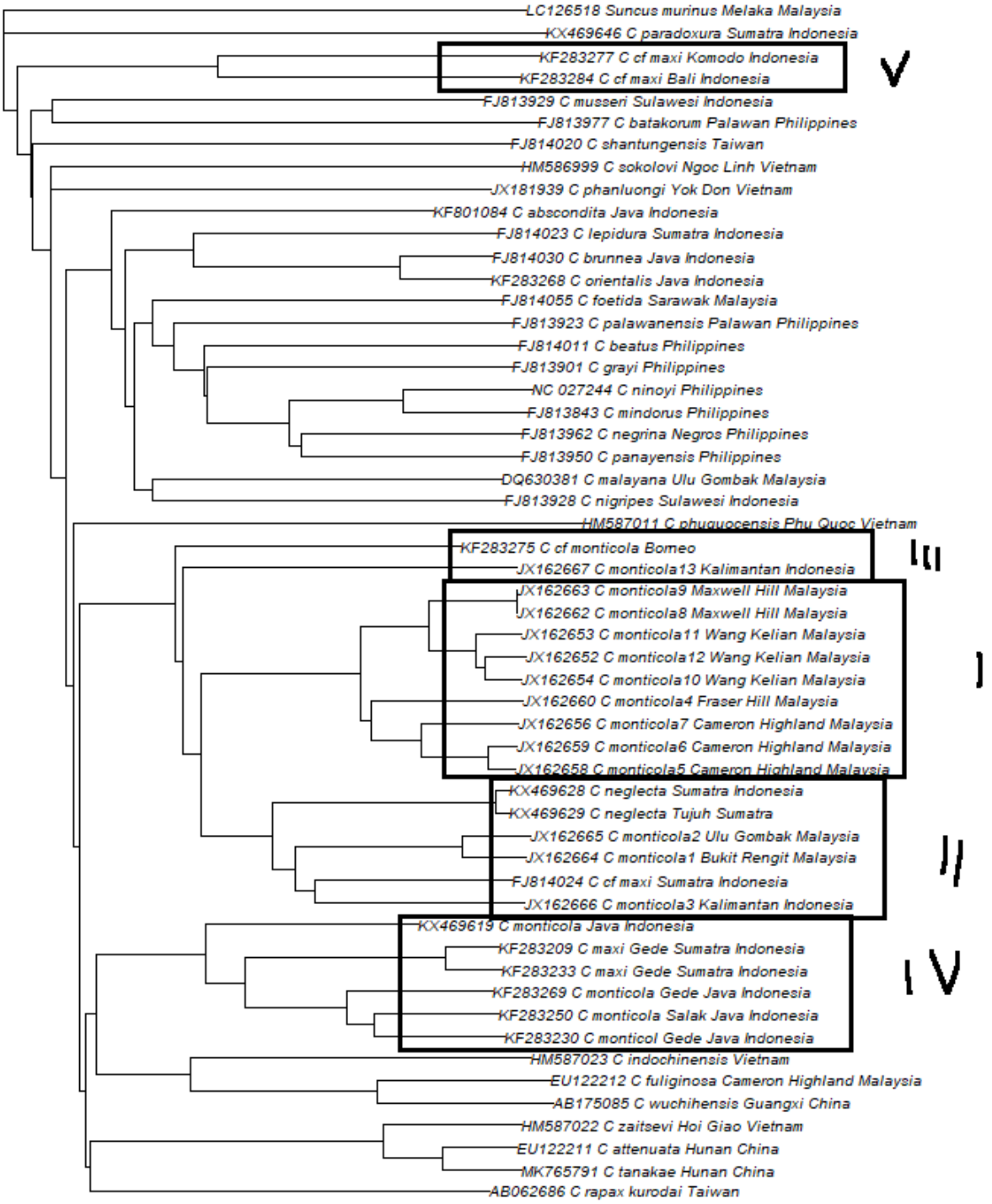


Figure 1 The BA phylogenetic tree for the genus *Crocidura* inferred from 1139 base-pairs of cyt *b* gene sequences and *Suncus murinus* was rooted as an outgroup. Bayesian posterior

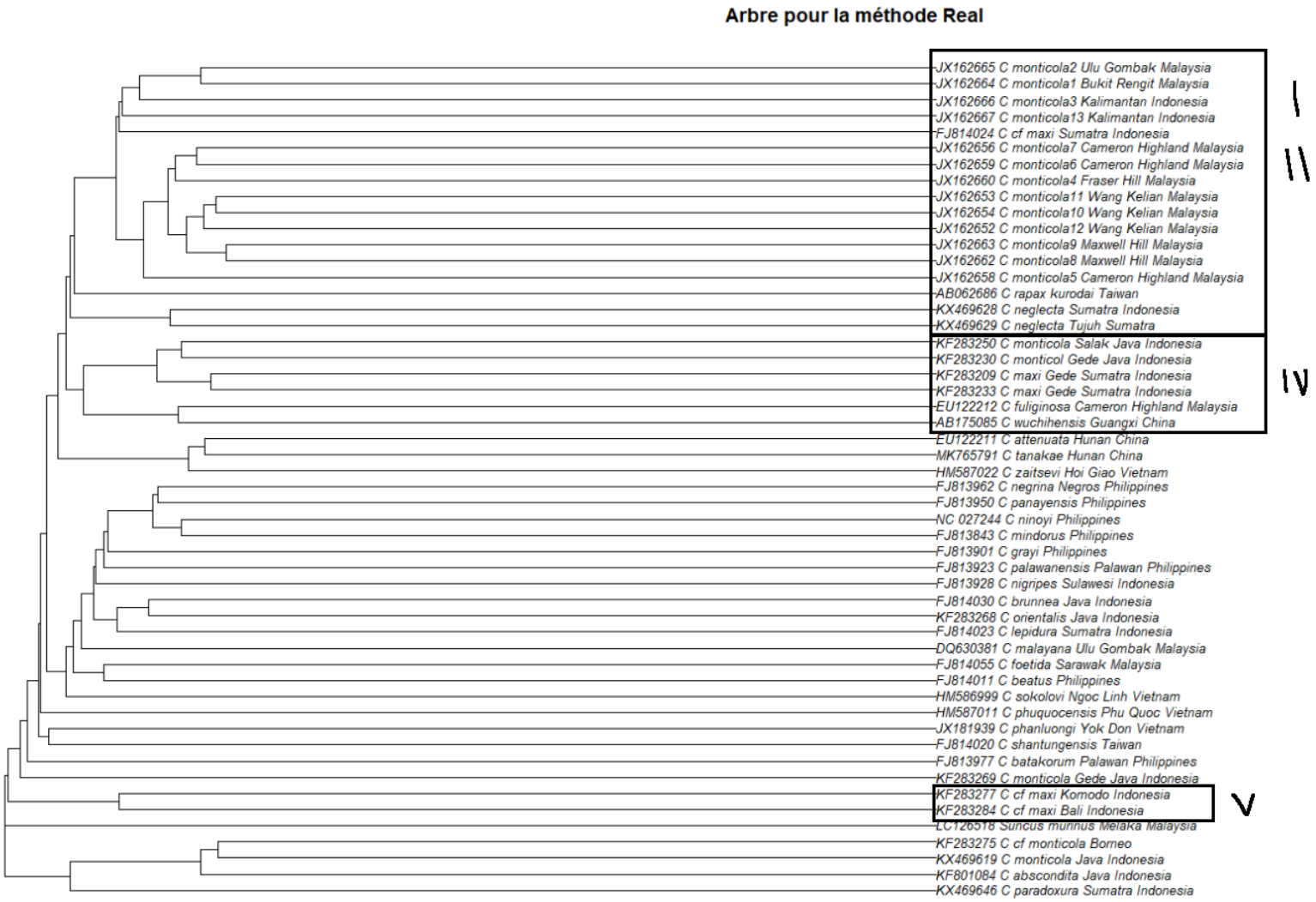
Les chercheurs ont effectué la fonction NJ sous le logiciel Mega, nous avons donc essayé d'utiliser la fonction NJ de R, et nous pouvons constater que les groupes retrouvés sont les mêmes que ceux des chercheurs.



Passons maintenant à la méthode qui transforme les données génétiques en numérique. Avec l'aide de plusieurs packages R pour la phylogénétique, nous obtenons les arbres suivants pour les deux méthodes de conversion choisies (Real et EIIP).
Pour cela nous avons fait la démarche suivante :

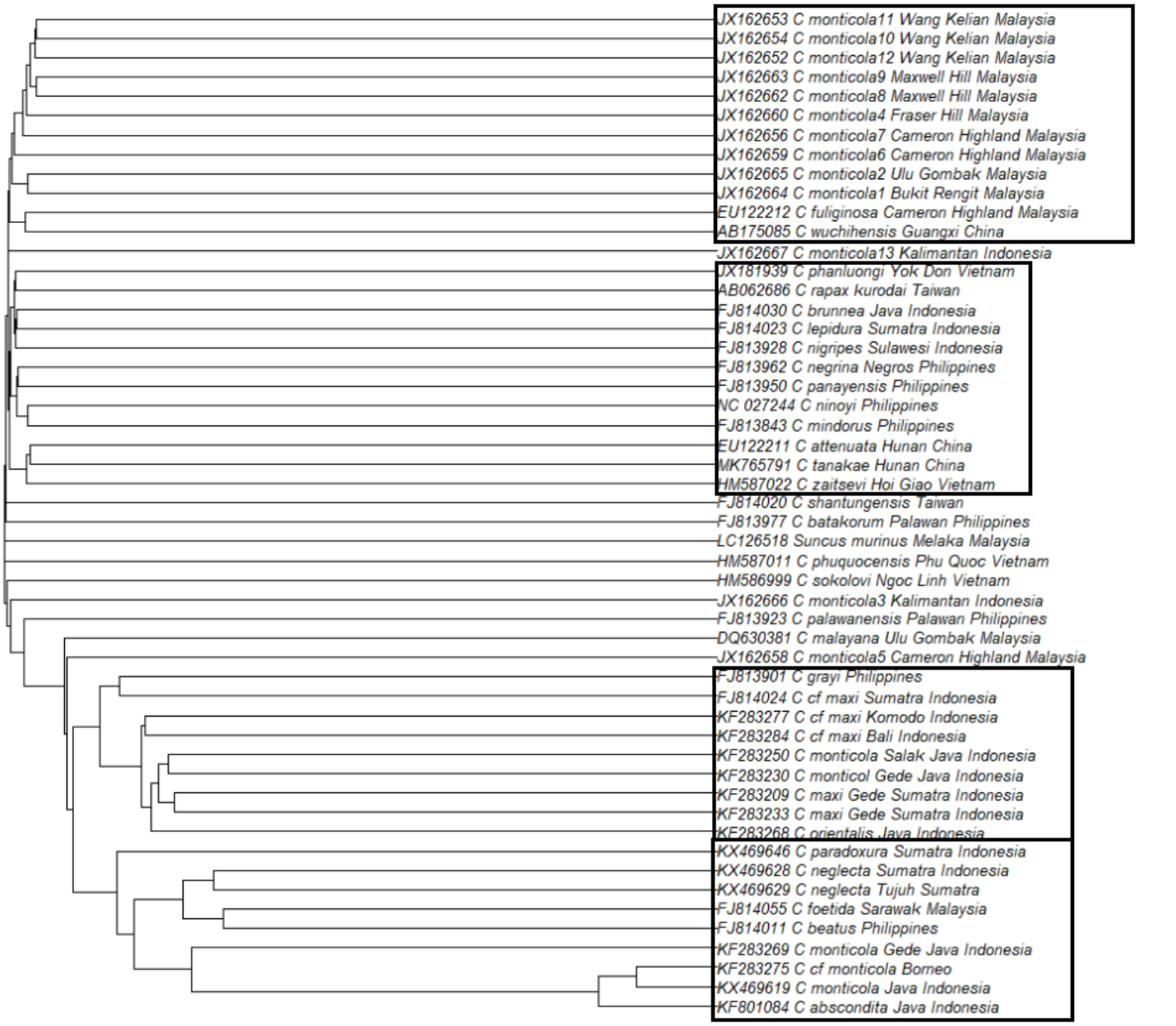
- on importe la base de données et on retire les individus gênants (ceux qui présentent d'autres lettres que A,C,T,G et N dans leur séquence, qui représentent une incertitude du nucléotide à cette position)
- on convertit les bases azotées en valeurs numériques selon la méthode
- on calcule les distances entre les individus
- on définit le(s) ancêtre(s) à la base de l'arbre et on effectue la classification à l'aide de la fonction "root".

On retrouve plus ou moins les mêmes classes que la partition témoin mais on ne parvient pas réellement à les séparer en autant de classes que dans la classification précédente.



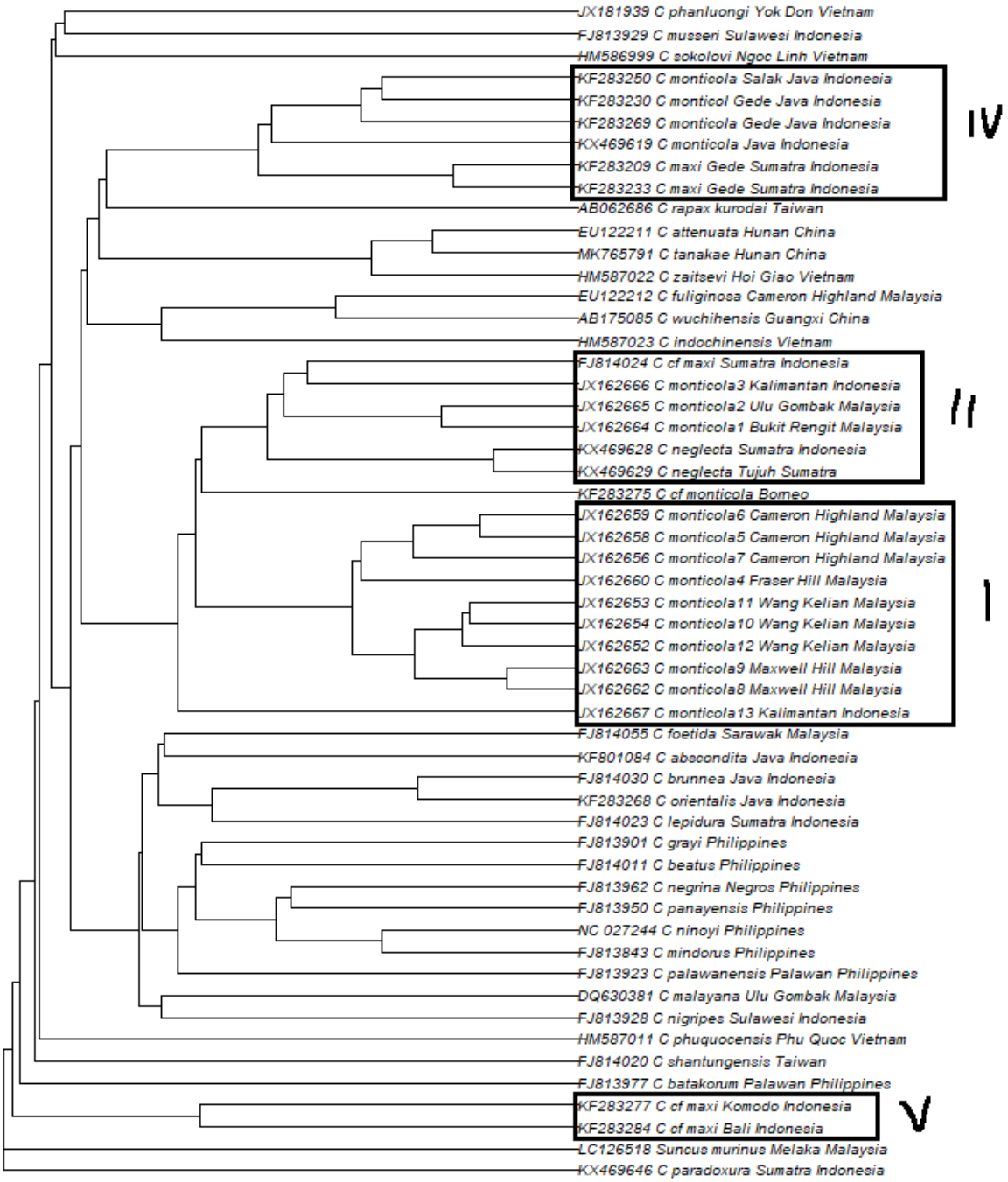
Malgré le fait que la méthode EIIP est censée être la meilleure parmi les méthodes de conversion numérique, on ne parvient toujours pas à distinguer précisément les classes de référence, on a toujours des trop gros groupes comparé à la méthode NJ.

Arbre pour la méthode EIIP

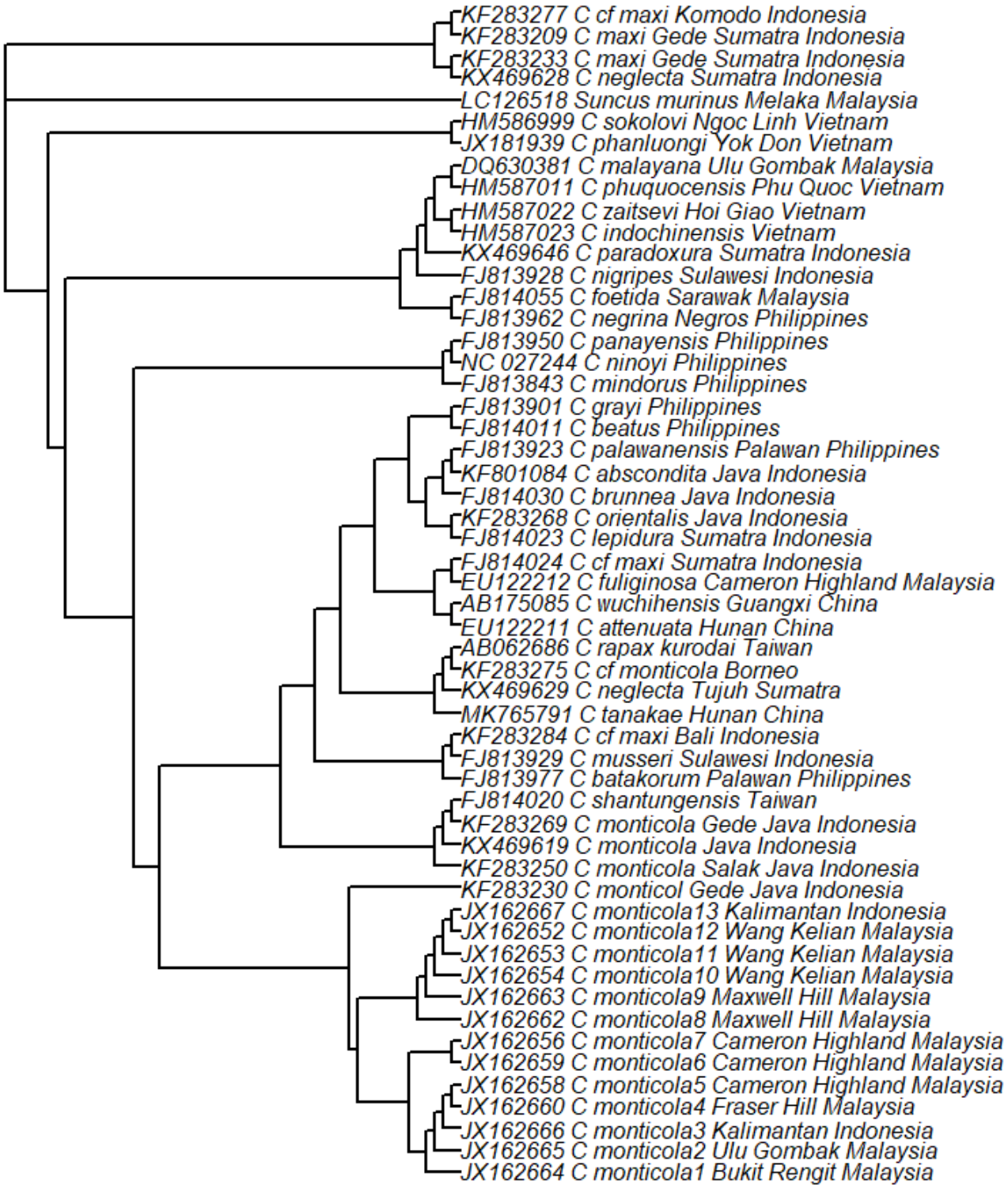


Finalement, nous avons programmé une dernière méthode qui semble être plus proche des résultats des scientifiques.

Cette méthode se nomme Unweighted pair group method with arithmetic mean (UPGMA), elle permet la transformation d'une matrice de distances (entre différents organismes, populations, ou séquences de nucléotides) en un arbre enraciné.
On a donc calculé cette matrice sur le tableau disjonctif complet légèrement modifié (on a remplacé les valeurs N par la moyenne arithmétique des autres individus à cette même position), puis on a effectué la classification sur ces données.
On constate que cette classification est celle qui correspond le plus à la classification des chercheurs. Cependant, on ne distingue pas le groupe III.

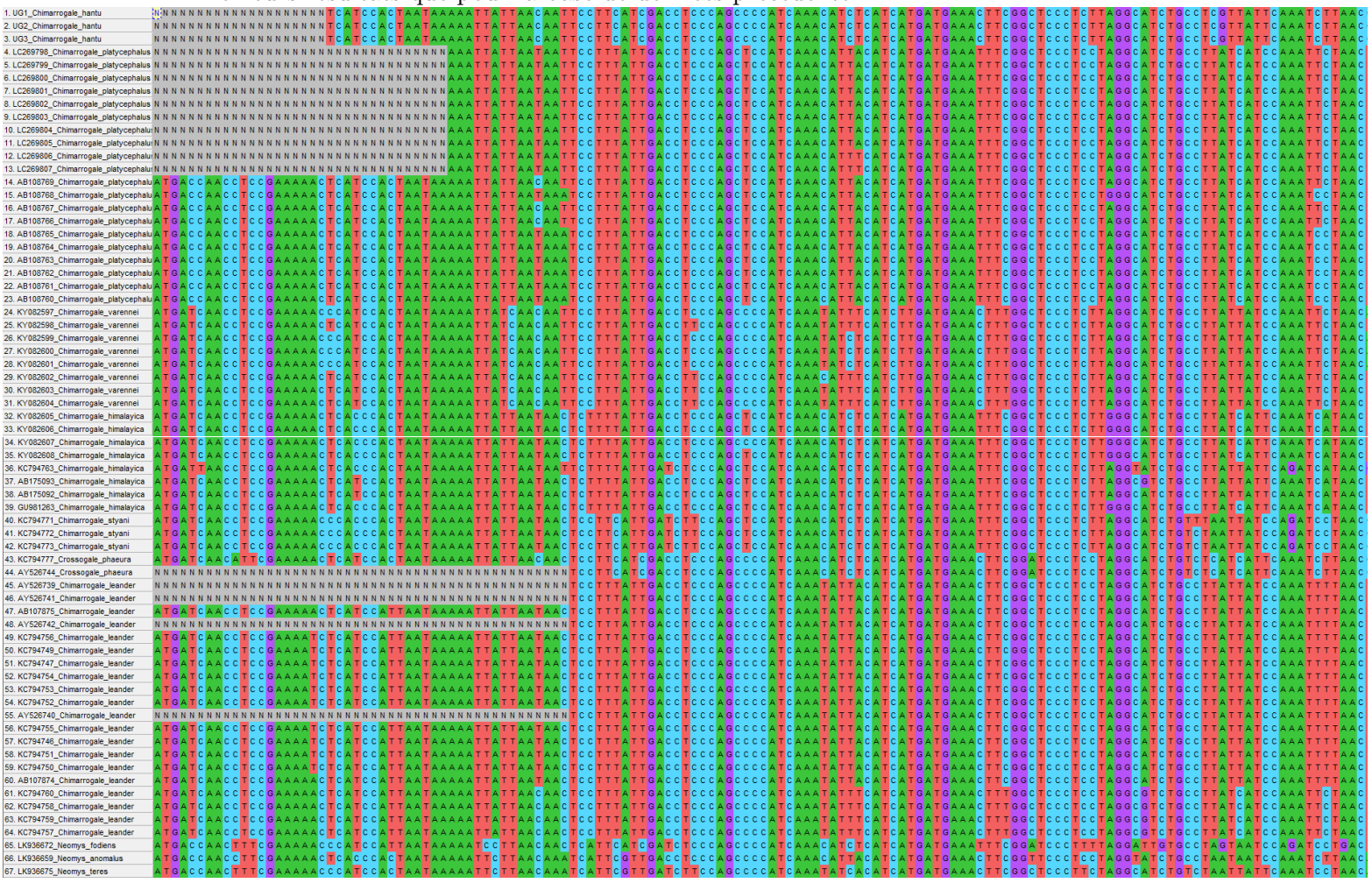


Maitenant nous avons effectué la classification fonctionnelle pour cette base de données, et nous obtenons le résultat suivant. Nous ne retrouvons pas les groupes principaux, ils sont mélangés (par exemple le groupe 4, avec les individus KF).

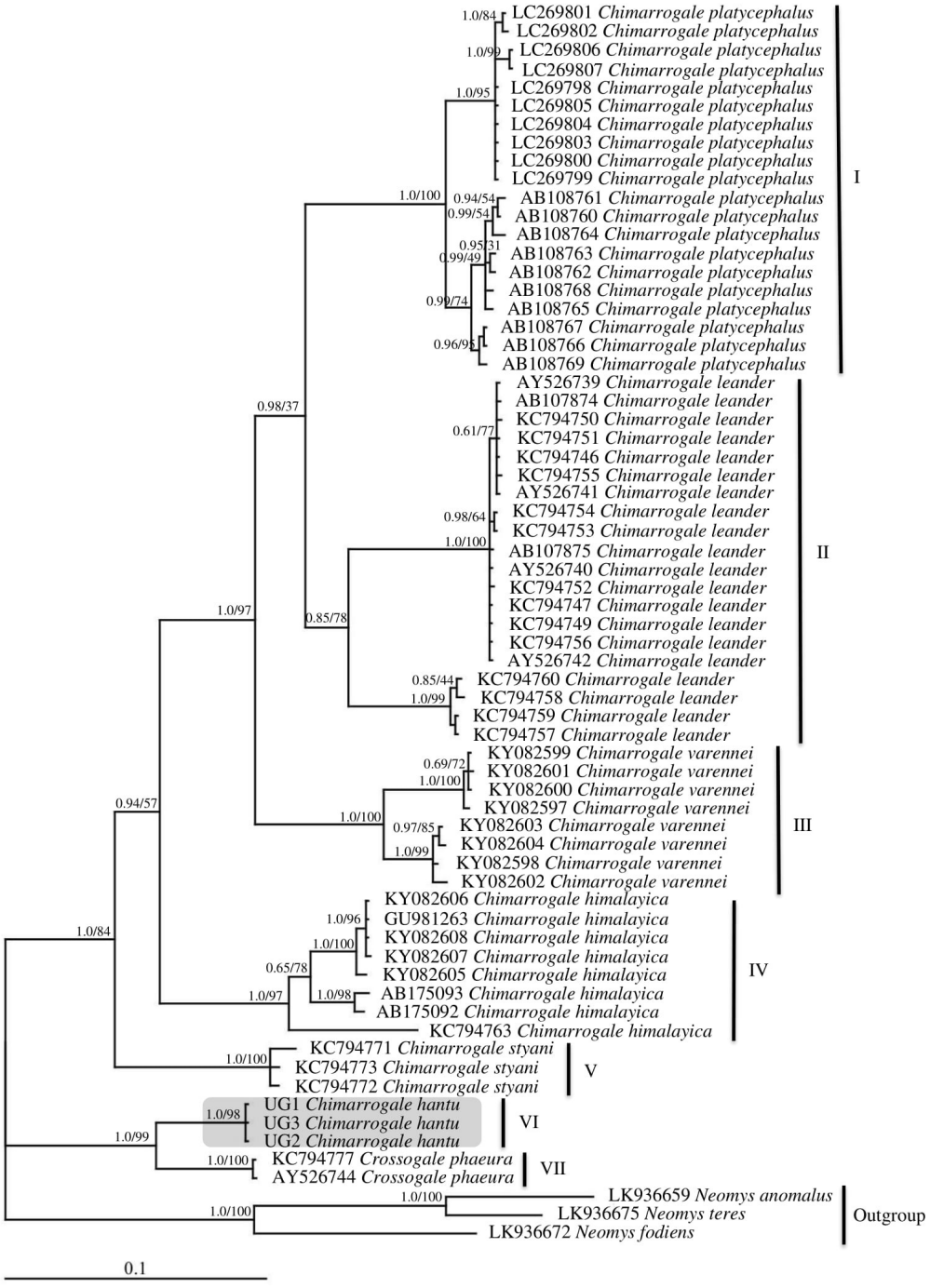


4.5 Application à la deuxième base de données : chimseq

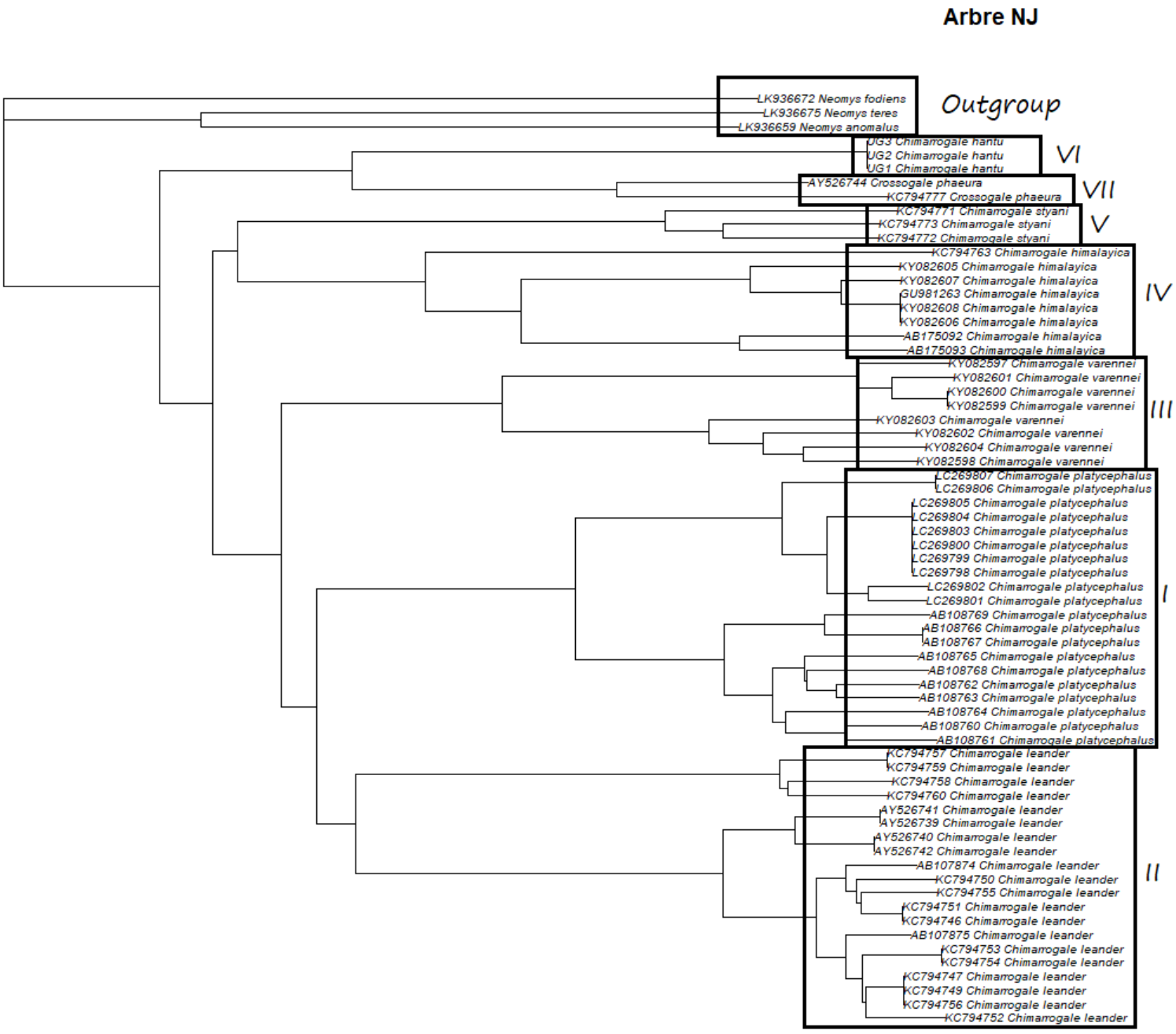
Ici, la base de données porte sur des musaraignes d'eau (cf. [7]). Comme pour la base crocinew notre but est de classifier les 67 individus pour lesquels nous avons une séquence ADN de longueur 1137. Ici, nous connaissons l'existence de trois ancêtres : "Neomys fodiens", "Neomys anomalus", "Neomys teres". Cette fois-ci, la classification des chercheurs a déjà été validée par des experts, et on devrait donc parvenir à de meilleurs résultats que pour la base de données précédente.



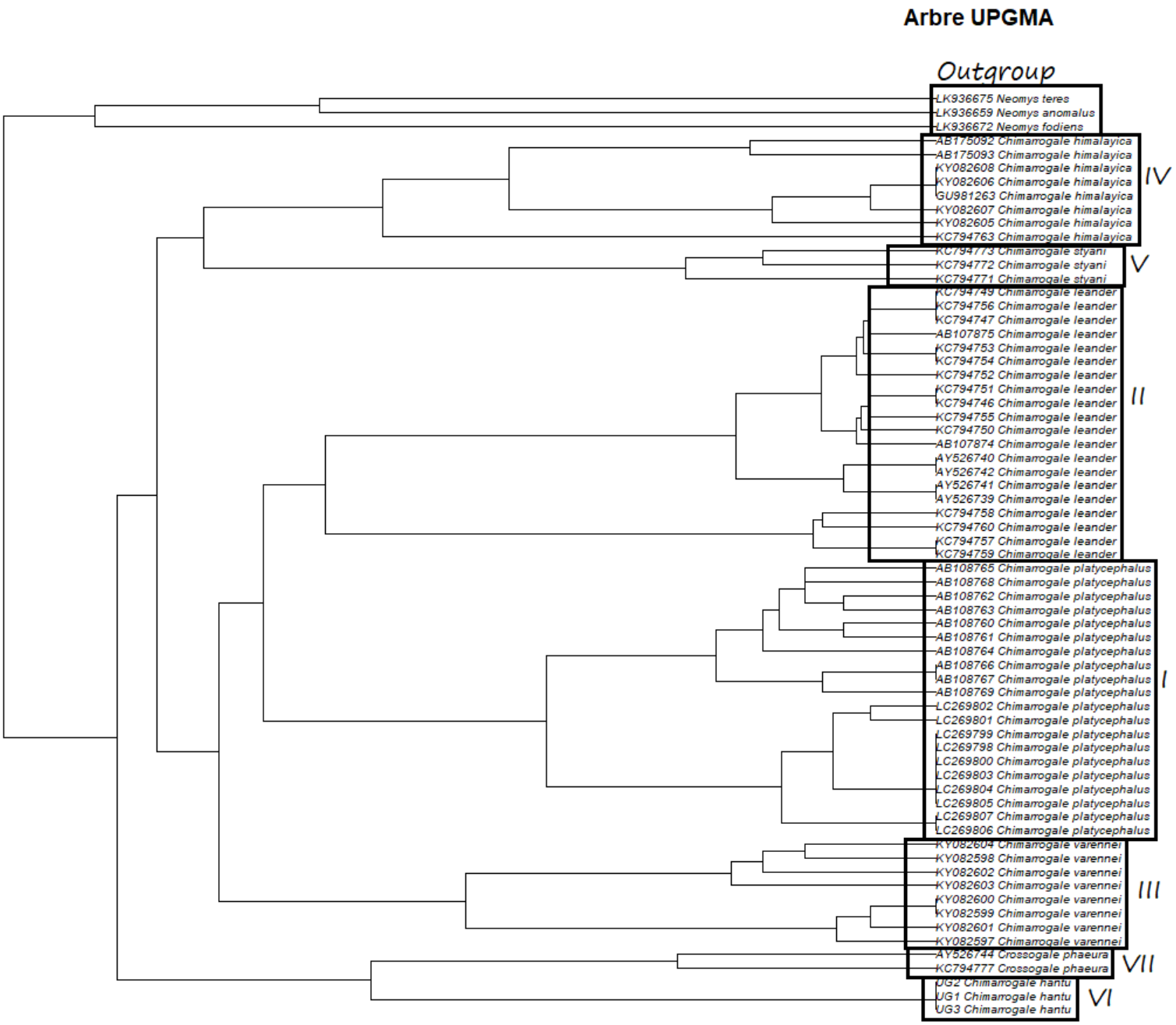
Pour cette base de données, les chercheurs ont obtenu la classification suivante :



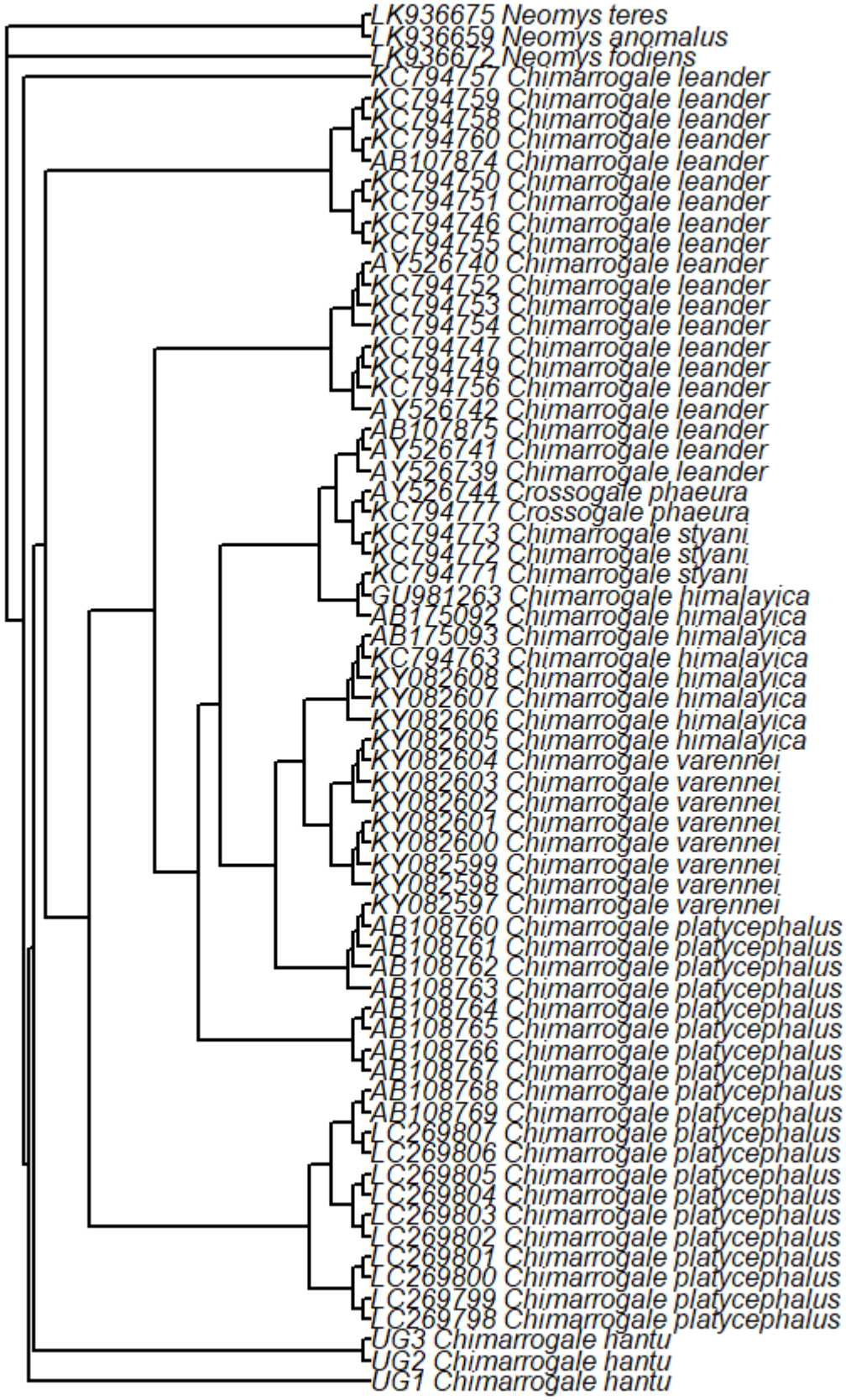
La méthode NJ de R nous donne cette classification qui correspond parfaitement à celle des chercheurs.



Avec la méthode UPGMA (c'est une méthode CAH), nous obtenons la classification suivante. Elle correspond aussi à la clasification des chercheurs.



Pour la classification fonctionnelle de cette base de données, nous constatons que l'arbre obtenu est exactement le même que celui obtenu par les chercheurs en biologie de l'université de Malaya (Malaisie). Ceci est normal car contrairement à crocinew, la base de données chimseq à été validée par des experts.



Pour les deux bases de données, la classification UPGMA était la plus proche de celle des chercheurs parmi celles que nous avons réalisées, nous pouvons donc conclure que c'est cette méthode qui est similaire à celle (NJ) de Mega : la distance utilisée est donc probablement la distance euclidienne appliquée au tableau disjonctif complet (avec la moyenne au lieu des valeurs manquantes).
Enfin, la classification fonctionnelle de chimseq est excellente, mais celle de crocinew n'est pas bonne : cela montre que cette méthode n'est pas toujours adaptée mais elle présente une alternative à la classification multivariée.

5 Conclusion

Rappelons que le but de ce travail est de voir si la classification génétique des espèces via la FDA est une alternative à la classification classique (multivariée) en phylogénétique.
Tout d'abord nous disposions de deux bases de données comportant chacune un certain nombre d'individus avec environ 1000 observations pour chacun de ces individus. Nous aurions pu traiter ces données dans $\mathbb{R}^{1000}$ mais nous souhaitions les voir comme des observations d'une fonction. Ainsi ces données, vues sous cet angle, sont dans un espace de dimension infinie (l'ensemble des fonctions) et il nous est impossible de travailler concrètement dessus.
C'est pourquoi il a fallu réduire la dimension, grâce à l'analyse en composantes principales fonctionnelles (en choisissant convenablement les fonctions propres).
Ensuite, cela nous ramène donc à faire de l'analyse multivariée puisque nous avons suffisamment réduit la dimension, et nous obtenons donc les classifications des sections 5.4 et 5.5.
La classification de la base de données chimseq nous montre que l'analyse de données fonctionnelles peut être une bonne alternative aux méthodes classiques phylogénétiques (NJ), car cette classification est exactement la même que celle obtenue par les biologistes, et qui nous sert de référence.
Cependant, il faut faire très attention aux choix faits lors de l'analyse fonctionnelle, par exemple pour le lissage des données ou pour la distance entre les individus choisie, car sinon nous pouvons obtenir une classification faussée. La classification pour la base de données crocinew est un exemple des limites de l'analyse de données fonctionnelles : en effet, la classification fonctionnelle obtenue en section 5.4 ne correspond pas du tout à la classification de référence (début section 5.4).

Références

[1] Anuj Srivastava et Eric P. Klassen, Functional and Shape Data Analysis, Springer, 2016

[2] J.O. Ramsay . Giles Hooker . Spencer Graves, Functional Data Analysis with R and MATLAB, Springer, 2009

[3] Piotr Kokoszka et Matthew Reimherr, Introduction to Functional Data Analysis, CRC Press, 2017

[4] Xiaoyan Leng et Hans-Georg Müller, Classification using functional data analysis for temporal gene expression data, Oxford University Press, 2005

[5] Gerardo Mendizabal-Ruiz et Israel Roman-Godınez et Sulema Torres-Ramos et Ricardo A. Salido-Ruiz et J. Alejandro Morales, On DNA numerical representations for genomic similarity computation, PLOS ONE, 21 Mars 2017

[6] Swarna Bai Arniker, Numerical Representation of DNA sequences, ResearchGate, Juillet 2009

[7] Muhammad Farhan Abd Wahab. Dharini Pathmanathan. Masaharu Motokawa. Faisal Ali Anwarali Khan. et Hasmahzaiti Omar, Taxonomic assessment of the Malayan water shrew, Chimarrogale hantu Harrison, Mammalian Biology

[8] Exemple du Neighbor-Joining de Wikipedia `https://en.wikipedia.org/wiki/Neighbor_joining#Example`

[9] Jones, N. S., Moriarty, J., Evolutionary inference for function-valued traits : Gaussian process regression on phylogenies, Journal of The Royal Society Interface, 2013

[10] Leng, X., Müller, H. G., Classification using functional data analysis for temporal gene expression data, Bioinformatics, 2005.

[11] Revell, Liam J., Size-correction and principal components for interspecific comparative studies, International Journal of Organic Evolution 63, 2009

[12] Wu, P. S., Müller, H. G., Functional embedding for the classification of gene expression profiles. Bioinformatics, 2010

[13] Srivastava, A., Klassen, E. P., Functional and shape data analysis., New York Springer, 2016

6 Annexe : Code R

6.1 L'exemple de la base de Fourier

```
library(fda)

daytime = (1:365)-0.5
JJindex = c(182:365, 1:181)
tempmat = daily$tempav[JJindex,]
tempbasis = create.fourier.basis(c(0,365),65)
tempfd = smooth.basis(daytime, tempmat, tempbasis)$fd
tempfd$fdnames = list("Day (July 2 to June 30)","Weather Station","Mean temperature (deg. C)")
plot(tempfd, col=1, lty=1)
lines(mean(tempfd),col="red", lwd=3)

tempmat=eval.bifd(daytime,daytime,var.fd(tempfd))
persp(daytime,daytime, tempmat)
par(mfrow=c(1,1), mar=c(5.1, 4.1, 4.1, 2.1))
```

6.2 Simulation de données génétiques et classification

```
sim1 <-function(p=100, freq=0.45){
  #freq= sequences de GC
    freq1 <- 1-freq ## freq de AT
    sequence <- as.vector(sample(c("A", "C", "G", "T"), size=p, replace=TRUE, prob=c(freq1/2, freq/2, freq/2, freq1/2)))
  }
# 100 sequences de 1000 gènes
n=50
genes=1000
seq=matrix(numeric(n*genes),nrow=n)
for (i in 1:n){
  seq[i,]=sim1(p=1000)
  #ACGT=1234
  f=factor(seq[i,])
  levels(f)=c(1,2,3,4)
  seq[i,]=f
}

liste=1:genes
plot(liste,seq[1,])
for (i in 2:n){
  points(liste,seq[i,])
}

#mean(seq[1,]) ne marche pas


library("FactoMineR")
library("factoextra")

# matrice des distances
dist <- dist(seq, method = "euclidean")
# CAH avec Ward (option ward.D2 avec les distances au carrÃ© ou ward avec la distance)
hc <- hclust(dist, method = "ward.D2")
# Les regroupements pas Ã  pas
hc$merge
# La hauteur des branches est la distance
hc$height
# Arbre
plot(hc, cex = 0.5)
```

6.3 Classification des données (méthodes numériques)

```
fastatodataframe <- read.fasta(file="crocinew.fas", as.string = FALSE)
t=fastatodataframe
t=t[-c(20,44)]

m=length(t[[1]])
n=length(t)

for (i in 1:length(t)){
  a=t[[i]]
  f=factor(a)
  levels(f)=list("0"="n", "1"="a", "2"="t", "3"="c", "4"="g")
  t[[i]]=as.numeric(paste(f)) #si on ne met pas paste, les 4 se transforment en 5 (je ne sais pas pourquoi)
  }

df=as.data.frame(t) #data frame des données

df=t(df) #pour bien faire la classification

# Différentes méthodes de modélisation
# On a mis avant
# 0 pour n
# 1 pour a
# 2 pour t
# 3 pour c
# 4 pour g

methode=3

df3=df
for (i in 1:length(df[,1])){
  a=df[i,]
  f=factor(a)
  if (methode==2){
    levels(f)=list("0"="0", "-1.5"="1", "1.5"="2", "0.5"="3", "-0.5"="4")
  }
  if (methode==3){
    levels(f)=list("0"="0", "0.1260"="1", "0.1335"="2", "0.1340"="3", "0.0806"="4")
  }
  df3[i,]=as.numeric(paste(f))
  }

dist3 <- dist(df3, method = "euclidean")
dend3 <- hclust(dist3, method = "average")

dendphylo3=as.phylo.hclust(dend3,use.labels=TRUE, directed=FALSE)
phyp3 <- root(dendphylo3,"LC126518_Suncus_murinus_Melaka_Malaysia")

plot(phyp3,align.tip.label=1, cex=0.8)
title("Arbre pour la méthode EIIP")
```

6.4 Classification des données (UPGMA)

```
dna <- fasta2DNAbin("crocinew.fas")

dna

X <- DNAbin2genind(dna)

##Conversion numerique, frequence de "a" "c" "g" "t" ?? la position
#et calcul moyenne de "n" et "w"

binary_num=tab(X,freq = TRUE, NA.method ="mean")

view(binary_num)


##Distance euclidienne
D <- dist(binary_num)
#x11()
temp <- as.data.frame(as.matrix(D))
table.paint(temp, cleg = 0, clabel.row = .5, clabel.col = .5)

tre <- nj(D)


## root
phy <- root(tre, "LC126518_Suncus_murinus_Melaka_Malaysia")
#x11()
plot(phy, cex = .6)


########### Est-ce-que l'arbre est approprie ?? la distance
x <- as.vector(D)
y <- as.vector(as.dist(cophenetic(phy)))
plot(x, y, xlab = "Distances euclidiennes", ylab = "Distance utilisee dans la construction de l'arbre",
     main = "NJ appropriee ?", pch = 20, col = transp("black",.1), cex = 3)
abline(lm(y~x), col = "red")
##oui
cor(x,y)^2

## dendogramme

tre2 <- as.phylo(hclust(D,method = "average"))
phy2 <- root(tre2, "LC126518_Suncus_murinus_Melaka_Malaysia")
plot(phy2,align.tip.label=1, cex = .6)

title("Arbre UPGMA")

########### Est-ce-que l'arbre est approprie ?? la distance

y <- as.vector(as.dist(cophenetic(tre2)))
plot(x, y, xlab = "Distances euclidiennes", ylab = "Distance utilisee dans la construction de l'arbre",
     main = "UPGMA appropriee ?", pch = 20, col = transp("black",.1), cex = 3)
abline(lm(y~x), col = "red")
cor(x,y)^2
```

6.5 Lissage + ACP des données

```
#dna <- fasta2DNAbin("crocinew.fas")

dna <- fasta2DNAbin("chimseq.fas")

####################
X <- DNAbin2genind(dna)

bin=tab(X,freq = TRUE, NA.method ="mean")

#########
bin=t(bin)

############# Fourier transform of bin

Z = fft(bin)
PSZ = abs(Z)^2
PSZ2=PSZ[-1,]


#############

n=dim(PSZ2)[2]
m=dim(PSZ2)[1]


x=1:m

####
daybasis <- create.fourier.basis(c(1, m), nbasis=51)

smoothf <- smooth.basis(x,PSZ2,daybasis)$fd

plot(smoothf)
################
```

```
BM.Mean <- rowMeans(PSZ2)

#ACP
BM.pca     <- pca.fd(smoothf, nharm=3)
par(mfrow=c(1,3), mar=c(2.1, 1.1, 4.1, 1.1))
# Les valeurs propres
invisible(plot(BM.pca$values[1:3], type="o", ylab="", main="Valeurs propres estimÃ©es (p=3)"))
# Les valeurs propres
invisible(plot(BM.pca$varprop[1:3], type="o", ylab="", main="Proportions de variance expliqu?e (p=3)"))
par(mfrow=c(1,1))
#Les fonctions propres
invisible(plot(BM.pca$harmonics, lwd=3, ylab="", main="Les composantes (p=3)"))


## Approximation des courbes sur la base composÃ©e de p premiÃ¨res fonctions propres

v_hat_mat  <- eval.fd(x, BM.pca$harmonics)
xi_hat_mat <- BM.pca$scores

# ACP-approx
X_fpca_fit <- BM.Mean + (xi_hat_mat %*% t(v_hat_mat))
# plot
par(mfrow=c(1,1), mar=c(5.1, 4.1, 2.1, 2.1))
invisible(plot(smoothf[1], lwd=1.3, main="Approximation de la 1ere donnee avec p=3"))
lines(y=X_fpca_fit[1,], x=x, col="red", lwd=1.3)
legend("topright", lty = c(1,1), col=c("black","red"), legend = c("Vraie","Approx"))

#### Utilisation des scores pour la classification
X2=as.data.frame(xi_hat_mat)
rownames(X2)=colnames(bin)

dist <- dist(X2, method = "canberra")
dist <- dist(X2, method = "maximum")
##########

dend <- hclust(dist, method = "average")


labels(dend)=colnames(bin)

#convert as phylotree to include the outgroup

dendphylo=as.phylo.hclust(dend,use.labels=TRUE, directed=FALSE)

#phyp <- root(dendphylo, "LC126518_Suncus_murinus_Melaka_Malaysia")

phyp <- root(dendphylo, c("LK936672_Neomys_fodiens", "LK936659_Neomys_anomalus", "LK936675_Neomys_teres"))

##

plot(phyp,edge.width = 2,cex = 1, use.edge.length = FALSE)
```

6.6 Simulation du mouvement Brownien

```
## Simulation de processus Brownien
Il y a 100 courbes discrétisées en 10000 points
```

```{r, echo=FALSE, fig.align='center',  out.width="90%"}
set.seed(23)
n       <- 100
J       <- 100
BM.mat  <- matrix(0, ncol=n, nrow=J)
for(i in 1:n){BM.mat[,i] <- cumsum(rnorm(J, sd = 1/100))}
BM.Mean <- rowMeans(BM.mat)
BM.SD   <- apply(BM.mat,1,sd)
xx          <- seq(0,1,len=J)
par(mfrow=c(1,2))
matplot(x=xx, y=BM.mat, xlab="", ylab="", type="l", col=gray(.7), lty=1, main="Mouvements Brownien")
lines(x=xx, y=BM.Mean)
lines(x=xx, y=BM.SD, lty=2)
legend("topleft", lty = c(1,2), legend = c("Moyenne Empirique","Déviation empirique"))
matplot(x=xx, y=BM.mat, xlab="", ylab="", type="l", col=gray(.7), lty=1, main="Mouvements Brownien")
lines(x=c(0,1), y=c(0,0), lty=1)
lines(x=c(0,1), y=c(0,sqrt(J*(1/100)^2)), lty=2)
legend("topleft", lty = c(1,2), legend = c("Vraie moyenne","Vraie Ecart-type"))
par(mfrow=c(1,1))
```

```
## Fonction de Covariance empirique
```

```{r, echo=FALSE, fig.align='center', out.width="100%"}
BM.cov <- var(t(BM.mat))
slct   <- c(seq.int(1,100,by=20),100)
par(mfrow=c(1,2), mar=c(1, 1.1, 1.2, 1.1))
persp(xx[slct], xx[slct], BM.cov[slct,slct], xlab="s", ylab="t", zlab="",main="Fonction de Covariance empirique",
      theta = -40, phi = 20, expand = .75, col = "blue", shade = 1.05)
x <- seq(0, 1, length= 30); y <- x
f <- function(x, y){min(x,y)}
f <- Vectorize(f)
z <- outer(x, y, f)
persp(x, y, z, xlab="s", ylab="t", zlab="",
      main="Vraie Fonction de Covariance",
      theta = -40, phi = 20, expand = .75, col = "blue", shade = 1.05)
par(mfrow=c(1,1), mar=c(5.1, 4.1, 4.1, 2.1))
```

```
## Lissage des données pour les rendre fonctionnelles
```

```{r echo=FALSE, fig.align='center',out.width="100%"}
# Lissage des données pour les rendre fonctionnelles avec la base de Splines
BSPL.basis <- create.bspline.basis(rangeval=c(0,1), nbasis=90)
# Tranformation des données en courbes
BM.fd      <- smooth.basis(argvals=xx, y=BM.mat, fdParobj=BSPL.basis)
#ACP
BM.pca     <- pca.fd(BM.fd$fd, nharm=3)
par(mfrow=c(1,3), mar=c(2.1, 1.1, 4.1, 1.1))
persp(xx[slct], xx[slct], BM.cov[slct,slct], xlab="s", ylab="t", zlab="",
      main="Fonction de Covariance Empirique", theta = -40, phi = 20, expand = .75, col = "blue", shade = 1.05)
# Les valeurs propres
invisible(plot(BM.pca$values[1:3], type="o", ylab="", main="Valeurs propres estimées (p=3)"))
# Les valeurs propres
invisible(plot(BM.pca$varprop[1:3], type="o", ylab="", main="Proportions de variance expliquée (p=3)"))
par(mfrow=c(1,1))
#Les fonctions propres
invisible(plot(BM.pca$harmonics, lwd=3, ylab="", main="Les composantes (p=3)"))
```

Approximation des courbes sur la base composée de p premières fonctions propres

```{r echo=TRUE, echo=FALSE, fig.align='center', fig.height=4}
BM.pca <- pca.fd(BM.fd$fd, nharm=10)
v_hat_mat  <- eval.fd(xx, BM.pca$harmonics)
xi_hat_mat <- BM.pca$scores

# ACp-approx
X_fpca_fit <- BM.Mean + (xi_hat_mat %*% t(v_hat_mat))

# plot
par(mfrow=c(1,1), mar=c(5.1, 4.1, 2.1, 2.1))
invisible(plot(BM.fd$fd[1], lwd=1.3, main="Approximation de la 1er donnée avec p=10"))
lines(y=X_fpca_fit[1,], x=xx, col="red", lwd=1.3)
legend("topright", lty = c(1,1), col=c("black","red"), legend = c("Vraie","Approx"))
```