Commandes « R » et explications

Charger la bibliothèque « ape » permettant de travailler avec des données phylogénétiques sur « R »[1].

```
library(ape)
```

Diriger « R » vers le répertoire contenant le(s) fichier(s) texte de l'arbre phylogénétique étudié (Fichier→Changer le répertoire courant).

Charger le fichier texte (par exemple « treefile ») contenant les données relatives à l'arbre phylogénétique que l'on souhaite étudier en lui donnant un nom (par exemple « arbre »).

```
arbre<-read.tree("tree")
```

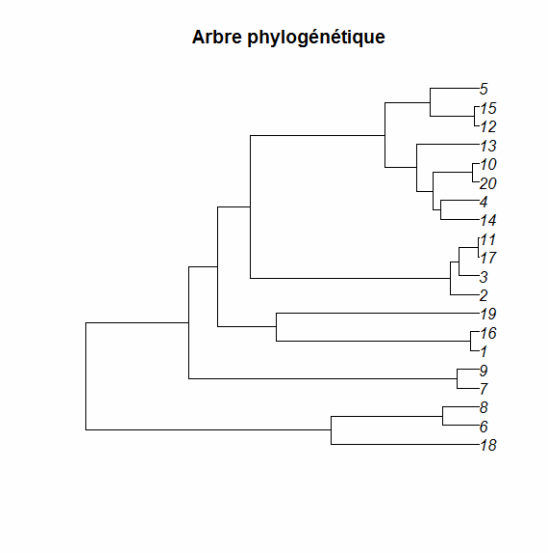
Définir le vecteur (nommé par exemple « t ») contenant les temps propre à chacune des branches de notre arbre (pour être exact, ce vecteur contient les distances séparant chaque nœud de notre arbre du sommet des branches de ce dernier, autrement dit les distances séparant chaque nœud du temps zéro).

> La comparaison de séquences ADN appartenant à des organismes d'espèces différentes peut être utilisée dans le but d'estimer l'âge relatif du dernier ancêtre commun à ces espèces. On part alors de principe que les substitutions nucléotidiques survenues entre ces espèces se sont accumulées au hasard et de manière régulière si bien que leur nombre est proportionnel au temps s'étant écoulé depuis l'événement de spéciation.

```
t<-branching.times(arbre)
```

On peut maintenant réaliser une représentation graphique de notre arbre.

```
plot(arbre,main="Arbre phylogénétique")
```



[1] Il est possible, dans le but de gagner un peu de temps, de créer un script source (fichier de type R) contenant les commandes principales que nous utilisons (comme les trois fonctions et le chargement de la librairie ape) et de le faire lire par « R » lorsque nous ouvrons la console (Fichier→Sourcer du code R…).

Commande pour la fonction de base (nommée « p(l,m,t) ») que nous allons utiliser et qui est celle relative au processus dit de « Birth and Death ». Dans cette formule, l est défini comme le taux de spéciation, m comme le taux d'extinction et t comme les temps séparant chaque nœud de notre arbre du sommet des branches de ce dernier (à savoir ceux contenu dans notre vecteur « t »). Relevons que, quoi qu'il arrive, ces trois variables sont comprises dans l'ensemble [0 ;∞[.

> Birth and Death Process : $p_1(t)=[(l-m)^2 \times e^{(l-m)t}] / [(l \times e^{(l-m)t} - m)^2]$ → donne la probabilité que pour un temps t une espèce donnée ne connaisse ni événement de spéciation ni événement d'extinction.

```
p<-function(l,m,t) {
num<-(l-m)^2*exp((l-m)*t)
den<- (l*exp((l-m)*t)-m)^2
return(num/den)}
```

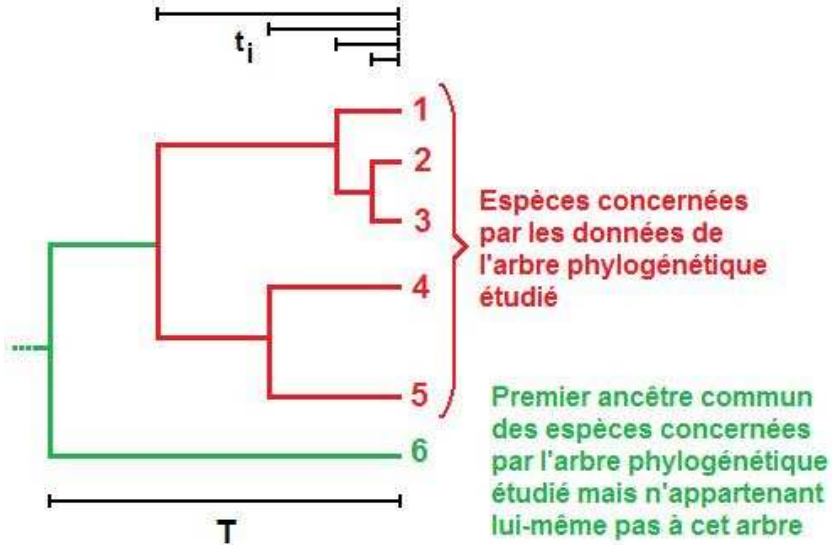
Commande pour la fonction prenant en compte les exceptions mathématiques pouvant se produire dans notre fonction de base (et nommée « prob(l,m,t) »). Principalement il faut prendre en compte deux cas particulier :

(a) Si l=m, alors le dénominateur vaut zéro. Dans ce cas, on souhaite utiliser une version simplifiée de la formule de processus de Birth and Death (nommée « like »).
(b) Si l=m et que le résultat obtenu est un « NaN » ou un « Infinite », alors on souhaite que le programme nous retourne une très grande valeur négative dans le but d'obtenir au final un probabilité nulle.

Suivant les valeurs de l, m et t introduites dans notre fonction de base, on peut obtenir des valeurs extrêmes (très petites et très grandes) qui sont de ce fait difficilement comparables et tout aussi difficilement représentables. C'est pourquoi on introduit dans cette fonction un logarithme en base dix.

```
prob<-function(l,m,t) {
if(m==l) {
like<-log(1/((1+l*t)^2))}
else{ like<-log(p(l,m,t))}
if(is.nan(like)| is.infinite(like)) {return(-1000000000)}
else  { return(like)}
}
```

Commande pour la formule dite du maximum de vraisemblance (Maximum likelihood estimation of speciation rates). Cette formule nous donne la probabilité que pour un l et un m donné le processus d'évolution ait effectivement produit le nombre d'espèces que l'on a dénombrées au temps présent (ou temps zéro, soit le sommet des branches de l'arbre). Une nouvelle variable apparaît dans cette formule, il s'agit de la variable T (cf. illustration ci-dessous). T représente la distance (~temps) séparant le sommet des branches de notre arbre (à savoir le temps zéro) du premier nœud ancestral n'étant pas compris dans l'arbre en question (la valeur de T nous est en principe transmise avec l'arbre étudié –et figure dans le fichier texte correspondant– et il est alors nécessaire de la définir dans « R »).



Maximum likelihood estimation of speciation rates (où a=nombre de taxa observé au temps présent et que l'on peut lier à la variable t de la manière suivante ; t_i=a-1):

$prob(a, t_1, t_2,\ldots, t_{a-1}| l, m, T) = p_1(T) \times l^{(a-1)} \times \mathbf{\Pi} p_1(t_i)$

Dans notre cas, l'introduction d'un logarithme entraîne la modification du produit en somme et des exposants en facteurs multiplicatifs :

$\log_{10}[prob(a, t_1, t_2,\ldots, t_{a-1}| l, m, T)]= p_1(T) + \mathbf{\Sigma} p_1(t_i)+ l(a-1)$

```
vrais<- function(l,m,t,T) {
a<-   prob(l,m,T)   +   sum(sapply(t,   prob,   l=l,   m=m))   +
length(t)*log(l)
return(a)}
```

Nous avons maintenant codé la fonction d'intérêt (Maximum likelihood estimation of speciation rates), et nous connaissons les variables t et T. Le but est donc de générer une matrice contenant toute les combinaisons des valeurs de l et m que nous souhaitons tester afin de l'introduire dans notre fonction « vrais(l, m, t, T) ». La probabilité maximale obtenue suite à l'insertion de cette matrice dans la fonction nous donnera les valeurs de l et de m les plus probables pour expliquer le nombre d'espèces dénombrées au temps présent ainsi que les différents t trouvés suite aux séquençages ADN.

Création d'une matrice 100x100 et définitions des valeurs de l et m que nous souhaitons tester[2] (en l'occurrence nous allons tester des taux de spéciation et d'extinction allant de 0.1 à 10 en allant par pas de 0.1 afin d'obtenir des vecteurs de l et m ayant la même taille que notre matrice). Il est logique de noter que plus le nombre des l et m testé ainsi que la précision de ces valeurs (soit la longueur du pas introduite) est importante et plus les valeurs de l et de m obtenues par le maximum de vraisemblance pourront être considérées comme étant exactes.

```
mat<-matrix(numeric(10000),ncol=100)
```

[2] Il est important de préciser ici que des taux de spéciation et d'extinction nulles ne sont biologiquement pas concevables. De ce fait, les valeurs l=0 et m=0 ne sont jamais testées.

```
l<-seq(0.1,10,0.1)
m<-l
```

ou alors

```
l<-seq(0.1,10,length.out=100)
m<-l
```

[Vérif. :

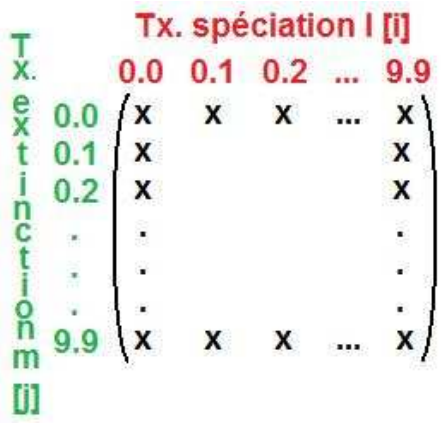
```
length(l)
[1] 100 ]
```

La commande length.out=x utilisée ici permet de stipuler que l'on souhaite que la séquence générée soit de longueur x.

Création d'une boucle permettant de tester à l'aide de notre fonction du maximum de vraisemblance toutes les combinaisons des valeurs l et m présentes dans la matrice précédemment crée.

```
for(i in 1:100) {
for(j in 1:100) {
mat[i,j]<-vrais(l[i], m[j], t, T)}}
```

Pour résumer, on a maintenant une matrice 100x100 se présentant ainsi :

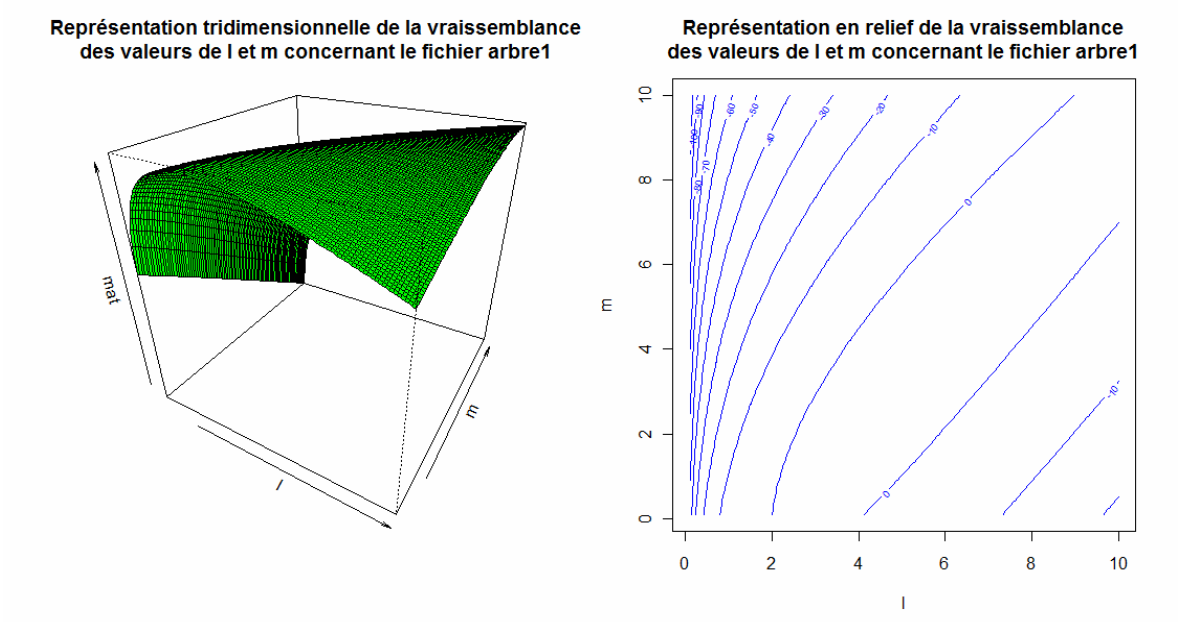


Où x_{ij} est la probabilité avec laquelle les valeurs des taux de spéciation l et d'extinction m correspondants expliquent les temps t (autrement dit x_{ij} est la vraisemblance de l_i et m_j).

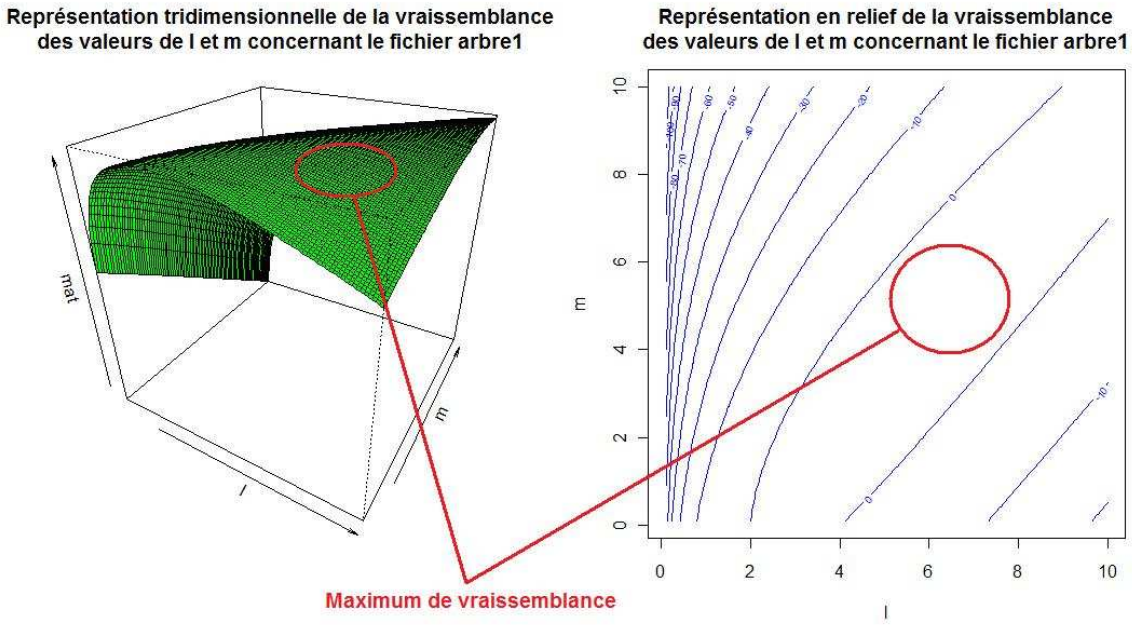
On peut donc représenter ces valeurs (l, m et la vraisemblance qui y est maintenant associée) à l'aide de graphiques.
Le premier graphe représente la surface de distribution des valeurs de vraisemblance en fonction de l et m.
Le second est une représentation « en relief » (analogie avec les courbes de niveaux d'une carte topographique) où les valeurs de vraisemblance sont indiquées le long des courbes du graphe.

```
persp(l,m,mat,main="Représentation tridimensionnelle de la
vraisemblance des valeurs de l et m concernant le fichier
arbre1",col="green",theta=30,psi=30)

contour(l,m,mat,xlab="l",ylab="m",main= "Représentation en
relief de la vraisemblance des valeurs de l et m concernant le
fichier arbre1",col="blue")
```



A partir de ces représentations, on peut donc faire une estimation graphique des valeurs de l et m correspondantes au maximum de vraisemblance.



Les commandes suivantes vont nous permettre de passer d'une estimation graphique des valeurs de l et m correspondants au maximum de vraisemblance à des valeurs calculées mathématiquement. Pour ce faire, une des méthodes envisageable aurait été de calculer la dérivée de la formule donnant le maximum de vraisemblance et de l'égaliser à zéro afin de trouver les maximum de notre surface de distribution de probabilité (point de la surface présentant une pente nulle). Malgré le fait que cette méthode ait l'avantage de fournir des résultats exacts, elle s'avère difficilement applicable à notre cas (dérivation trop complexe). Nous avons donc opter pour l'utilisation d'un processus plus approximatif, à savoir

l'application d'une méthode itérative reposant sur l'algorithme dit de « la méthode de Newton[3] ».

```
vrais2<-function(x){
return(-vrais(x[1],x[2],t,T))}
```

Cette première commande est nécessaire pour stipuler à R que l'on connaît les variables t et T (elles ont été définies précédemment) et que l'on souhaite qu'il nous retourne, lorsque l'on utilisera la fonction `vrais2()`, un vecteur de deux valeur `x[1],x[2]` qui seront les valeurs correspondant aux l et m qu'il aura calculé. Par la suite, la seconde commande que nous allons utiliser repose sur une méthode de minimisation itérative (Non Linear Minimization), or nous recherchons des maximums. C'est pourquoi il est important de définir `vrais2` comme l'opposé de `vrais()`.

Cette seconde commande est l'application par R de la méthode de Newton, et repose sur la fonction `nlm()` (Non Linear Minimization). Dans cette commande, les deux valeurs numériques sont en fait les approximations obtenues graphiquement de respectivement l et m.

```
nlm(vrais2,c(6,4))
$minimum
[1] -2.928932

$estimate
[1] 5.766176 4.496907

$gradient
[1]  1.173420e-08 -1.210728e-08

$code
[1] 1

$iterations
[1] 9
```

Le résultat obtenu sous `minimum` est en fait le logarithme de la probabilité que les valeurs de l et m calculées par la fonction `nlm()` soient effectivement les plus vraisemblables (il convient donc d'en prendre l'exponentiel pour obtenir la valeur exact de cette probabilité). Ces valeurs de l et m sont données dans l'ordre sous `estimate`.

On peut maintenant vérifier la pertinence de ce résultat en traçant sur le graphe `contour()` correspondant à l'arbre duquel découlent ces calculs des droites passant par les l et m obtenus.

[3] Algorithme efficace pour trouver des approximations d'un zéro d'une fonction d'une variable réelle à valeurs réelles. L'algorithme consiste à linéariser une fonction f en un point et de prendre le point d'annulation de cette linéarisation comme approximation du zéro recherché. Partant d'une valeur approximative raisonnable d'un zéro d'une fonction d'une variable réelle, on approxime au premier ordre la fonction par sa tangente en ce point. Cette tangente est une fonction affine dont on sait trouver l'unique zéro (analyse élémentaire). Ce zéro de la tangente sera généralement plus proche du zéro de la fonction. Par cette opération, on peut donc espérer améliorer l'approximation par itérations successives. *(Wikipedia)*

```
contour(l,m,mat,xlab="l",ylab="m",main="Représentation en
relief de la vraisemblance
des valeurs de l et m concernant arbre",col="blue")

abline(h=4.496907,v=5.766176,col="red")
```