Chaînes de Markov et génome

Etienne Pardoux

Introduction

Le but de ces quelques leçons est de tenter d'expliquer le "pourquoi" et le "comment" des algorithmes basés sur les chaînes de Markov cachées pour l'annotation du génome. On parlera aussi des chaînes semi–markoviennes cachées, et on effleurera d'autres applications.

1 Comment lire l'ADN ?

On considère un fragment d'ADN, sous la forme d'un simple brin constitué d'une succession de nucléotides, que nous considérerons comme des lettres dans l'alphabet `a, c, g, t`, par exemple

`a c c g t a a t t c g g a` $\cdots$ `t t g c`

"Lire" ou "annoter" cette séquence consiste essentiellement à la décomposer en *régions codantes à l'endroit* ou *à l'envers* (sachant que l'ADN est constitué en réalité de deux brins complémentaires, avec un `a` toujours apparié avec un `t`, un `c` avec un `g`, qui ne sont pas lus dans le même sens), et *régions non codantes*; dans le cas des génomes eukaryotes il faut en outre découper les *régions codantes* en *introns* et *exons*. Notons que les régions codantes sont lues par codons, i.e. triplets de nucléotides, chaque codon étant ensuite traduit en un *acide aminé*. La succession des acides aminés cosntitue une *protéine*. Il est donc essentiel de lire chaque région codante dans la bonne *phase de lecture*. Oublier un codon n'est pas forcément très grave, mais se tromper en décalant la lecture d'un ou de deux nucléotides est catastrophique !

On est aidé dans cette démarche par la présence d'un codon START (resp. STOP) au début (resp. à la fin) de chaque région codante. Mais tout START potentiel n'en est pas forcément un. Par contre, un STOP potentiel, dans une région codante, rencontré dans la bonne phase de lecture, est un STOP. Et il n'y a pas de signaux aussi nets marquant la transition entre *intron* et *exon*.

Une première possibilité est que les proportions respectives de `a`, de `c`, de `g` et de `t` soient nettement différentes entre plage codante et non codante. Une seconde possibilité est que ces proportions ne sont pas vraiment nettement différentes, et qu'il faut compter les *di* ou *trinucléotides*.

Dans le premier cas, on va distinguer entre région codante et non codante en comparant les proportions de a, de c, de g et de t. Dans le second cas, il faudra compter les paires ou les triplets. Et quelque soit le critère adopté, le plus difficile est de localiser correctement les ruptures (ou changements de plage).

Les méthodes que nous venons d'évoquer pour décomposer une séquence d'ADN en ses différentes plages – dans le but de détecter les gènes – peuvent être vues comme des procédures statistiques associées à une modélisation probabiliste. Cette modélisation n'est pas la même suivant que l'on regarde des fréquences de *nucléotides*, de *bi–* ou de *tri-nucléotides*.

Avant de faire un détour par les modèles probabilistes possibles pour une séquence d'ADN, considérons deux problèmes parmi les plus simples d'analyse de séquences.

1.1 Les ilôts cpg

On note cpg le dinucléotide c *suivi de* g (la notation c g ou c-g désigne plutôt la paire de bases c et g appariée, chacune sur un brin de l'ADN). Dans le génome humain, les dinucléotides ont tendance à disparaître, parce que lorsque la cytosine c est suivie par la guanine g, elle a tendance à être modifiée par méthylation, et méthyl–c mute facilement en thymine t. D'où le fait que les dinucléotides cpg sont plus rares que ce que donne le produit de la fréquence des c par celle des g. Mais le processus de méthylation est inhibé dans certaines portions du génome, autour des promoteurs et des codons START. Dans ces régions on trouve beaucoup de

cpg (en fait plus que le produit de la fréquence des c par celle des c). On appelle ces régions des "ilôts cpg".

On peut donc se poser deux questions. Etant donnée une petite portion de génome, comment décider s'il provient ou non d'un ilôts cpg ? Etant donnée une longue séquence, comment isoler les ilôts cpg ?

1.2 Recherche de gènes dans un génome prokaryote

Dans un génome prokaryote, un gène est une succession de codons (de trinucléotides, qui chacun code pour un acide aminé), encadré par un codon START et un codon STOP. Il y a trois codons START possibles, et trois codons STOP possibles. Mais alors qu'un codon STOP putatif dans la bonne phase de lecture est forcément un STOP, un codon START possible au milieu d'une région non codante n'est pas forcément un START.

On a donc dans un génome prokaryote des *gènes putatifs* consitués d'un codon START, un nombre entier de codons (ie. un multiple de 3 nucléotides), un codon STOP. Comment distinguer, parmi une collection de *gènes putatifs*, les *vrais gènes* des *faux gènes* ? Comment trouver les gènes dans un génome prokaryote ?

2 Le modèle i.i.d

i.i.d. veut dire "indépendants et identiquement distribués". Ici on suppose que les nucléotides d'une sous–séquence donnée sont tirés indépendemment les uns des autres, tous avec la même loi de probabilité. La sous–séquence en question est une "plage homogène" (région codante, région intergénique, ...).

Définissons tout d'abord la notion d'*espace de probabilité* $(\Omega, \mathcal{F}, \mathbb{P})$.

• Ω est l'ensemble de toutes les réalisations possibles de l'expérience aléatoire, ou ensemble de tous les états du monde possibles, ou ensemble de toutes les suites possibles de nucléotides.

• $\mathcal{F}$ est la tribu des événements. En première approximation, on peut ici choisir $\mathcal{F}$ = ensemble des toutes les parties de Ω.

Exemples d'événement :

"Le 1er nucléotide est une purine"

"le triplet en position 7–8–9 est un codon START".

• $\mathbb{P}$ est la *probabilité*, qui à chaque événement $F \in \mathcal{F}$ associe un nombre réel $\mathbb{P}(F)$ de l'intervalle [0,1], et qui vérifie les deux axiomes :

i) $\mathbb{P}(\Omega) = 1$

ii) Si $\{F_n, n \geq 1\} \subset \mathcal{F}, F_n \cap F_m = \emptyset$ dès que $n \neq m$, $\mathbb{P}(\bigcup_n^\infty F_n) = \sum_1^\infty \mathbb{P}(F_n)$

Une *variable aléatoire* à valeurs dans E (par exemple $E = \{\mathtt{a}, \mathtt{c}, \mathtt{g}, \mathtt{t}\}$, ou $E = \mathbb{N}, \ldots$) est une application

$$X = \Omega \to E$$

qui à $\omega \in \Omega$ associe $X(\omega)$ (qui doit vérifier la condition $\{\omega; X(\omega) = x\} \in \mathcal{F}$ pour tout $x \in E$).

Exemples de variable aléatoire

"le 5ème nucléotide de la séquence"

"le rang du 1er codon START dans la séquence"

"le nombre le `t a t a` dans la séquence"

Définition 2.1 *Une suite $(X_1, \ldots, X_n)$ de v.a. est dite indépendante si pour tout $x_1, \ldots, x_n \in E$,*

$$\mathbb{P}(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n) = \prod_{i=1}^n \mathbb{P}(X_i = x_i)$$

♢

Exemple particulier avec $k = 5$:

$$\begin{aligned}
&\mathbb{P}(X_1 = \mathtt{a}, X_2 = \mathtt{c}, X_3 = \mathtt{a}, X_4 = \mathtt{t}, X_5 = \mathtt{g}) \\
&= \mathbb{P}(X_1 = \mathtt{a})\mathbb{P}(X_2 = \mathtt{c})\mathbb{P}(X_3 = \mathtt{a})\mathbb{P}(X_4 = \mathtt{t})\mathbb{P}(X_5 = \mathtt{g}) \\
&= \mathbb{P}(X_1 = \mathtt{a})\mathbb{P}(X_1 = \mathtt{c})\mathbb{P}(X_1 = \mathtt{a})\mathbb{P}(X_1 = \mathtt{t})\mathbb{P}(X_1 = \mathtt{g}).
\end{aligned}$$

Soit X_1 le premier nucléotide de notre séquence. Sa *loi de probabilité* est définie par le vecteur $p = (p_{\mathtt{a}}, p_{\mathtt{c}}, p_{\mathtt{g}}, p_{\mathtt{t}})$ donné par

$$p_{\mathtt{a}} = \mathbb{P}(X_1 = \mathtt{a}),\ p_{\mathtt{c}} = \mathbb{P}(X_1 = \mathtt{b}),\ p_{\mathtt{g}} = \mathbb{P}(X_1 = \mathtt{g}),\ p_{\mathtt{t}} = \mathbb{P}(X_1 = \mathtt{t})$$

Notons que $p_{\mathtt{a}}, p_{\mathtt{c}}, p_{\mathtt{g}}, p_{\mathtt{t}} \geq 0$ et $p_{\mathtt{a}} + p_{\mathtt{c}} + p_{\mathtt{g}} + p_{\mathtt{t}} = 1$.

On dit que les v.a. $(X_1, \ldots, X_n)$ sont i.i.d. (indépendantes et identiquement distribuées) si elles sont indépendantes et toutes de même loi. On dit aussi (dans le langage des statisticiens) que la suite $(X_1, \ldots, X_n)$ est un échantillon de taille n de la loi commune des X_i. A cet échantillon, on associe la loi de probabilité empirique

$$p^n_{\mathtt{a}} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}_{\{X_i=\mathtt{a}\}},\ p^n_{\mathtt{c}} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}_{\{X_i=\mathtt{c}\}},\ p^n_{\mathtt{g}} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}_{\{X_i=\mathtt{g}\}},\ p^n_{\mathtt{t}} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}_{\{X_i=\mathtt{t}\}}.$$

$p^n = (p^n_{\mathtt{a}}, p^n_{\mathtt{c}}, p^n_{\mathtt{g}}, p^n_{\mathtt{t}})$ est une probabilité sur E.

En pratique, la loi commune $p = (p_{\mathtt{a}}, p_{\mathtt{c}}, p_{\mathtt{g}}, p_{\mathtt{t}})$ des X_i est inconnue. Du moins si n est suffisamment grand, p^n est une bonne approximation de p. En effet, il résulte de la loi des grands nombres que

$$p^n_{\mathtt{a}} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}_{\{X_i=\mathtt{a}\}} \to \mathbb{E}(\mathbf{1}_{\{X_1=\mathtt{a}\}}) = \mathbb{P}(X_1 = \mathtt{a})$$

quand $n \to \infty$ (même résultat pour $\mathtt{c}, \mathtt{g}, \mathtt{t}$), et en outre d'après le théorème de la limite centrale,

$$\sqrt{n}(p_{\mathtt{a}} - p^n_{\mathtt{a}}) \xrightarrow{\mathcal{L}} N(0, p_{\mathtt{a}}(1 - p_{\mathtt{a}})),$$

c'est à dire pour tout $\delta > 0$,

$$\mathbb{P}\left(-\delta\sqrt{\frac{p_{\mathtt{a}}(1-p_{\mathtt{a}})}{n}} \leq p_{\mathtt{a}} - p^n_{\mathtt{a}} \leq \delta\sqrt{\frac{p_{\mathtt{a}}(1-p_{\mathtt{a}})}{n}}\right) \to \frac{1}{\sqrt{2\pi}}\int_{-\delta}^{\delta} e^{-\frac{x^2}{2}}dx,$$

et donc puisque $\sqrt{p_{\mathtt{a}}(1-p_{\mathtt{a}})} \leq \dfrac{1}{2}$,

$$\mathbb{P}\left(|p_{\mathtt{a}} - p^n_{\mathtt{a}}| > \frac{\delta}{2\sqrt{n}}\right) \leq \sqrt{\frac{2}{\pi}}\int_{\delta}^{\infty} e^{-\frac{x^2}{2}}dx$$

On peut donc estimer la loi inconnue p, sous l'hypothèse que les nucléotides sont i.i.d., donc en particulier que la plage considérée est *homogène*.

L'hypothèse d'indépendance n'est pas forcément vérifiée, mais en réalité elle n'est pas absolument nécessaire pour que la démarche ci–dessus puisse être justifiée.

3 Le modèle de Markov

Supposer que les nucléotides sont indépendants les uns des autres n'est pas très raisonnable. On peut penser par exemple que, dans une région codante, la loi du 2ème nucléotide d'un codon dépend de quel en est le premier nucléotide.

D'où l'idée de supposer que la suite $(X_1, \ldots, X_n)$ forme une chaîne de Markov.

Rappelons la notion de probabilité conditionnelle $\mathbb{P}(A|B) = \dfrac{\mathbb{P}(A \cap B)}{\mathbb{P}(B)}$

Définition 3.1 *Une suite de v.a. $(X_1, \ldots, X_n)$ à valeurs dans l'ensemble E est une chaîne de Markov d'ordre 1 (modèle $M1$) si pour tout $1 < k \leq n$, $x_1, x_2, \ldots, x_k \in E$,*

$$\mathbb{P}(X_k = x_k | X_1 = x_1, \ldots, X_{k-1} = x_{k-1}) = \mathbb{P}(X_k = x_k | X_{k-1} = x_{k-1}).$$

Plus généralement, la suite $(X_1, \ldots, X_n)$ est une chaîne de Markov d'ordre ℓ (≥ 1) (Modèle $M\ell$) si $\forall k > \ell$,

$$\mathbb{P}(X_k = x_k | X_1 = x_1, \ldots, X_{k-1} = x_{k-1}) = \mathbb{P}(X_k = x_k | X_{k-\ell} = x_{k-\ell}, \ldots, X_{k-1} = x_{k-1})$$

♢

Notons qu'une suite indépendante constitue un modèle $M0$. On va maintenant étudier le modèle $M1$, qui constitue le modèle de référence.

4 Chaîne de Markov homogène d'ordre 1

Même si notre but ultime est précisément d'étudier des situations non homogènes, il est essentiel de comprendre d'abord le cas homogène.

Définition 4.1 *Une chaîne de Markov $(X_1, \ldots, X_n)$ à valeurs dans l'ensemble fini E est dite homogène si pour tous $x, y \in E$, la quantité*

$$\mathbb{P}(X_{k+1} = y | X_k = x)$$

ne dépend pas de $1 \leq k < n$

♢

Notons $P = (P_{xy})_{x,y \in E}$ la *matrice de transition* de la chaîne définie par

$$P_{xy} = \mathbb{P}(X_{k+1} = y | X_k = x), \;\; x, y \in E,$$

et $\mu_x = \mathbb{P}(X_1 = x), \;\; x \in E$ la *loi initiale*.

Lemme 4.2 *Soit F un autre ensemble fini, $f = E \times F \to E$, $\{Y_2, Y_3, \dots Y_n\}$ une suite de v.a. i.i.d. à valeurs dans F, indépendante de X_1, avec X_1 = v.a. à valeurs dans E, de loi μ. Alors la suite $\{X_1, \dots, X_n\}$ définie par la formule de récurrence*

$$X_k = f(X_{k-1}, Y_k), \ 2 \leq k \leq n$$

définit une chaîne de Markov de loi initiale μ et de matrice de transition

$$P_{xy} = \mathbb{P}(f(x, Y_2) = y).$$

Proposition 4.3 *La suite $(X_1, \dots, X_n)$ est une chaîne de Markov de loi initiale μ et de matrice de transition P ssi pour tout $1 < k \leq n$, la loi de $(X_1, \dots, X_k)$ est donnée par*

$$\mathbb{P}(X_1 = x_1, X_2 = x_2, \dots, X_k = x_k) = \mu_{x_1} P_{x_1 x_2} \times \cdots \times P_{x_{k-1} x_k}.$$

Preuve : La CN s'établit en utilisant $k - 1$ fois la formule

$$\mathbb{P}(A \cap B) = \mathbb{P}(A|B)\mathbb{P}(B),$$

et $k - 2$ fois la propriété de Markov.

Corollaire 4.4 *Si $(X_1, \dots, X_n)$ est une chaîne de Markov de loi initiale μ, et de matrice de transition P, alors pour $k \geq 1$, $\mathbb{P}(X_{1+k} = y | X_1 = x) = (P^k)_{xy}$ et la loi de X_k $(1 < k \leq n)$ est la probabilité*

$$\mu(k) = \mu P^k,$$

i.e. $\forall x \in E$,

$$\mu(k)_x = \sum_y \mu_y (P^k)_{yx}$$

5 Chaîne de Markov homogène irréductible

On veut maintenant énoncer l'équivalent de la loi des grands nombres, pour les chaînes de Markov. Il nous faut d'abord ajouter une condition.

Définition 5.1 *On dit que la chaîne de Markov $(X_1, \dots, X_n)$ de matrice de transition P est irréductible si $\forall x, y \in E$, $\exists k \geq 1$ tel que*

$$(P^k)_{xy} > 0,$$

i.e. si $\exists k$ tel que la chaîne passe avec probabilité non nulle de x à y en k itérations.

Théorème 5.2 *Soit* $(X_k,\ k \geq 1)$ *une chaîne de Markov à valeurs dans un ensemble fini* E*, de matrice de transition* P *irréductible. Alors il existe une unique probabilité* π *invariante par* P*, i.e. telle que* $\pi = \pi P$*, qui vérifie* $\pi_x > 0$*,* $\forall x \in E$*. Si* $(X_k,\ k \geq 1)$ *est une chaîne de Markov de loi initiale* π *et de matrice de transition* P*, alors* π *est la loi de* X_k *pour tout* $k \geq 1$*. Enfin on a le théorème ergodique (généralisation de la loi forte des grands nombres) : pour tout* $f : E \to \mathbb{R}$*,*

$$\frac{1}{n}\sum_{k=1}^{n} f(X_k) \to \sum_{x\in E} f(x)\pi_x \quad p.s.,$$

quelle que soit la loi de X_1*.*

Si on note à nouveau $\mu(k)$ la loi de X_k, il résulte du théorème ergodique, en prenant l'espérance, que pour toute probabilité $\mu(1)$ sur E,

$$\frac{1}{n}\sum_{k=1}^{n} \mu(k) \to \pi,$$

quand $n \to \infty$. Une question naturelle à se poser est de savoir si l'on a ou non $\mu(n) \to \pi$, quand $n \to \infty$. Ce n'est pas toujours vrai sous les hypothèses ci–dessus. Il faut en outre supposer que

Définition 5.3 *On dit qu'une chaîne de Markov de matrice de transition* P *irréductible est apériodique ssi pour un couple* (x, y) *dans* $E \times E$ *(et alors pour tous les couples, par l'irréductibilité), il existe* N *tel que* $(P^n)_{xy} > 0$*, pour tout* $n \geq N$*.*

Remarque 5.4 *On peut admettre que la chaîne des nucléotides successifs est irréductible et apériodique. Mais les chaînes périodiques ne sont pas que des curiosités mathématiques que l'on ne rencontrerait pas dans des modèles simples. Considérons la chaîne cachée (i.e. dont les valeurs ne sont pas données par la lecture des nucléotides), qui indique dans quelle plage (non codante, codante à l'endroit, codante à l'envers) se trouve le nucléotide que l'on lit. Codons par 0 l'état "non codant", 1 l'état "codant à l'endroit", 2 l'état "codant à l'envers". Quand on est à l'état 0, soit on y reste, soit on passe dans l'état 1, soit on passe dans l'état 2. Il est assez raisonnable de supposer que l'on ne passe pas directement de l'état 1 dans l'état 2 (et vice versa), sans repasser dans l'état 0. Considérons maintenant la chaîne qui décrit la suite des plages visitées, en "gommant" les longueurs de ces plages (donc en particulier la matrice de transition correspondante* P *a tous ses termes diagonaux nuls). Alors*

$$(P^n)_{00} = \begin{cases} 0, & \text{si } n \text{ est impair} \\ 1, & \text{si } n \text{ est pair,} \end{cases}$$

et la chaîne que nous venons de décrire est périodique.

Théorème 5.5 *Soit* $(X_k, k \geq 1)$ *une chaîne de Markov de matrice de transition* P *irréductible et apériodique. On désigne pour tout* $n \geq 1$ *par* $\mu(n)$ *la loi de probabilité de* X_n*. Alors* $\mu(n) \to \pi$ *quand* $n \to \infty$*.*

Donc dans le cas irréductible et apériodique on peut admettre que, en tout cas à partir d'un certain rang, la suite des X_k est identiquement distribuée, de loi commune l'unique probabilité invariante π de la chaîne.

6 Chaîne de Markov réversible

Soit $(X_1, \ldots, X_n)$ une chaîne de Markov de matrice de transition P. Posons $\hat{X}_k = X_{n+1-k}$. C'est un exercice facile sur les probabilités conditionnelles de montrer que la *suite retournée* $(\hat{X}_1, \ldots, \hat{X}_n) = (X_n, \ldots, X_1)$ est une chaîne de Markov. En général cette nouvelle chaîne de Markov n'est pas homogène. $(\hat{X}_1, \ldots, \hat{X}_n)$ est une chaîne homogène si $(X_1, \ldots, X_n)$ est initialisée avec sa probabilité invariante π. Dans ce cas, d'après la formule de Bayes, la matrice $\hat{P}$ de la chaîne retournée est donnée par

$$\hat{P}_{xy} = \frac{\pi_y P_{yx}}{\pi_x}.$$

On dit que la chaîne $(X_1, \ldots, X_n)$ est *réversible* si $\hat{P} = P$, ce qui est équivalent à ce que la *relation d'équilibre ponctuel* (en Anglais *detailed balance equation*) suivante soit satisfaite

$$\pi_x P_{xy} = \pi_y P_{yx}, \quad \forall x \neq y,$$

autrement dit si la quantité $\pi_x P_{xy}$ est symmétrique en x, y.

Remarque 6.1 *Etant donnée une chaîne irréductible de matrice de transition P, cette chaîne possède une unique probabilité invariante, qui forcément satisfait la relation $\pi P = \pi$. Le couple (π, P) satisfait ou non la relation d'équilibre ponctuel (i.e. toutes les chaînes irréductibles ne sont pas réversibles). Pour construire une chaîne irréductible et non réversible, il suffit de choisir une matrice P irréductible, telle pour un certain couple $x \neq y$, $P_{xy} = 0 < P_{yx}$.*

D'un autre coté, si P est une matrice de transition et π une probabilité sur E, telles que la relation d'équilibre ponctuel soit satisfaite, alors π est une probabilité invariante, comme on le vérifie en sommant la relation d'équilibre ponctuel par rapport à x (ou à y).

7 Statistique des chaînes de Markov homogènes $M1$

7.1 Estimation de la mesure invariante

On sait que $\frac{1}{n}\sum_1^n \mathbf{1}_{\{X_k=x\}} \to \pi_x$ p.s. quand $n \to \infty$. Donc

$$\frac{1}{n}\sum_1^n \mathbf{1}_{\{X_k=x\}}$$

est un estimateur de π_x, $x \in E$, au vu des observations $X_1, X_2, \ldots, X_n$.

7.2 Estimation de la matrice de transition

On va montrer que

$$\frac{\sum_1^n \mathbf{1}_{\{X_k=x,X_{k+1}=y\}}}{\sum_1^n \mathbf{1}_{\{X_k=x\}}} \to P_{x,y}, n \to \infty.$$

Notons tout d'abord que la fraction ci–dessus vaut

$$\frac{\frac{1}{n}\sum_1^n \mathbf{1}_{\{X_k=x,X_{k+1}=y\}}}{\frac{1}{n}\sum_1^n \mathbf{1}_{\{X_k=x\}}}.$$

Il suffit de considérer le numérateur. Plus précisément, il vaut

$$\frac{1}{n}\sum_{k \text{ pair}} \mathbf{1}_{\{X_k=x,X_{k+1}=y\}} + \frac{1}{n}\sum_{k \text{ impair}} \mathbf{1}_{\{X_k=x,X_{k+1}=y\}}$$

Considérons par exemple la quantité

$$\frac{2}{n}\sum_{k \text{ pair, } 1\le k\le n} \mathbf{1}_{\{X_k=x,X_{k+1}=y\}}$$

On remarque que la chaîne $\{(X_1,X_2),(X_3,X_4),\ (X_5,X_6),\ldots\}$ est une chaîne de Markov à valeurs dans $E\times E$, de matrice de transition de (x,y) à (x',y') donnée par $P_{y,x'}P_{x'y'}$, et de probabilité invariante $\tilde{\pi}_{xy}=\pi_x P_{xy}$.

Donc

$$\frac{2}{n}\sum_{k \text{ pair, } 1\le k\le n} \mathbf{1}_{\{X_k=x,X_{k+1}=y\}} \to \pi_x P_{xy}.$$

Finalement

$$\frac{\sum_1^n \mathbf{1}_{\{X_k=x,X_{h+1}=y\}}}{\sum_1^n \mathbf{1}_{\{X_k=x\}}} \to P_{xy}$$

p.s., quand $n\to\infty$.

7.3 Application aux ilôts cpg

Les données ci–dessous sont reprises de [2]. On estimé deux matrices de transition de Markov, d'une part sur des ilôts cpg, et d'autre part sur des séquences qui ne sont pas des ilôts cpg. Dans le premier cas, on obtient la matrice de transition estimée

$$P^+ = \begin{array}{c|cccc} & \texttt{a} & \texttt{c} & \texttt{g} & \texttt{t} \\ \texttt{a} & 0.180 & 0.274 & 0.426 & 0.120 \\ \texttt{c} & 0.171 & 0.368 & 0.274 & 0.188, \\ \texttt{g} & 0.161 & 0.339 & 0.375 & 0.125 \\ \texttt{t} & 0.079 & 0.355 & 0.384 & 0.182 \end{array}$$

et dans le second cas

$$P^- = \begin{array}{c} \\ \texttt{a} \\ \texttt{c} \\ \texttt{g} \\ \texttt{t} \end{array} \begin{array}{cccc} \texttt{a} & \texttt{c} & \texttt{g} & \texttt{t} \\ 0.300 & 0.205 & 0.285 & 0.210 \\ 0.322 & 0.298 & 0.078 & 0.302. \\ 0.248 & 0.246 & 0.298 & 0.208 \\ 0.177 & 0.239 & 0.292 & 0.292 \end{array}$$

Etant donnée une séquence $x = (x_1, \ldots, x_n)$ de nucléotides, on calcule un *score* qui est en fait un *log–rapport de vraisemblance* sous la forme

$$\begin{aligned} S(x) &= \log \frac{\mathbb{P}_{\text{modèle+}}(X = x)}{\mathbb{P}_{\text{modèle−}}(X = x)} \\ &= \sum_{i=2}^{n} \log \left(\frac{P^+_{x_{i-1}x_i}}{P^-_{x_{i-1}x_i}} \right) \\ &= \sum_{i=2}^{n} R_{x_{i-1}x_i}. \end{aligned}$$

On peut en fait normaliser ce score en le divisant par la longueur n de la séquence. La matrice des R est donnée par

$$R = \begin{array}{c} \\ \texttt{a} \\ \texttt{c} \\ \texttt{g} \\ \texttt{t} \end{array} \begin{array}{cccc} \texttt{a} & \texttt{c} & \texttt{g} & \texttt{t} \\ -0.740 & 0.419 & 0.580 & -0.803 \\ -0.913 & 0.302 & 1.812 & -0.685. \\ -0.624 & 0.461 & 0.331 & -0.730 \\ -1.169 & 0.573 & 0.393 & -0.679 \end{array}$$

Lorsque l'on compare les $S(x)$ pour diverses séquences dont on sait s'il s'agit ou non d'ilôt `cpg`, on voit que le score ci–dessus discrimine bien les ilôts `cpg` des autres types de séquence.

7.4 Recherche de gènes dans un génome prokaryote

On procède de façon analogue à ce que l'on vient de faire à la sous–section précédente. On a à notre disposition des *vrais gènes* et des *faux gènes*. On va en utiliser une partie pour l'*apprentissage du modèle*, et une autre pour tester le caractère discriminant ou non de tel *score*.

On utilise un premier sous–ensemble des *vrais gènes* pour estimer la matrice de transition d'une chîne de Markov. Notons P^+ l'estimé obtenu. Le modèle "–" est un modèle i.i.d., la probabilité jointe des 4 nucléotides étant donnée par les 4 fréquences, calculée à partir d'un pool contenant à la fois des *vrais gènes* et des *faux gènes*. Notons π l'estimé obtenu.

Maintenant si x est un *gène putatif*, on calcule son score

$$\begin{aligned} S(x) &= \log \frac{\mathbb{P}_{\text{modèle+}}(X = x)}{\mathbb{P}_{\text{modèle−}}(X = x)} \\ &= \sum_{i=2}^{n} \log \left(\frac{P^{+}_{x_{i-1}x_i}}{\pi_{x_i}} \right) \\ &= \sum_{i=2}^{n} R_{x_{i-1}x_i}. \end{aligned}$$

Il s'avère que cette statistique discrimine très mal les *vrais gènes* des *faux gènes*.

Par contre, si l'on prend comme modèle + un modèle de Markov $M1$ sur les *codons*, et que l'on procède comme ci–dessus, alors la nouvelle statistique $S(x)$ discrimine bien les *vrais gènes* des *faux gènes*.

8 Statistique des chaînes de Markov Mk

Pour simplifier, on va se contenter de décrire le modèle $M2$. Dans ce cas, ce qui remplace la matrice P est une matrice de transition de $E \times E$ dans E, qui donne la loi de probabilité de X_{k+1}, sachant le couple (X_{k-1}, X_k). Dans le cas $E = \{\mathtt{a}, \mathtt{c}, \mathtt{g}, \mathtt{t}\}$, on a donc une matrice de transition à 16 lignes (indexées par les dinucléotides $\{\mathtt{aa}, \mathtt{ac}, \ldots, \mathtt{gt}, \mathtt{tt}\}$) et 4 colonnes (indexées par $\{\mathtt{a}, \mathtt{c}, \mathtt{g}, \mathtt{t}\}$).

Remarque 8.1 *On peut aussi se ramener à un modèle $M1$ sur l'espace d'état $E \times E$, puisque si $(X_1, X_2, \ldots, X_n)$ est une chaîne de Markov d'ordre 2 à valeurs dans E, $((X_1, X_2), (X_2, X_3), \ldots, (X_{n-1}, X_n))$ est une chaîne de Markov d'ordre 1 à valeurs dans $E \times E$. On se ramène à une matrice de transition carrée, et on peut introduire la notion de probabilité invariante...*

On estime la probabilité de transition $P_{xy,z}$ à l'aide de la quantité

$$\frac{\sum_{k=1}^{n-2} \mathbf{1}_{\{X_k = x, X_{k+1} = y, X_{k+2} = z\}}}{\sum_{k=1}^{n-2} \mathbf{1}_{\{X_k = x, X_{k+1} = y\}}},$$

qui converge p.s. vers $P_{xy,z}$ quand $n \to \infty$. Remarquons que cette statistique inclut le décompte des trinucléotides, donc en particulier des codons, ce qui fait que les chaînes d'ordre 2 sont très utilisées pour modéliser les régions codantes de l'ADN.

9 Chaîne de Markov non homogène

Plusieurs types de non homogénéités sont pertinents dans la modélisation de l'ADN.

9.1 Chaîne de Markov phasée

Dans une "plage codante", on peut penser que la probabilité de transition n'est pas indépendante du site, mais périodique de période 3. Comme la notion de "chaîne de Markov périodique" désigne tout autre chose (à savoir une chaîne qui n'est pas "apériodique" au sens de la définition 5.3), nous utiliserons, à la suite de [4], la terminologie *chaîne de Markov phasée* pour désigner une chaîne de Markov $(X_n,\ 1 \leq n \leq N)$ telle que pour tous $x, y \in E$, l'application $n \to P(X_{n+1} = y | X_n = x)$ est périodique. Dans le cas qui nous occupe, on peut y compris songer à une chaîne de Markov d'ordre 2, telle que pour tout $y \in E$, la quantité $P(X_{n+1} = y | X_n = x, X_{n-1} = x')$ ne dépende pas de x, x' pour $n = 3k$, que de x pour $n = 3k + 1$ et dépende de x, x' pour $n = 3k + 2$, k entier. Cela veut dire en particulier que les codons successifs sont i.i.d. On pourraît aussi supposer que les codons successifs forment une chaîne de Markov d'ordre 1.

9.2 Chaîne de Markov localement homogène

Si l'on regarde plus globalement la séquence génomique, on s'attend à ce que la chaîne de Markov qui décrit celle–ci soit homogène dans la réunion des régions non codantes, dans celle des régions codantes à l'endroit, celle des régions codantes à l'envers, la réunion des introns, mais pas globalement homogène, et c'est d'ailleurs cette inhomogénéité qui doit nous permettre de réaliser l'annotation. Le principal problène est bien de détecter ce que l'on appelle les "ruptures de modèle".

Il existe une importante littérature statistique sur ces problèmes de rupture de modèle, mais il n'est pas clair que les algorithmes correspondant sont adaptables à la situation qui est la nôtre ici, où il est essentiel d'exploiter l'homogénéité de la chaîne sur la réunion des plages de même type (non codant, codant à l'endroit,...), et pas seulement sur chacune de ces plages prise isolément.

Cependant, Audic et Claverie ont mis au point un algorithme pour l'annotation des génomes prokaryotes, que nous allons maintenant décrire. On suppose que notre modèle (qui peut être $M0$, $M1$, $M2$,...) est décrit par un paramètre θ (qui est une probabilité sur E dans le cas $M0$, une probabilité de transition dans le cas $M1$, ...), lequel prend trois valeurs distinctes (toutes trois inconnues !) $(\theta_0, \theta_1, \theta_2)$, suivant que l'on est dans une plage non codante, codante à l'endroit ou codante à l'envers.

- •*Étape d'initialisation* On découpe la séquence en plages de longueur 100 (éventuellement, la dernière plage est de longueur > 100). On décide au hasard de placer chaque plage dans l'une des trois "boîtes" 0, 1, et 2. Sur la base de tous les X_n se trouvant dans la boîte 0, on estime une valeur du paramètre θ, soit $\theta_0^{(1)}$. On estime de même les valeurs $\theta_1^{(1)}$ et $\theta_2^{(1)}$.
- •*Étape de mise à jour* Supposons que nos trois "boîtes" 0, 1, et 2 contiennent chacune des plages distinctes de longueur ≥ 100, sur la base desquelles on a estimé les valeurs $\theta_0^{(n)}$, $\theta_1^{(n)}$ et $\theta_2^{(n)}$. On commence par vider ces boîtes, et on reprend la séquence complète $\{X_n,\ 1 \leq n \leq N\}$. On extrait la sous–suite $\{X_n,\ 1 \leq n \leq 100\}$. On estime le paramètre

θ sur la base de cette sous–suite, et on choisit laquelle des trois valeurs $\theta_0^{(n)}$, $\theta_1^{(n)}$ et $\theta_2^{(n)}$ est la plus proche de cette nouvelle valeur estimée. Puis on se pose le même problème avec la suite $\{X_n,\, 10 \leq n \leq 110\}$, avec la suite $\{X_n,\, 20 \leq n \leq 120\}$,.. jusqu'à ce que la valeur estimée devienne plus proche d'une autre des trois valeurs de l'étape précédente. Alors on revient en arrière de 50 nucléotides, et on place l'intervalle ainsi sélectionné depuis le début de la séquence dans la boîte 0, 1, ou 2, suivant le cas. On recommence, en prenant une plage de longueur 100, adjacente à l'intervalle que l'on vient de placer dans une des boîtes, et on répète les opérations précédentes. Lorsque l'on a épuisé la séquence, on se retrouve avec trois boîtes contenant chacune (du moins il faut l'espérer) des plages de longueur ≥ 100. On estime alors les trois nouvelles valeurs $\theta_0^{(n+1)}$, $\theta_1^{(n+1)}$ et $\theta_2^{(n+1)}$, sur la base du contenu des boîtes 0, 1, et 2 respectivement.

Si la séquence initiale est effectivement constituée de plages dont les compositions statistiques sont de trois types différents, l'algorithme converge rapidement, et quand on s'arrête, on a un découpage de la séquence initiale en sous–séquences de trois types différents. Il ne reste plus qu'à décider "qui est qui", ce qui requiert des connaissances a priori, acquises en observant des séquences qui ont déjà été annotées.

10 Chaîne de Markov cachée

Le point de vue Bayésien consiste à se donner une loi de probabilité a priori sur les paramètres inconnu θ_i, et leur évolution. Plus précisément on va maintenant se donner une nouvelle chaîne de Markov $(Y_1, \ldots, Y_N)$, dite "cachée" parce que non observée. Dans le cas des génomes prokaryotes, la chaîne (Y_n) prend par exemple ses valeurs dans l'ensemble à trois éléments $F = \{0, 1, 2\}$, et dans le cas eukaryote il faut différencier les états 1 et 2 entre les parties *intron* et *exon*. En réalité c'est encore un peu plus compliqué, car il faudrait prendre en compte les codons START et STOP, mais on verra cela un peu plus loin. L'avantage de cette approche est que l'on dispose d'algorithmes pour répondre aux questions que nous nous posons. On note F l'espace dans lequel la chaîne cachée prend ses valeurs, et $d = \mathrm{card}(F)$. Rappelons que $d \geq 3$.

Pour simplifier la présentation succincte de ces algorithmes, on va supposer que $(Y_1, \ldots, Y_N)$ est une chaîne de Markov (μ, P) à valeurs dans F, et que, connaissant les (Y_n), la suite des nucléotides $(X_1, \ldots, X_N)$ est indépendante, la loi de chaque X_n dépendant uniquement du Y_n correspondant, i.e. pour tout $1 \leq n \leq N$,

$$\begin{aligned}\mathbb{P}(X_1 = x_1, \ldots, X_n = x_n | Y_1 = y_1, \ldots, Y_n = y_n) &= \prod_{k=1}^{n} \mathbb{P}(X_k = x_k | Y_k = y_k) \\ &= \prod_{k=1}^{n} Q_{y_k x_k}.\end{aligned}$$

Le problème que l'on cherche à résoudre est le suivant : ayant observé la suite des nucléotides $(x_1, \ldots, x_N)$, quelle est la suite des états cachés $(y_1^*, \ldots, y_N^*)$ qui "explique le

mieux" ces observations ? Autrement dit, il s'agit de calculer la suite qui maximise la vraisemblance de la loi a posteriori sachant les observations, i.e.

$$(y_1^*, \ldots, y_N^*) = \text{argmax}_{y_1,\ldots,y_N} \mathbb{P}(Y_1 = y_1, \ldots, Y_N = y_N | X_1 = x_1, \ldots, X_N = x_N).$$

Notons que, dans ce modèle, on a comme paramètres inconnus le triplet (μ, P, Q). Pour résoudre le problème ci–dessus, on est obligé d'estimer d'abord les paramètres (mais nous discuterons ce problème à la fin). Si l'on admet que l'on connaît les paramètres, notre problème est résolu par :

10.1 L'algorithme de Viterbi

Définissons la suite de vecteurs ligne $\delta(n)$ par :

$$\delta_y(n) = \max_{y_1,y_2,\ldots,y_{n-1}} \mathbb{P}_\theta(Y_1 = y_1, \ldots, Y_{n-1} = y_{n-1}, Y_n = y, X_1 = x_1, \ldots, X_n = x_n)$$

$\delta_y(n)$ est en quelque sorte la plus forte probabilité d'une trajectoire des $\{Y_k, 1 \leq k \leq n-1\}$, qui se termine par $Y_n = y$, et correspondant à la suite des nucléotides observés $x_1, \ldots, x_n$. On a la formule de récurrence suivante entre les vecteurs $\delta(n)$:

$$\delta_y(n+1) = (\delta(n) * P)_y Q_{yx_{n+1}}$$

où l'opération $*$ qui à un vecteur ligne de dimension d et une matrice $d \times d$ associe un vecteur ligne de dimension d est définie comme suit :

$$(\delta * P)_y = \sup_{z \in F} \delta_z P_{zy}.$$

L'algorithme de Viterbi consiste à calculer les $\delta(n)$ de $n = 1$ à $n = N$, puis à retrouver la trajectoire optimale en cheminant pas à pas dans le sens "rétrograde" : connaissant y_n^*, on en déduit y_{n-1}^* par la formule :

$$y_{n-1}^* = \psi_{y_n^*}(n),$$

avec

$$\psi_y(n) = \text{argmax}_{z \in F} \delta_z(n-1) P_{zy}.$$

L'algorithme de Viterbi est décrit comme suit :

1. *Initialisation :*

$$\delta_y(1) = \mu_y Q_{yx_1}, \ \ y \in F;$$
$$\psi(1) = 0.$$

2. *Récurrence :* pour $1 < n \leq N$,

$$\delta_y(n) = (\delta(n-1) * P)_y Q_{yx_n},$$
$$\psi_y(n) = \text{argmax}_{z \in F} \delta_z(n-1) P_{zy}, \ \ y \in F.$$

3. *Etape finale :*

$$\delta^* = \max_{y \in F} \delta_y(N)$$
$$y_N^* = \arg\max_{y \in F} \delta_y(N).$$

4. *Récurrence rétrograde*

$$y_n^* = \psi_{y_{n+1}^*}(n+1), \ 1 \leq n < N.$$

10.2 Estimation des paramètres

Il y a deux stratégies possibles. L'une consiste à estimer les paramètres sur une séquence d'apprentissage déjà annotée. Dans ce cas, on estime les paramètres d'un modèle où toute la suite $\{(X_n, Y_n),\ 1 \leq n \leq N\}$ est observée. On utilise les algorithmes d'estimation bien connus que nous avons présentés dans les sections précédentes.

L'autre stratégie consiste à estimer les paramètres sur la base des seules observations de la suite des nucléotides. L'avantage est de faire l'estimation à partir du génome étudié, et non pas à partir d'un génome différent. L'inconvénient est bien sûr que l'on estime un modèle avec des observations très partielles. Il existe cependant des algorithmes maintenant classiques (l'algorithme EM, et sa variante SEM), qui permettent de résoudre ce problème.

Nous allons présenter de façon très sommaire l'algorithme SEM, qui est le plus utile dans les situations que nous décrirons plus loin. Pour chaque valeur du paramètre inconnu θ, on considère la loi conditonnelle des états cachés, sachant la suite des nucléotides, notée

$$\mathbb{P}_\theta \left(Y_1 = y_1, \ldots, Y_N = y_N | X_1 = x_1, \ldots, X_N = x_N \right),$$

ou plutôt

$$\mathbb{P}_\theta \left(Y_1^N = y_1^N | X_1^N = x_1^N \right).$$

L'algorithme SEM est un algorithme itératif, que l'on initie avec une valeur θ_0. L'itération qui remplace θ_n par θ_{n+1} se décompose en deux étapes comme suit :

- *Simulation* On tire au hasard une réalisation de la suite aléatoire Y_1^N, suivant la loi $\mathbb{P}_{\theta_n}(Y_1^N = \cdot | X_1^N = x_1^N)$. Notons $y_1^N(n)$ la suite obtenue ainsi.
- *Maximisation* On choisit

$$\theta_{n+1} = \text{argmax}_\theta \mathbb{P}_\theta \left(Y_1^N = y_1^N(n), X_1^N = x_1^N \right).$$

Dans l'algorithme EM, l'étape de simulation est remplacée par le calcul de $\mathbb{E}_{\theta_n}(Y_1^N | X_1^N = x_1^N)$.

11 Modèle semi–markovien caché

11.1 Les limites du modèle de Markov caché

Un conséquence de la propriété de Markov est que les temps de séjour d'une chaîne de Markov dans chacun des états visités suivent des lois géométriques. Le modèle de la

section 10 implique donc que les longueurs des plages codantes et non codantes d'un génome prokaryote suivent des lois géométriques. Or cette hypothèse ne cadre pas avec les données dont on dispose. Il y a là un premier argument pour envisager un modèle plus général, mais on va voir maintenant un argument encore plus convainquant pour abandonner le modèle de Markov caché.

Examinons plus précisément notre problème, en nous limitant à nouveau pour simplifier au génome prokaryote. Il est bien sûr essentiel de prendre en compte l'information contenue dans les codons START et STOP. Si l'on renonce à un modèle phasé, on est obligé d'introduire 3 états START, 3 états codants et 3 états STOP, chacun correspondant à une des trois phases de lecture, le tout doit être multiplié par deux pour tenir compte du brin complémentaire. On ajoute un état non codant. Cela fait en tout 19 états. Certes, la plupart des termes de la matrice de transition sont nuls, mais cela fait quand même beaucoup d'états, et dans le cas eukaryote la situation est bien pire. On peut réduire ce nombre avec un modèle phasé, mais on récupère la même complexité en multipliant par trois le nombre de matrices de transition à estimer. Enfin on pourrait penser travailler sur la suite des codons plutôt que sur celle des nucléotides, mais ceci ne serait pas valable pour les parties non codantes.

On va voir ci–dessous que le modèle semi–markovien permet de réduire le nombre d'états à trois dans le cas prokaryote, en outre qu'il permet de choisir une loi plus réaliste que la loi géométrique pour la longueur des plages codantes.

11.2 Qu'est–ce qu'une chaîne semi–markovienne ?

La réponse dépend des auteurs. Je vais donner ma définition. Comme son nom l'indique, une chaîne semi–markovienne est "un peu moins markovienne" (i.e. oublie un peu moins son passé) qu'une chaîne de Markov. Étant donnée une suite aléatoire $(X_1, \ldots, X_N)$, et $1 < n < N$, on définit pour chaque $1 < n < N$ la v. a. η_n de la façon suivante

$$\eta_n = \sup\{k \geq 0,\ X_{n-k} = X_{n-k+1} = \cdots = X_n\}.$$

Dans l'application qui nous intéresse, c'est le nombre de sites à gauche du site n, qui sont dans la même plage que celui–ci. Bien entendu, si l'on connaît la réalisation de la suite $(X_1, \ldots, X_n)$, on connaît la valeur de η_n. On notera $\varphi_n(x_1, \ldots, x_n)$ la valeur de η_n quand $(X_1, \ldots, X_n) = (x_1, \ldots, x_n)$. C'est à dire

$$\varphi_n(x_1, \ldots, x_n) = \sup\{k; x_{n-k} = \cdots = x_n\},$$

d'où $\eta_n = \varphi_n(X_1, \ldots, X_n)$.

Définition 11.1 *Une suite aléatoire $(X_1, \ldots, X_N)$ à valeurs dans E est une chaîne semi–markovienne ssi pour tout $1 < n \leq N$, pour tout $(x_1, \ldots, x_{n-1}, x, y) \in E^{n+1}$,*

$$\begin{aligned}&\mathbb{P}(X_{n+1} = y | X_1 = x_1, \ldots, X_{n-1} = x_{n-1}, X_n = x)\\ &\quad = \mathbb{P}(X_{n+1} = y | X_n = x, \eta_n = \varphi_n(x_1, \ldots, x_{n-1}, x)).\end{aligned}$$

Le fait que l'état suivant d'une chaîne semi–markovienne dépende non seulement de l'état courant, mais de la longueur de la "visite" dans cet état jusqu'au site considéré fait que les lois des temps de séjour dans les divers états sont complètement arbitraires.

Plus précisément, une façon "générique" de préciser la loi d'une chaîne semi–markovienne (et aussi d'indiquer comment la simuler) est la suivante.

- D'une part on associe à chaque point $x \in E$ une loi de probabilité $(d_x(n),\ n \in \mathbb{N}\backslash\{0\})$ sur les entiers privés de 0, qui précise les lois des longueurs des "plages" sur lesquelles la chaîne est constante.
- D'autre part on se donne une matrice de transition P sur $E \times E$ d'une chaîne de Markov, dont tous les termes diagonaux sont nuls. Cette matrice décrit suivant quelle loi la chaîne change d'état, quand elle en change.

Voyons comment simuler une chaîne semi–markovienne dont la loi est caractérisée par les données : pour tout $x \in E$, d_x désigne la loi du temps de séjour à l'état x, et $P_{x\cdot}$ la loi du prochain état visité après l'état x. Si x est le point de départ $(X_1 = x)$, on tire une variable alétoire T_1 à valeurs dans $\mathbb{N}^*$ de loi d_x. Notons n la valeur simulée. Alors $X_1 = X_2 = X_3 = \cdot = X_n = x$. On tire une v.a. Z_1 de loi $P_{x\cdot}$ sur $E\backslash\{x\}$. Supposons que le résultat du tirage soit $Z_1 = y$. Alors $X_{n+1} = y$, et on recommence en tirant une v. a. T_2 de loi d_y, et une v.a. Z_2 de loi $P_{y\cdot}$, et ainsi de suite. Tous les tirages successifs sont bien entendu indépendants les uns des autres.

11.3 Le modèle semi–markovien caché

Encore une fois, limitons–nous pour simplifier au cas prokaryote. On considère 3 états cachés, l'état 0 pour *non codant*, l'état 1 pour *codant à l'endroit*, l'état 2 pour *codant à l'envers*. L'état 0 est un état *markovien* (on verra ci–dessous la raison de cette restriction), ce qui veut dire que la loi des longueurs des plages non codantes est une loi géométrique de paramètre q (à estimer). Les états 1 et 2 sont dits *semi–markoviens*. On choisira comme loi des longueurs des plages codantes à l'endroit et à l'envers l'image par l'application $x \to 3x$ d'une loi binomiale négative de paramètres $m \in \mathbb{N}^*$ et $0 < p < 1$ (i. e. cette loi décrit le nombre de codons plutôt que le nombre de nucléotides).

Définition 11.2 *On dit que la v.a. T suit la loi binomiale négative de paramètres m et p si T est le nombre de jets de pile ou face nécessaires pour obtenir exactement m piles, p désignant la probabilité d'obtenir pile à chaque coup. Soit*

$$\mathbb{P}(T = k) = C_{m-1}^{k-1}(1-p)^{k-m}p^m,$$

qui vaut la probabilité d'obtenir $m-1$ piles au cours des $k-1$ premiers coups, multipliée par la probabilité d'obtenir pile au k–ième coup.

La logique voudrait que l'on choisisse comme valeur du paramètre m le plus petit nombre d'acides aminés que contient un gène, plus deux (pour les codons START + STOP). Malheureusement, ce nombre minimal peut être extrêmement réduit dans des cas tout à fait exceptionnels, alors qu'il est de l'ordre de la dizaine hormis ces cas tout à fait exceptionnels.

Un choix raisonnable semble être $m = 10$, mais ce choix doit être critiqué–validé par le Biologiste (et/ou confronté à la séquence étudiée).

Quant au paramètre p, il doit être estimé.

Quant à la loi de probabilité qui régit les changements d'état, elle est définie comme suit. On admet que tout plage codante (à l'endroit comme à l'envers) est suivie d'une plage non codante. Donc $P_{10} = P_{20} = 1$. En outre, $P_{01} + P_{02} = 1$, et on peut soit supposer que $P_{01} = 1/2$, soit chercher à estimer cette quantité.

Discutons maintenant de la loi des nucléotides, sachant l'état caché. On peut admettre que dans les plages non codantes, les nucléotides sont i. i. d., la loi commune étant à estimer. Dans une plage codante, on suppose par exemple que les codons sont mutuellement indépendants, le premier prenant ses valeurs dans l'ensemble des codons START possibles, le dernier dans l'ensemble des codons STOP possibles, les autres codons étant i. i. d., à valeurs dans les codons possibles qui codent pour un acide aminé (en particulier ces codons ne peuvent pas prendre comme valeur un des codons STOP). La description de la loi des codons d'une plage codante à l'envers se déduit aisément de celle que nous venons de donner pour les plages codantes à l'endroit.

11.4 Algorithme de Viterbi dans le cas semi–markovien caché

Nous allons maintenant décrire comment l'algorithme de Viterbi s'écrit dans notre nouvelle situation (qui est en fait une situation mixte markov caché – semi–markov caché).

Comme pour les chaînes de Markov cachées, l'idée est de calculer des quantités $\delta_y(n)$, pour tous les états cachés y, et $1 \leq n \leq N$. Pour $y = 0$, on pose comme précédemment

$$\delta_y(n) = \max_{y_1,y_2,\ldots,y_{n-1}} \mathbb{P}_\theta(Y_1 = y_1, \ldots, Y_{n-1} = y_{n-1}, Y_n = y, X_1 = x_1, \ldots, X_n = x_n).$$

Pour $y = 1$ (et de même pour $y = 2$), on pose

$$\delta_y(n) = \max_{y_1,\ldots,y_{n-1}} P_\theta(Y_1 = y_1, ..., Y_{n-1} = y_{n-1}, Y_n = y, Y_{n+1} \neq y, X_1 = x_1, ..., X_n = x_n)$$

La formule de récurrence est la suivante :

$$\delta_y(n) = \max \Big\{ P_\theta(X_1^n = x_1^n | Y_1^n = y, Y_{n+1} \neq y) d_y(n) \mu_y,$$

$$\max_{1\leq k\leq n-1} \left[P_\theta(X_{n-k+1}^n = x_{n-k+1}^n | Y_{n-k} \neq y, Y_{n-k+1}^n = y, Y_{n+1} \neq y) \max_{z\neq y} [\delta_z(n-k) P_{zy}] d_y(k) \right] \Big\}$$

Remarquons que le fait qu'à chaque étape on ait à calculer un max sur $1 \leq k \leq n$ rend l'algorithme a priori quadratique en N, ce qui est une mauvaise nouvelle. Cependant on peut limiter la portée de ce max, en arguant du fait qu'une région codante se termine au premier codon STOP rencontré. Cette remarque est encore vraie pour un exon dans le cas eukaryote : celui–ci se termine au plus loin lors de la rencontre du premier codon STOP (soit que ce codon marque effectivement la fin du gène, soit qu'il soit situé dans un intron, ou encore dans la mauvaise phase de lecture, mais au delà du premier intron rencontré). La même remarque ne s'applique pas aux régions non codantes, d'où le choix de prendre l'état 0 markovien.

11.5 Application : recherche de gènes dans un génome prokariote

11.5.1 La loi a priori des Y

On a 3 états, codant $+ = 1$, codant $- = 2$, non codant $= 0$.

- • La loi de Y_1 est une loi $\mu = (\mu_0, \mu_1, \mu_2)$.
- • Quand $Y_n = 0$, on tire Y_{n+1} suivant la loi $p = (p_0, p_1, p_2)$ (avec $p_0 > 0$, $p_1 > 0$, $p_2 > 0$, $p_0 + p_1 + p_2 = 1$).
- • Si $Y_{n-1} = 0$ et $Y_n = 1$ (description analogue en remplaçant 1 par 2), on tire la longueur de la région codante suivant une loi qui ne charge que les multiples entiers de 3. Juste après une plage codante, on est dans l'état non codant (état 0).

11.5.2 La loi conditionnelle des X, sachant les Y

- • Sachant que $Y_n = 0$, X_n est indépendant de tout le reste, et suit la loi $q = (q_a, q_c, q_g, q_t)$ (avec $q_a > 0$, $q_c > 0$, $q_g > 0$ et $q_t > 0$, $q_a + q_c + q_g + q_t = 1$).
- • Si $Y_n = 1$ (description analogue si $Y_n = 2$), tout dépend de où on en est dans la plage codante
  1. Le 1er codon suit une loi à déterminer portée par les codons START.
  2. Les codons intermédiaires sont tirés suivant une loi qui charge les divers codons qui codent pour des Acides Aminés, les codons STOP possibles étant exclus.
  3. Le dernier codon est tiré suivant une loi à déterminer portée par les codons STOP.

11.5.3 Le calcul des δ

Pour $y = 1$ et $y = 2$, on définit

$$\delta_y(n) = \max_{y_1,\dots,y_{n-1}} \mathbb{P}(Y_1^{n-1} = y_1^{n-1}, Y_n = y, Y_{n+1} = 0, X_n^1 = x_n^1).$$

Pour $y = 0$, on définit

$$\delta_0(n) = \max_{y_1,\dots,y_{n-1}} \mathbb{P}(Y_1^{n-1} = y_1^{n-1}, Y_n = 0, X_n^1 = x_n^1).$$

• Pour $n < 3m$, $\delta_1(n) = \delta_2(n) = 0$. En outre si $Y_n = 0$, nécessairement toute la suite $Y_1, \dots, Y_n$ est constante et égale à 0. Donc il n'y a pas de maximum à prendre et

$$\delta_0(n) = \mathbb{P}(Y_1^n = 0, X_1^n = x_1^n) = \mu_0 p_0^{n-1} \prod_{k=1}^{n} q_{x_k}.$$

• Pour $n \geq 3m$,

- Voyons d'abord la formule de récurrence dans le cas $y = 0$. Si $n > 3m$, la collection des $y_1, \dots, y_{n-1}$ sur laquelle on prend le maximum se subdivise en trois classes, suivant la valeur de y_{n-1}. On a

$$\delta_0(n) = \max[\delta_0(n-1)p_0, \delta_1(n-1), \delta_2(n-1)]q_{x_n}.$$

– Considérons maintenant le cas $y = 1$ (on traiterait de la même façon le cas $y = 2$).

1. soit $Y_1^n = 1$ et $Y_{n+1} \neq 1$ (ce qui suppose que n est un multiple entier de 3). Cela s'écrit :
$$\mathbb{P}(X_1^n = x_1^n \mid Y_1^n = 1, Y_{n+1} \neq 1) d_1(n)\mu_1$$

2. soit la séquence cachée est en 1 sur la plage $[n-k+1, n]$, alors (avec $y_{n-k} \neq 1$, donc nécessairement $y_{n-k} = 0$, ce qui impose que k soit un multiple entier de 3 ; et la quantité qui suit est nulle sauf si à la fois $(x_{n-k}, x_{n-k+1}, x_{n-k+2})$ est un codon START et (x_{n-2}, x_{n-1}, x_n) est un codon STOP)
$$\max_{y_1,\ldots,y_{n-k-1}} \mathbb{P}(Y_1^{n-k} = y_1^{n-k}, Y_{n-k+1}^n = 1, Y_{n+1} = 0, X_1^n = x_1^n)$$
$$= \delta_0(n-k)p_1\mathbb{P}(X_{n-k+1}^n = x_{n-k+1}^n, Y_{n-k+1}^n = 1, Y_{n+1} = 0 | Y_{n-k+1} = 1, Y_{n-k} = 0)$$
$$= \delta_0(n-k)p_1\mathbb{P}(X_{n-k+1}^n = x_{n-k+1}^n | Y_{n-k} = 0, Y_{n-k+1}^n = 1, Y_{n+1} = 0) d_1(k).$$
La probabilité conditionnelle ci–dessus se factorise en
$$\prod_{j=0}^{k/3} \mathbb{P}(X_{n-k+3j+1}^{n-k+3j+3} = x_{n-k+3j+1}^{n-k+3j+3} | Y_{n-k} = 0, Y_{n-k+1}^n = 1, Y_{n+1} \neq 1)$$
$$= \mathbb{P}_{\text{START}}(x_{n-k+1}^{n-k+3}) \prod_{j=1}^{k/3-1} \mathbb{P}_{\text{CODANT}}(x_{n-k+3j+1}^{n-k+3j+3}) \mathbb{P}_{\text{STOP}}(x_{n-2}^n)$$
$$:= \mathbb{P}_{\text{C+}}(x_{n-k+1}^n).$$
Finalement, avec la convention $\delta_0(0) = \mu_1/p_1$,
$$\delta_1(n) = \max_{k \text{ multiple entier de } 3; 3m \leq k \leq n} \delta_0(n-k)p_1\mathbb{P}_{\text{C+}}(x_{n-k+1}^n) d_1(k).$$

12 Alignement de deux séquences

Considérons deux séquneces de nucléotides (ou d'acides aminés)
$$x_1^n = (x_1, \ldots, x_n)$$
$$y_1^m = (y_1, \ldots, y_m).$$

Considérons pour fixer les idées le cas de séquences de nucléotides, et considérons les deux séquences

a a c g g t t c c c a g t t

a c g t t t c c a g t c.

On a bien envie de les aligner comme suit

```
a a c g g t t c c c a g t t
a − c g t t t c c − a g t c.
```

Pour faire cela, on créé des *trous* (*gap* en Anglais) (dans l'une des deux séquences (on aurait pu vouloir créer des *trous* dans chacune des deux séquences). Notons que du moment que $n \neq m$, on aura un minimum de $|n-m|$ *trous*.

On peut associer à chaque alignement des deux séquences x_1^n et y_1^m un *score*, qui sera d'autant plus élevé que l'alignement est *bon*. Supposons que la longueur commune des séquences alignées (en comptant les *trous*) est T. Nécessairement, $T \geq \sup(n,m)$. Pour $1 \leq i \leq T$, si en position i on trouve le nucléotide a_i dans la première séquence et le nucléotide b_i dans la seconde séquence (on n'a pas nécessairement $a_i = x_i$, sauf s'il n'y a pas de *trou* dans la première séquence à gauche de la position i ; idem pour la seconde séquence), la position i contribue $s(a_i, b_i)$ au score global de l'alignement. Si à la position i on a l'ouverture d'un *trou* dans l'une des deux séquences (au sens où il y a un trou dans la première (resp. la seconde) séquence à la position i, et pas de trou dans la même séquence à la position $i-1$), cette position contribue $-d$ au score global, et si on a le prolongement d'un *trou*, i contribue $-e$ au score global. Finalement, le score global d'un alignement de longueur T vaut

$$\sum_{1\leq i\leq T, \text{pas de trou en } i} s(a_i, b_i) - \sum_{\text{trous 1ère séquence}} (d + (\ell_j - 1)e) - \sum_{\text{trous 2ème séquence}} (d + (\ell'_k - 1)e),$$

si ℓ_j (resp. ℓ'_k) désigne la longueur du j–ième *trou* de la première séquence (resp. du k–ième *trou* de la seconde séquence). s est une application de $\{\texttt{acgt}\}^2$ dans $\mathbb{R}$, qui est maximal sur la diagonale. Dans le cas de séquences d'acides aminés, on remplace bien sûr $\{\texttt{acgt}\}$ par l'ensemble des 20 acides aminés. Dans les deux cas, s sera choisi de telle sorte que, pour $a \neq b$, $s(a,b)$ sera d'autant plus grand qu'une mutation entre a et b est plus probable.

12.1 Algorithme de Needleman–Wunsch

La recherche de l'alignement optimal peut se faire à l'aide d'un algorithme de programation dynamique. Si l'on définit $M(i,j)$ comme le meilleur score possible parmi tous les débuits d'alignements qui aboutissent à aligner x_i et y_j, $I_x(i,j)$ le meilleur score parmi tous les débuts d'alignements qui aboutissent à aligner x_i avec un *trou* dans la seconde séquence, y_j étant le dernier nucléotide de la seconde séquence à gauche du *trou*, et $I_y(i,j)$ le meilleur score parmi tous les débuts d'alignements qui aboutissent à aligner y_j avec un *trou* dans la première séquence, x_i étant le dernier nucléotide de la première séquence à gauche du *trou*,

alors on a les formules de récurrence suivantes :

$$M(i,j) = \max \begin{cases} M(i-1,j-1) + s(x_i, y_j), \\ I_x(i-1,j-1) + s(x_i, y_j), \\ I_y(i-1,j-1) + s(x_i, y_j); \end{cases}$$

$$I_x(i,j) = \max \begin{cases} M(i-1,j) - d, \\ I_x(i-1,j) - e; \end{cases}$$

$$I_y(i,j) = \max \begin{cases} M(i,j-1) - d, \\ I_y(i,j-1) - e. \end{cases}$$

On a exclu dans ces formules qu'un trou dans une séquence puisse être immédiatement suivi par un trou dans l'autre, ce qui est vrai pour l'alignement optimal si $-d - e < \inf_{a,b} s(a,b)$. Dans le cas $e = d$, la triple récurrence ci–dessus se réduit à l'unique récurrence

$$F(i,j) = \max \begin{cases} F(i-1,j-1) + s(x_i, y_j), \\ F(i-1,j) - d, \\ F(i,j-1) - d. \end{cases}$$

Il existe de nombreuses variantes de cet algorithme. En particulier, on peut s'intéresser à la recherche du meilleur alignement local (et non nécessairement global). Celui–ci s'obtient dans le cas $d = e$ en modifiant la récurrence comme suit

$$F(i,j) = \max \begin{cases} 0, \\ F(i-1,j-1) + s(x_i, y_j), \\ F(i-1,j) - d, \\ F(i,j-1) - d. \end{cases}$$

Le fait que le maximum soit atteint par 0 indique le début d'un alignement local. Notons que dans ce cas, $F(\cdot, 0) = F(0, \cdot) \equiv 0$. L'algorithme d'alignement local que nous venons de décrire s'appelle l'algorithme de Smith–Waterman.

12.2 Modélisation par chaîne de Markov cachée ("Pair–HMMs")

Nous allons maintenant replacer la recherche d'un alignement optimal entre deux séquences dans le cadre des chaînes de Markov cachées. Ici la chaîne cachée sera notée $Z_1, \ldots, Z_T$. Notons que T n'est pas donné à priori, il est aléatoire. Cette chaîne prend ses valeurs dans un espace à quatre états, que l'on désignera par $\{A, I, S, F\}$, A pour *aligner*, I pour *insérer* un *trou* dans la première séquence, S pour *supprimer*, soit insérer un *trou* dans la seconde séquence, F pour *fin* qui est un état absorbant de la chaîne.

Il faut maintenant préciser d'une part la loi a priori de la chaîne $Z_1, \ldots, Z_T$, et d'autre part la loi conditionnelle de la double suite

$$\left(\begin{pmatrix} \hat{X}_1 \\ \hat{Y}_1 \end{pmatrix}, \ldots, \begin{pmatrix} \hat{X}_{T-1} \\ \hat{Y}_{T-1} \end{pmatrix} \right)$$

à valeurs dans l'ensemble $\{\mathtt{a}, \mathtt{c}, \mathtt{g}, \mathtt{t}, -\}$, sachant la suite $Z_1, \ldots, Z_T$ (la valeur de T, la longueur de la suite, faisant partie des inconnues qui sont précisées par l'alignement).

On va considérer $\{Z_t,\ t \geq 1\}$ comme une chaîne de Markov à valeurs dans l'espace $\{A, I, S, F\}$, et $T := \inf\{t \geq 1,\ Z_t = F\}$. La matrice de transition de cette chaîne est la matrice

$$\begin{pmatrix} 1-2\delta-\tau & \delta & \delta & \tau \\ 1-\varepsilon-\tau & \varepsilon & 0 & \tau \\ 1-\varepsilon-\tau & 0 & \varepsilon & \tau \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

On choisit comme loi de Z_1 la probabilité

$$(1-\tau)^{-1}(1-2\delta-\tau \quad \delta \quad \delta \quad 0).$$

La loi de $(Z_1, \ldots, Z_T)$ est complètement spécifiée.

Notons que la loi de T est donnée par

$$\mathbb{P}(T = t) = (1-\tau)^{t-2}\tau,\ t \geq 2.$$

On peut préciser la loi des transitions, conditionnées au fait qu'elles n'aboutissent pas en F. Si l'on pose

$$P_{GH} := \mathbb{P}(Z_{t+1} = H | Z_t = G, T > t+1),$$

$G, H \in \{A, I, S\}$, on a

$$P = \begin{pmatrix} \frac{1-2\delta-\tau}{1-\tau} & \frac{\delta}{1-\tau} & \frac{\delta}{1-\tau} \\ \frac{1-\varepsilon-\tau}{1-\tau} & \frac{\varepsilon}{1-\tau} & 0 \\ \frac{1-\varepsilon-\tau}{1-\tau} & 0 & \frac{\varepsilon}{1-\tau} \end{pmatrix}.$$

Précisons maintenant la loi conditionnelle de la suite des $\hat{X}$ et des $\hat{Y}$, sachant $(Z_1, \ldots, Z_T)$. A nouveau, on écrit les choses dans le cas de séquences de nucléotides. La transposition aux séquences d'acides aminés ne pose pas de difficulté. On se donne d'une part une probabilité $\{p_{a,b},\ (a,b) \in \{\mathtt{acgt}\}^2\}$, et d'autre part une probabilité $\{q_a,\ a \in \{\mathtt{acgt}\}\}$. Conditonnellement en $(Z_1, \ldots, Z_T)$, les v. a. $(\hat{X}_t, \hat{Y}_t)$, $1 \leq t < T$ sont indépendantes. Sachant que $1 \leq t < T$ et $Z_t = A$, $(\hat{X}_t, \hat{Y}_t)$ est distribué suivant la loi p. Sachant que $1 \leq t < T$ et $Z_t = I$, $\hat{X}_t = -$ et $\hat{Y}_t$ suit la loi q. Sachant que $1 \leq t < T$ et $Z_t = S$, $\hat{X}_t$ suit la loi q et $\hat{Y}_t = -$. $(\hat{X}_t, \hat{Y}_t)$ n'est pas défini pour $t \geq T$. Autrement dit,

$$\mathbb{P}\left(\begin{pmatrix} \hat{X}_1^{T-1} \\ \hat{Y}_1^{T-1} \end{pmatrix} = \begin{pmatrix} x_1^{t-1} \\ y_1^{t-1} \end{pmatrix} \middle| Z_1^T = z_1^t\right) = \prod_{1 \leq i < t; z_i = A} p_{x_i y_i} \prod_{1 \leq j < t; z_j = I} \mathbf{1}_{\{x_j = -\}} q_{y_j} \prod_{1 \leq k < t; z_k = S} q_{x_k} \mathbf{1}_{\{y_k = -\}}$$

On peut maintenant calculer la probabilité

$$\begin{aligned}\mathbb{P}\left(\begin{pmatrix}\hat{X}_1^{T-1}\\ \hat{Y}_1^{T-1}\end{pmatrix}=\begin{pmatrix}x_1^{t-1}\\ y_1^{t-1}\end{pmatrix}, Z_1^T=z_1^t\right) &= \mu_{z_1}(1-\tau)^{t-2}\tau \prod_{1\le i\le nx_i\neq -,y_i\neq -} p_{x_iy_i} \prod_{1\le j\le nx_j=-} q_{y_j} \prod_{1\le k\le ny_k=-} q_{x_k}\\ &\quad\times\left(\frac{1-2\delta-\tau}{1-\tau}\right)^{\ell_1}\left(\frac{\delta}{1-\tau}\right)^{\ell_2}\left(\frac{\varepsilon}{1-\tau}\right)^{\ell_3}\left(\frac{1-\varepsilon-\tau}{1-\tau}\right)^{\ell_4}\\ &= \mu_{z_1}\tau \prod_{1\le i\le n;x_i\neq -,y_i\neq -} p_{x_iy_i} \prod_{1\le j\le n;x_j=-} q_{y_j} \prod_{1\le k\le n;y_k=-} q_{x_k}\\ &\quad\times(1-2\delta-\tau)^{\ell_1}\delta^{\ell_2}\varepsilon^{\ell_3}(1-\varepsilon-\tau)^{\ell_4},\end{aligned}$$

où ℓ_1 est le nombre de transitions de la chaîne $\{Z_\cdot\}$ de A vers A, ℓ_2 le nombre de transitions de A vers I ou S (ouverture d'un *trou*), ℓ_3 le nombre de transitions de I ou S vers lui–même (prolongement d'un *trou*), et ℓ_4 le nombre de transition de I ou S vers A (fermeture d'un *trou*).

Notons que l'observation est constituée des réalisations des deux suites $(X_1,\dots,X_N)$ et $(Y_1,\dots,Y_M)$ obtenues à partir de la double suite

$$\left(\begin{pmatrix}\hat{X}_1\\ \hat{Y}_1\end{pmatrix},\dots\begin{pmatrix}\hat{X}_{T-1}\\ \hat{Y}_{T-1}\end{pmatrix}\right),$$

en supprimant les *trous*.

Il nous faut encore préciser la quantité

$$\mathbb{P}(X_1^N=x_1^n, Y_1^M=y_1^m, Z_1^T=z_1^t).$$

En fait on va plutôt calculer la quantité suivante, donc la maximisation par rapport à z_t^1 produira le même résultat :

$$\mathbb{P}^*(X_1^N=x_1^n, Y_1^M=y_1^m, Z_1^T=z_1^t)=\frac{\mathbb{P}(X_1^N=x_1^n, Y_1^M=y_1^m, Z_1^T=z_1^t)}{\prod_{i=1}^n q_{x_i}\prod_{j=1}^m q_{y_j}}.$$

On a

$$\mathbb{P}^*(X_1^N=x_1^n, Y_1^M=y_1^m, Z_1^T=z_1^t)=\frac{\tau}{1-\tau}\prod_{i=1}^{t-1}v(i),$$

avec

$$v(i)=\begin{cases}(1-2\delta-\tau)\frac{p_{x_{k(i)}y_{\ell(i)}}}{q_{x_{k(i)}}q_{y_{\ell(i)}}}, & \text{si } z_i=A \text{ et } i=1 \text{ ou } z_{i-1}=A,\\ (1-\varepsilon-\tau)\frac{p_{x_{k(i)}y_{\ell(i)}}}{q_{x_{k(i)}}q_{y_{\ell(i)}}}, & \text{si } z_i=A,\ i>1 \text{ et } z_{i-1}=I \text{ ou } S,\\ \delta\frac{q_{y_{\ell(i)}}}{q_{y_{\ell(i)}}}=\delta, & \text{si } z_i=I, \text{ et } i=1 \text{ ou } z_{i-1}=A,\\ \delta\frac{q_{x_{k(i)}}}{q_{x_{k(i)}}}=\delta, & \text{si } z_i=S, \text{ et } i=1 \text{ ou } z_{i-1}=A,\\ \varepsilon\frac{q_{y_{\ell(i)}}}{q_{y_{\ell(i)}}}=\varepsilon, & \text{si } z_i=I,\ i>1 \text{ ou } z_{i-1}=I,\\ \varepsilon\frac{q_{x_{k(i)}}}{q_{x_{k(i)}}}=\varepsilon, & \text{si } z_i=S,\ i>1 \text{ ou } z_{i-1}=S,\end{cases}$$

où

$$k(i) = i + \sum_{j=1}^{i-1} \mathbf{1}_{\{Z_j = I\}}, \quad \ell(i) = i + \sum_{j=1}^{i-1} \mathbf{1}_{\{Z_j = S\}}.$$

On note que $\prod_{i=1}^{t-1} v(i) = \prod_{i=1}^{t} v'(i)$, où

$$v'(i) = \begin{cases} (1-2\delta-\tau)\frac{p_{x_{k(i)} y_{\ell(i)}}}{q_{x_{k(i)}} q_{y_{\ell(i)}}}, & \text{si } Z_i = A, \\ \delta\frac{1-\varepsilon-\tau}{1-2\delta-\tau}, & \text{si } z_i = I, \text{ et } i = 1 \text{ ou } z_{i-1} = A, \\ \delta\frac{1-\varepsilon-\tau}{1-2\delta-\tau}, & \text{si } z_i = S, \text{ et } i = 1 \text{ ou } z_{i-1} = A, \\ \varepsilon, & \text{si } z_i = I,\ i > 1 \text{ ou } z_{i-1} = I, \\ \varepsilon, & \text{si } z_i = S,\ i > 1 \text{ ou } z_{i-1} = S, \end{cases}$$

pour $1 \leq i < t$, et

$$v(t) = \begin{cases} 1, & \text{si } z_{t-1} = A, \\ \frac{1-\varepsilon-\tau}{1-2\delta-\tau}, & \text{si } z_{t-1} = I \text{ ou } S. \end{cases}$$

Chercher un $(z^*)_1^t$ qui maximise $\mathbb{P}^*(X_1^N = x_1^n, Y_1^M = y_1^m, Z_1^T = z_1^t)$ est équivalent à chercher un $(z^*)_1^t$ qui maximise

$$\log \mathbb{P}^*(X_1^N = x_1^n, Y_1^M = y_1^m, Z_1^T = z_1^t) = \log\left(\frac{\tau}{1-\tau}\right) + \sum_{i=1}^{t} \log v'(i).$$

Il est alors facile de voir que l'algorithme de Viterbi pour résoudre ce probème est exactement l'algorithme de Needleman–Wunsch, à condition de poser

$$\begin{aligned} s(a,b) &= \log\frac{p_{ab}}{q_a q_b} + \log(1-2\delta-\tau), \\ d &= -\log\frac{\delta(1-\varepsilon-\tau)}{1-2\delta-\tau}, \\ e &= -\log\varepsilon. \end{aligned}$$

Notons que l'algorithme calcule une trajectoire z_1^t qui maximise la probabilité a posteriori. En particulier, la valeur t de la longueur de l'alignement optimal est donnée par

$$t = \inf\{s, k(s-1) = n, \ell(s-1) = m\}.$$

12.3 Loi a posteriori de l'alignement

Nous venons de donner un cadre probabiliste à l'algorithme de Needleman–Wunsch. Nous allons maintenant exploiter ce cadre probabiliste, pour faire des choses nouvelles.

L'idée de chercher à aligner deux séquences est liée à la croyance qu'il y a une similitude entre ces deux séquences, par exemple parce qu'elles dérivent d'une même séquence ancestrale. Lorsqu'un alignement n'est pas performant, cela peut provenir de ce que cet alignement

n'est pas le bon, ou qu'il n'y en a pas de bon, par exemple parce que ces séquences n'ont rien à voir entre elles. Il peut être intéressant de disposer d'un critère qui permette de décider dans quelle mesure deux séquences x_1^n et y_1^m sont *alignables*. Un tel critère (d'*alignabilité ?*) est donné par la probabilité que notre modèle de chaîne de Markov cachée produise la paire de séquences observée, i. e. par la quantité

$$\mathbb{P}(X_1^N = x_1^n, Y_1^M = y_1^m) = \sum_{z_1^t \in \text{alignements}} \mathbb{P}(X_1^N = x_1^n, Y_1^M = y_1^m, Z_1^T = z_1^t).$$

Quel est l'*ensemble des alignements* que parcourt z_1^t ? C'est l'ensemble des suites de longueur arbitraire t, dont les $t-1$ premiers termes appartiennent à l'ensemble $\{A, I, S\}$, et le dernier terme est un F. Evidemment, pour que la probabilité correspondante soit non nulle, une contrainte très forte liant z_1^t à n et m doit être satisfaite, à savoir

$$t = \inf\{s, k(s-1) = n, \ell(s-1) = m\}.$$

En particulier, on doit avoir $t \geq \sup(n, m) + 1$.

On va maintenant voir qu'il existe un algorithme *progressif* qui calcule la quantité $\mathbb{P}(x_1^n, y_1^m)$. Il s'agit tout simplement de la partie progressive de l'algorithme de Viterbi–Needelman–Wunsch, où l'on remplace la maximisation par la somme. Plus précisément, on calcule $(f^A(i,j), f^I(i,j), f^S(i,j))$ par une **récurrence progressive** sur (i,j) comme suit

$$\begin{aligned}
f^{\cdot}(i,-1) &= f^{\cdot}(-1,j) = 0, \quad \forall i,j \\
f^A(0,0) &= 1,\ f^I(0,0) = 0,\ f^S(0,0) = 0 \\
f^A(i,j) &= p_{x_i y_j}[(1-2\delta-\tau)f^A(i-1,j-1) + (1-\varepsilon-\tau)(f^I(i-1,j-1) + f^S(i-1,j-1))] \\
f^I(i,j) &= q_{y_j}[\delta f^A(i,j-1) + \varepsilon f^I(i,j-1)] \\
f^S(i,j) &= q_{x_i}[\delta f^A(i-1,j) + \varepsilon f^I(i-1,j)]
\end{aligned}$$

Finalement

$$\mathbb{P}(X_1^N = x_1^n, Y_1^M = y_1^m) = \tau[f^A(n,m) + f^I(n,m) + f^S(n,m)].$$

On peut maintenant considérer la loi a posteriori de l'alignement, i. e. la probabilité conditionnelle

$$\mathbb{P}(Z_1^T = z_1^t | X_1^N = x_1^n, Y_1^M = y_1^m) = \frac{\mathbb{P}(X_1^N = x_1^n, Y_1^M = y_1^m, Z_1^T = z_1^t)}{\mathbb{P}(X_1^N = x_1^n, Y_1^M = y_1^m)}.$$

Le cas le plus favorable est celui où cette loi est très concentrée autour de l'alignement optimal, i.e. où la quantité $\mathbb{P}(Z_1^T = (z^*)_1^t | X_1^N = x_1^n, Y_1^M = y_1^m)$ est sinon proche de 1, du moins nettement non nulle. Lorsque ce n'est pas le cas, il peut être intéressant de savoir si l'ensemble des alignements *proches de l'alignement optimal* concentre ou non une masse significative de la loi a posteriori. La loi a posteriori contient donc beaucoup d'information sur la qualité et la pertinence de l'alignement optimal. Nous allons maintenant décrire un

algorithme *rétrograde* pour simuler suivant cette loi a posteriori, qui utilise le calcul des quantités $(f^A(i,j), f^I(i,j), f^S(i,j))$ fait ci dessus.

On commence par tirer au hasard avec la probabilité

$$(f^A(n,m) + f^I(n,m) + f^S(n,m))^{-1}(f^A(n,m) \quad f^I(n,m) \quad f^S(n,m))$$

si l'alignement se termine au site $t-1$ par l'alignement de x_n avec y_m (choix de A), ou par l'alignement de y_m avec un *trou* dans la première séquence (choix de I), ou par l'alignement de x_n avec un trou dans la seconde séquence (choix de S).

Décrivons maintenant l'itération de cet algorithme. Supposons qu'à un moment donné l'algorithme nous ait conduit à aligner x_i et y_j. On examine alors la quantité

$$f^A(i,j) = p_{x_i y_j}[(1-2\delta-\tau)f^A(i-1,j-1) + (1-\varepsilon-\tau)(f^I(i-1,j-1) + f^S(i-1,j-1))].$$

On décide alors

– d'aligner x_{i-1} avec y_{j-1} avec la probabilité $\frac{p_{x_i y_j}(1-2\delta-\tau)f^A(i-1,j-1)}{f^A(i,j)}$,

– d'aligner y_{j-1} avec un *trou* dans la première séquence avec la probabilité $\frac{p_{x_i y_j}(1-\varepsilon-\tau)f^I(i-1,j-1)}{f^A(i,j)}$,

– d'aligner x_{i-1} avec un *trou* dans la seconde séquence avec la probabilité $\frac{p_{x_i y_j}(1-\varepsilon-\tau)f^S(i-1,j-1)}{f^A(i,j)}$.

Si par contre l'algorithme nous avait conduit à aligner y_j avec un *trou* dans la première séquence, x_i étant le dernier nucléotide de celle–ci *non encore utilisé*, on décide

– d'aligner x_i avec y_{j-1} avec la probabilité $\frac{q_{y_j}\delta f^A(i,j-1)}{f^I(i,j)}$,

– d'aligner y_{j-1} avec un *trou* dans la première séquence avec la probabilité $\frac{q_{y_j}\varepsilon f^I(i,j-1)}{f^I(i,j)}$.

On a une description analogue dans le cas où x_i est aligné avce un *trou* dans la seconde séquence.

12.4 La probabilité a posteriori d'alignements particuliers

Si la probabilité de n'importe quel alignement est faible, il se peut que celle de certains alignements partiels soit élevé. On va voir que l'on peut calculer la probabilité que l'alignement mette en correspondance certaines paires (x_i, y_j). On notera $x_i \diamondsuit y_j$ l'ensemble des alignements z_1^t qui placent x_i en face de y_j. On va maintenant calculer la probabilité

$$\mathbb{P}(X_1^N = x_1^n, Y_1^M = y_1^m, x_i \diamondsuit y_j) = \sum_{z_1^t \in x_i \diamondsuit y_j} \mathbb{P}(X_1^N = x_1^n, Y_1^M = y_1^m, Z_1^T = z_1^t).$$

On a en fait

$$\mathbb{P}(X_1^N = x_1^n, Y_1^M = y_1^m, x_i \diamondsuit y_j) = \mathbb{P}(X_1^i = x_1^i, Y_1^j = y_1^j, x_i \diamondsuit y_j) \times \mathbb{P}(X_i^N = x_i^n, Y_j^M = y_j^m | x_i \diamondsuit y_j).$$

Le premier facteur dans cette formule n'est autre que la quantité $f^A(i,j)$ que l'on a calculée par un algorithme progressif à la sous–section précédente.

Le second facteur est la quantité $b^A(i,j)$, que l'on calcule par une **récurrence rétrograde** comme suit.

$$b^A(n,m) = b^I(n,m) = b^S(n,m);$$
$$b^{\cdot}(\cdot, m+1) \equiv b^{\cdot}(n+1, \cdot) \equiv 0.$$

Pour tous les $(i,j) \neq (n,m)$,

$$b^A(i,j) = (1-2\delta-\tau)p_{x_{i+1}y_{j+1}}b^A(i+1,j+1) + \delta[q_{x_{i+1}}b^S(i+1,j) + q_{y_{j+1}}b^I(i,j+1)],$$
$$b^I(i,j) = (1-\varepsilon-\tau)p_{x_{i+1}y_{j+1}}b^A(i+1,j+1) + \varepsilon q_{y_{j+1}}b^I(i,j+1),$$
$$b^S(i,j) = (1-\varepsilon-\tau)p_{x_{i+1}y_{j+1}}b^A(i+1,j+1) + \varepsilon q_{x_{i+1}}b^S(i+1,j).$$

On notera alors

$$\mathbb{P}(x_i \diamondsuit y_j | X_1^N = x_1^n, Y_1^M = y_1^m) = \frac{\mathbb{P}(X_1^N = x_1^n, Y_1^M = y_1^m, x_i \diamondsuit y_j)}{\mathbb{P}(X_1^N = x_1^n, Y_1^M = y_1^m)},$$

que l'on écrira en abrégé $\mathbb{P}(x_i \diamondsuit y_j)$.

Etant données deux séquences x_1^n et y_1^m, et un alignement z_1^t de ces séquences, on notera $(i,j) \in z_1^t$ si et seulement si $z_1^t \in x_i \diamondsuit y_j$, i. e. si l'alignement z_1^t place x_i et y_j l'un en face de l'autre. La quantité suivante est une sorte d'espérance, pour deux suites x_1^n et y_1^m données, du recouvrement entre l'alignement z_1^t et un alignement tiré au hasard suivant la loi a posteriori des alignements :

$$\mathcal{A}_{x_1^n, y_1^m}(z_1^t) = \sum_{(i,j) \in z_1^t} \mathbb{P}(x_i \diamondsuit y_j).$$

Il s'agit d'un nouveau critère de qualité d'un alignement, pour lequel on peut calculer l'alignement optimal, par l'algorithme de programmation dynamique classique associé à la récurrence progressive (on abandonne ci–dessous l'indice (x_1^n, y_1^m)).

$$A(i,j) = \max \begin{cases} A(i-1,j-1) + \mathbb{P}(x_i \diamondsuit y_j), \\ A(i-1,j), \\ A(i,j-1). \end{cases}$$

Notons que tous les $\mathbb{P}(x_i \diamondsuit y_j)$, au facteur multiplicatif constant $\mathbb{P}(X_1^N = x_1^n, Y_1^M = y_1^m)$ près, se déduisent immédiatement des quantités calculées par les deux algorithmes progressif et rétrograde ci–dessus.

13 ProbCons, un algorithme d'alignement multiple

En bioinformatique, on a souvent besoin d'aligner plus de deux séquences. Ce problème est très difficile. Tous les algorithmes qui résolvent ce problème en un temps raisonnable (c'est à dire tous ceux qui sont utlisés en pratique) procèdent peu ou prou par alignements deux à deux successifs.

Nous allons présenter le dernier né de ces algorithmes, qui est celui qui à ce jour semble donner les meilleurs résultats en terme de précision, tout en étant plus rapide que certains de ses concurrents, cf. [1]. Cet algorithme utilise, comme on va le voir, le cadre des "PairHMMs" que nous venons d'étudier à la section précédente. On reprendra les notations introduites ci–dessus sans les redéfinir. Cependant, nous allons introduire un changement de notation : dorénavant les alignements seront notés a, et non plus z. En effet, puisque nous serons obligés de désigner trois séquences à la fois, nous les noterons $x_1^{|x|}$, $y_1^{|y|}$ et $z_1^{|z|}$.

Etant donné m séquences $\{s^1, \ldots, s^m\}$, l'algorithme effectue les opérations successives suivantes :

– Etape 1 : Calcul des matrices de probabilités a posteriori.
Pour chaque paire $x, y \in S$, on calcule la matrice

$$P(x,y) = \left(P_{ij}(x,y)\right), \quad 1 \leq i \leq |x|,\ 1 \leq j \leq |y|$$

donnée par

$$P_{ij}(x,y) = \mathbb{P}(x_i \diamondsuit y_j | X_1^N = x_1^{|x|}, Y_1^M = y_1^{|y|}).$$

– Etape 2 : Alignements deux à deux.
Pour chaque paire $x, y \in S$, on calcule l'alignement a^* qui maximise la quantité

$$\mathcal{A}_{x_1^{|x|}, y_1^{|y|}}(a),$$

et on pose

$$E(x,y) = \frac{1}{\min(|x|,|y|)} \mathcal{A}_{x_1^{|x|}, y_1^{|y|}}(a^*).$$

– Etape 3 : Transformation par consistence.
Aux trois séquences $x, y, z \in S$, on associe en particulier la matrice $|x| \times |z|$ $P(x,z)$ et la matrice $|z| \times |y|$ $P(z,y)$. On a alors bien sûr pour $1 \leq i \leq |x|$ et $1 \leq j \leq |y|$,

$$\left(P(x,z)P(z,y)\right)_{ij} = \sum_{k=1}^{|z|} P_{ik}(x,z) P_{kj}(z,y),$$

et on définit une nouvelle matrice $P'(x,y)$ par

$$P'(x,y) = \frac{1}{|S|} \sum_{z \in S} P(x,z)P(z,y).$$

Notons que par définition

$$P_{ij}(x,x) = \begin{cases} 1, & \text{si } i = j, \\ 0, & \text{si } i \neq j. \end{cases}$$

Dans chacune des matrices ci–dessus, un grand nombre de termes sont très petits. Tous les termes inférieurs à un seul fixé sont mis à zéro, et un algorithme de multiplication de matrices creuses est utilisé. La transformation $P \to P'$ peut être itérée un nombre arbitraire de fois. Par défaut, ProbCons l'effectue deux fois.

– Etape 4 : Construction d'un arbre.
Ici on va utiliser les quantités $E(x, y)$ calculées à l'étape 2. La construction est effectuée par la méthode itérative suivante. Pour commencer, chaque séquence est identifiée à un groupe. A chaque paire de groupes x et y, on associe la quantité $E(x, y)$ calculée à l'étape 2. A chaque étape, on cherche le couple de groupes (x, y) qui maximize la quantité $E(x, y)$. On fusionne ces deux groupes en un seul noté xy, et on définit pour tout autre groupe z la quantité

$$E(xy, z) := E(x, y)\frac{E(x, z) + E(y, z)}{2}.$$

– Etape 5 : Alignement progressif.
On aligne les séquences deux par deux, en utilisant l'algorithme de Needleman–Wunsch, avec le score

$$s(x_i, y_j) := P'_{ij}(x, y),$$

et des pénalités pour les *trous* (d et e) égales à zéro. Deux groupes sont alignés selon le meilleur alignement de deux séquences, une prise dans chaque groupe.

– Raffinement itératif.
On partitionne de façon aléatoire l'ensemble S en deux sous–groupes, auxquels on fait subir les étapes 4 et 5.

Références

[1] C. B. Do, M. S. P. Mahabhashyam, M. Brudno, S. Batzoglou, ProbCons : Probabilistic consistency–based multiple sequence alignment, *Genome Research* **15**, 330–340, 2005.

[2] R. Durbin, S. Eddy, A. Krogh, G. Mitchison, *Biological sequence analysis. Probabilistic models of proteins and nucleic acids*, Cambridge Univerfsity Press, 1998.

[3] F. Muri–Majoube, B. Prum, Une approche statistique de l'analyse des génomes, *Gazette des Mathématiciens* **89**, 63–98, 2001.

[4] S. Robin, F. Rodolphe, S. Schbath, *ADN, mots, modèles*, Belin 2003.

[5] V. Plagnol, Mémoire de DEA, 2001.