Université du Québec à Chicoutimi

Mémoire présenté à
l'Université du Québec à Chicoutimi
comme exigence partielle
de la maîtrise en informatique

par
ISMAËL COULIBALY

CONCEPTION D'ALGORITHMES PROBABILISTES POUR L'ESTIMATION DES GÉNOTYPES D'UN CORPUS DE GÉNÉALOGIE PAR CHAINES DE MARKOV

29 mars 2009

Bibliothèque
Paul-Émile-Boulet

UQAC
Université du Québec
à Chicoutimi

Mise en garde/Advice

Afin de rendre accessible au plus grand nombre le résultat des travaux de recherche menés par ses étudiants gradués et dans l'esprit des règles qui régissent le dépôt et la diffusion des mémoires et thèses produits dans cette Institution, **l'Université du Québec à Chicoutimi (UQAC)** est fière de rendre accessible une version complète et gratuite de cette œuvre.

Motivated by a desire to make the results of its graduate students' research accessible to all, and in accordance with the rules governing the acceptation and diffusion of dissertations and theses in this Institution, the **Université du Québec à Chicoutimi (UQAC)** is proud to make a complete version of this work available at no cost to the reader.

L'auteur conserve néanmoins la propriété du droit d'auteur qui protège ce mémoire ou cette thèse. Ni le mémoire ou la thèse ni des extraits substantiels de ceux-ci ne peuvent être imprimés ou autrement reproduits sans son autorisation.

The author retains ownership of the copyright of this dissertation or thesis. Neither the dissertation or thesis, nor substantial extracts from it, may be printed or otherwise reproduced without the author's permission.

Résumé

La fréquence d'un trait génétique dans une population contemporaine dépend de la fréquence de ce trait chez les ancêtres. Cependant, la seule information génotypique disponible sur ces ancêtres est essentiellement constituée des analyses d'ADN sur un échantillon de contemporains. Aussi, l'intégration des données moléculaires et des données généalogiques se heurtent naturellement à un problème de complétude. L'un des principaux problèmes en génétique des populations est alors d'inférer sur les génotypes de ces ancêtres en se basant sur un échantillon d'individus contemporains. On ne peut cependant reproduire de façon exacte les génotypes des individus. En effet, les lois de transmission des gènes sont telles qu'il est rarement possible d'obtenir un génotype certain pour un ancêtre ; au mieux il est possible d'obtenir qu'une estimation de la loi de probabilité des génotypes ancestraux. L'utilisation d'algorithmes déterministes permet dans des cas simples de trouver les solutions à ce problème. Néanmoins, certains algorithmes probabilistes, particulièrement les méthodes de Monte Carlo par Chaînes de Markov (MCMC), sont particulièrement adaptés à l'analyse des génotypes liée à une structure généalogique. Elizabeth A. Thompson utilise cette technique pour effectuer des analyses de liaisons génétiques (linkage) sur des noyaux familiaux étendus (Wijsman et al. [28]). Le contexte général d'un algorithme probabiliste en informa-

tique étant l'utilisation d'un générateur de nombres pseudo aléatoires, la mise en place de tels algorithmes est donc relativement facile et s'adapte très bien à des problèmes d'inférence. Notre objectif est alors de concevoir des algorithmes efficaces pour l'estimation des génotypes d'un corpus de généalogie en utilisant la technique des MCMC. Le caractère théorique de ces algorithmes et leur efficacité ont été largement étudiés dans plusieurs ouvrages sur lesquelles nous nous sommes basés. Dans ce mémoire, nous nous appliquons principalement à utiliser les méthodes de Monte Carlo par chaînes de Markov (MCMC) à l'analyse de corpus de généalogie. Nous montrons comment nous réussissons à adapter la méthode de l'échantillonnage de Gibbs à l'élaboration d'algorithmes efficaces pour l'inférence des distributions de probabilité génotypique dans le contexte de généalogies profondes.

Liste des algorithmes

LISTE DES FIGURES

Liste des tableaux

TABLE DES MATIÈRES

CHAPITRE 1

INTRODUCTION

La génétique a connu de grandes avancées dans les dernières décennies. Depuis la redécouverte des lois de Mendel par les scientifiques Hugo de Vries, Carl Correns et Erich Von Tschemark au début du XX$^{\text{ième}}$ siècle, en passant par la découverte de l'ADN par James D. Watson et Francis H.C Crick dans les années 1950 (Russel [24]), la génétique n'a cessé de révolutionner le monde à travers différents domaines de recherche. Cela en mettant à la disposition des chercheurs de plus en plus d'informations et de données, plus particulièrement au niveau moléculaire. Combiné à cela, les percées en informatique, autant au niveau des machines que des techniques de traitement de grandes masses de données, ont permis d'identifier et de cartographier des traits génétiques simples ainsi que quelques traits complexes. Ces derniers sont cependant beaucoup plus difficiles à étudier et l'apport d'information sur la structure de la population est nécessaire pour mieux cerner l'effet spécifique des gènes. Pour incorporer l'information sur la structure de la population, il faut dans un premier temps reconstituer les généalogies concernant les individus pour lesquels l'information moléculaire est disponible et ensuite combiner ces deux informations.

Depuis le début des années 1970, le projet Balsac collige des actes de naissance, de mariage et de décès disponibles sur la population du Québec depuis l'arrivée

des premiers colons. Il en a résulté la création de bases de données informatisées permettant de reconstituer des généalogies québécoises sur plusieurs générations avec un taux d'erreur minime (BALSAC [1]). Plusieurs projets ont été mis en place autour de ces bases de données permettant ainsi de lier des données moléculaires à des structures généalogiques. Les bases de données Balsac et les données d'ADN permettent alors d'explorer, de façon plus poussée, certains traits à caractère génétique, ce qui était auparavant impossible dû au manque d'informations sur les populations étudiées ainsi qu'à l'inaccessibilité de certaines données.

La quantité d'informations disponibles et la facilité d'accès à ces informations amènent à nous intéresser aux deux faits suivants. Le premier concerne l'utilité de ces informations et le second la manière de les utiliser. Une remarque rapide que nous pouvons faire relativement à ces faits peut être résumée par la phrase suivante : La disponibilité de l'information combinée à la puissance de calcul des ordinateurs actuels et l'intelligence avec laquelle nous les utilisons peuvent permettre, par une lecture probabiliste du passé, d'inférer sur certaines données. En effet, l'exécution de calculs auparavant impossibles à réaliser permet de résoudre des problèmes complexes puisque certaines méthodes purement théoriques peuvent maintenant être implantées sur ordinateur.

Ainsi, les fichiers de population permettent de reconstituer des ascendances généalogiques sur plusieurs générations. Cependant, les données moléculaires ne sont disponibles que pour quelques individus, habituellement les points de départ des ascendances généalogiques. Le problème est alors d'inférer sur les génotypes des ancêtres. Pour cela, il faudrait être capable de reconstituer l'histoire génétique des individus. La reconstitution génotypique d'un corpus de généalogie apparaît dès lors comme une problématique.

Dans ce mémoire, nous tentons donc de résoudre la problématique de la reconstitution des gènes d'un corpus généalogique. Nous tentons d'inférer sur le génotype des individus d'une généalogie, en accolant une probabilité aux différentes possibilités génotypiques de ces individus, quelle que soit leur position dans la généalogie. Les méthodes de Monte Carlo par chaînes de Markov permettent de concevoir des algorithmes efficaces pour l'inférence sur des distributions de probabilité génotypiques dans le contexte de corpus de généalogies profondes. Dans le second chapitre, nous commencerons par définir quelques termes et préciser certaines notions qui permettront de comprendre et d'expliquer la problématique. Nous faisons aussi un exposé sur les chaînes de Markov d'une part et les méthodes de Monte Carlo d'autre part, cela pour comprendre le contexte de leurs applications en génétique des populations. Le troisième chapitre sera alors l'occasion d'exposer le contexte des algorithmes utilisés et de les optimiser. Enfin, dans le chapitre quatre, nous testons ces algorithmes à l'aide de données théoriques, ce qui permettra de calibrer les paramètres du modèle pour ensuite les appliquer à des données réelles.

Chapitre 2

Généralités et Problématique

2.1 Notations et définitions

Dans cette partie, nous formalisons certaines notions et quelques principes qui découlent du modèle mendélien. Cela permettra d'établir les bases de nos analyses, pour ensuite formaliser la problématique.

2.1.1 Le modèle Mendélien

Le modèle mendélien repose sur deux principes fondamentaux établis par Mendel, vers la fin du XIX$^{\text{ième}}$ siècle. Le premier principe concerne la ségrégation des caractères selon laquelle les gamètes sont dits purs dans le sens où ils ne portent qu'un allèle de chaque gène ; dans le cas contraire, ils sont dits ségrégés et alors les individus descendants sont formés par combinaison au hasard de deux gamètes provenant de chacun des parents (Russel [24]). Le deuxième principe découle de la ségrégation indépendante ou du réassortiment indépendant qui affirme que des gènes situés sur des chromosomes non homologues[1] se répartissent indépendamment les uns des autres au moment de la formation des gamètes (Russel [24]). Voir aussi Jenkins [16] et Suzuki et al. [25] pour plus de détails sur les lois de Mendel. Nous reviendrons un peu plus

1. Chromosomes sexuels chez l'être humain.

loin sur le concept de modèle mendélien ; avant cela déterminons certains éléments que nous utiliserons comme support dans nos analyses.

2.1.2 Le concept de généalogie

Définition 2.1 *Une généalogie est la reconstitution de la descendance ou de l'ascendance, pour des caractères ou des traits, indiquant différents membres d'une famille, leurs rapports, et leur statut en ce qui concerne le caractère ou le trait.*

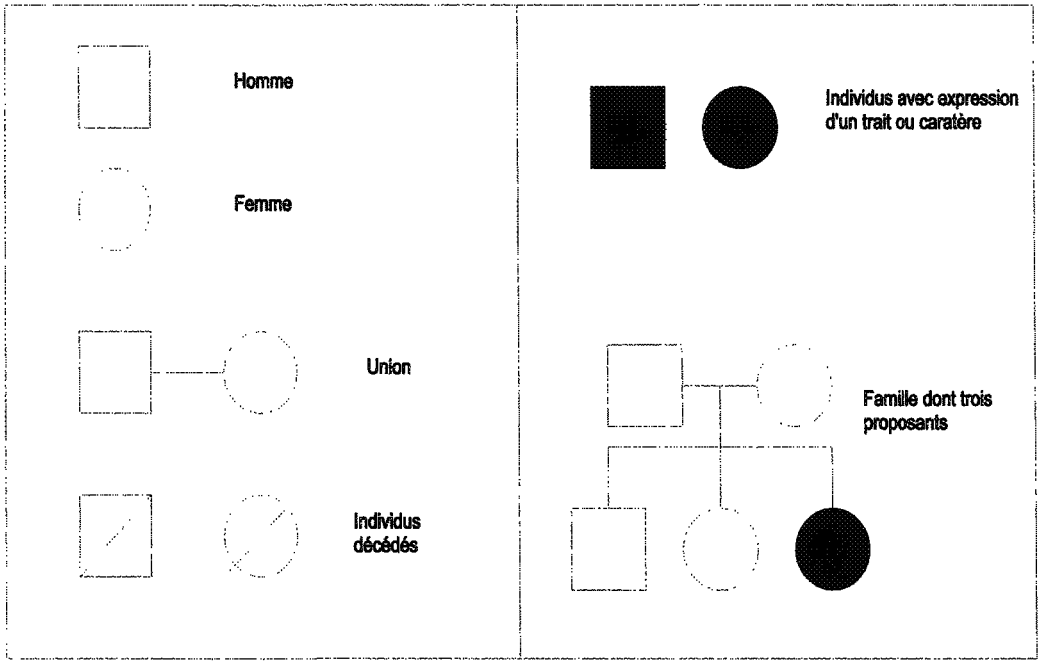


FIGURE 2.1 – Différents éléments de composition d'une généalogie

Une généalogie est divisée en strates ou générations appelé profondeur. Les éléments présentés à la Figure 2.1 font consensus pour la représentation graphique des données d'une généalogie.

Définition 2.2 *On appelle généalogie profonde, une généalogie qui s'étend sur plusieurs générations, généralement supérieur à 10.*

Un individu appartenant alors à la $k^{ième}$ génération, a ses parents à la $(k+1)^{ième}$ génération et ses enfants à la $(k-1)^{ième}$ génération. Les profondeurs sont donc établies en remontant l'arbre du bas (des feuilles) vers le haut.

Définition 2.3 *Les individus qui permettent de remonter l'arbre à partir de la base sont appelés les proposants.*

En d'autres mots, un proposant est un individu de bout d'arbre ayant un père et une mère, mais aucune descendance.

Définition 2.4 *Les individus qui permettent de parcourir l'arbre dans le sens contraire, donc les individus pour lesquels il a été impossible de retrouver les parents, ayant au moins une descendance sont appelés les fondateurs.*

D'après le modèle mendélien, la transmission d'un caractère ou d'un trait se fait de parents à enfants par l'entremise du gène. Le gène est l'unité héréditaire contrôlant un caractère ou un trait particulier. Il s'exprime en deux ou plusieurs versions provenant de la mère et du père. Ces versions sont appelées allèles. Ainsi, lors de la transmission, un allèle provient de la mère et l'autre provient du père.

Définition 2.5 *L'ensemble du patrimoine héréditaire, donc génétique, propre à un individu est appelé génotype.*

Lorsque nous parlons de génotype dans la suite, nous faisons référence au trait ou caractère étudié et non à l'ensemble du patrimoine génétique de l'individu. Pour mieux comprendre les notions précédentes, aidons-nous de la Figure 2.2. Cette figure[2] représente une généalogie de 18 individus répartis sur 4 générations. Les individus ombragés

2. Les numéros attribués aux individus de la généalogie sont purement arbitraires.

ont l'expression d'un trait spécifique (les individus 15 et 17). Ce trait peut être dû à la présence d'au moins un allèle lié au trait étudié dans le génotype de l'individu, dans ce cas la on parlera d'allèle de référence. Dans le cas contraire ou l'expression du trait est dû à la présence de deux allèles liés au trait étudié, on parlera d'allèle complémentaire. En effet, le système de transmission des gènes peut être multiallélique, mais l'idée est plutôt d'observer un seul allèle (donc l'allèle de référence).

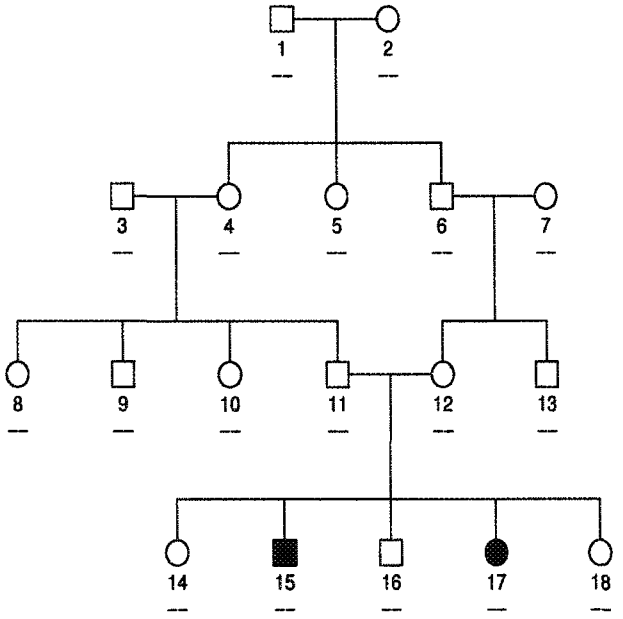


FIGURE 2.2 – Généalogie de 18 individus, repartie sur 4 générations (profondeur) dont 4 fondateurs (les individus 1, 2, 3 et 7) et 2 proposants (les individus 15 et 17) exprimant un trait particulier.

2.2 La problématique de la reconstitution des génotypes

À partir de la Figure 2.2, nous pouvons poser la question suivante : sachant que les individus 15 et 17 ont l'expression d'un certain trait, pouvons-nous connaître la provenance de ce caractère ? Pour répondre à cette question, il faudrait être capable de reconstituer le portrait génotypique de la généalogie. Nous arrivons alors à un problème en analyse génétique des populations : la reconstitution des génotypes. Cette problèma-

tique n'en serait pas une si toute l'information génotypique était disponible. Cela n'est pas le cas, dû au manque de données moléculaires sur les individus qui n'existent plus. Plusieurs approches peuvent être utilisées pour essayer de trouver une solution. Nous étudions dans cette partie, les différentes approches pouvant être utilisées, à travers leur mécanisme, leurs avantages et leurs limites.

2.2.1 Approche déterministe

Dans un premier temps, nous présentons l'approche la plus intuitive. L'approche déterministe va comme suit : à partir des individus 15 et 17 de la Figure 2.2, nous essayons de déduire dans l'ordre suivant, les génotypes des parents, des frères et sœurs, des grands-parents, des frères et sœurs des parents, et ainsi de suite. Cette reconstitution intuitive se base sur les principes du modèle mendélien (définis dans les sections précédentes). Essayons de réaliser cette tâche à partir de la généalogie de la Figure 2.2. Avant cela, faisons quelques suppositions pour simplifier le mécanisme. Supposons qu'on nomme arbitrairement « a » l'allèle de référence. On aurait pu supposer le contraire et faire la même démarche ; dans ce cas là, l'allèle « a » aurait été l'allèle complémentaire. L'allèle complémentaire est alors noté « A ». Les possibilités pour le génotype d'un individu sont donc : « aa », « Aa » ou « AA ».

On commence alors par attribuer les génotypes aux individus 15 et 17. Leur génotype respectif sont « aa » et « aa » (allèle récessif). Selon le modèle mendélien, ils reçoivent chacun, un allèle de leur père et un allèle de leur mère. Donc, l'allèle transmis par le père (individu 11) est « a » et l'allèle transmis par la mère (individu 12) est aussi « a ». Les génotypes possibles pour chacun des parents sont alors « aa » ou « Aa ». Les frères et sœurs (les individus 14, 16 et 18) n'ont pas l'expression du trait étudié, leur génotype peut être « Aa » ou « AA », parce qu'encore une fois l'allèle « a » est l'allèle de référence. On arrive donc aux remarques suivantes :

- Le génotype de chacun des frères et sœurs est identifié par un seul allèle « A- », puisque nous n'avons aucune certitude concernant le deuxième allèle ;
- Le génotype de chacun des parents est alors « Aa », puisqu'ils donnent avec certitude un allèle de type « a » ou un de type « A », à chacun de leurs enfants.

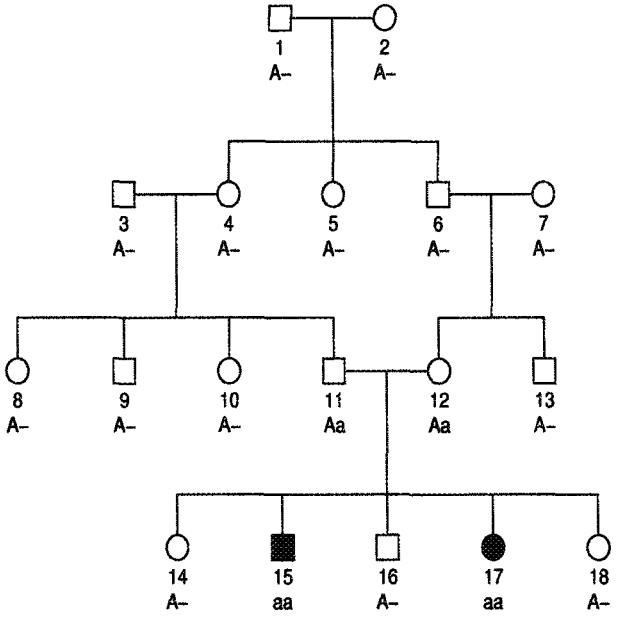


FIGURE 2.3 – Reconstitution déductive des génotypes des individus à partir des génotypes des individus 15 et 17.

On continue alors ce mécanisme pour reconstituer l'information génétique de la généalogie de la Figure 2.2. Cette reconstitution génotypique est illustrée à la Figure 2.3.
Le premier constat concerne la simplicité de cette méthode par déduction. On part d'un fait et on en déduit les prédicats. Le second constat relève de la lourdeur des déductions. Même avec une petite généalogie (l'exemple de la Figure 2.2), il est souvent difficile de faire des déductions. Le troisième constat concerne l'incapacité de compléter l'information génotype pour certains individus. En effet, le génotype de certains des individus de

la généalogie ne peut être déduit avec certitude. Nous n'avons pas assez d'informations pour faire certaines analyses. Par exemple, nous ne pouvons pas avoir d'information sur la provenance d'un trait étudié. Aussi, pour de grandes généalogies, il est facile de constater que la reconstitution par approche déterministe se révèle inefficace. D'autres méthodes utilisant une approche probabiliste ont fait l'objet d'études et apportent une amélioration significative. En effet, dans le meilleur des cas, nous pouvons avoir un ou plusieurs chemins de transmission (relativement au trait étudié) en inférant sur le génotype des individus.

2.2.2 Méthode du gene-dropping

L'une des méthodes utilisant les simulations informatiques, appliquées en génétique des populations, est le *gene-dropping*. La méthode du *gene-dropping* consiste donc à simuler la transmission mendélienne d'un gène à travers une généalogie. Cette méthode s'avère efficace pour l'étude des fréquences alléliques dans une population, suite à l'introduction d'un gène par un ou plusieurs ancêtres (Évelyne Heyer [27]). Elle utilise un algorithme assez simple pour simuler et mesurer la transmission des gènes dans des populations. Sachant que chaque parent possède deux allèles et qu'un seul de ces allèles

Algorithme 2.1 Gene-Dropping

Préconditions: Généalogie
Postconditions: Tableau des fréquences d'apparition des allèles

```
Attribuer un génotype unique à chaque fondateur (2 allèles différents)
pour i allant de 1 à n faire
    tant que il existe des individus sans génotype faire
        Attribuer un génotype à chaque descendant à partir des génotypes de ses parents
    fin tant que
    Noter les fréquences d'apparition des allèles fondateurs dans un tableau T[ ]
fin pour
Retourner T[ ].
```

est transmis à son enfant, chaque allèle a donc une chance sur deux d'être transmis

aux descendants et ce, indépendamment de l'héritage de l'autre parent. Il s'agit alors d'appliquer une simulation descendante du modèle mendélien en prenant comme origine les génotypes ancestraux pour obtenir celui des individus contemporains. L'algorithme (illustré par l'Algorithme 2.1) commence d'abord par attribuer un numéro unique à chacun des individus, tel que les parents ont un numéro de rang inférieur à ceux de leurs enfants. On attribue ensuite un génotype unique, formé de deux allèles différents, à chaque fondateur. Le génotype des descendants (génération par génération) est enfin déterminé, en choisissant un allèle à partir du génotype du père et un autre allèle à partir de celui de la mère, de façon aléatoire (application du modèle mendélien). Cette attribution est faite de génération en génération jusqu'à ce que tous les individus de l'arbre généalogique aient un génotype pour constituer alors la fin d'une simulation. Ainsi, on s'assure de respecter la transmission « parents vers les enfants » en simulant le génotype des individus des parents avant ceux de leurs enfants. Le processus est répété plusieurs fois et à la fin de chaque simulation, on note les différentes fréquences d'apparition des allèles des fondateurs, pour dresser un tableau de distribution des probabilités des allèles de ceux-ci. Un exemple simple de la méthode du *Gene-Dropping* est présenté à la Figure 2.4 par l'attribution des génotypes fondateurs (initialisation) et à la Figure 2.5, par la reconstitution des génotypes de tous les individus qui illustre la fin d'une simulation.

Ainsi à partir de la Figure 2.4, nous déterminons les fondateurs et leur attribuons un génotype unique :

- Individu 1 avec comme génotype A_1/A_2 ;
- Individu 2 avec comme génotype A_3/A_4 ;
- Individu 3 avec comme génotype A_5/A_6 ;
- Individu 4 avec comme génotype A_7/A_8 ;

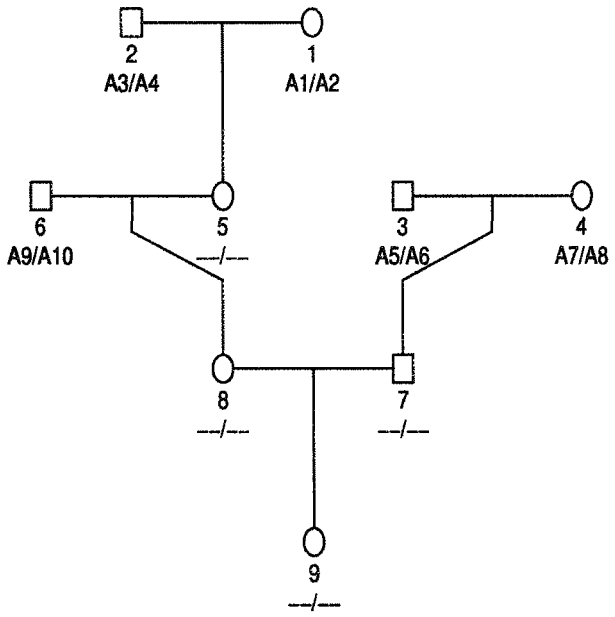


FIGURE 2.4 – Gene-Dropping : Attribution des gènes aux fondateurs.

– Individu 6 avec comme génotype A_9/A_{10}.

Ensuite, à partir des génotypes de leurs parents, les génotypes des individus 5 et 7 sont déterminés, celui de l'individu 8 et, enfin, celui de l'individu 9. Le résultat final est présenté à la Figure 2.5.

Cette technique est très simple à mettre en place, son fondement étant basé sur le concept du modèle mendélien. Elle va donc servir à l'analyse des ascendances d'un corpus de généalogie, en identifiant un allèle fondateur (donc un fondateur) comme étant vraisemblablement à l'origine de la propagation du caractère étudié.

La méthode du *gene-dropping* reste cependant assez limitée dans le contexte de notre étude. Elle reste intéressante pour la description du pool génétique d'une population (Tremblay et al. [26]), mais elle ne permet pas d'associer la généalogie directement à des données moléculaires (génotypes observés chez les proposants). Nous cherchons

plutôt à inférer sur les génotypes ancestraux à partir des informations obtenues sur les proposants.

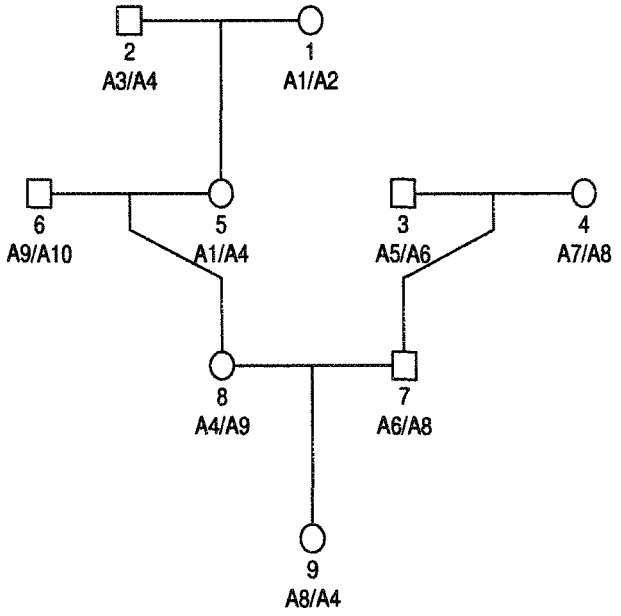


FIGURE 2.5 – Gene-Dropping : Reconstitution des gènes de la généalogie à partir des fondateurs. On reconstitue les génotypes des individus de haut en bas.

2.2.3 Approche probabiliste utilisant les chaînes de Markov

L'utilisation des chaînes de Markov combinée aux méthodes de Monte Carlo (dont nous discuterons plus loin), permettent, à partir des probabilités à priori sur plusieurs simulations, d'intégrer les données moléculaires aux simulations en conditionnant sur les observations (génotype disponible de certains individus). On obtient alors un ensemble d'observations compatibles avec les génotypes connus. Chacune de ces observations est constituée des génotypes simulés de tous les individus « inconnus » de la généalogie. L'algorithme de Hasting-Métropolis et l'échantillonnage de Gibbs consti-

tuant les méthodes de Monte Carlo par chaînes de Markov *(MCMC)* se sont montrées utiles et efficaces. Ces méthodes ont été utilisés par Elizabeth A. Thompson lors d'études pour des analyses de liaisons génétiques à partir de généalogies restreintes (Wijsman et al. [28]). Elles ont permis de simuler une distribution génotypique chez les individus d'un corpus généalogique tout en restant compatible avec les génotypes réellement observés. Cependant, les MCMC utilisées dans ce contexte sont utilisées pour maximiser une fonction de vraisemblance liée à un paramètre. Dans ce mémoire, nous adaptons la méthode de l'échantillonnage de Gibbs pour permettre de répondre à la question de la reconstitution des génotypes.

Les progrès techniques des deux dernières décennies en informatique, tant du côté de la conception des ordinateurs que de celui des algorithmes - le premier entraînant le deuxième dans la plupart des cas - ont contribué fortement à l'évolution des méthodes en statistique. La mise en place de méthodes qui se basent sur les calculs par ordinateur et par conséquent demandent d'énormes ressources de calcul est grandement facilitée de nos jours puisqu'elles peuvent enfin être implantés de façon efficace pour produire des résultats optimaux. On peut citer, entre autres, plusieurs théories déterministes émergentes comme l'estimation fonctionnelle non paramétrique. Ces méthodes déterministes ont fait leur preuve surtout en modélisation mathématique, mais elles peuvent se révéler inefficaces dans certains contextes d'application où le problème exploré est souvent de complexité élevée (Robert [22]). Il s'est alors développé d'autres méthodes basées sur le paradigme de la théorie probabiliste utilisant la simulation informatique. Précisons que la simulation informatique de modèle consiste à considérer un modèle de la forme $Y = f(X)$, avec Y variable observée et X variable explicative. Pour simuler f, nous pouvons alors construire un estimateur pour X en utilisant la

probabilité conditionnelle $P(X|Y)$. Cet estimateur est de la forme :

$$\textbf{Moyenne :}\ x_m = E_{P(.|Y)}[X] = \int x\, P(X|Y)\, dx \tag{2.1}$$

ou

$$\textbf{Extremum :}\ x_M = \arg[\max_X(P(X|Y))] \tag{2.2}$$

On est alors capable d'estimer des modèles à partir de calculs numériques d'intégrales et/ou d'optimisation. Mais cette tâche étant souvent impossible à réaliser de façon déterministe, s'impose alors l'utilisation de méthodes probabilistes comme celles étudiées dans ce chapitre : les méthodes de Monte Carlo par chaînes de Markov. Dans la suite de ce chapitre, nous explorons la notion de chaînes de Markov, pour ensuite nous intéresser aux algorithmes probabilistes, en particulier aux méthodes de Monte Carlo par chaînes de Markov (MCMC), et enfin terminer par une introduction de leur application aux généalogies.

2.3 Chaînes de Markov

Nous présentons les chaînes de Markov de façon générale. Pour un exposé plus élaboré sur les chaînes de Markov, le lecteur est référé à Revuz [21], Kemeny and Snell [17], Chung [5], Freedman [10], Romanovskii [23] et Isaacson [15]. Pour la théorie des probabilités, le lecteur peut consulter Feller [9] et Doob [8] pour une étude des processus stochastiques. Nous nous concentrons sur les définitions, les propriétés et les différentes relations qui concernent les chaînes de Markov et surtout sur le concept de convergence dans le contexte des méthodes de Monte Carlo par chaînes de Markov. Aussi, il faut rappeler que nous ferons la différence entre chaînes de Markov à temps

continu et chaînes de Markov à temps discret. Dans ce mémoire, nous nous intéressons uniquement aux chaînes de Markov à temps discret, c'est-à-dire les chaînes de Markov dont l'ensemble des états E des systèmes étudiés est $\mathbb{N}$ (ensemble des entiers naturels) ou une partie finie de $\mathbb{N}$. Nous omettrons donc volontairement le terme « temps discret » dans la suite de ce document.

2.3.1 Définitions

Définition 2.6 *Une chaîne de Markov est une suite de variables aléatoires X_n (avec $n \in \mathbb{N}$) telle que pour tout $n \geq 1$, X_{n+1} dépend uniquement de X_n.*

L'expression suite de variables aléatoires fait ici référence à un processus stochastique, qui n'est rien d'autre que l'évolution dans le temps d'une variable aléatoire.
Notons $P(.|.)$, la distribution marginale reliée à une chaîne de Markov. On a ainsi :

$$P[X_{n+1} = j | X_0 = i_0; X_1 = i_1; ...; X_n = i_n] = P[X_{n+1} = j | X_n = i_n] \tag{2.3}$$

On dit alors que le processus est sans mémoire ou non héréditaire. C'est dire que l'état du processus au temps t ne dépend que du temps à l'instant $t-1$.
Notons p_{ij}, la probabilité, pour le système, de passer de l'état i à l'état j en un seul temps n. On parle de *probabilité de passage*. On aura alors :

$$p_{ij} = P[X_{n+1} = j | X_n = i_n] \tag{2.4}$$

On dira ainsi d'une chaîne de Markov qu'elle est *homogène*, c'est-à-dire qu'elle ne dépend pas du temps n représentant le temps mis pour passer d'une étape à une autre. On considère que ce temps est constant et le même quel que soit l'état dans lequel se trouve la chaîne.

Remarque 2.1 *L'étude des chaînes de Markov est souvent restreinte aux chaînes de Markov homogènes. Ainsi, nous ne parlerons pas des chaînes de Markov hétérogènes auxquelles les propriétés de convergence classique (que nous verrons plus loin) ne s'appliquent pas. Pour plus de détails voir Robert [22].*

Définition 2.7 *On appelle* matrice de passage ou noyau de transition ou noyau Markovien, *la matrice P dont les coefficients sont les probabilités de passage p_{ij}.*

$$P = \begin{bmatrix} p_{11} & p_{12} & p_{13} & \cdots & \cdots & p_{1j} \\ p_{21} & p_{22} & p_{23} & \cdots & \cdots & p_{2j} \\ p_{31} & p_{32} & p_{33} & \cdots & \cdots & p_{3j} \\ p_{41} & p_{42} & p_{43} & \cdots & \cdots & p_{4j} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ p_{i1} & p_{i2} & p_{i3} & \cdots & \cdots & p_{ij} \end{bmatrix}$$

On aura alors :

1. $\sum_{j \in \mathbb{N}} p_{ij} = 1$ et
2. $\forall (i,j) \in \mathbb{N}^2, p_{ij} \geq 0$.

P est donc une matrice stochastique.

Remarque 2.2 *Dans la suite du document, nous ne faisons pas de différence entre chaîne de Markov comme telle et noyau Markovien, même si en principe, la chaîne de Markov représente le processus et le noyau est représenté par la matrice P.*

En résumé, une chaîne de Markov évolue dans le temps. On a alors la définition suivante :

Définition 2.8 *On appelle la distribution (ou la loi) de la chaîne de Markov à l'instant t, le vecteur colonne* $q^{(t)} = (q_1^{(t)}, q_2^{(t)}, ..., q_n^{(t)})$ *tel que le* $i^{ième}$ *élément du vecteur représente la probabilité que la chaîne soit dans l'état i à l'instant t.*

2.3.2 Relation de Chapman-Kolmogorov et Loi stationnaire

La relation de Chapman-Kolmogorov relie la probabilité de passage en n étapes à la probabilité de passage en une seule étape. Considérons la chaîne de Markov homogène $(X_n)\,(n \geq 0)$ avec E l'ensemble des états et $P = (p_{ij})$ où $(i,j) \in E^2$ la matrice de passage. Posons :

$$P^{(n)} = P_{ij}^{(n)} = P\left[X_n = j | X_0 = i\right] \text{ avec } (i,j) \in E^2$$

On aura alors selon le théorème de Chapman-Kolmogorov :

$$\forall n \geq 0, P^{(n)} = (P)^n \tag{2.5}$$

La matrice de passage en n étapes est donc égale à la puissance $n^{\text{ième}}$ de la matrice de passage en une seule étape.

Notons :

- la loi de X_n par π_n ;
- avec π_0 la loi initiale lorsque $X_n = x_0$

À partir de la définition (de la loi de X), on a la loi marginale de X_n donnée par :

$$\pi_n = P[X_n]. \tag{2.6}$$

La distribution de probabilité à l'étape $n+1$ est alors donnée par :

$$\pi_{n+1} = \pi_n \, P \tag{2.7}$$

Définition 2.9 *On appelle* loi stationnaire de la chaîne de Markov *(ou du noyau markovien P), la loi π telle que :*

$$\pi = \pi \, P \tag{2.8}$$

On arrive alors à prouver facilement que si la chaîne suit la loi π à l'étape n, elle suit la même loi à l'étape $n+1$

2.3.3 Classes et irréductibilité

La propriété d'irréductibilité est l'une des plus importantes des chaînes de Markov dans le contexte des méthodes de Monte Carlo. En effet, cette propriété garantit la convergence des chaînes de Markov vers un état stationnaire (Robert [22]). Pour commencer, essayons de représenter une chaîne de Markov en remplaçant chaque état du processus par un sommet et chaque passage (transition) par un arc orienté. On aboutit donc à une représentation graphique de la chaîne de Markov appelée graphe orienté du processus (Figure 2.6). Les états d'une chaîne de Markov se répartissent en classes que l'on obtient à partir de la matrice de passage. Avant de donner une définition de la notion de classe, précisons certaines relations : (*accessibilité* et *communication*).

- On dit que l'état j est accessible à partir de l'état i, s'il existe un n tel que la probabilité de passage de i à j est non nul ($P_{ij}^{(n)} \geq 0$) ; On écrit alors $i \rightarrow j$ (à partir de la Figure 2.6, on a par exemple $E_1 \rightarrow E_2$).
- On dit aussi que deux états i et j communiquent, si on a : $i \rightarrow j$ et $j \rightarrow i$; On écrit alors $i \leftrightarrow j$ (à partir de la Figure 2.6, on a par exemple $E_1 \leftrightarrow E_3$). Ainsi la relation de communication est une relation d'équivalence.

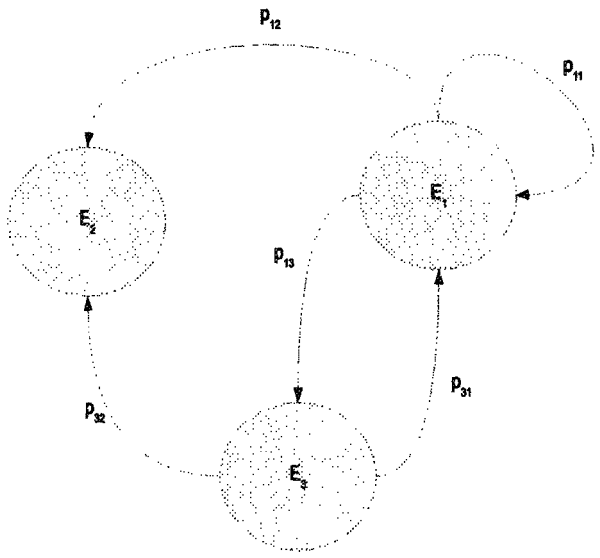


FIGURE 2.6 – Chaîne de Markov de 3 états : E_1, E_2 *et* E_3

Définition 2.10 Une classe *(aussi appelée classe d'équivalence) est une partition des états d'une chaîne de Markov, telle que les états d'une même classe communiquent entre eux et que deux états de classes différentes ne communiquent pas.*

On dit alors d'une classe qu'elle est :

- *récurrente*, s'il n'est pas possible d'en sortir (la classe E_2 de la Figure 2.6).
- *transiente*, s'il est possible d'en sortir, mais impossible d'y revenir.

Un *état est dit absorbant* lorsqu'il est le seul état d'une classe récurrente. Une *chaîne de Markov est dite irréductible* lorsqu'elle est composée d'une seule classe récurrente.

2.3.4 Périodicité

En règle générale, une chaîne apériodique garantit toujours l'irréductibilité de celle-ci (Robert [22]). Ainsi, le temps minimal donné pour revenir à un état i donné d'une chaîne de Markov est appelé *temps de retour* de l'état i.

Définition 2.11 *On appelle* période *de i, le p.g.c.d. de tous les entiers $n \geq 1$, tels que $p_{ii}^{(n)} > 0$.*

On note T_i la période de l'état i. On dit alors que :

1. l'état i est *périodique* de période T_i si $T_i > 1$;
2. l'état i est *apériodique* si $T_i = 1$

Si i est un *état de non retour* (donc dans un état récurrent) $\forall n \geq 1, p_{ii}^{(n)} = 0$, alors on pose $T_i = +\infty$. Ainsi, une chaîne de Markov (un noyau markovien), qui possède uniquement des états apériodiques, est dite *apériodique.*

2.3.5 Ergodicité et Convergence

La propriété d'ergodicité est la propriété qui va garantir la convergence d'une chaîne de Markov (Robert [22]). Nous avons les définitions suivantes :

Définition 2.12 *Un état apériodique et récurrent non-nul est dit ergodique.*

Définition 2.13 *Une chaîne de Markov est ergodique lorsque tous ses états sont ergodiques.*

La notion de convergence insiste alors sur le fait que, quelle que soit la loi initiale de la chaîne X_n, le noyau markovien converge toujours vers l'unique loi stationnaire π. À partir du théorème de Perron-Frobenuis, on arrive à donner l'équivalence suivante : *Une chaîne ergodique admet une unique loi stationnaire π vers laquelle elle converge quelle que soit sa distribution initiale π_0.*

Nous utilisons donc une chaîne de Markov ergodique dans ce mémoire pour la conception de nos algorithmes parce que celle-ci garantit toujours la convergence du noyau markovien vers la loi stationnaire (la loi d'intérêt). Nous verrons dans les sections suivantes comment, à partir de la propriété d'ergodicité d'une chaîne de Markov, nous arrivons à simuler la loi d'intérêt.

2.4 Les Méthodes de Monte Carlo par Chaînes de Markov

Dans cette partie, nous nous intéressons aux algorithmes probabilistes, plus particulièrement aux méthodes de Monte Carlo par Chaînes de Markov (MCMC). Nous aborderons les algorithmes probabilistes en général. Pour des études plus approfondies sur les algorithmes probabilistes, le lecteur peut consulter Motwani and Raghavan [20]. Pour des études sur les MCMC, le lecteur est référé à Robert [22] et Hammersley [13]. Nous nous basons sur ces ouvrages pour présenter cette partie. Après une brève description des algorithmes probabilistes, nous nous intéressons à deux algorithmes qui constituent les Méthodes de Monte Carlo par Chaînes de Markov : l'algorithme de Métopolis-Hasting (Metropolis et al. [19] et Hastings [14]) et l'échantillonnage de Gibbs (Geman and Geman [11]).

2.4.1 Les Algorithmes probabilistes

Un algorithme probabiliste est un algorithme qui fait usage d'un générateur de nombres aléatoires pour le traitement des données pendant son exécution. Il existe deux principales catégories d'algorithmes probabilistes : les algorithmes dits de *Las Vegas* et ceux de *Monte Carlo.*

Un algorithme est de type Las Vegas lorsqu'il retourne toujours une réponse exacte, avec un temps d'exécution, la plupart du temps, inconnu. La nature probabiliste du temps d'exécution d'un tel algorithme ne découle pas du fait que ce temps soit borné, mais plutôt de la garantie de l'existence d'une espérance mathématique finie de la complexité du temps d'exécution. Ceci assure alors une fin à ce genre d'algorithme. Un bon exemple d'utilisation d'algorithme de Las Vegas est son adaptation dans le problème du *Tri Rapide (Quick Sort).* En effet, la question primordiale dans ce genre de tri est le choix du pivot. Dépendamment de l'ordre des éléments du tableau (ou de la liste), le choix du pivot peut avoir une conséquence désastreuse sur le temps d'exécution de

l'algorithme. Par exemple, l'implémentation naïve d'un tel algorithme, qui consiste à choisir le premier élément du tableau comme pivot, va se révéler inefficace (par rapport à un tri classique) avec un tableau (ou une liste) partiellement trié. Ainsi, une solution à ce problème revient à choisir la médiane comme pivot (Brassard and Bratley [3]) ou à choisir le pivot de façon aléatoire (Motwani and Raghavan [20]). La première solution ajoute une complexité linéaire (la recherche de la médiane), alors que la seconde (choix aléatoire) se fait en temps constant. Cette dernière solution a pour avantage non seulement de rendre l'implémentation de cet algorithme plus simple mais aussi d'en améliorer considérablement les performances, surtout si les permutations des nombres du tableau en entrée sont équiprobables (voir Cormen et al. [6] pour plus de détails). On arrive alors à prouver, avec une grande probabilité, que le temps d'exécution d'un tel algorithme n'est pas plus élevé que son espérance (Motwani and Raghavan [20]). En effet, le temps d'exécution du tri rapide à pivot aléatoire (RandQS) est au plus $O(nlogn)$.

Un algorithme de Monte Carlo est, quant à lui, un algorithme probabiliste qui retourne une réponse exacte avec une certaine probabilité. Avec ce genre d'algorithme, le temps d'exécution est toujours borné mais l'exactitude de la réponse est incertaine. Le problème de la *Coupe Minimale*, illustré dans Motwani and Raghavan [20], résume bien l'utilisation de tels algorithmes. Soit G un graphe connexe non orienté de n nœuds. Une coupe de G est un ensemble d'arcs qui déconnectent le graphe lorsqu'ils sont retirés. Une coupe minimale est ainsi une coupe de cardinalité minimale. Ce problème peut se résoudre avec des algorithmes déterministes de calcul de flot. Toutefois, un algorithme très simple et efficace, donné dans Motwani and Raghavan [20], consiste à choisir une arête (un arc) de façon uniformément aléatoire et à contracter (fusionner) les 2 nœuds correspondants à chacune des extrémités, cela tant que le graphe contient plus que deux

nœuds. Cette solution se révèle simple d'implémentation et les auteurs ont prouvé que la probabilité que cet algorithme ne trouve pas de coupe minimale est exponentiellement petite. Cela veut donc dire que répéter cet algorithme plusieurs fois et ensuite choisir le minimum dans toutes les coupes minimales trouvées peut permettre de minimiser la probabilité d'erreur, caractéristique principale des algorithmes de types Monte Carlo. Comme nous l'avons signifié plus haut, le lecteur peut consulter Motwani and Raghavan [20], ouvrage dans lequel plusieurs algorithmes probabilistes sont abordés, avec de nombreux exemples très simples. Après une brève description des algorithmes probabilistes, intéressons-nous au cas particulier des MCMC.

Comme nous l'avons mentionné plus haut, il ne s'agit pas ici de démontrer l'intérêt ou l'utilité des MCMC. Le lecteur peut consulter Hammersley [13], Robert [22] Brooks [4] dans ce but. Nous étudions plutôt comment, à partir de l'utilisation de ces méthodes, nous pouvons simuler des chaînes de Markov convergeant vers une loi stationnaire. Commençons par donner la définition générale des MCMC.

Définition 2.14 *Un algorithme de type Monte Carlo par chaînes de Markov est une méthode simulant une chaîne de Markov ergodique (Robert [22]).*

Les méthodes MCMC vont non seulement permettre de construire des chaînes de Markov ergodiques, mais elles vont aussi garantir les propriétés nécessaires à la convergence du noyau markovien vers la loi stationnaire. Cette convergence est atteinte après un certain nombre d'itérations plus ou moins important (Belisle [2]). Nous verrons, dans un premier temps, l'algorithme de Métropolis-Hasting et ensuite, celui de l'échantillonnage de Gibbs. Rappelons que pour simuler la loi d'intérêt $Y = f(X)$, les MCMC utilisent la loi conditionnelle de X sachant Y. Alors, dans les sections suivantes, nous désignerons par $q(X|Y)$ cette loi conditionnelle.

2.4.2 Algorithme de Métropolis-Hasting

Rappelons que l'objectif est de construire une chaîne de Markov de loi stationnaire π (de loi d'intérêt f). L'algorithme de Métropolis-Hastings propose l'utilisation de la loi $q(Y|X)$ à certaines conditions :

- $q(.|X)$ doit être facilement simulable ;
- $q(.|X)$ est disponible de façon analytique ou symétrique $(q(Y|X) = q(X|Y))$;
- $q(.|X)$ doit approcher la loi d'intérêt f ;
- Le support de q doit inclure celui de f, puisque dans le cas contraire la chaîne $(X_n, n \in \mathbb{N})$ ne visitera pas certains états appartenant au support de f et absents de celui de q.

Le principe de cet algorithme consiste alors à générer une suite de variables aléatoires $(X^{(0)}, X^{(1)}, X^{(2)}, ..., X^{(i)}, ...)$ telle que $X^{(i+1)} = X^{(i)}$ avec probabilité ρ ou $X^{(i+1)} = \hat{x}^{(i)}$ avec probabilité 1-ρ où $\hat{x}^{(i)}$ suit $q(X|X^{(i-1)})$ et $\rho = \rho_{f,q}(X^{(t)}, \hat{x}^{(i)})$. On a ainsi l'algorithme donné par l'Algorithme 2.2.

Algorithme 2.2 Algorithme de Métropolis-Hastings

1. Initialiser $X^{(0)}$ aléatoirement
2. Produire $\hat{x}^{(0)} \sim q(X|X^{(i-1)})$
3. Accepter

$$X^{(i+1)} = \begin{cases} \hat{x}^{(t)} & \text{avec probabilité } \rho \text{ ;} \\ X^{(i)} & \text{sinon.} \end{cases}$$

où

$$\rho = \rho_{fg}(X^{(t)}, \hat{x}^{(i)}) = min\left\{\frac{f(\hat{x}^{(t)})}{f(X^{(t)})}\frac{q(X^{(t)}|\hat{x}^{(t)})}{q(\hat{x}^{(t)}|X^{(t)})}, 1\right\}$$

4. Itération de i : i ← i+1
5. Retourner à l'étape 2

Remarquons que la qualité du choix de la loi q(.|X) est primordiale pour garantir la convergence de la chaîne $(X_n, n \in \mathbb{N})$ vers la loi stationnaire π. Dans la plupart

des cas, restreindre q aux conditions établies plus haut peut s'avérer fastidieux (Robert [22]), plusieurs variantes de l'algorithme de Hastings-Métropolis ont alors été proposées. Dans les lignes qui suivent, nous donnons deux adaptations de l'algorithme de Hastings-Métropolis :

- L'algorithme de Hastings-Métropolis indépendant : dans ce cas, la loi q est indépendante de $X^{(i)}$. On a alors $q(\hat{x}^{(i)}|X) = q(\hat{x}^{(i)})$. L'Algorithme 2.3 se révèle alors plus efficace que l'Algorithme 2.2 (Robert [22]) ;
- L'algorithme de Métropolis-Hasting à marche aléatoire : cette technique se base sur le paradigme de la marche aléatoire et permet d'écrire $\hat{x}_t$ sous la forme $X_t + \varepsilon_t$. La procédure à suivre est donnée par l'Algorithme 2.4.

Algorithme 2.3 Algorithme de Métropolis-Hastings indépendant

1. Initialiser $X^{(0)}$ aléatoirement
2. Produire $\hat{x}^{(0)} \sim q(X)$
3. Accepter

$$X^{(i+1)} = \begin{cases} \hat{x}^{(t)} & \text{avec probabilité } \rho \text{ ;} \\ X^{(i)} & \text{sinon.} \end{cases}$$

où

$$\rho = \rho_{fg}(X^{(t)}, \hat{x}^{(i)}) = min\left\{ \frac{f(\hat{x}^{(t)})}{f(X^{(t)})} \frac{q(X^{(t)})}{q(\hat{x}^{(t)})}, 1 \right\}$$

4. Itération de i : i $\leftarrow$ i+1
5. Retourner à l'étape 2

2.4.3 L'échantillonnage de Gibbs

L'échantillonnage de Gibbs, contrairement à celui de Métropolis-Hasting, cherche à reproduire la loi d'intérêt f de manière automatique, donc le plus fidèlement possible, en utilisant les propriétés de cette loi à un degré beaucoup plus évolué (Robert [22]). L'échantillonnage de Gibbs est alors un cas particulier de l'algorithme de Métropolis-Hasting. La différence découle de l'ajout de deux principes. Le premier principe résulte

Algorithme 2.4 Algorithme de Métropolis-Hastings à marche aléatoire

1. Initialiser $X^{(0)}$ aléatoirement
2. Produire $\hat{x}^{(0)} \sim q(X - X^{(t)})$
3. Accepter

$$X^{(i+1)} = \begin{cases} \hat{x}^{(t)} & \text{avec probabilité } \rho\ ; \\ X^{(i)} & \text{sinon.} \end{cases}$$

où

$$\rho = \rho_{fg}(X^{(t)}, \hat{x}^{(i)}) = min\left\{\frac{f(\hat{x}^{(t)})}{f(X^{(i)})}, 1\right\}$$

4. Itération de i : i ← i+1
5. Retourner à l'étape 2

du fait que le taux d'acceptation des variables simulées est égal à 1. Ainsi, contrairement à l'algorithme de Métropolis-Hasting, l'échantillonnage de Gibbs accepte toutes les variables simulées. Le deuxième principe repose sur une connaissance préalable de la loi d'intérêt f.

Le principe général de l'échantillonnage de Gibbs est donc le suivant : si la chaîne (X_n) peut se décomposer en $(X_0, X_1, ..., X_n)$, et que $\pi_i(i = 1 \text{ à } n)$ est connu et simulable, alors nous sommes en mesure de générer une suite de variables aléatoires $(X_i^{(t)})$ telle que $X_{i\neq j}^{(t+1)} \sim \pi_i(X_i|X_{j>i}^{(t)}, X_{j<i}^{(t+1)})$. Nous avons alors l'Algorithme 2.5.

Algorithme 2.5 Algorithme de l'échantillonnage de Gibbs

1. Initialiser avec $X^{(0)} = (X_1^{(0)}, X_2^{(0)}, ..., X_n^{(0)})$ aléatoirement
2. Produire
   - $X_1^{(t+1)} \sim \pi_1(X_1|X_2^{(t)}, X_3^{(t)}, ..., X_n^{(t)})$
   - $X_2^{(t+1)} \sim \pi_2(X_2|X_1^{(t+1)}, X_3^{(t)}, ..., X_n^{(t)})$
   - ... ...
   - ... ...
   - $X_n^{(t+1)} \sim \pi_n(X_n|X_1^{(t+1)}, X_2^{(t+1)}, ..., X_{n-1}^{(t+1)})$
3. Itération de i : i ← i+1
4. Retourner à l'étape 2

L'échantillonnage de Gibbs est principalement fondé sur la connaissance des lois conditionnelles. L'un des avantages de cette méthode est l'utilisation de la propriété de réversibilité des chaînes de Markov qui permet de renforcer les propriétés de convergence vers la loi stationnaire (Robert [22]). Une chaîne de Markov est réversible si pour tout $k \geq 1$, la loi de $(X_0, X_1, ..., X_k)$ est égale à la loi de $(X_k, X_{k-1}, ..., X_1, X_0)$. Deux variantes de cet algorithme permettent de garantir cette propriété. Il s'agit de l'échantillonnage de Gibbs à balayage symétrique (algorithme 2.6) et de celui à balayage aléatoire (algorithme 2.7). L'échantillonnage de Gibbs à balayage aléatoire est une amélioration de celui à balayage symétrique; ainsi la simulation de X_n se fait de façon aléatoire.

Algorithme 2.6 Algorithme de l'échantillonnage de Gibbs à balayage symétrique

1. Initialiser avec $X^{(0)} = (X_1^{(0)}, X_2^{(0)}, ..., X_n^{(0)})$ aléatoirement
2. Produire
   - $X_1^* \sim \pi_1(X_1 | X_2^{(t)}, X_3^{(t)}, ..., X_n^{(t)})$
   - $X_2^* \sim \pi_2(X_2 | X_1^*, X_3^{(t)}, ..., X_n^{(t)})$
   - ... ...
   - ... ...
   - $X_{n-1}^* \sim \pi_{n-1}(X_{n-1} | X_1^*, X_2^*, ..., X_{n-2}^*, X_n^{(t)})$
   - $X_n^{(t+1)} \sim \pi_n(X_n | X_1^*, X_2^*, ..., X_{n-2}^*, X_{n-1}^*)$
   - $X_{n-1}^{(t+1)} \sim \pi_{n-1}(X_{n-1} | X_1^*, X_2^*, ..., X_{n-2}^*, X_n^{(t+1)})$
   - ...
   - $X_1^{(t+1)} \sim \pi_1(X_1 | X_2^{(t+1)}, X_3^{(t+1)}, ..., X_n^{(t+1)})$
3. Itération de i : i $\leftarrow$ i+1
4. Retourner à l'étape 2

Dans les sections précédentes, nous avons montré comment à partir des méthodes de Monte Carlo, nous pouvons simuler des chaînes de Markov ergodiques. Nous remarquons cependant que certaines méthodes, en particulier les algorithmes concernant l'échantillonnage de Gibbs, sont beaucoup plus simples d'implémentation (simulation à partir de la loi à posteriori) et puissantes (Robert [22]). En effet, l'échantillonnage

Algorithme 2.7 Algorithme de l'échantillonnage de Gibbs à balayage aléatoire

1. Initialiser avec $X^{(0)} = (X_1^{(0)}, X_2^{(0)}, ..., X_n^{(0)})$ aléatoirement
2. Produire une permutation k des états de transition de n éléments
3. Produire
   - $X_{k_1}^{(t+1)} \sim \pi_{k_1}(X_{k_1}|X_j^{(t)}, j \neq k_1)$
   - ... ...
   - ... ...
   - $X_{k_n}^{(t+1)} \sim \pi_{k_n}(X_{k_n}|X_j^{(t+1)}, j \neq k_n)$
4. Itération de i : i $\leftarrow$ i+1
5. Retourner à l'étape 2

de Gibbs renforce certaines propriétés des chaînes de Markov (comme la propriété de réversibilité), ce qui permet d'en garantir la convergence, avec un taux d'erreur faible, vers une loi stationnaire (la loi d'intérêt). Dans la section suivante, nous verrons alors pourquoi et comment ces méthodes peuvent être adaptées à des analyses sur des généalogies.

2.4.4 Applications des MCMC aux généalogies

Les MCMC ont fait leur preuve dans différents contextes (en inférence bayésienne, en analyse numérique, ...). L'une des questions que nous nous posons alors, concerne l'utilisation de telles méthodes dans le concept de la reconstitution des génotypes traité dans la première partie de notre étude. Comment ces méthodes peuvent-elles aider à reconstituer, à partir des informations de liens familiaux, l'ensemble des génotypes des individus composant une généalogie profonde ? Nous tentons de répondre à cette question dans les lignes qui suivent.

Plusieurs auteurs ont appliqués les MCMC en génétique. Citons, par exemple Lange [18] et Geyer and Thompson [12]. Dans ce mémoire, nous cherchons à déterminer les génotypes des individus d'une généalogie profonde. Dans les sections précédentes, nous avons montré que la reconstitution déterministe des génotypes est quasiment im-

possible pour une généalogie s'étendant sur plusieurs générations. Nous utilisons alors les MCMC pour obtenir une distribution de probabilités pour le génotype de chacun des individus de la généalogie.

Dénotons un système quelconque par : $(X_n, n \in \mathbb{N})$ tel que $X_i = (X_{i_1}, X_{i_2}, ..., X_{i_p})$, avec $i = 1$ à n, où X_{i_k} représente le génotype de la $k^{\text{ième}}$ personne dans une généalogie de p individus. Rappelons que l'état d'une chaîne de Markov au temps t_i dépend uniquement de son état au temps t_{i-1}. Ainsi, pour obtenir une distribution de probabilité pour le génotype de chaque individu, il faut déterminer le génotype de chacun des individus au temps t_i. Déterminer le génotype de l'individu k revient à choisir de façon aléatoire ce génotype parmi un ensemble de possibilités relativement à la position de l'individu dans la généalogie. En effet, le génotype d'un individu dépend de ses voisins immédiats, à savoir ses enfants et ses parents (père et mère). Cela revient alors à simuler $X_{i_k}^{(t+1)}$ selon une loi à posteriori, donc une loi conditionnelle $\pi_{i_k}(X_{i_k}|X_j^{(t+1)}, j \neq i_k)$. Or, cette loi représente la probabilité d'avoir X_{i_k} comme génotype pour l'individu k sachant que les autres individus de la généalogie ont comme génotype $X_j^{(t+1)}, j \neq i_k$. Notons que les $j \neq i_k$ peuvent se restreindre aux individus voisins, c'est-à-dire sa parenté directe (ses partenaires et ses enfants, ses frères et sœurs et enfin ses deux parents). Cela est principalement dû au fait que le génotype d'un individu est indirectement lié aux génotypes de ses voisins éloignés par translation de ses voisins proches. Nous remarquons alors un cas d'application des méthodes de Monte Carlo par chaînes de Markov. En plus, ayant une bonne connaissance de la loi d'intérêt π, l'utilisation de la méthode de l'échantillonnage de Gibbs se révèle plus intéressante que celle de Métropolis-Hastings. Pour matérialiser nos idées, nous définissons le concept de probabilité liée aux généalogies par un exemple. Considérons un gène donné composé de deux allèles, avec l'ensemble génotypique suivant :

- Génotype 1 : AA (État 1)
- Génotype 2 : Aa (État 2)
- Génotype 3 : aa (État 3)

Le génotype d'un individu peut donc être dans un des trois états précédents. On appelle π_{i_k}, la probabilité que le génotype de l'individu k soit dans l'état i. C'est dire que $\pi_{i_k} = Pr[X_k = i]$. Par conséquent, on aura :

$$\sum_{i=1}^{3} \pi_{i_k} = 1, \ \forall \ k \in \{1, ..., p\} \tag{2.9}$$

Les mesures qui nous intéressent pour chacun des individus sont :

1. π_{1_k} : probabilité d'avoir AA comme génotype
2. π_{2_k} : probabilité d'avoir Aa comme génotype
3. π_{3_k} : probabilité d'avoir aa comme génotype

Cela nous permet alors de construire une chaîne de Markov à partir des lois conditionnelles $\pi_k(X_k|X_j(j \neq k))$. Appliquons l'échantillonnage de Gibbs de façon générale à ce cas. Nous avons comme entrée une généalogie composée de p individus dont s proposants (le génotype de ces individus étant connu). On initialise $X^{(0)} = (X_1, X_2, ..., X_p)$ en attribuant le génotype Aa à tous les individus excepté les proposants. On remarquera que ce choix est judicieux, puisque le croisement de deux parents ayant chacun comme génotype Aa (le génotype 2) conduit à des enfants de génotypes AA, Aa ou aa (donc tous les cas possibles). On dira alors que le modèle est dans un *état compatible.* Un état compatible est une condition nécessaire, puisqu'il garantit la non nullité de la loi conditionnelle. La chaîne de Markov du modèle est donc irréductible.

Définition 2.15 *Un état compatible pour un modèle de généalogie est un état dans lequel :*

1. *Tous les individus de la généalogie ont un génotype ;*
2. *Pour tout individu k, il y a compatibilité entre son génotype et les génotypes de ses voisins proches.*

Le modèle que nous avons ici est donc dans un état compatible, tous les individus ayant le génotype 2 (excepté les proposants).

Remarque 2.3 *On remarque qu'il n'y a pas d'état compatible unique pour une généalogie.*

On décide ensuite du génotype de chaque individu à partir des lois $\pi_k(X_k|X_j(j \neq k))$. Une simulation consiste alors à choisir de façon aléatoire le génotype de tous les individus successivement. Nous reproduisons cet effet un certain nombre de fois, ce qui conduit à reproduire la loi d'intérêt (propriétés de MCMC). Mais combien de simulations devrons-nous effectuer pour simuler la loi d'intérêt ? Nous introduisons trois mesures qui nous permettent de calibrer les paramètres du modèle afin de respecter les règles d'application des MCMC. Le premier paramètre est relié à la loi stationnaire. La question posée est la suivante : à partir de combien de simulations, le modèle se trouve t-il dans un état stationnaire ?

Définition 2.16 *On appelle état stationnaire, l'état dans lequel le modèle simule la loi d'intérêt.*

Les paramètres des simulations sont liés entre eux et une valeur raisonnable pour chacun est essentielle pour obtenir des valeurs réalistes. Il faut ainsi déterminer le nombre de simulations qui permet de passer d'un état quelconque, souvent hors du champ des états réalistes, à un état réaliste dans l'ensemble des états stables. Par la suite, il faut déterminer le nombre de simulations pour passer d'un état stable à un autre état stable indépendant du dernier. Il est évident qu'un état stable légèrement modifié conduit à

un état légèrement différent mais qui dépend du dernier état stable. Une répétition du processus de simulation devrait permettre de passer d'un état stable à un autre état stable sans qu'il y ait une dépendance suffisante pour influer sur les résultats. C'est le deuxième paramètre.

Définition 2.17 *Un état indépendant pour un modèle de généalogie, est un état stationnaire qui a été perturbé par simulation.*

Paramètre	Description	État du modèle espéré
b	nombre de simulations pour obtenir un état stationnaire	État stationnaire
m	nombre de simulations pour obtenir un état indépendant	État stationnaire et indépendant
n	nombre de blocs de m simulations pour obtenir une meilleure précision	Précision

TABLEAU 2.1 – Tableau récapitulatif des paramètres du modèle

Le troisième paramètre concerne la précision des inférences sur le modèle. Le fait est que le modèle dans un état stationnaire donne une estimation avec une certaine erreur (propriété propre au MCMC). Pour avoir une estimation plus précise, nous allons appliquer le théorème central limite. Ainsi, le modèle étant dans un état stationnaire, nous observons les mesures et nous recommençons les simulations pour observer un autre état stationnaire. Ce processus est répété un certain nombre de fois dépendamment de la précision désirée. Les mesures introduites sont alors consignées dans le Tableau 2.1. La première colonne désigne le paramètre, la deuxième sa description et le troisième l'état désiré du modèle. Ainsi, ces paramètres permettent de calibrer notre modèle afin d'assurer d'une part l'existence d'une loi stationnaire et, d'autre part, d'en garantir la convergence vers cette loi. Il faut aussi tenir compte de la vitesse de convergence vers

la loi stationnaire, puisque selon la méthode utilisée cette vitesse aura un impact sur l'efficacité de nos algorithmes. Plusieurs questions méritent alors d'être posées :

Quelles sont les valeurs des paramètres b, m et n ?
Quelle version de l'échantillonnage de Gibbs utiliser ?
Comment implémenter un tel algorithme ?

Voila là, des questions auxquelles nous chercherons à répondre dans les chapitres qui suivent.

Chapitre 3

Algorithmes, Architecture et Optimisation

La théorie sur les MCMC a permis de définir les méthodes pertinentes pour résoudre la problématique de notre étude. Nous estimons que la méthode de l'échantillonnage de Gibbs est plus appropriée dans la problématique de la reconstitution génotypique en ce qui concerne des généalogies profondes. Néanmoins, le cadre d'application de ces méthodes reste à être défini au regard des objectifs de la recherche. Nous définissons, dans cette partie, les algorithmes, l'environnement dans lequel ils sont conçus et ensuite leur optimisation.

3.1 Algorithmes

Les algorithmes doivent permettre d'effectuer les simulations de Monte Carlo. L'objectif premier étant d'estimer les génotypes des individus d'une généalogie profonde, nous devons avant tout simuler la transmission des gènes à l'intérieur de la généalogie. Comme nous l'avons mentionné dans l'introduction, nous nous baserons sur le modèle mendélien pour simuler cette transmission. Pour cela, nous définissons le concept de *modèle génétique* dans les pages suivantes.

3.1.1 Modèle génétique

La transmission des gènes basée sur le modèle mendélien simule de façon naturelle la transmission des gènes de parents à enfants. Nous réduisons le concept de gène

à celui de génotype pour faciliter la simulation. Un individu enfant a donc un génotype (gène) composé de deux allèles. Un provenant de son père et l'autre de sa mère. Chacun des individus parents a un génotype et transmet de façon aléatoire un allèle à un enfant. Notons alors E, P et M, les génotypes respectifs de l'enfant, de son père et de sa mère. On a :

- $P- > P_1P_2$ (le génotype P est composé des allèles P_1 et P_2)
- $M- > M_1M_2$ (le génotype M est composé des allèles M_1 et M_2)

La distribution des probabilités pour le génotype de l'enfant est donnée au Tableau 3.1. Nous obtenons ces résultats en supposant que le choix des allèles des parents est équiprobable.

Possibilité pour E	Probabilité
P_1M_1	1/4
P_1M_2	1/4
P_2M_1	1/4
P_2M_2	1/4

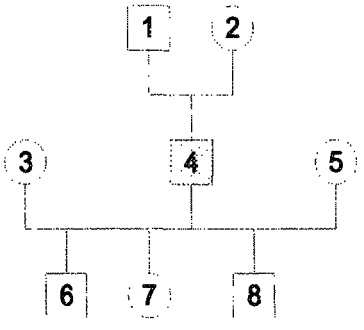
TABLEAU 3.1 – Distribution des probabilités pour le génotype de l'enfant

Remarque 3.1 *Notons que l'ordre de parution des allèles ne compte pas. Par exemple, les génotypes P_1M_1 et M_1P_1 sont les mêmes.*

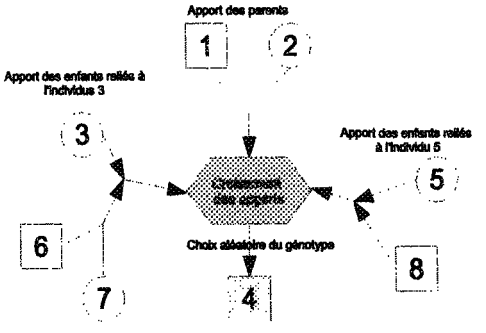
Déterminer le génotype d'un individu revient ainsi à choisir un génotype de façon aléatoire parmi les différentes combinaisons de taille 2, formées d'un allèle du père et d'un allèle de la mère. Notons quand même que cette situation est la plus simple, lorsqu'il s'agit de choisir le génotype d'un individu (généalogie de deux parents et d'un ou plusieurs enfants). En réalité, le choix du génotype est plus complexe. En effet, un individu, en plus d'avoir des parents, a non seulement plusieurs enfants, mais aussi

peut avoir plusieurs partenaires (avec qui il a eu ses enfants). Le choix du génotype de cet individu dépend alors des génotypes de ses parents, mais aussi de ceux de ses enfants et de ses différents partenaires. Un exemple est donné à la Figure 3.1, qui illustre la complexité du choix du génotype de l'individu 4 en fonction de celui des voisins immédiats (ses parents, ses partenaires et ses enfants). L'individu 4 a eu 2 partenaires, les individus 3 et 5. Pour déterminer le génotype de cet individu, il faudra, d'une part, ternir compte de l'apport génétique de ses parents, mais aussi considérer son apport génétique aux enfants qu'il a eu avec ses 2 partenaires. Cela garantira alors la compatibilité de la généalogie. Ainsi, le génotype de l'individu 4 est déterminé par l'intersection des apports génétiques reliés aux 3 groupes d'individus suivants :

- Ses parents ;
- Groupe composé du partenaire de gauche (individu 3) et de la descendance (les enfants : individus 6 et 7) ;
- Groupe composé du partenaire de droite (individu 5) et de la descendance (les enfants : individu 8).



(a) Identification de l'individu 4



(b) Choix du génotype de l'individu 4 sachant le génotype de ses voisins proches

FIGURE 3.1 – Complexité du choix du génotype

3.1.2 Simulation du génotype d'un individu

La simulation du génotype d'un individu se fait par étape. La première étape consiste à déduire des génotypes des parents, les différentes formes (possibilités) que peut prendre le génotype de l'individu. Le résultat est confiné dans un tableau $T_1[\ ]$. La deuxième étape consiste à déterminer les possibilités du génotype de l'individu en tenant compte des combinaisons partenaires-enfants, cela pour chacun d'entre eux. Ces résultats sont contenus dans un tableau $T_2[\ ]$. À la troisième étape, un croisement (l'intersection) est effectué entre $T_1[\ ]$ et $T_2[\ ]$ pour aboutir à un tableau $T[\ ]$. Enfin, la dernière étape revient à déterminer le génotype de l'individu de façon aléatoire parmi $T[\ ]$. Ainsi, chaque génotype à une probabilité d'être choisi égale à sa proportion d'apparition. L'algorithme 3.1 en illustre le fonctionnement.

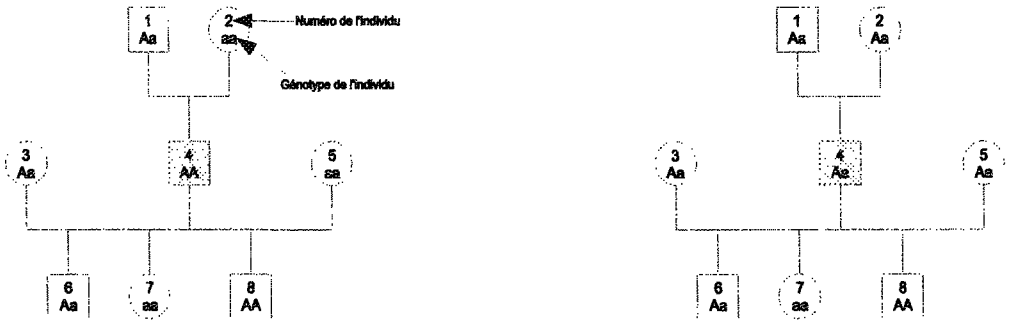
Algorithme 3.1 Simulation du génotype d'un individu

Préconditions: identification de l'individu
Postconditions: le génotype de l'individu
$T_1[\ ] \leftarrow$ les génotypes possibles en fonction des parents
$T_2[\ ] \leftarrow$ les génotypes possibles en fonction des partenaires et enfants
$T[\ ] \leftarrow$ On fait l'intersection de $T_1[\ ]$ et de $T_2[\ ]$
Réajuster les probabilités d'être choisi pour chacun des génotypes
Choisir le génotype de l'individu
Retourner $T[i]$.

3.1.3 État compatible

Cette étape consiste à trouver un état compatible pour la généalogie à partir du génotype des proposants (condition à priori). La première approche consiste, comme nous l'avons fait dans le chapitre précédent, à attribuer le génotype 2 (Aa) à tous les individus de la généalogie. La Figure 3.2 illustre un exemple d'état compatible dans le cas particulier d'un gène à deux allèles (différents). En effet, un gène à deux allèle peut être formé à partir de plus que deux allèles. Donc, au lieu d'avoir uniquement

deux allèles A et a, nous pouvons avoir trois allèles (ou plus) identifiés par les lettres A, B et C par exemple. Dans ce cas là, la méthode utilisée dans cette approche ne fonctionne pas. Nous proposons plutôt une approche qui peut être utilisée dans tous les cas possibles.



(a) État non compatible : l'individu 4 ne peut être de ce génotype.

(b) État compatible : les génotypes de tous les individus sont initialisés à Aa.

FIGURE 3.2 – Exemple d'état compatible

La deuxième approche utilisée consiste à parcourir l'arbre de généalogie du bas vers le haut (des enfants vers les parents). Les proposants ayant leur génotype déjà connu, on commence par déterminer le génotype des parents. Ensuite, on détermine les génotypes des parents de ces derniers et ainsi de suite, jusqu'à ce que tous les individus de la généalogie aient un génotype. Cependant, un problème existe avec l'utilisation de cette méthode déterministe. Un individu peut appartenir à plusieurs générations en même temps. Ainsi, il existe des situations pour lesquelles il faudra déterminer le génotype de cet individu avant celui de ses enfants. Cette situation vient alors contredire la priorité avec laquelle on détermine les génotypes des individus, c'est-à-dire enfant d'abord et parent ensuite. L'utilisation d'une telle méthode produit la plupart du temps

une généalogie incompatible. Le problème ne se poserait pas si tous les arbres de généalogie étaient des arbres conventionnels dans lesquels un nœud appartient à une seule génération. Nous utilisons plutôt une approche probabiliste simulant la transmission des gènes sur le postulat du modèle mendélienne.

Avec cette approche, nous commençons par attribuer un génotype à chacun des fondateurs. Ensuite, les génotypes de leurs enfants sont déterminés, puis ceux de leurs petits-enfants et ainsi de suite (même les proposants se font attribuer de nouveaux génotypes). Le processus s'arrête lorsque tous les individus ont un génotype. Nous procédons ensuite à des simulations sur la généalogie, en utilisant l'algorithme 3.1, pour changer le génotype de tous les individus de la généalogie. Ce processus est alors répété jusqu'à ce que tous les proposants aient le génotype souhaité (c'est-à-dire leur génotype réel). L'état dans lequel la généalogie se trouve à la fin des simulations est donc un état compatible. Le seul problème avec cette méthode est le temps d'exécution qui reste inconnu et aléatoire. Si nous le remarquons bien, cet algorithme est un algorithme de Las Vegas :

- Il ne retourne pas toujours une réponse ;
- Le temps d'exécution est aléatoire ;
- La solution qu'il retourne est toujours exacte, lorsqu'il en retourne une.

L'Algorithme 3.2 donne un exemple. Lors des tests effectués, cet algorithme a donné de très bonnes performances. Il a toujours retourné une solution dans des temps acceptables, de quelques secondes pour une petite généalogie de moins de dix générations, à quelques minutes pour une grosse généalogie s'étendant sur des dizaines de générations.

Algorithme 3.2 État compatible

Préconditions: généalogie indiquant le génotype des proposants
Postconditions: généalogie dans un état compatible
tant que tous les proposants n'ont pas le génotype désiré **faire**
Attribuer un génotype à tous les individu commençant par les fondateurs, leurs enfants ensuite, et ainsi de suite
fin tant que
Retourner la généalogie compatible.

3.1.4 Échantillonnage de Gibbs

Ainsi, après avoir déterminé un état compatible pour la généalogie, nous pouvons réaliser les simulations de Monte Carlo. En effet, à partir de l'état compatible obtenu à la section précédente, la méthode de l'échantillonnage de Gibbs, illustrée par l'Algorithme 3.3, utilise l'Algorithme 3.1 (choix du génotype) pour effectuer plusieurs simulations basées sur les probabilités à priori. Ainsi, simuler une généalogie équivaut à changer les génotypes de tous les individus de la généalogie.

Algorithme 3.3 Échantillonnage de Gibbs adapté aux généalogies

Préconditions: généalogie dans un état compatible
Préconditions: b
Préconditions: m
Préconditions: n
Postconditions: Matrice des probabilités
pour i allant de 1 à b **faire**
On simule la généalogie
fin pour
pour i allant de 1 à n **faire**
pour j allant de 1 à m **faire**
On simule la généalogie
fin pour
On note le génotype des individus
fin pour
5 Retourner la matrice des probabilités.

3.2 Architecture logicielle

La plupart des analyses effectuées sur les résultats des algorithmes conçus sont des analyses statistiques. Nous aurons alors besoin d'un environnement dans lequel ces analyses pourront être effectuées. Nous utilisons une plateforme Unix (Solaris 8). Les simulations probabilistes sont assez gourmandes en temps CPU, par conséquent elle peuvent prendre un temps d'exécution important (de l'ordre de quelques jours). Les algorithmes ont une complexité qui limite leur conception avec un logiciel d'analyse statistique, nous utilisons donc un langage de programmation qui nous permet de concevoir ces algorithmes. Nous utilisons alors comme langage de programmation le C++ (avec le compilateur g++) et le logiciel R comme outil d'analyse statistique. La Figure 3.3 illustre le fonctionnement du système. Les entrées et sorties sont effectuées à l'aide des procédures en R. Ces procédures sont responsables d'effectuer les appels aux fonctions (algorithmes) en C++. Les paramètres sont passés par pointeurs, des procédures aux fonctions (cela permet de garantir leur intégrité). Après traitement, les fonctions retournent les résultats aux procédures qui les ont appelées. Les résultats sont alors disponibles pour être analysés. Les opérations sont exécutées en mode commande, un projet futur pourrait améliorer la présentation de l'écran utilisateur, pour permettre une plus grande transparence.

3.3 Optimisation

Dans la section précédente, nous avons défini les bases de nos algorithmes en matérialisant leur structure générale. Il nous incombe maintenant de définir la stratégie de leur mise en oeuvre. Cependant, cette définition ne peut se faire sans une étude préalable des structures de données à utiliser, mais aussi des différentes améliorations à apporter aux algorithmes par optimisation. Ces différentes étapes sont alors traitées

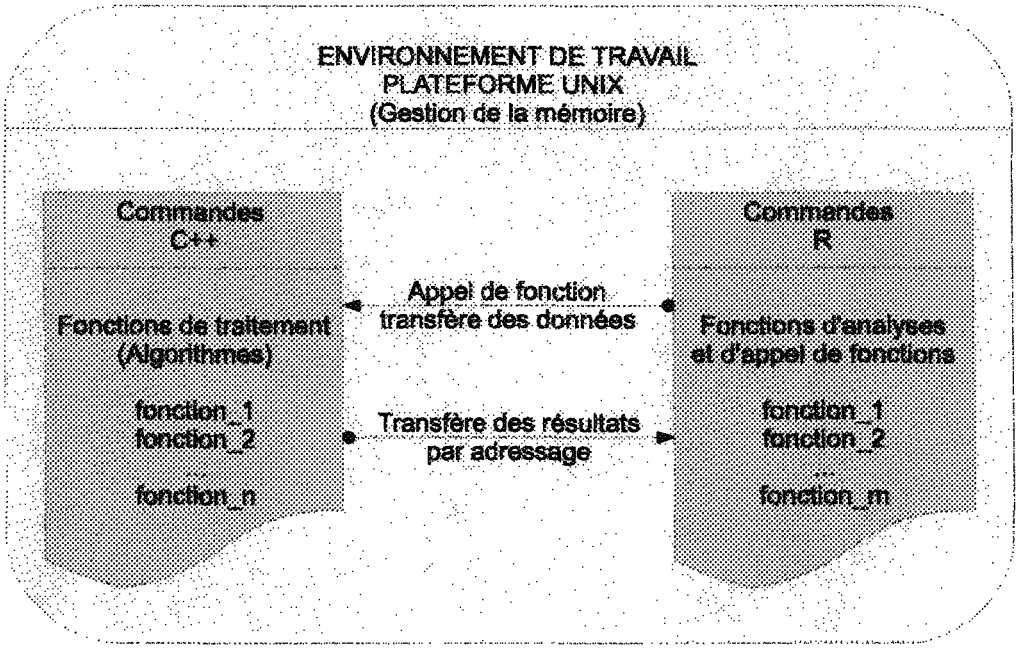


FIGURE 3.3 – Environnement de travail

dans cette section.

3.3.1 Structures de données

Les structures de données définies ici correspondent aux classes d'objets utilisées, permettant de réaliser les simulations. Nous retrouvons alors cinq classes : *CList, ProbaSimul, CindSimul, Geno_Type, Gen.* La Figure 3.4 donne un aperçu de l'environnement objet défini comme suit :

1. *CList* - cette classe représente la liste des enfants d'un individu (une liste chaînée) :
   - CList * next : pointeur vers le nœud suivant ;
   - CindSimul * nœud : pointeur vers le nœud courant.
2. *ProbaSimul* - classe représentant la probabilité d'avoir un génotype donné :
   - double probabilité1 : probabilité d'avoir un génotype donné pour la simulation courante ;
   - double probabilité2 : probabilité d'avoir un génotype donné pour l'ensemble des simulations déjà effectuées.
3. *CindSimul* - classe représentant un individu :
   - long numero : numéro de l'individu ;
   - CindSimul * père : pointeur vers le père (individu) de l'individu ;
   - CindSimul * mère : pointeur vers la mère (individu) de l'individu ;
   - CList * enfant : pointeur vers la liste des enfants de l'individu (Nul sinon) ;
   - string sexe : sexe de l'individu ;
   - string allele : génotype de l'individu ;
   - string copyallele : copie du génotype, permettant d'effectuer les simulations ;
   - int statut : flag permettant de définir le statut de l'individu dépendamment du fait que son génotype peut être changé ou pas ;
   - ProbaSimul * Proba : tableau des probabilités pour chacun des génotypes.

4. *Geno_Type* - classe représentant un génotype :
   - string gene : codage du génotype ;
   - double poids : poids (probabilité) du génotype.
5. *Gen* - classe représentant une généalogie :
   - int nb_individu : nombre d'individus de la généalogie ;
   - int nb_genotype : nombre de génotypes possibles ;
   - CindSimul * Cind : tableau des individus ;
   - Geno_Type * Geno : tableau des génotypes.

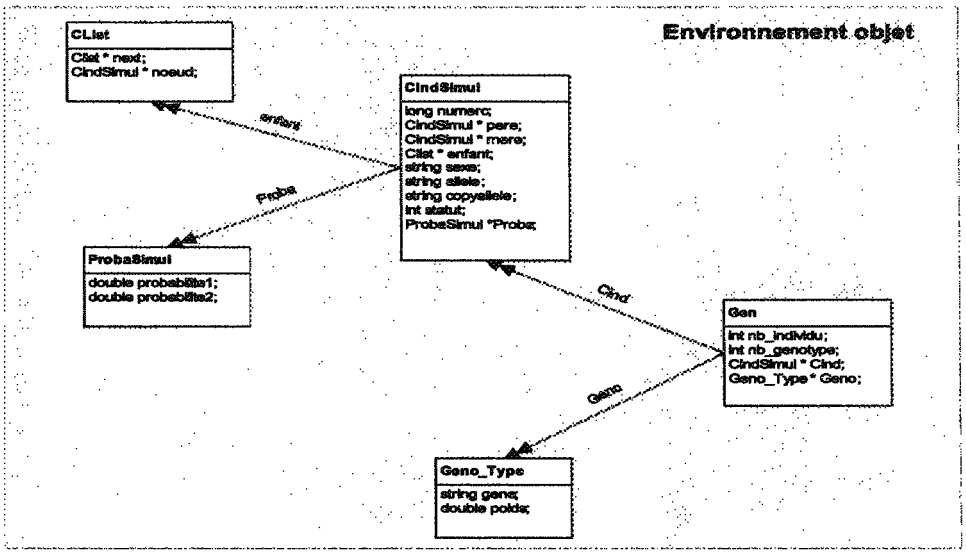


FIGURE 3.4 – Modèle des classes objets

3.3.2 Optimisation probabiliste

Une amélioration est faite à la version de l'algorithme de Gibbs utilisée. En effet, comme nous l'avons défini dans la section sur les MCMC (précisément dans la partie concernant l'échantillonnage de Gibbs), une des versions de l'algorithme de Gibbs

permet de renforcer certaines propriétés des chaînes de Markov. Pour cela, nous utilisons la version de cet algorithme garantissant la réversibilité du noyau markovien. La version qui correspond à cet algorithme est l'échantillonnage de Gibbs à balayage aléatoire. En effet, pour simuler une chaîne de Markov ergodique, dans le cas des généalogies profondes, nous choisissons d'abord un individu dans la généalogie. Ensuite, nous déterminons aléatoirement et changeons le génotype de cet individu. Puis, des simulations sont réalisées et l'ordre dans lequel nous choisissons les individus devant se faire perturber (détermination aléatoire et changement du génotype) est le même pour chaque simulation. Cette façon de faire permet de garantir l'ergodicité de la chaîne de Markov. Cependant, si nous voulons renforcer la propriété d'ergodicité et par le même fait garantir la réversibilité de la chaîne de Markov, l'ordre dans lequel les individus sont choisis doit être différent à chaque simulation, c'est-à-dire aléatoire. L'adaptation de la version aléatoire de l'algorithme de Gibbs amène à apporter une modification à notre algorithme. À chaque début de simulation, il faudra produire une permutation des n individus de la généalogie. La nouvelle version de l'algorithme est donnée par l'Algorithme 3.4.

Algorithme 3.4 Échantillonnage de Gibbs à balayage aléatoire adapté aux pedigree

Préconditions: généalogie dans un état compatible
Préconditions: b
Préconditions: m
Préconditions: n
Postconditions: Matrice des probabilités

1 - On simule b fois la généalogie;
2 - On génère une permutation des n individus de la généalogie;
3 - On simule m fois la généalogie;
4 - On note le génotype des individus;
5 - On refait les étapes 2, 3 et 4 n-1 fois;
6 - Retourner la matrice des probabilités.

3.3.3 Optimisation informatique

La deuxième amélioration apportée découle de l'optimisation probabiliste présentée plus haut. En effet, la première version de notre algorithme (non amélioré) traite les individus dans l'ordre dans lequel ils étaient classés dans le tableau d'individus (CindSimul * Cind). Donc, le choix du prochain individu à traiter se fait en incrémentant le rang de l'individu courant. Cependant, avec l'amélioration apportée (algorithme 3.4), le prochain individu à traiter a un rang quelconque (aléatoire). Ainsi, accéder au prochain individu à traiter se fait en temps linéaire, puisqu'il faut parcourir tout le tableau afin de le trouver. Il faut donc pouvoir trouver une astuce pour accéder à cet individu en temps constant. Nous avons alors opté pour une indexation de la généalogie. Ainsi, à la création de la généalogie, nous créons une nouvelle version indexée de cette généalogie. Les numéros des individus sont alors remplacés par les numéros de l'index et par conséquent accéder à un individu se fait en un temps constant.

Le cadre d'application des MCMC étant défini dans ce chapitre, il nous revient de mettre en application les théories énoncées plus haut. Nous avons démontré que les MCMC sont bien adaptées à l'analyse de données généalogiques. Nous avons aussi montré comment ces méthodes pouvaient être utilisées en pratique avec des généalogies profondes. Dans le chapitre suivant, nous nous concentrons à les mettre en pratique. Cela permettra dans un premier temps de calibrer les paramètres du modèle à partir de petites généalogies conçues à cet effet. Dans un second temps, nous validons notre approche en appliquant les simulations à des généalogies plus grandes et comparer les résultats obtenus avec ceux des petites généalogies. Enfin, nous terminerons le chapitre par l'applications des algorithmes à des données provenant d'une généalogie réelle.

Chapitre 4

Ajustements et Application

Dans un premier temps, nous allons calibrer les paramètres utilisés dans le modèle : b, m et n (définis plus haut). Pour cela, nous utilisons deux généalogies théoriques. Une généalogie de 30 individus (nous en donnons un aperçu à la Figure 4.1) et une autre de 95 individus consistant en 30 proposants dont le génotype est connu par rapport à un allèle (Fichier présenté à l'Annexe A). Nous fixons des contraintes à ces généalogies et nous procédons à la simulation, pour ensuite observer les paramètres. Notons que le paramètre n ne nécessite pas d'étude, puisque c'est le paramètre de précision. Plus il est grand, plus les estimations sont précises. Nous mettons donc l'accent sur les paramètres b et m.

Dans un second temps, nous essayons de valider notre approche en comparant les calculs théoriques de probabilités et les résultats des simulations sur une généalogie de sept individus (on comprendra que les calculs théoriques à effectuer sont assez laborieux, raison pour laquelle nous considérons ici une généalogie très simple).

Enfin, nous testons nos algorithmes sur une généalogie réelle et nous effectuons des analyses sur cette généalogie, à partir des résultats de simulations.

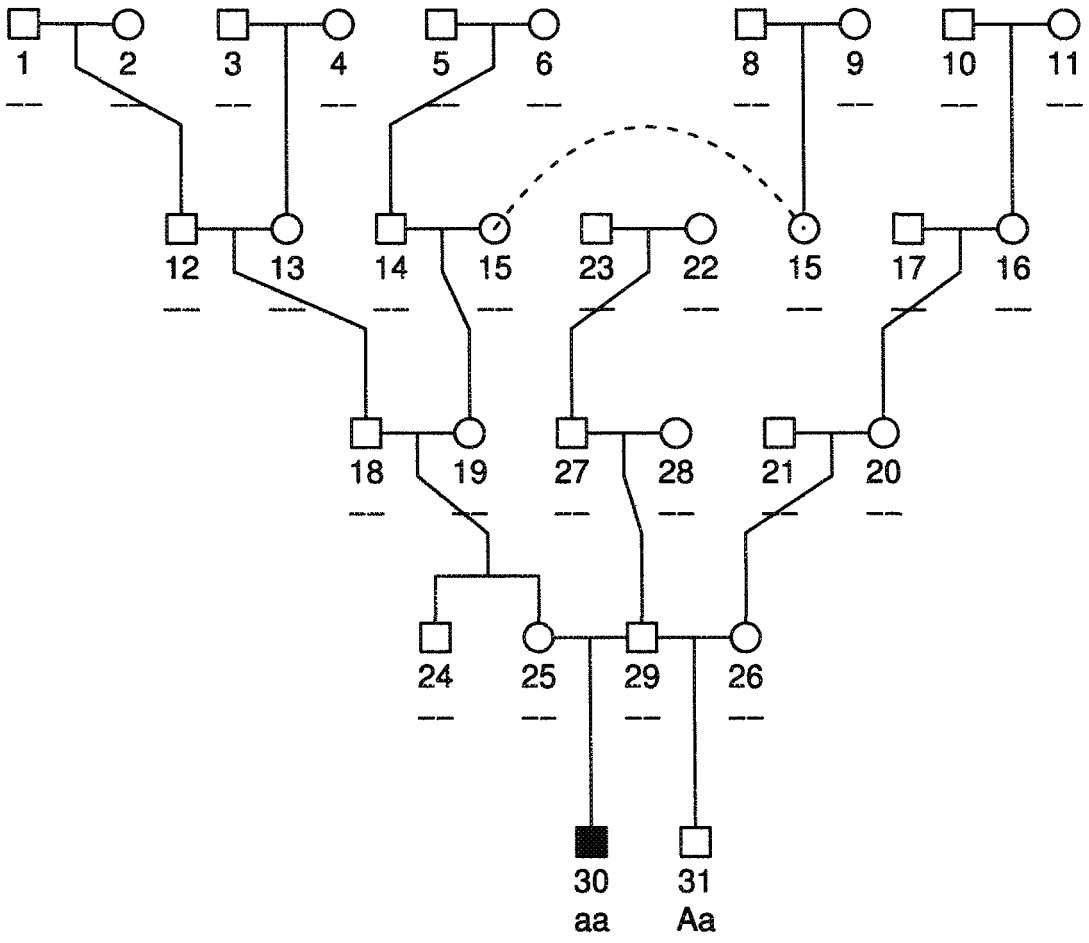


FIGURE 4.1 – Généalogie de 30 individus

4.1 Ajustement des paramètres

4.1.1 Principe retenu

Le principe d'ajustement est simple ; des simulations sont effectuées jusqu'à ce que toutes simulations supplémentaires ne change pas fondamentalement les résultats. Pour déterminer si les résultats sont changés, un ou plusieurs ancêtres sont ciblés en fonction de leur importance dans la généalogie. Le nombre de simulations augmentent alors jusqu'à ce que les paramètres d'intérêt se stabilisent. Cela permet de déterminer les valeurs optimales de ces paramètres. Ainsi, le principe de l'ajustement suit l'algorithme suivant :

Algorithme 4.1 Algorithme d'ajustement des paramètres

1. Définir des génotypes
2. Effectuer un certain nombre de simulations
3. Observation des génotypes des cibles
4. Estimation de la probabilité des génotypes en fonction du nombre de simulations

Certaines probabilités sont plus importantes que d'autres. Ainsi, un fondateur qui est à 2 générations d'un proposant ne peut pas avoir introduit un allèle pour tous les proposants. La calibration a été effectuée en choisissant un individu qui a une importance stratégique du point de vue de sa contribution au patrimoine de l'ensemble des proposants. Ainsi :

- Pour la généalogie de 30 individus, les individus 30 et 31 sont utilisés comme références pour la simulation au niveau des porteurs connus et l'individu 22 est utilisé comme standard. Cet dernier est assez profond par rapport aux autres et les 2 premiers sont représentatifs d'un apparentement réaliste pour la population québécoise ;
- Pour la généalogie des 95 individus, les individus 92, 93, 94 et 95 sont utilisés

comme références pour les individus identifiés comme porteurs tandis que, l'individu 72 est utilisé comme standard pour l'estimation des génotypes. Cela permet d'obtenir une bonne appréciation des valeurs des deux paramètres des simulations pour obtenir une estimation stable des probabilités selon les génotypes des ancêtres ;

- Le génotype est basé sur l'identification d'un allèle en particulier. Cela se traduit par les 3 génotypes suivants : génotype 1 (AA), génotype 2 (Aa) et génotype 3 (aa) ;
- Les génotypes des proposants sont identifiés et fixés au préalable ;
- Nous effectuerons 250 simulations ;

4.1.2 Description des généalogies

Ainsi pour chacune des généalogies utilisées, nous avons établi les paramètres suivants :

1. Généalogie de 30 individus : (voir Figure 4.1)
   - Les proposants sont les individus 30 et 31 ayant pour génotype respectif (fixé) aa et Aa ;
   - les observations seront effectuées sur l'individu 22 ;
2. Généalogie de 95 individus : (voir l'annexe A)
   - Les proposants sont les individus 92, 93, 94 et 95 ayant pour génotype respectif (fixé) Aa, Aa, aa et Aa ;
   - les observations seront effectuées sur l'individu 72 ;

4.1.3 Résultats

Les résultats sont la probabilité empirique que l'individu ciblé soit porteur hétérozygote (deux allèles différents) selon l'allèle ciblé. La précision de cet estimateur

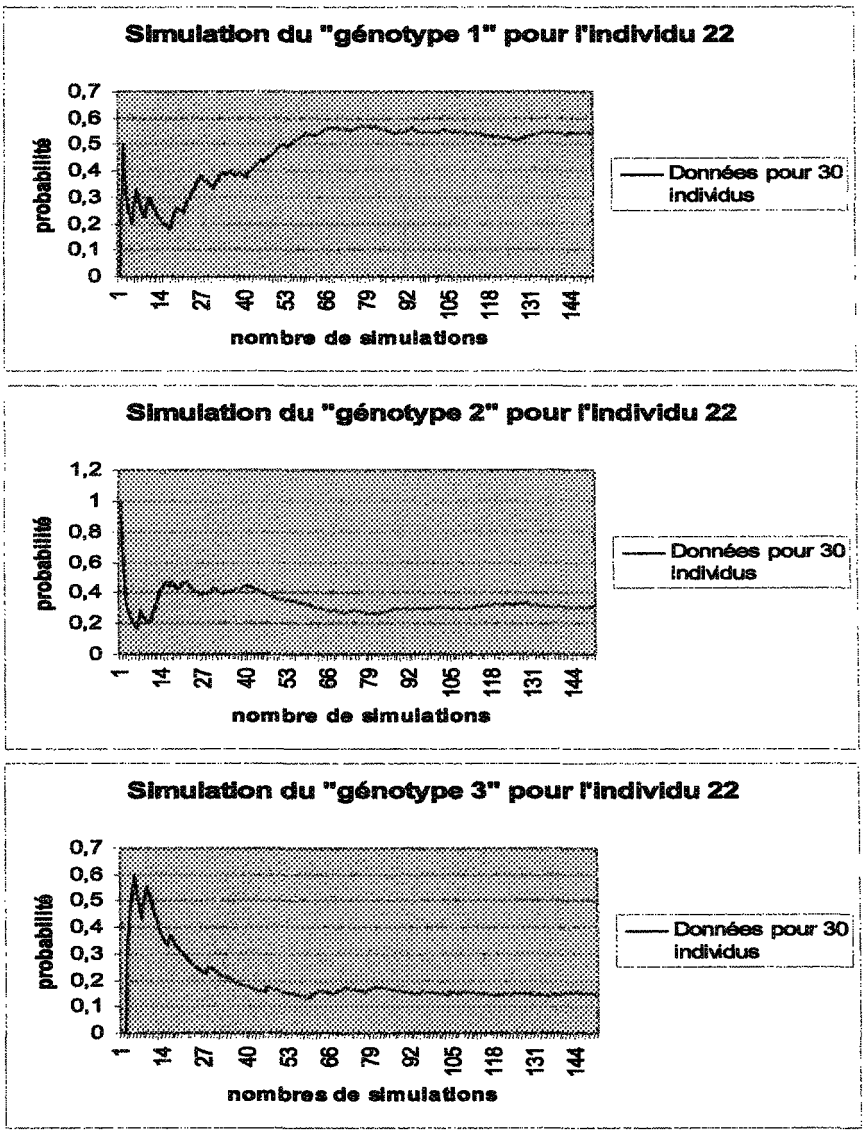


FIGURE 4.2 – Estimation des génotypes de l'individu 22 pour une généalogie de 30 individus

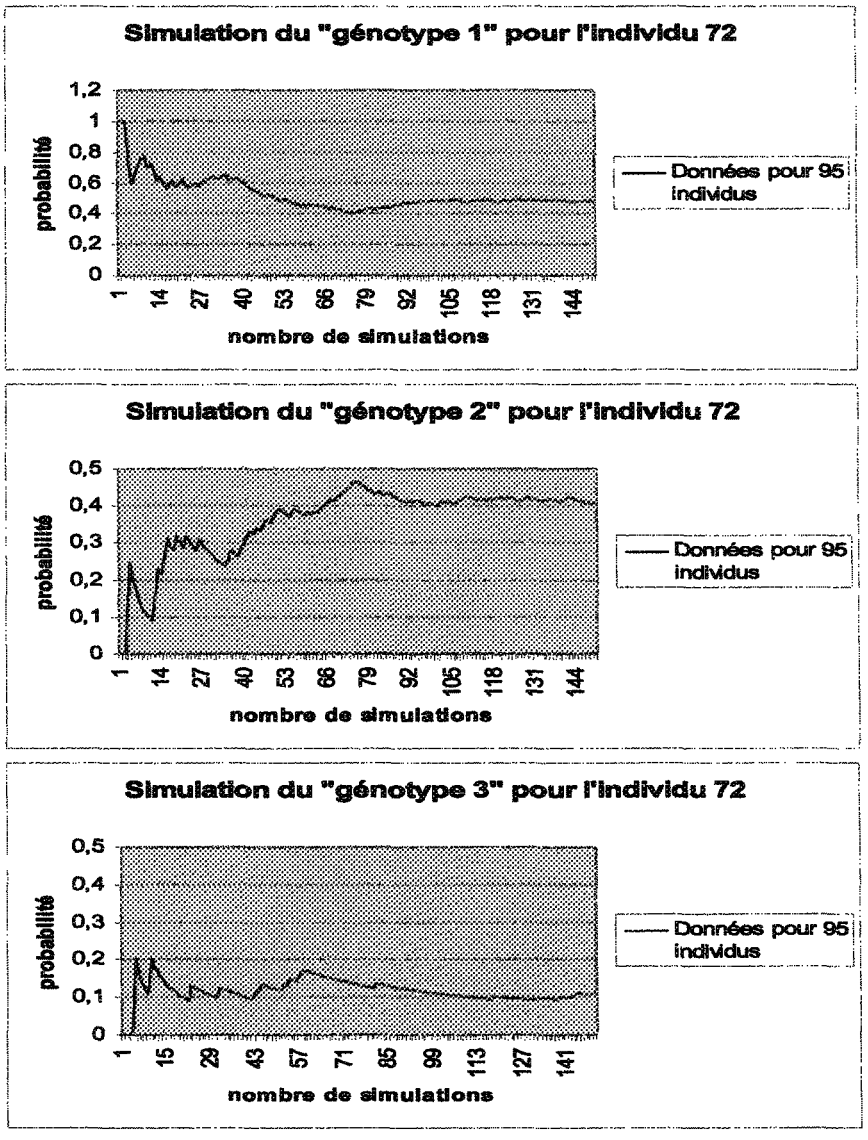


FIGURE 4.3 – Estimation des génotypes de l'individu 72 pour une généalogie de 95 individus

dépend du nombre de simulations effectuées en considérant un modèle binomial pour les simulations tandis que, l'adéquation du modèle dépend des valeurs de n et m pour assurer qu'il y a indépendance. Le paramètre est assez grand si les estimations sont stables. Cela veut dire que l'augmentation du nombre de simulations ne change pas les résultats. Il y aura nécessairement une fluctuation entre les différentes valeurs des paramètres mais elles sont considérées comme normales, c'est-à-dire en accord avec les théorèmes de convergence sur les suites de variables aléatoires. Un graphique des estimations en fonction du nombre de répétition permet d'obtenir une valeur raisonnable pour les différents paramètres. Ces graphiques sont présentés aux Figures 4.2 et 4.3, pour respectivement les populations de 30 et 95 individus.

Nous remarquons alors que le système a tendance à se stabiliser pour $k \in [50; 100]$ simulations, tant pour la généalogie de 30 individus que celle de 95 individus. Ainsi, on peut penser que dans cet intervalle, le système passe d'un état indépendant à un état stable. On pourra donc considérer la valeur de $b = 50$ comme fiable, quand à la garantie de la propriété d'indépendance de notre modèle et pour une valeur de $m = 100$ minimum afin de garantir la stabilité du modèle. Ainsi dans les simulations, pour garantir les propriétés d'indépendance et de stabilisation du modèle, il faut :

- $k \geq 50$;
- $m \geq 100$.

4.2 Validation théorique

Une première validation consiste à comparer les valeurs obtenues par simulations par rapport aux valeurs théoriques. Dans le cas d'une généalogie très petite, les probabilités conditionnelles exactes sont faciles à évaluer à l'aide de la règle de Bayes

étant donné deux évènements A_i et A_j (i, j, k_1 et k_2 entiers) :

$$P(A_i = k_1 | A_j = k_2) = \frac{P(A_j = k_2 | A_i = k_1) P(A_i = k_1)}{P(A_j = k_2)}$$

Il est alors possible de comparer les valeurs obtenues par simulation par rapport aux valeurs exactes. Encore une fois, il est difficile de faire une comparaison de toutes les possibilités et ainsi, une seule configuration sera retenu pour la généalogie retenue. Étant donné la complexité des calculs en jeu, seul la généalogie la plus simple sera retenue. Nous considérons donc une généalogie très simple de 7 individus présentée à la Figure 4.4. Cette figure représente la généalogie étudiée avec le génotype du proposant retenu et l'identification de l'individu retenu. Nous identifions alors un individu à observer (ici l'individu 5) ; nous relèverons ensuite les résultats produits par simulation (relativement à cet individu) que nous comparerons aux résultats réels (calculs algébriques).

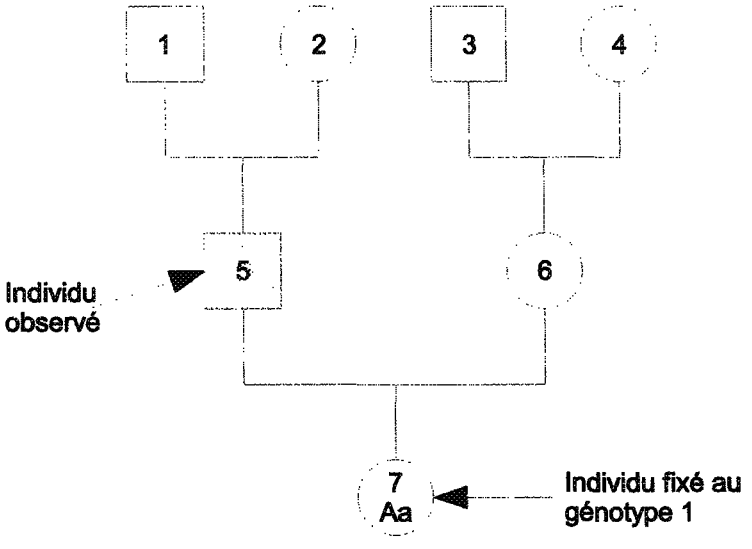


FIGURE 4.4 – Généalogie de 7 individus

Avant de commencer les calculs, commençons par poser certaines hypothèses et

effectuer quelques calculs préliminaires dont nous aurons besoin tout au long de notre démarche.

Paramètres initiaux :

- Nos observations sont réalisées sur un gène composé de 2 allèles, soient AA, Aa et aa. L'allèle que nous voulons observer est l'allèle a. Nous notons ainsi :

  AA = génotype 0 ;

  Aa = génotype 1 ;

  aa = génotype 2 ;

- la généalogie étant simple, nous avons fixé le génotype d'un individu, l'individu se trouvant en fin de liste. Ce génotype est alors initialisé et fixé au génotype 1 ;
- Soit A_i l'événement « avoir l'allèle de type a pour l'individu i ». Nous avons :
  1. P($A_i = 0$) -> l'individu i n'a pas d'allèle de type a ;
  2. P($A_i = 1$) -> l'individu i a 1 allèle de type a ;
  3. P($A_i = 2$) -> l'individu i a 2 allèles de type a ;
- Pour les fondateurs, nous donnerons : $p_0 = 1/2$; $p_1 = 1/4$; $p_2 = 1/4$, avec :

  p_0 : probabilité d'avoir 0 allèle de type a ;

  p_1 : probabilité d'avoir 1 allèle de type a ;

  p_2 : probabilité d'avoir 2 allèles de type a ;

Calculs préliminaires :

Partant de la structure d'une famille (père-mère-enfants : illustrée à la Figure 4.5) pour créer une généalogie, nous calculons différentes probabilités relatives à l'individu « enfant » (individu 3), puisque les probabilités relatives aux parents sont déjà connues (probabilités attribuées au fondateurs).

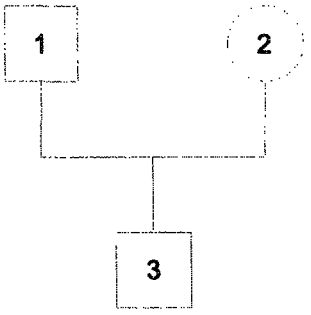


FIGURE 4.5 – Structure d'une famille simple

Ainsi, à partir du théorème de Bayes et en conditionnant les calculs relativement au génotype du père, nous avons les résultats suivants (les résultats auraient été les mêmes si nous avions conditionné les calculs relativement au génotype de la mère). Nous donnons juste les résultats et ramenons le lecteur à l'annexe B pour le détail des calculs.

On a ainsi :

$P(A_3 = 0) = 43/93 \approx 0.4479166667$;

$P(A_3 = 1) = 79/192 \approx 0.4114583333$;

$P(A_3 = 2) = 9/64 \approx 0.1406250000$;

On remarque alors que :

$P(A_3 = 0) + P(A_3 = 1) + P(A_3 = 2) = 43/96 + 79/192 + 9/64 = 1$. Les hypothèses et les calculs préliminaires étant établis, nous pouvons passer aux calculs désirés. Les probabilités qui nous intéressent sont alors :

- $P(A_5 = 0|A_7 = 1)$: probabilité que l'individu 5 ait le génotype 0 sachant que l'individu 7 a le génotype 1 ;
- $P(A_5 = 1|A_7 = 1)$: probabilité que l'individu 5 ait le génotype 1 sachant que l'individu 7 a le génotype 1 ;

- $P(A_5 = 2|A_7 = 1)$: probabilité que l'individu 5 ait le génotype 2 sachant que l'individu 7 a le génotype 1.

Calculs théoriques : Les calculs théoriques sont présentés à l'annexe C.

Résultats :

1. *Calculs Théoriques :*
   - $P(A5 = 0|A7 = 1) = P(A6 = 0|A7 = 1) \approx 0.3527556$
   - $P(A5 = 1|A7 = 1) = P(A6 = 1|A7 = 1) \approx 0.4909553$
   - $P(A5 = 2|A7 = 1) = P(A6 = 2|A7 = 1) \approx 0.1562891$
2. *Calculs par simulation avec* $n = 10000$ *;* $m = 100$ *et* $b = 100$ *:*
   - $P^*(A_5 = 0|A_7 = 1) = 0.360125$
   - $P^*(A_5 = 1|A_7 = 1) = 0.485224$
   - $P^*(A_5 = 0|A_7 = 1) = 0.154657$
3. *Erreur absolue (différence entre calcul théorique et calcul de simulation en valeur absolue) :*
   - E_0

$$= |P(A5 = 0|A7 = 1) - P^*(A5 = 0|A7 = 1)|$$

$$= |0.352756 - 0.360125|$$

$$= 0.007369$$

   - E_1

$$= |P(A5 = 1|A7 = 1) - P * (A5 = 1|A7 = 1)|$$

$$= |0.490955 - 0.485224|$$

$$= 0.005731$$

- E_2

$$= |P(A5 = 2|A7 = 1) - P*(A5 = 2|A7 = 1)|$$

$$= |0.156289 - 0.154657|$$

$$= 0.001632$$

Conclusion : L'erreur absolue dans le calcul des 3 probabilités est très faible, de l'ordre de 0.5%. Cela correspond à la précision espérée pour un modèle binomial avec 10000 répétitions et ainsi nous pouvons supposer que les calculs effectués par simulation reflètent assez bien les calculs réels. Les simulations peuvent donc donner des résultats avec une précision d'environ 3 chiffres significatifs.

4.3 Application aux données réelles

Les liens généalogiques sont réels mais les génotypes ne sont pas disponibles, c'est le but de ce travail de donner une estimation de ces génotypes. L'idée est de simuler un trait génétique, à partir des algorithmes obtenus, selon une configuration donnée et d'en évaluer la performance. Pour évaluer les performances des méthodes, nous devons disposer d'une généalogie dont le génotype de chacun des individus est connu. Cela est impensable pour une généalogie profonde. Cependant, il est assez facile par la méthode du « gene dropping » (Évelyne Heyer [27]) de simuler la transmission d'un caractère génétique selon les lois de Mendel pour obtenir une répartition « réaliste » des allèles dans l'ensemble de la population. Nous proposons alors de considérer une généalogie profonde et de créer une répartition des génotypes pour cet ensemble par la méthode du « gene dropping ». Il est alors possible de comparer les résultats de MCMC avec les valeurs obtenues. Nous appliquons ainsi les algorithmes à une généalogie réelle anonymisée provenant de la base de données du projet Balzac. Elle est constitué de 30 proposants, 145326 ancêtres dont 8319 distincts. On dispose alors d'une généalogie

de 8349 individus au total. Pour extraire la généalogie de la base de donnée, nous introduisons un trait quelconque. Nous choisissons au hasard un ancêtre suffisamment influent, mais pas trop, pour générer au moins 12 hétérozygotes dans la population et après quelques essais, une généalogie virtuelle est créée par la méthode du « gene dropping ». Cela mène à 14 porteurs sur 30 proposants. L'étude sera menée sur cette généalogie dont les individus et les liens sont réels mais dont les génotypes sont virtuels. Cela permettra de comparer les résultats des MCMC à des résultats réels.

4.3.1 Caractéristique de la généalogie

Les caractéristiques générales de la généalogie sont les suivants :

- Nombre d'individus total : 8349 ;
- Nombre de fondateurs : 2109 ;
- Nombre de proposants : 30 ;
- Nombre d'ancêtres : 145326 dont 8319 distincts (8349 - 30) ;
- Nombre de générations : 15.

Aussi, à partir de la généalogie, nous procédons à des analyses démogénétiques illustrées aux Figures 4.6 et 4.7. La Figure 4.6 donne la densité de la profondeur généalogique. On remarque bien une répartition adéquate des individus (entre les générations 7 et 12). La généalogie correspond alors à une généalogie profonde pour laquelle les individus sont repartis convenablement. La Figure 4.7 vient confirmer cette remarque. Elle donne la complétude par profondeur (exprimée en pourcentage d'individus).

4.3.2 Les simulations

Chaque simulation donne des génotypes compatibles avec les observations. Après n simulations on a pour chaque individu dans la généalogie, le nombre de simulations pour lesquelles l'individu était porteur d'une copie de l'allèle d'intérêt, porteur

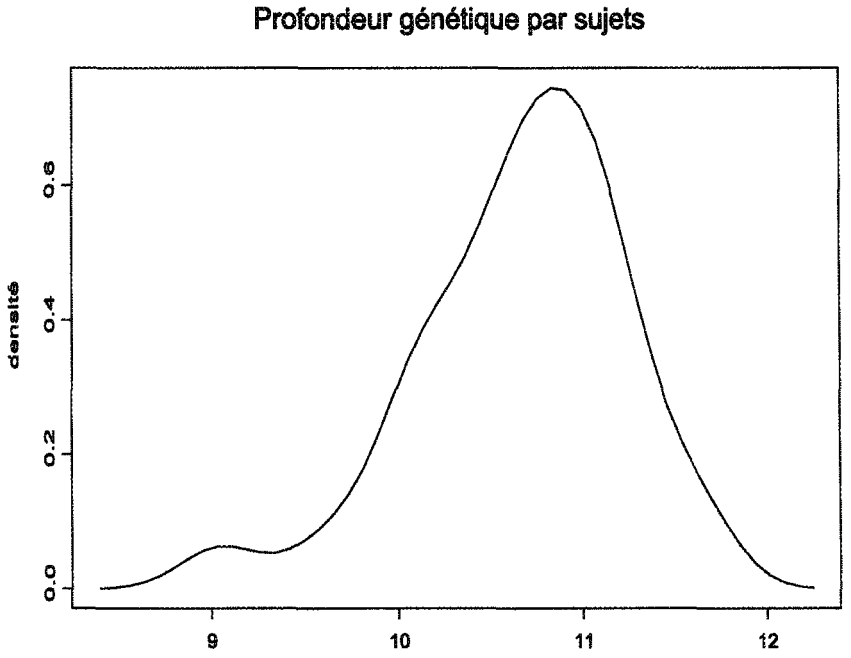


FIGURE 4.6 – Profondeur par sujets (généalogie réelle).

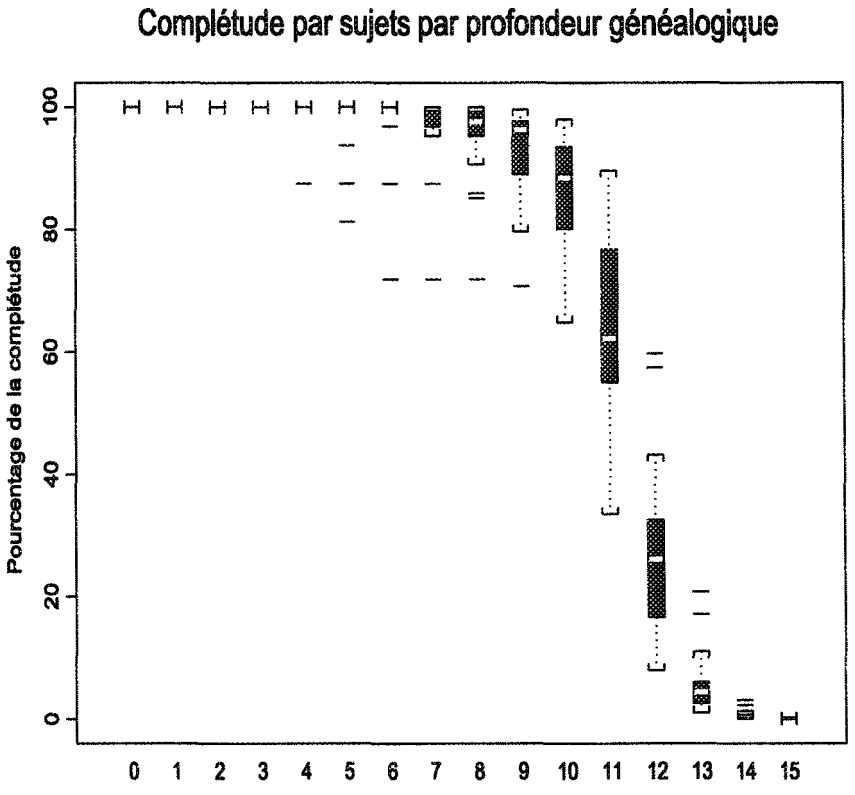


FIGURE 4.7 – Complétude par sujets par profondeur (généalogie réelle).

de deux copies ou non porteur. On a donc pour chaque individu une estimation de la probabilité que l'individu soit non porteur, porteur d'un allèle ou porteur de deux allèles. Cela veut dire que, pour chaque individu et pour chaque simulation, le génotype est simulé selon :

- Génotype 0 : noté « 00 » (non porteur) ;
- Génotype 1 : noté « 01 » (une copie de l'allèle d'intérêt) ;
- Génotype 2 : noté « 11 » (deux copie de l'allèle d'intérêt).

Comme nous l'avons signifié plus haut, nous introduisons une trait quelconque. Ce trait sera caractérisé par le génotype « 01 ». Nous fixons alors le génotype de l'individu 74052 au génotype « 01 », et nous simulons ensuite une propagation. Nous obtenons ainsi 14 proposants porteurs du génotype « 01 » sur 30. Nous appliquons enfin l'échantillonnage de Gibbs à la généalogie, avec les paramètres tels que déterminés plus haut, c'est-à-dire :

- b = 100 ;
- m = 100 ;
- n = 1000 (nombre de simulations).

4.3.3 Résultats

Dans la généalogie, il y a 8319 ancêtres distincts dont on a l'estimation de la probabilité d'être porteur suite aux simulations et les génotypes réels ou de référence (ceux obtenu par la méthode du « gene dropping »). L'objectif est de comparer les simulations et les génotypes réels. Nous retenons deux indices d'adéquation : la distribution des probabilités d'être porteur selon le génotype de référence et l'indice kappa d'association entre les génotypes de référence et les génotypes obtenus par simulation. En effet, les simulations permettent d'obtenir le nombre de génotypes « 00 », « 01 » et « 11 » pour chaque individu par simulation. Une compilation de ces résultats donne une estimation de la probabilité des génotypes « 00 », « 01 » et « 11 » pour chacun

des individus de la généalogie. Aussi, la probabilité d'homozygotes relativement à l'allèle d'intérêt est très faible et donc non informative pour un nombre de simulations raisonnables, c'est pourquoi les résultats sont analysés sur la base de la probabilité de porteur et de non porteur ou si on préfère la probabilité d'être porteur puisque l'autre probabilité est le complémentaire.

Ainsi, dans un premier temps le nombre de porteurs chez les individus dont le génotype connu est « 00 » est comparé au nombre de non porteurs chez les individus dont le génotype connu n'est pas « 00 ». Or, il est intuitivement plus difficile d'obtenir des informations sur les ancêtres éloignés des proposants pour lesquels l'information génotypique est connue que sur les ancêtres proches. Pour tenir compte de cette distance, les indices sont calculés par génération. Chaque individu a une probabilité estimée d'être porteur et, en considérant l'ensemble des ancêtres à une génération donnée, on obtient une distribution des valeurs des probabilités. La méthode sera efficace si la distribution des probabilités pour les ancêtres à une génération donnée donne des valeurs plus petites pour les non porteurs selon les génotypes de référence que pour les porteurs. L'idée est de comparer ces deux distributions de probabilité. Les distributions sont des estimations plus ou moins variables selon le nombre d'ancêtres considérés à une génération donnée, c'est pourquoi un lissage de ces distributions est effectué par la méthode du noyau. Dans ce cas-ci, la méthode du noyau permet d'estimer la densité des distributions de probabilité par lissage. Les deux paramètres importants sont le noyau et la fenêtre. le noyau est généralement choisi comme étant la densité d'une fonction gaussienne normale (elle peut être sous une autre forme : cosinus, biweight,...). La fenêtre est le paramètre qui optimise le lissage de la fonction de densité. Ainsi, la densité en chacun des points des deux distributions est estimé par l'estimateur non-paramétrique de la méthode du noyau. On utilise alors les même paramètres (noyau et fenêtre) pour

estimer les deux distributions (porteurs et non-porteurs). Pour plus de détails sur la méthode du noyau, on pourra se référer à (Devroye [7]). Le but étant simplement de comparer les distributions, aucune optimisation n'a été effectuée sur la forme du noyau ou sur la fenêtre. Le noyau utilisé est le noyau normal et la fenêtre a été déterminée par essais pour obtenir une distribution relativement lisse. L'autre idée est de vérifier s'il existe une bonne discrimination entre les deux densités à différentes profondeurs et, cela s'avère si la discrimination est dans le bon sens. Ainsi, on constate que les non porteurs se divisent en 2 groupes (graphe rouge Figure 4.8), un qui est très similaire aux porteurs (densité très proche ou formes des graphes rouge et noir coïncident sur la Figure 4.8) et un autre qui donne des valeurs de probabilité plus proche de 0 (ce qui est souhaitable). C'est dire alors qu'il existe des individus de la généalogie qui sont non porteurs et qu'on ne peut distinguer des porteurs. En effet, cela était à prévoir puisqu'un enfant qui a hérité d'un allèle peut l'avoir hérité de son père ou de sa mère et sans autres informations, ils sont aussi probables l'un que l'autre. Cependant, le plus souvent il n'y en aura qu'un seul des deux parents qui est porteur. Il est donc encore difficile de déterminer le génotype de certains individus non porteurs. L'élément important étant le fait de réussir à distinguer les génotypes de non porteur pour une partie des ancêtres, on remarque que cette discrimination est aussi présente sur les générations profondes et qu'il n'y a que les génération 5 et 6 (Figure 4.8) pour lesquels cette discrimination n'est pas très nette.

Dans un second temps, nous considérons une classification des individus selon les résultats des simulations. Or, pour chaque ancêtre le résultat de la simulation est une probabilité donc il faut choisir un point en deçà duquel un individu sera considéré comme non porteur. Plusieurs techniques similaires comme la régression logistique ont comme point critique a priori la valeur de 0.5. Dans le contexte des analyses de géno-

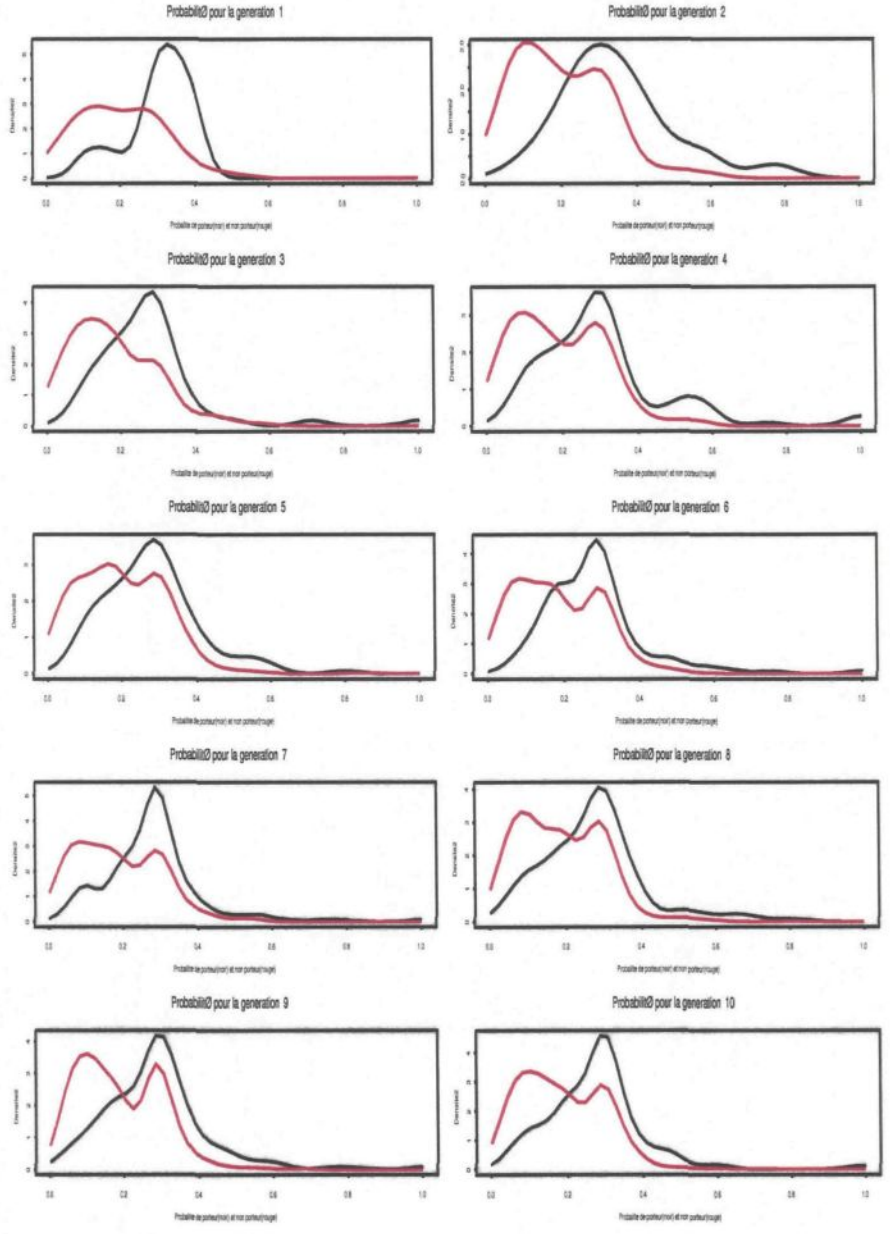


FIGURE 4.8 – Distributions (estimation par la méthode du noyau) pour les porteurs « 01 » ou « 11 » et les non porteurs « 00 » par génération (généalogie réelle).

typage, cette valeur est beaucoup trop grande et il est préférable de choisir une valeur plus petite. Pour déterminer la valeur limite il faut considérer l'indice kappa. Prenons $a = 0.22$ le paramètre de coupure. Cela veut dire que tous les ancêtres pour lesquels la probabilité est plus petite que ce point sont considérés comme non porteurs et les autres porteurs. On obtient la table de contingence au Tableau 4.1. La technique est bonne si tous les ancêtres se trouvent sur la diagonale ou du moins une grande partie. En effet, l'indice de kappa mesure l'intensité de l'adéquation entre les génotypes de référence, obtenus par la méthode du « gene dropping », et les génotypes obtenus par simulation. Le point a qui est utilisé est celui qui maximise la valeur de kappa. Pour le déterminer, l'indice est calculé pour toutes les valeurs de a allant de 0 à 1 avec une précision de 0.01 (Figure 4.9). On observe ainsi une meilleure estimation pour les générations proches de 0 et une valeur stable pour les génération 6 et plus. On remarque aussi que la génération 5 est celle avec la plus mauvaise valeur de kappa. Cela correspond à ce que l'on voit dans les densités des probabilités.

	Porteurs réels	Non porteurs réels
Porteurs simulés	160	2503
Non porteurs simulés	0	5686

TABLEAU 4.1 – Tableau de contigence (Porteurs Vs Non porteurs)

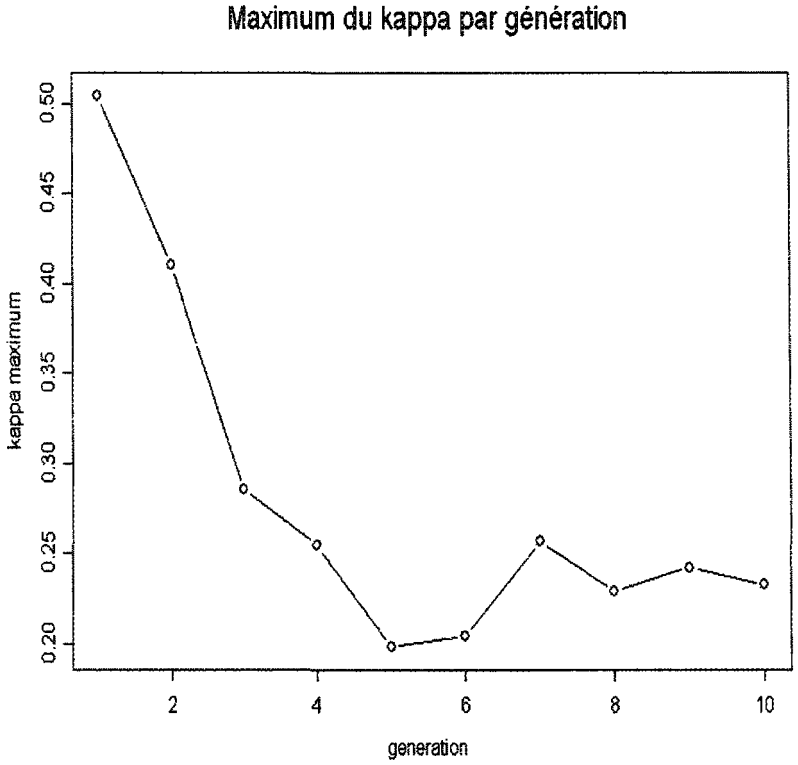


FIGURE 4.9 – Maximun de l'indice du Kappa par génération (généalogie réelle).

Conclusion et discussion

La problématique de la reconstitution des génotypes d'un ensemble de généalogies a motivé l'étude réalisée dans le cadre de ce mémoire. En effet, les solutions déterministes à cette problématique n'ont pas été satisfaisantes lorsqu'il s'agissait de les appliquer sur des généalogies profondes. Nous avons donc utilisé les méthodes probabilistes, nous permettant ainsi d'inférer sur les génotypes des individus. Ces méthodes probabilistes, connues sous le nom de méthodes de Monte Carlo par chaîne de Markov, se sont révélées très efficaces, même avec des généalogies de plus d'une dizaine de générations.

La première partie de ce mémoire (chapitre 2) a été l'occasion de passer en revue les différentes solutions qui auraient pu s'appliquer à la problématique. Nous avons démontré qu'une solution déterministe ne pouvait s'appliquer de façon efficace. Cependant, certaines solutions utilisant une approche probabiliste se révélaient plus intéressantes. Ces méthodes probabilistes avaient déjà été utilisées dans des études similaires, mais elles s'avéraient limitées dans le contexte de notre étude puisque soit leur application dans le contexte d'analyse sur des généalogies profondes devenait fastidieuses, soit les résultats obtenus ne nous donnaient pas les informations désirées. Nous nous sommes alors intéressé aux MCMC, à savoir l'algorithme de Hastings-Métropolis

et celui de l'échantillonnage de Gibbs. Ces méthodes ont prouvé leur efficacité dans certaines études, notamment dans des analyses de liaisons génétiques sur des noyaux familiaux étendus.

Dans la deuxième partie du mémoire (chapitre 3), nous avons étudié les MCMC pour en extraire les fondements, les applications et les règles d'utilisation. Nous avons alors prouvé que les MCMC permettaient de simuler des chaîne de Markov ergodique dont le noyau convergeait vers une loi stationnaire : la loi d'intérêt. Ce qui pouvait alors nous permettre d'inférer sur les génotype des individus d'une généalogie quelconque. Nous avons donc adapté les MCMC généralisées pour aboutir à des algorithmes s'appliquant à notre étude. Il a fallu alors renforcer les propriétés des chaînes de Markov pour garantir d'une part l'existence d'une loi stationnaire, et d'autre part accélérer la convergence vers cette loi.

La troisième section (Chapitre 4) se ramenait à montrer comment les MCMC généralisées que nous avions adaptées à notre étude pouvaient être implantées. Nous avons procédé à certains ajustements au niveau de notre modèle et au calibrage des paramètres. Cela a permis d'élaborer un modèle équilibré à partir duquel les algorithmes de simulation se sont précisés. Nous pouvions alors passer à l'étape des tests pour prouver la validité de notre modèle en pratique. Nous avons testé les différentes propriétés souhaitées plus haut. Ainsi, pour des paramètres bien ajustés, le modèle converge vers la loi d'intérêt. Ce constat a pu être validé en comparant les résultats des simulations et les calculs réels, et aussi, en validant les algorithmes à partir de tests sur des données réelles. En effet, en introduisant un allèle de référence dans la généalogie, trait que nous avons assigné à un individu (fondateur) choisi au hasard dans la généalogie (l'individu 74052), nous avons obtenu deux patterns de génotypes : les génotypes de référence obtenus par la méthode du « gene dropping » et ceux obtenus par simulation MCMC. La

synthèse des résultats obtenus nous a permis d'établir, pour chaque individu, les distributions de probabilité pour les génotypes « 00 », « 01 » et « 11 ». Nous avons alors regrouper les individus en deux catégories : les porteurs (« 01 » et « 11 ») et les non porteurs (« 00 »). L'idée était ensuite de comparer les génotypes de référence à ceux obtenus par simulation. Pour cela, nous utilisons l'estimation par la méthode du noyau et la mesure de Kappa. L'estimation par la méthode du noyau donnant la distribution pour les porteurs et les non porteurs par génération, nous a permis d'obtenir une idée de la répartition des probabilités selon l'état des individus. Cela a permis de constater qu'il existe un grand nombre d'individus pour lesquels le génotype peut être déterminé. Cependant, il est encore difficile de déterminer le génotype de certains individus non porteurs. L'indice de Kappa a permis de réaliser une analyse génération par génération et ainsi voir l'effet des profondeurs. Bien que perfectible, cet indice a été une bonne façon de voir si la méthode utilisée est bonne, et pour quelles générations. Pour ce faire, on a établi la génération moyenne en nombre entier pour chaque ancêtre puis, pour une génération donnée, le Kappa a été calculé en fonction de tous les « points de rupture » possibles (de 0 à 1 par pas de 0.001). Le point a été déterminé comme le point optimal à chaque génération. Nous observions ainsi une meilleure estimation pour les générations proches de 0 et une valeur stable pour les génération 6 et plus.

Ainsi, les algorithmes parviennent à donner un portrait génotypique global de la généalogie étudiée. Les analyses effectuées sur les simulations ont permis de constater que la méthode utilisée dans le contexte d'une estimation des génotypes d'une généalogie profonde est bonne et permet d'apporter des éléments intéressants à l'analyse des génotypes d'une généalogie profonde. En effet, plusieurs points intéressants sont à retenir. Le premier point est la simplicité des méthodes utilisées. L'utilisation astucieuse des nombres aléatoires permet de concevoir des algorithmes simples à implémenter

pour résoudre une problématique difficile à modéliser de façon déterministe. Il est intéressant de constater comment l'utilisation d'un algorithme de type Las Vegas permet l'obtention d'une généalogie compatible dans des temps très acceptables. Le second point concerne le temps d'exécution des algorithmes dans le contexte des MCMC. Les algorithmes s'exécutent dans des temps très satisfaisants (par estimation). Enfin, le troisième point est la performance (qualité) des méthodes utilisées. Pour une généalogie profonde, les méthodes permettent d'identifier deux groupes d'individus. Ceux qui n'influencent pas la généalogie en terme de participation génotypique (non porteurs) et ceux qui l'influencent considérablement (les porteurs). Cependant, il est encore difficile de connaître avec un taux d'erreur minime, les ancêtres non porteurs reliés aux individus porteurs. L'explication intuitive est qu'en réalité, l'allèle de référence transmis à un individus peut provenir d'un seul des parents, mais dans les simulations les parents ont une chance équiprobable de transmettre cet allèle. Nous donnons ici quelques pistes de solutions qui pourraient être étudier lors de prochaines études. La première piste serait d'identifier et d'isoler les individus qui ne contribuent aucunement à la généalogie, cela permettrait ainsi de mieux cibler les porteurs réels. L'autre point intéressant serait d'inclure dans la généalogie plusieurs individus dont le génotype est connu, pas seulement des proposants mais aussi des ancêtres. Dans ce cas là, les estimations au niveau des génotypes seront plus précises. Aussi, comme dernière piste, nous suggérons de regarder plus qu'un gène à la fois. En effet, lors des simulations, les observations sont faites sur un seul gène, plus précisément sur l'allèle de référence. Cela à tendance à minimiser l'effet global d'un ensemble d'allèles sur la transmission des gènes. Nous pensons que cette manière de faire permettrait de mieux comprendre et analyser la transmission, surtout si les gènes sont liés génétiquement.

Ainsi, toutes ces améliorations devraient permettre de faire des analyses supplémen-

taires et de les pousser un peu plus loin. Cependant, pour permettre ces améliorations, il pourrait être intéressant d'apporter certains ajustements techniques. Par exemple, au niveau de l'écran utilisateur, toutes les exécutions étant réalisées en mode commande, cela rend difficile et fastidieux l'analyse des résultats. On pourra aussi chercher à optimiser les algorithmes pour permettre une meilleur discrimination des individus dans les paramètres initiaux et ainsi, minimiser le taux d'erreur. Voici là, quelques pistes de recherche qui promettent de parfaire les méthodes utilisées. Nous espérons donc que de prochaines études s'y pencheront.

Annexe A : Généalogie de 95 individus

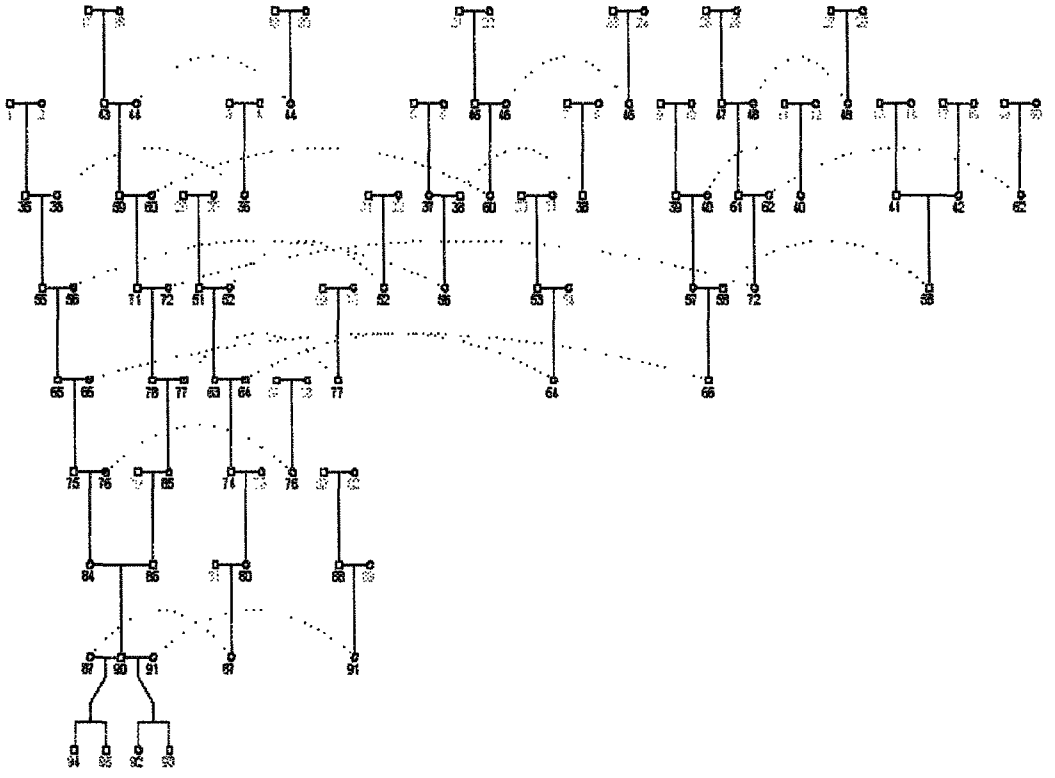


Figure 10 – Généalogie de 95 individus

Annexe B : Calculs préliminaires

$$P(A_3 = 0) =$$

$$P(A_1 = 0)[\ P(A_2 = 0)P(A_3 = 0|A_1 = 0, A_2 = 0)+$$

$$P(A_2 = 1)P(A_3 = 0|A_1 = 0, A_2 = 1)\ ] +$$

$$P(A_1 = 1)[\ P(A_2 = 0)P(A_3 = 0|A_1 = 1, A_2 = 0)+$$

$$P(A_2 = 1)P(A_3 = 0|A_1 = 1, A2\_1)\ ]$$

$P(A_3 = 0) = 1/2\ (1/2 + 1/6) + 1/4\ (\ 1/3 + 1/8)$

$P(A_3 = 0) = 43/96$

$$P(A_3 = 1) =$$

$$P(A_1 = 0)[\ P(A_2 = 1)P(A_3 = 1|A_1 = 0, A_2 = 1)+$$

$$P(A_2 = 2)P(A_3 = 1|A_1 = 0, A_2 = 2)\ ] +$$

$$P(A_1 = 1)[\ P(A_2 = 0)P(A_3 = 1|A_1 = 1, A_2 = 0)+$$

$$P(A_2 = 1)P(A_3 = 1|A_1 = 1, A_2 = 1)+$$

$$P(A_2 = 2)P(A_3 = 1|A_1 = 1, A_2 = 2)\ ]$$

$$P(A_1 = 2)[\ P(A_2 = 1)P(A_3 = 1|A_1 = 2, A_2 = 1)+$$

$$P(A_2 = 2)P(A_3 = 1|A_1 = 2, A_2 = 2)\ ]$$

$$P(A_3 = 1) = 1/2\ (1/12 + 1/4) + 1/4\ (1/6 + 1/16 + 1/8) + 1/4\ (1/2 + 1/8)$$

$$P(A_3 = 1) = 79/192$$

$$P(A_3 = 2) =$$

$$P(A_1 = 1)(P(A_2 = 1)P(A_3 = 2|A_1 = 1, A_2 = 1)+$$

$$P(A_2 = 2)P(A_3 = 2|A_1 = 1, A_2 = 2)\ ] +$$

$$P(A_1 = 2)(P(A_2 = 1)P(A_3 = 2|A_1 = 2, A_2 = 1)+$$

$$P(A_2 = 2)P(A_3 = 2|A_1 = 2, A_2 = 2)\ ]$$

$$P(A_3 = 2) = 1/4\ (1/16 + 1/8) + 1/4\ (1/8 + 1/4)$$

$$P(A_3 = 2) = 9/64$$

Annexe C : Calculs théoriques

$$P(A_5 = 0|A_7 = 1) =$$

$$1/2[P(A_5 = 0|A_6 = 1, A_7 = 1) + P(A_5 = 0|A_6 = 2, A_7 = 1) + P(A_5 = 0)]$$

$$P(A_5 = 0|A_7 = 1) = 1/2(79/192 * 43/96 + 9/64 * ((43/96)/(79/192 + 43/96)) + 43/96)$$

$$P(A_5 = 0|A_7 = 1) = 0.3527555832$$

$$P(A_5 = 1|A_7 = 1) =$$

$$1/2[P(A_5 = 1|A_6 = 0, A_7 = 1) + P(A_5 = 1|A_6 = 1, A_7 = 1) +$$

$$P(A_5 = 1|A_6 = 2, A_7 = 1) + P(A_5 = 1)]$$

$$P(A_5 = 1|A_7 = 1) =$$

$$1/2(43/96 * ((79/192)/(9/64 + 79/192)) + 79/192 *$$

$$79/192 + 9/64 * ((79/192)/(43/96 + 79/192)) + 79/192)$$

$$P(A_5 = 1|A_7 = 1) = 0.4909552622$$

$$P(A_5 = 2|A_7 = 1) =$$

$$1/2[P(A_5 = 2|A_6 = 0, A_7 = 1) + P(A_5 = 2|A_6 = 1, A_7 = 1) + P(A_5 = 2)]$$

$$P(A_5 = 2|A_7 = 1) = 1/2(43/96 * ((9/64)/(79/192 + 9/64)) + 79/192 * 9/64 + 9/64)$$

$$P(A_5 = 2|A_7 = 1) = 0.1562891546$$

BIBLIOGRAPHIE

[1] Projet BALSAC. Rapport annuel - projet balsac. Technical report, Université du Québec à Chicoutimi (UQAC), 1997-1998.

[2] Claude Belisle. Slow convergence of the gibbs sampler. *The Canadian Journal of Statistics*, Vol. 26, No 4 :pp : 629–641, 1998.

[3] Gilles Brassard and Paul Bratley. *Algorithmique : conception et analyse.* Montréal : Presse de l'Université de Montréal, 1987.

[4] Stephen P. Brooks. Markov chain monte carlo method and its application. *The Statistician*, Vol. 47, Part 1 :pp : 69–100, 1998.

[5] Kai Lai Chung. *Markov Chains with stationary transition probabilities.* Berlin ; New York : Springer-Verlag, 1967.

[6] Thomas Cormen, Charles Leiserson, and Ronald Rivest. *Introduction à l'algorithmique.* Paris : Dunod, 2002.

[7] Luc Devroye. *A course in density estimation.* Birkhäuser, 1987.

[8] Joseph L. Doob. *Stochastic processes.* London : Wiley, 1953.

[9] William Feller. *An introduction to probability theory and its applications.* New York : J. Wiley, 1968.

[10] David Freedman. *Markov chains.* San Francisco : Holden-Day, 1971.

[11] Stuart Geman and Donald Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE transactions on pattern analysis and machine intelligence*, Vol. 6 :pp : 721–741, 1984.

[12] Charles J. Geyer and Elizabeth A. Thompson. Annealing markov chain monte carlo with applications to ancestral inference. *Journal of the American Statistical Association*, Vol. 90, No. 431, Septembre 1995.

[13] John Michael Hammersley. *Monte Carlo methods.* London : Chapman and Hall, 1979.

[14] W. K. Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, Vol. 57 (1) :pp : 97–109, 1970.

[15] Dean L Isaacson. *Markov chains, theory and applications.* New York ; Toronto : J. Wiley, 1976.

[16] John B. Jenkins. *Human Genetics.* Menlo Park, Calif. ; Don Mills, Ont. : Benjamin/Cummings, 1983.

[17] John George Kemeny and James Laurie Snell. *Finite Markov chains.* New York : Springer, 1976.

[18] Kenneth Lange. *Mathematical and Statistical Methods for Genetic Analysis.* Springer-Verlag New York, Inc ; 1 édition, 1997.

[19] Nicholas Metropolis, Ariana W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, Vol. 21 (6) :pp : 1087–1092, 1953.

[20] Rajeev Motwani and Prabhakar Raghavan. *Randomized Algorithms.* Cambridge University Press, 1995.

[21] Daniel Revuz. *Markov chains.* New York : Ameriacan Elsevier, 1975.

[22] Christian Robert. *Méthodes de Monte Carlo par chaînes de Markov.* Economica, 1996.

[23] Vsevolod Ivanovich Romanovskii. *Discrete Markov chains.* Groningen : Wolters-Noordhoff, 1970.

[24] Peter J. Russel. *Essential Genetics.* Blackwell Scientific Publications, 1987.

[25] David T. Suzuki, Anthony J.F. Griffiths, and Richard C. Lewontin. *An introduction to genetic analysis.* New York : W. H. Freeman, 1981.

[26] Marc Tremblay, Julie Arsenault, and Évelyne Heyer. Les probabilités de transmission des gènes fondateurs dans cinq populations régionales du québec. *Population (French Edition)*, Vol. 58e Année, No. 3 :pp : 403–423, 2003.

[27] Évelyne Heyer. One founder/one gene hypothesis in a new expanding population : Saguenay (quebec, canada). *Human biology*, vol. 71, no1 :pp : 99–109, 1999.

[28] Ellen M. Wijsman, Joseph H. Rothstein, and Elizabeth A. Thompson. Multipoint linkage analysis with many multiallelic or dense diallelic markers : Markov chain monte carlo provides practical approaches for genome scans on general pedigrees. *American Journal of Human Genetics*, Vol. 79 :pp : 846–858, Novembre 2006.