UNIVERSITÉ D'ÉVRY
École Doctorale des Génomes aux Organismes

THÈSE

pour l'obtention du titre de
DOCTEUR EN SCIENCES

Spécialité : Biomathématiques

soutenue par
Pierre NICOLAS
le jeudi 18 décembre 2003

Titre :

Mise au point et utilisation de modèles de chaînes de Markov cachées pour l'étude des séquences d'ADN.

Directeurs de thèse :
Bernard PRUM et Philippe BESSIÈRES

JURY

Christian GAUTIER		Rapporteur
Alain VIARI		Rapporteur
Philippe BESSIÈRES		Examinateur
Bernard PRUM		Examinateur
Christian ROBERT	Président du Jury	Examinateur
Claude THERMES		Examinateur

Un grand merci à,

Un grand merci à Bernard PRUM et à Philippe BESSIÈRES qui ont dirigé ce travail. Merci à tous les deux pour votre disponibilité, vos bons conseils et votre soutien constant. Merci à Bernard pour son plaisir à partager ses connaissances, toujours prêt à « improviser » un petit cours de math ! Merci à Philippe sans qui je n'aurai peut-être jamais entendu parler de HMM et qui n'a pas d'égal pour l'organisation de collaborations.

Un grand merci aussi à Florence MURI et à Anne-Sophie TOCQUET qui ont consacré beaucoup de temps au co-encadrement de ce travail et qui ont joué un grand rôle dans sa réalisation.

Je souhaite vivement remercier François RODOLPHE pour son incroyable disponibilité et son inépuisable enthousiasme lorsqu'il s'agit de discuter de modélisation.

Je remercie Sophie SCHBATH pour son investissement dans la vie du groupe de travail *Statistique des Séquences Biologiques* et du Laboratoire *Mathématique, Informatique et Génome*. Merci aussi pour la Californie !

Je tiens à remercier Christian GAUTIER et Alain VIARI qui ont accepté d'être les rapporteurs de cette thèse. Je remercie aussi Christian P. ROBERT et Claude THERMES pour leur participation au jury.

Un immense merci à l'ensemble des membres (présents et passés) des laboratoires *Mathématique, Informatique et Génome* et *Statistique et Génomes* qui m'ont accueilli durant ces trois années ainsi que les autres membres du groupe *Statistique des Séquences Biologiques*. Merci non seulement pour l'environnement scientifique très stimulant mais aussi pour l'ambiance vraiment exceptionnelle !

Merci à tous les autres avec qui j'ai eu la chance de travailler pendant ma thèse. Merci à Yves d'AUBENTON, Claude THERMES, Ludovic COTTRET et Élodie GUILLAUME pour m'avoir initié aux problèmes de la prédiction de gènes chez les eucaryotes. Merci à Katia MARROCCO et Philippe GUERCHE. Merci aux bulgariciens (bulgaricistes ?) Maarten VAN DE GUCHTE et Stéphanie PENAUD. Merci aux sakeistes Stéphane CHAILLOU, Sophie BEAUFILS et Véronique MARTIN. Merci encore à Stéphane pour m'avoir fait découvrir la randonnée dans les Alpes. Merci à Mariam IBRAHIM, Rozen GARDAN et à Vincent SANCHIS de leur intérêt pour l'étude des petits gènes.

Merci enfin à tous ceux qui m'ont soutenu pendant ces trois années et

particulièrement lors des derniers mois. Pour ça encore, merci à tous les membres des deux laboratoires. Merci à Fred, Marc, Gauthier, Alexis et Corinne. Merci à Nils, Brieuc et Nicole.

Résumé

Les modèles de chaînes de Markov cachées (HMM) font partie des principaux modèles statistiques utilisés pour l'analyse des séquences d'ADN. Ce travail a porté sur la mise au point et l'utilisation d'approches d'estimation non supervisée de ces modèles. D'une manière générale, ces approches présentent l'intérêt de pouvoir mettre en évidence de nouvelles caractéristiques des séquences. Trois domaines d'application des HMM pour l'interprétation des génomes bactériens ont été abordés dans cette thèse.

Le premier domaine est l'utilisation d'un HMM pour la segmentation des séquences en régions de composition homogène. L'approche s'est montrée adaptée à l'identification des transferts génétiques horizontaux sur le chromosome de *Bacillus subtilis*. Elle a aussi permis de révéler d'autres niveaux d'hétérogénéités liés aux propriétés biologiques des gènes.

Le second domaine est la prédiction de gènes. Un logiciel de prédiction de gènes fondé sur l'estimation, sans constitution d'un ensemble d'apprentissage, des paramètres d'un HMM a été développé. Il présente aussi l'originalité d'associer une mesure de confiance aux prédictions. Ses prédictions sont de très bon niveau et la mesure de confiance révèle de nombreux petits ORF (*Open Reading Frame*) dont les propriétés de composition ressemblent à celles des gènes. Une confirmation par comparaison de génomes de la nature codante de ces ORF a été recherchée. Chez *B. subtilis*, elle a conduit à proposer une trentaine de petits gènes de taille inférieure à 50 acides aminés qui s'ajoute à la vingtaine de petits gènes connus biologiquement.

Le troisième domaine concerne la recherche de motifs de sites de fixation de l'ARN polymérase. Ces motifs sont constitués de deux boîtes séparées par une distance variable mais contrainte. Un algorithme MCMC (Monte-Carlo par chaînes de Markov) de sélection bayésienne de modèle a été développé pour répondre à la question délicate du choix d'un modèle adapté. Cet algorithme détermine simultanément, la taille de chacune des boîtes, le support de la distribution de la distance séparant les deux boîtes, et l'ordre markovien du modèle de composition des séquences en dehors des boîtes. Les modèles sélectionnés sont en accord avec les descriptions de motifs connus. Étant donné l'enjeu que représente la recherche de motifs, l'algorithme présenté ici pour la mise en œuvre de l'approche bayésienne de sélection de modèle semble très prometteur.

Table des matières

Préambule

Au cours des dix dernières années, l'information génétique a été mise à portée de main des biologistes par un essor considérable des technologies de séquençage de l'ADN. La séquence complète du génome de nombreux organismes est déjà disponible ; et potentiellement, c'est la séquence de n'importe quel génome qui peut maintenant être déterminée. L'interprétation de ces séquences est donc naturellement devenue un enjeu central de la biologie. Mais cette interprétation pose, entre autres, d'importants problèmes méthodologiques pour lesquels les approches fondées sur une description de la séquence à travers un modèle probabiliste se sont déjà montrées pertinentes. Parmi les modèles utilisés, les modèles de chaînes de Markov cachées (HMM, *Hidden Markov Model*) ont pris une place particulière. En effet, ils constituent une solution simple pour permettre une description probabiliste de l'alternance de différents types d'éléments le long des séquences.

Cependant, l'interprétation des séquences reste largement tributaire des connaissances déjà acquises, en particulier grâce à des expériences. Dans le cadre de l'utilisation des modèles probabilistes, le tribut se traduit par la mise au point de modèles adaptés à la description des connaissances déjà disponibles. En forçant à peine le trait, les paramètres sont le plus souvent estimés sur un jeu de données le plus sûr possible en regard de ces connaissances. De façon encore plus nette, l'architecture des modèles est élaborée pour correspondre à ce que l'on connaît déjà.

L'objectif des travaux réalisés au cours de cette thèse est de participer au raffinement de l'utilisation des HMM. Plus précisément, l'axe suivi vise à contribuer à l'élargissement du champ d'application des méthodes d'estimation et de sélection de modèle « non supervisées ». Ces méthodes ne s'appuient pas sur un jeu de données supposé propre pour ajuster le modèle. Elles peuvent donc, dans une certaine mesure, proposer de nouvelles descriptions des données. C'est pourquoi, elles portent un espoir de sortir partiellement du processus de reproduction des connaissances.

Trois domaines d'application des HMM pour l'analyse des génomes bactériens ont été abordés dans cette thèse : la description et l'interprétation des hétérogénéités de composition ; la détection des régions traduites en protéines ; et enfin, la recherche des motifs structurés que sont les promoteurs. La méthodologie employée dans les deux premiers travaux repose essentiel-

lement sur l'estimation des paramètres d'un HMM par maximisation de la vraisemblance de la séquence observée. Le troisième problème est vu sous un angle de choix de modèle. Il se situe dans une approche bayésienne mise en œuvre grâce à un algorithme de Monte-Carlo par chaînes de Markov à sauts réversibles (*Reversible Jump* MCMC).

Le manuscrit est organisé en cinq chapitres. Le premier vise à une introduction large au problème de la modélisation des séquences de macromolécules biologiques. Le second chapitre détaille les principaux aspects mathématiques et algorithmiques de l'utilisation des chaînes de Markov cachées. Une place particulière y est faite à la présentation de l'approche bayésienne de sélection de modèle. Les trois derniers chapitres sont rédigés sous la forme d'articles et présentent les travaux réalisés au cours de cette thèse : étude des hétérogénéités, prédiction de gènes, puis recherche de motifs structurés.

Deux annexes sont aussi incluses. La première contient un chapitre de livre présentant les principales approches pour la prédiction de gènes. La seconde est la documentation du logiciel SHOW, développé pour permettre une utilisation souple des HMM dans le domaine de l'analyse des séquences biologiques.

Chapitre 1

Des macromolécules biologiques aux chaînes de Markov cachées

Ce chapitre présente le cadre de l'utilisation des modèles de chaînes de Markov cachées pour l'analyse des séquences des trois principaux types de macromolécules biologiques que sont l'ADN, l'ARN et les protéines. Il débute par une présentation de ces molécules visant à mettre en lumière le rôle joué par leurs séquences. La section 1.2 décrit ensuite comment le développement du séquençage à grande échelle de l'ADN a placé le problème de l'interprétation de ces séquences au coeur des préoccupations de la biologie actuelle. La section 1.3 constitue une introduction à l'analyse des séquences fondée sur leur modélisation probabiliste. Cette dernière section s'achève par une présentation des modèles de chaînes de Markov cachées qui sera poursuivie dans le chapitre 2.

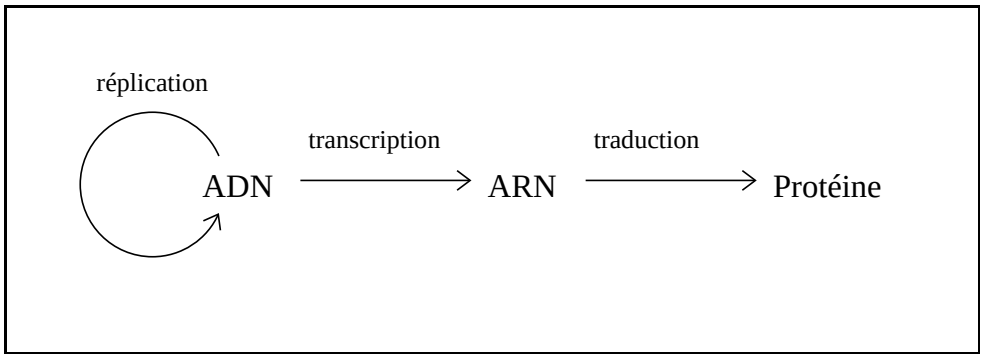

FIG. 1.1 – Le flux de l'information génétique

1.1 Macromolécules biologiques : ADN, ARN et protéines

Certains organismes vivants sont multicellulaires, d'autres sont unicellulaires mais, dans tous les cas, la cellule est l'unité vivante élémentaire. La cellule est séparée du reste du monde par au moins une membrane et elle possède la capacité de se reproduire. Selon leur organisation, on distingue deux grands types de cellules vivantes : les cellules procaryotes, relativement simples et constituées essentiellement d'un unique compartiment ; et les cellules eucaryotes, compartimentées, et souvent beaucoup plus grandes que les cellules procaryotes. Alors que les organismes multicellulaires (végétaux, champignons et animaux) sont tous des organismes eucaryotes, il existe des organismes unicellulaires aussi bien eucaryotes (algues unicellulaires, levures, protistes) que procaryotes (eubactéries et archées).

Au niveau chimique toutes les cellules vivantes contiennent les trois principaux types de macromolécules biologiques que sont l'acide désoxyribonucléique (ADN), l'acide ribo-nucléique (ARN) et les protéines. Tous trois sont des enchaînements non périodiques et orientés d'un petit nombre de molécules élémentaires. L'ADN et l'ARN sont des acides nucléiques constitués chacun de quatre types de nucléotides. Les nucléotides de l'ADN et de l'ARN diffèrent légèrement et sont identifiés respectivement par les lettres $\{a, g, c, t\}$ pour l'ADN et $\{a, g, c, u\}$ pour l'ARN. Les protéines sont quant à elles des chaînes d'acides aminés. Ceux-ci sont au nombre de vingt $\{A, C, D, \ldots\}$. La diversité des séquences de ces macromolécules est à l'origine de la diversité biochimique du monde vivant.

La description des relations entre les séquences d'ADN, d'ARN et de protéines est le cœur de la théorie de la biologie moléculaire (Fig. 1.1). Les molécules d'ADN sont le support de l'information génétique. Cette information se perpétue grâce au mécanisme de réplication de l'ADN. L'expression de l'information génétique repose sur la transcription de l'ADN en ARN et

sur la traduction de l'ARN en protéine. Les mécanismes de la transcription et de la traduction assurent un flux unidirectionnel de l'information de l'ADN à la protéine en passant par l'ARN. Les protéines sont la principale forme d'expression de l'information génétique ; elles jouent notamment un rôle crucial dans l'accélération et l'orientation des réactions chimiques (catalyse) nécessaires à la vie.

Cette présentation est légèrement simplificatrice car l'ARN constitue le support de l'information génétique de certains virus. L'ARN est alors soit rétrotranscrit en ADN, soit directement répliqué sous forme d'ARN. La rétrotranscription constitue une inversion du flux de l'information génétique.

Cette section aborde dans un premier temps la description de la nature chimique de l'ADN, de l'ARN et des protéines et se poursuit par une présentation de leurs séquences.

1.1.1 Nature chimique des macromolécules biologiques

La double hélice d'ADN

Un nucléotide est constitué d'une base, d'un groupe phosphate et d'un sucre. Les nucléotides de l'ADN diffèrent les uns des autres uniquement par leur base qui peut être l'adénine, la guanine, la cytosine ou la thymine. Les bases portent l'information génétique alors que le sucre et le phosphate ont un rôle de structure. Les nucléotides s'enchaînent en un long brin d'ADN par des liaisons covalentes entre le sucre et le phosphate des nucléotides adjacents.

La Figure 1.2 représente la structure en double hélice d'une petite molécule d'ADN obtenue par cristallographie aux rayons X. Enroulé en double hélice, la molécule d'ADN est une molécule d'un diamètre d'environ $0,2 \times 10^{-9}\,m$ mais d'une longueur surprenante. En effet, l'information génétique d'une cellule est contenue dans un petit nombre de molécules d'ADN : les chromosomes. Le chromosome de la bactérie *Escherichia coli* à une longueur physique de $1,4 \times 10^{-3}\,m$ pour un peu plus de quatre millions de paires de bases. Les chromosomes humains, de plusieurs dizaines de millions de paires de bases, font eux plusieurs centimètres. Ces longueurs sont à comparer à la dimension micrométrique des cellules (typiquement, $\sim 1 \times 10^{-6}\,m$ pour une bactérie et $10 - 30 \times 10^{-6}\,m$ pour une cellule animale).

La découverte de la structure en double hélice de l'ADN par Watson et Crick en 1953 a révolutionné la biologie. Malgré des résultats expérimentaux publiés dès 1944 et qui paraissent aujourd'hui très convaincants, le rôle de l'ADN comme support de l'information génétique n'a été admis par l'ensemble de la communauté scientifique qu'après 1953. Les protéines, présentant une grande diversité de propriétés chimiques, paraissaient de fait de meilleures candidates. Mais la description de la structure en double hélice mit en évidence deux propriétés essentielles à la compréhension de rôle de

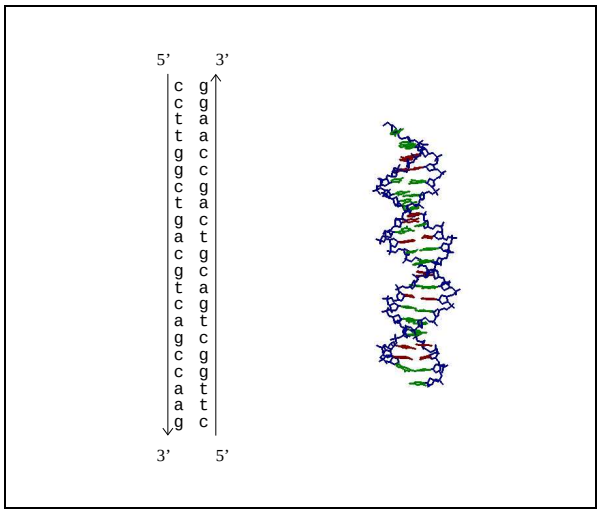

FIG. 1.2 – Structure de la double hélice d'ADN. Les liaison entre atomes (sauf hydrogènes) sont représentées par des barres. Le squelette sucre-phosphate est représenté en bleu, les bases adénine et thymine en rouge et les bases cytosine et guanine en vert.

l'ADN comme support de l'information génétique. D'une part, elle a montré le caractère ordonné de la séquence des nucléotides. Elle a d'autre part mis en évidence la nature double brin de la molécule d'ADN et la règle d'appariement des nucléotides dite de Watson et Crick. Celle-ci permet de déduire la séquence d'un brin en fonction de l'autre : a fait face à t et g à c. Cette observation a immédiatement suggéré le mécanisme de réplication de la séquence : chaque brin sert de modèle pour la synthèse d'un nouveau brin. À partir de cette description, les principaux mécanismes de l'expression de l'information génétique en ARN et en protéine ont été élucidés en une dizaine d'années.

ARN

Les nucléotides de l'ARN diffèrent de ceux de l'ADN par la nature du sucre (ribose au lieu de desoxyribose) et le remplacement d'une base (la thymine) par une autre (l'uracile). L'ARN est, comme l'ADN, une molécule linéaire mais elle ne comporte qu'un brin et ne s'enroule donc pas en double hélice.

La plupart des ARN sont des ARN messagers (ARNm) qui sont traduits en protéines et n'acquièrent pas de structure tridimensionnelle stable. Cependant grâce à l'appariement des bases d'un même brin, certains ARN dits de structure acquièrent une structure tridimensionnelle complexe pouvant notamment comporter des doubles hélices. La Figure 1.3 montre la structure

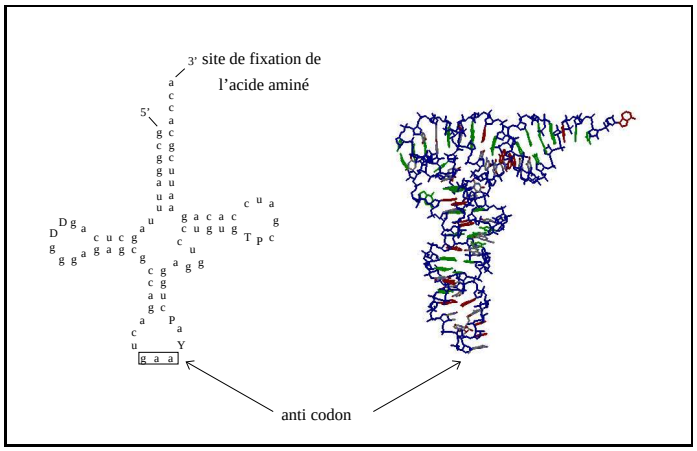

FIG. 1.3 – Structure d'un ARN de transfert. Certaines des bases sont modifiées.

d'un ARN de transfert (ARNt) obtenue par cristallographie aux rayons X. Les ARNt jouent un rôle essentiel dans la traduction ; en effet, chaque type d'ARNt porte un acide aminé et reconnaît un ou des triplets de nucléotides (codon) qui lui correspondent. La reconnaissance a lieu grâce à l'appariement du codon de l'ARNm avec trois bases successives de l'ARNt (l'anticodon).

Protéines

Les vingt acides aminés sont tous constitués d'un atome de carbone central auquel sont fixés un groupe carboxyle, un groupe amine et une chaîne latérale. Ils diffèrent les uns des autres par la nature chimique de leur chaîne latérale. Les acides aminés d'une protéine sont liés les uns aux autres par liaison covalente entre le groupe amine et le groupe carboxyle. Cette séquence en acide aminé est parfois appelée structure primaire d'une protéine.

La structure tridimensionnelle, ou repliement, d'une protéine est déterminée de façon plus ou moins unique (plutôt plus que moins) par sa structure primaire. D'un point de vue physique, la structure tridimensionnelle est un minimum d'une fonction d'énergie. Cette fonction dépend essentiellement des interactions de faibles énergies induites par la nature chimique des acides aminés mis en contact dans la structure tridimensionnelle. On distingue un niveau intermédiaire entre la structure tridimensionnelle et la structure primaire. Il s'agit de la structure secondaire qui décrit l'arrangement dans l'espace des acides aminés adjacents sur la séquence. Certains éléments de structure secondaire sont bien identifiés et constituent des structures ordonnées comme les hélices α et les feuillets β.

Deux exemples de protéines sont présentés ci-dessous avec pour objectif d'illustrer le rôle crucial du repliement dans la fonction des protéines.

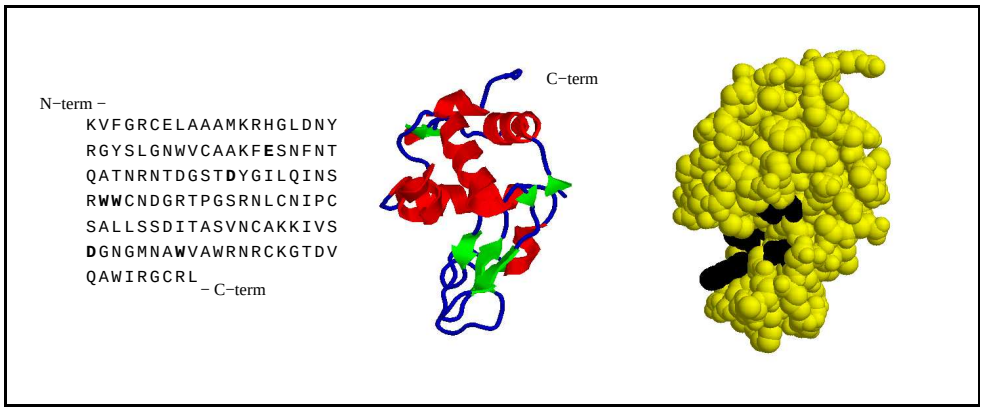

FIG. 1.4 – Structure d'une enzyme : le lysozyme. À gauche, la séquence des acides aminés de la protéine. Au milieu, la structure tridimensionnelle où sont représentés les éléments de structure secondaire (hélices alpha et feuillets béta). À droite, la structure tridimensionnelle où les atomes (à l'exception des hydrogènes) sont représentés par des sphères indiquant l'espace qu'ils occupent. Les acides aminés importants du site actif sont proches les uns des autres dans l'espace (en noir dans la structure) mais pas le long de la séquence (en gras dans la séquence).

La figure 1.4 montre la séquence d'une protéine et son repliement dans l'espace obtenu par cristallographie. Il s'agit d'une enzyme, c'est à dire d'une protéine capable de catalyser une réaction chimique. Sa fonction est de dégrader certains sucres des parois bactériennes. La réaction se fait en un site précis de la structure appelé site actif. Ce site actif est en général un creux dans la structure compacte de l'enzyme. On trouve au sein du site actif des acides aminés distants sur la séquence mais que le repliement tridimensionnel a rapprochés, permettant la fonction de catalyse.

Le deuxième exemple est destiné à illustrer un autre aspect de l'importance de la structure dans les interactions entre molécules. La figure 1.5 montre un complexe ADN - protéine qui correspond à la reconnaissance d'une séquence de régulation de la transcription. La protéine présentée ici est constituée de deux sous-unités (identiques dans cet exemple) qui ont été traduites séparément. Ces deux sous-unités sont associées l'une à l'autre par des interactions faibles entre certains de leurs acides aminés. Ces contacts sont rendus possibles par la structure tridimensionnelle adoptée par les sous-unités. De nombreuses protéines sont ainsi constituées de l'association de plusieurs sous-unités. C'est notamment le cas d'un grand nombre de protéines reconnaissant des signaux sur les séquences nucléiques. Les séquences reconnues présentent alors souvent une certaine symétrie. La structure tridimensionnelle permet également le contact entre quelques acides aminés et

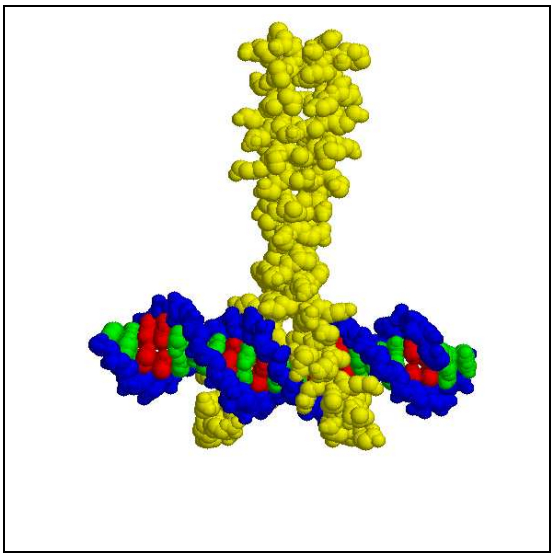

FIG. 1.5 – Structure d'un complexe ADN - protéine. Dimère de protéine à structure de type *leucine-zipper* : on trouve une leucine toute les sept positions dans le domaine de liaison entre les deux protéines.

des bases de la séquence d'ADN à l'origine d'une reconnaissance de cette séquence. Chaque sous-unité participe à cette reconnaissance. Deux séquences d'ADN (ici identiques mais de sens opposés) sont donc reconnues simultanément, ce qui augmente la force de la liaison et la longueur de la séquence reconnue.

Plus généralement, la fixation de protéines à l'ADN est souvent un phénomène coopératif : les interactions entre protéines facilitent la fixation de l'ensemble et participent à la reconnaissance de motifs complexes. Bien sûr, la fixation d'une protéine peut aussi gêner la fixation d'une autre.

La structure d'une protéine n'est souvent pas complètement figée et les changements de structures induits par les liaisons avec d'autres molécules jouent des rôles importants. Ainsi, dans l'exemple de l'opéron lactose présenté dans la section suivante, ce sont les modifications de structures des répresseurs consécutives à leur liaison avec de petites molécules qui changent leur capacité de liaison aux séquences d'ADN reconnues.

1.1.2 Séquences : l'information génétique

Réplication de l'ADN et variabilité de l'information génétique

Comme on l'a vu, l'ADN est une molécule constituée de deux brins appariés. Ces brins sont parallèles mais de sens opposés (antiparallèles). Les séquences des deux brins sont différentes mais se déduisent l'une de l'autre :

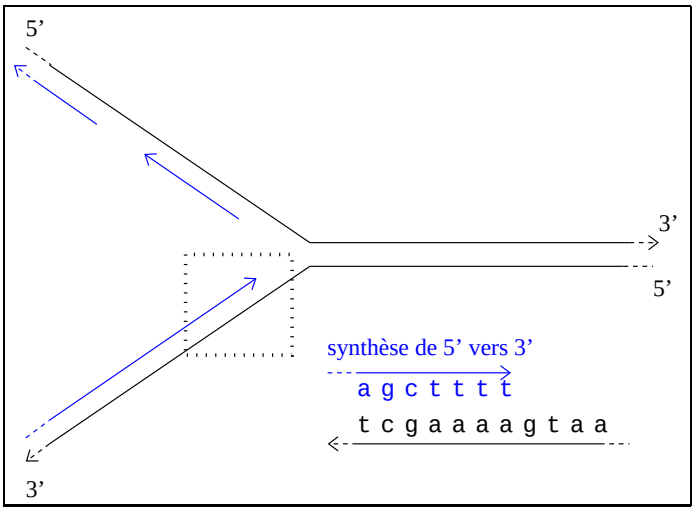

FIG. 1.6 – La réplication de l'ADN

elles sont dites inverses complémentaires. Les nucléotides face à face vérifient la règle de complémentarité de Watson et Crick qui définit les deux seuls appariements autorisés : a avec t et c avec g. Cette règle d'appariement permet la réplication de l'information génétique. Durant celle-ci (Fig 1.6), la séquence de chaque brin sert de modèle à la synthèse d'un brin fils, de séquence inverse complémentaire. Les séquences filles sont toujours synthétisées dans le sens dit 5'→3', la séquence modèle étant lue dans le sens 3'→5'. À partir de son origine, la réplication progresse en séparant les deux brins modèles tout en synthétisant les deux brins fils. Ce mécanisme permet aux brins de rester peu de temps non appariés mais implique une asymétrie de leur traitement. En effet, les deux séquences mères ne peuvent être lues simultanément en continu dans le sens 3'→5'. Ainsi, seul un brin est synthétisé de façon continue (*leading strand*) alors que l'autre est synthétisé par morceaux (*lagging strand*).

Bien que la réplication de l'ADN soit très fidèle, de petites erreurs, ou mutations ponctuelles, peuvent survenir. Elles sont une des origines de la variabilité de l'information génétique nécessaire à l'évolution. Des mutations peuvent aussi se produire entre les réplications et sont alors transmises à la descendance lors de la réplication de l'ADN muté. Parmi les mécanismes à l'origine de la variabilité génétique, les recombinaisons jouent aussi un rôle essentiel en permettant le déplacement de fragments d'ADN. Elles peuvent notamment entraîner des insertions, des délétions et des duplications de grande ampleur.

Expression de l'information génétique

La transcription de l'ADN en ARN. L'expression de l'information génétique passe par la transcription de la séquence d'ADN en séquence d'ARN, puis souvent par la traduction de la séquence d'ARN en séquence protéique. La figure 1.7 schématise les mécanismes de ces deux étapes.

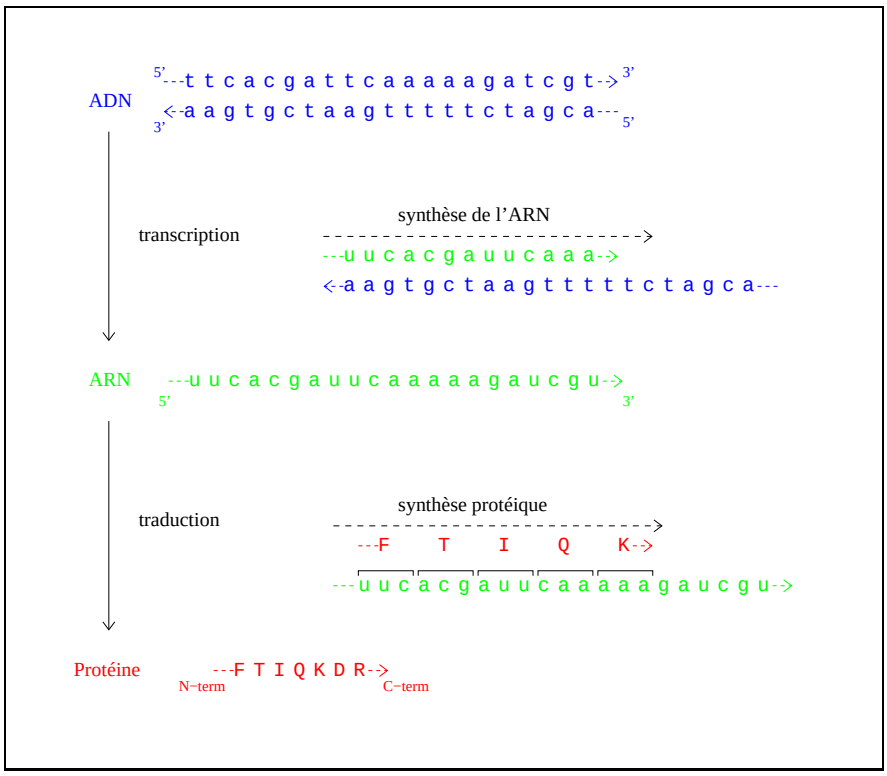

FIG. 1.7 – L'expression de l'information génétique

Le principe de la transcription est proche de celui de la réplication. Comme la synthèse de l'ADN, la synthèse de l'ARN se fait dans le sens 5'→3' et se fonde sur l'appariement des nucléotides de l'ARN synthétisé avec ceux de l'ADN modèle, mais en plaçant un u au lieu d'un t en face d'un a. À la différence de l'ADN, l'ARN est une molécule simple brin : sa synthèse ne prend donc pour modèle qu'un des deux brins de la molécule d'ADN. Pour déterminer la séquence de l'ARN transcrit à partir d'un segment d'ADN double brin, il est donc nécessaire d'identifier le brin transcrit. On ne regarde fréquemment que la séquence d'un des brins d'ADN, celle-ci contenant toute l'information. Dans ce cas, on parle parfois par commodité de sens de transcription plutôt que de brin transcrit.

La plupart des ARN sont traduits, au moins partiellement, en protéines : ce sont des ARN messagers (ARNm). La transcription des ARNm peut en

partie être interprétée comme une étape d'amplification de l'information génétique. En effet, une même séquence d'ADN est généralement transcrite en de multiples molécules d'ARNm qui seront elles-mêmes traduites en de multiples séquences protéiques.

Certains ARN ne sont pas traduits en protéines. La fonction de ces ARN est alors souvent liée à la structure de la molécule dans l'espace. La façon dont cette structure dépend de la séquence a été illustrée dans la section 1.1.1 (page 7).

La traduction de l'ARN en protéine. La traduction utilise un code, dit code génétique, présenté Table 1.1. Celui-ci permet de déterminer la séquence protéique, écrite dans un alphabet à vingt lettres, en fonction de la séquence nucléique, écrite dans un alphabet à quatre lettres. Ce code associe à chacun des 64 triplets de nucléotides l'un des vingt acides aminés ou un signal d'arrêt de la traduction. Les triplets de nucléotides traduits par le code génétique sont appelés codons. La séquence d'ARN est lue comme une suite de codons non chevauchants dans le sens 5'→3' et la séquence protéique est synthétisée dans le sens dit N-term→C-term. Pour déterminer la séquence protéique correspondant à une séquence d'ARN, il faut être capable de retrouver la phase de lecture. Les codons qui entraînent l'arrêt de la traduction (*uga*, *uaa* et *uag*) sont les codons *stop*. En plus des codons *stop*, il existe aussi des codons *start* qui, seuls, permettent de débuter la traduction. Chez les bactéries plusieurs codons *start* sont utilisés (*aug*, *auu* et *agu*). Au contraire chez les cellules eucaryotes, tels que les cellules des végétaux et des animaux, seul le codon *aug* permet de débuter la traduction. À la différence des codons *stop*, les codons *start* apparaissent non seulement aux bornes des régions traduites mais aussi en phase à l'intérieur de ces régions où ils sont traduits conformément au code génétique comme n'importe quel autre codon.

Dans les cellules des organismes « supérieurs » tels que les plantes et les animaux, les régions traduites d'une même protéine ne sont souvent pas d'un seul tenant sur l'ADN. En effet, dans ces cas, l'ARN transcrit subit un épissage au cours duquel les régions non traduites (les introns) sont excisées très précisément de façon à rassembler les régions codantes (les exons) avant la traduction.

Le code génétique est dégénéré puisqu'il fait correspondre 20 acides aminés à 61 codons. Comme le montre la Table 1.1, la plupart des codons synonymes diffèrent uniquement par le nucléotide en troisième position. Les fonctions des protéines synthétisées sont, comme on l'a vu dans la section 1.1.1, largement dépendantes des structures tridimensionnelles qu'elles acquièrent en se repliant dans l'espace. Ce repliement est essentiellement déterminé par la séquence en acide aminé et peut donc être considéré lui aussi comme un niveau d'expression de l'information génétique. Le repliement dépend beaucoup des propriétés physico-chimiques des acides aminés. Certaines de ces

uuu	Phe, F	ucu	Ser, S	uau	Tyr, Y	ugu	Cys, C
uuc	Phe, F	ucc	Ser, S	uac	Tyr, Y	ugc	Cys, C
uua	Leu, L	uca	Ser, S	uaa	STOP	uga	STOP
uug	Leu, L	ucg	Ser, S	uag	STOP	ugg	Trp, W
cuu	Leu, L	ccu	Pro, P	cau	His, H	cgu	Arg, R
cuc	Leu, L	ccc	Pro, P	cac	His, H	cgc	Arg, R
cua	Leu, L	cca	Pro, P	caa	Gln, Q	cga	Arg, R
cug	Leu, L	ccg	Pro, P	cag	Gln, Q	cgg	Arg, R
auu	Ile, I	acu	Thr, T	aau	Asn, N	agu	Ser, S
auc	Ile, I	acc	Thr, T	aac	Asn, N	agc	Ser, S
aua	Ile, I	aca	Thr, T	aaa	Lys, K	aga	Arg, R
aug	Met, M	acg	Thr, T	aag	Lys, K	agg	Arg, R
guu	Val, V	gcu	Ala, A	gau	Asp, D	ggu	Gly, G
guc	Val, V	gcc	Ala, A	gac	Asp, D	ggc	Gly, G
gua	Val, V	gca	Ala, A	gaa	Glu, E	gga	Gly, G
gug	Val, V	gcg	Ala, A	gag	Glu, E	ggg	Gly, G

TAB. 1.1 – Le code génétique. Les couleurs correspondent à certaines caractéristiques physico-chimiques des acides aminés : rouge - aliphatique, bleu - hydrophobe, cyan - groupe hydroxyle, vert - groupe sulfure, jaune - aromatique, magenta - acide.

propriétés sont indiquées dans la Table 1.1. On peut remarquer que le nucléotide en seconde position du codon a plus souvent une influence sur les propriétés physico-chimiques de l'acide aminé codé que le nucléotide en première position.

Signaux

La présence de signaux le long des séquences nucléiques et protéiques est essentielle à l'expression de l'information génétique. On peut citer notamment les signaux sur la séquence d'ADN qui permettent le choix des segments transcrits, ou les signaux sur l'ARN qui déterminent les limites des introns et le choix du site de début de traduction. Le début de l'ARNm, par exemple, contient le site de fixation du ribosome ou RBS (*Ribosome Binding Site*) qui joue un rôle déterminant dans le choix du site de démarrage de la traduction. Des signaux existent aussi sur les protéines : on peut citer notamment des signaux présents sur certaines séquences qui permettent la reconnaissance du compartiment de la cellule vers lequel la protéine doit être adressée.

Les signaux peuvent être décrits comme l'occurrence de séquences particulières. Cependant, bien souvent, un grand nombre de séquences correspondent à un même signal et le signal doit donc plutôt être décrit comme un motif plutôt qu'une séquence particulière. La situation est encore compliquée par le fait que des séquences interprétées comme un signal dans certains contextes peuvent aussi apparaître « par hasard » ailleurs le long de la sé-

quence sans pour autant jouer le rôle d'un signal.

Pour illustrer le rôle des signaux sur la séquence d'ADN, nous nous appuierons sur l'exemple de la régulation de la transcription de l'opéron lactose chez *E. coli*. Les gènes *lacZ*, *lacY* et *lacA* sont transcrits en un même ARNm et constituent donc ce qu'on appelle un opéron. L'expression de ces gènes en protéines permet à la bactérie d'utiliser le lactose comme source d'énergie. Cette expression est régulée au niveau de la transcription de telle sorte qu'en l'absence de lactose, les gènes ne sont pas transcrits ; en présence de lactose les gènes ne sont transcrits que si il n'y a pas de glucose disponible comme source d'énergie (le glucose est une meilleure source d'énergie que le lactose). Cette régulation, comme la plupart des régulations transcriptionnelles, s'exerce au niveau du démarrage de la transcription. Elle fait intervenir la fixation de protéines sur des sites particuliers de la séquence d'ADN, ces sites sont situés en amont de la région transcrite dans une région appelée promoteur. La figure 1.8 présente l'organisation des gènes sur la séquence d'ADN et montre la position des sites de fixation de l'ARN polymérase et de deux régulateurs transcriptionnels : un activateur, la protéine CAP, qui se fixe en l'absence de glucose, et un répresseur, la protéine LacI, qui se fixe en l'absence de lactose. Lorsqu'aucun des deux régulateurs n'est fixé à l'ADN, le niveau de transcription est bas car l'ARN polymérase n'a qu'une affinité relativement faible pour son site de fixation présent sur cette séquence. En l'absence de glucose, la fixation de la protéine CAP facilite la fixation de l'ARN polymérase ce qui augmente beaucoup le niveau de la transcription. En l'absence de lactose, la fixation du répresseur LacI empêche l'ARN polymérase de se positionner sur son site de fixation et donc la transcription n'a pas lieu, que la protéine CAP soit fixée ou non.

Le terme de gène a été introduit bien avant la compréhension des mécanismes moléculaires de l'hérédité pour désigner le support matériel d'un caractère héréditaire observable (macroscopique). Une fois identifié l'ADN comme étant le support de l'information génétique, le gène a naturellement désigné le segment d'ADN qui porte l'information responsable du caractère héréditaire.
En biologie moléculaire, le terme est utilisé dans un sens général pour désigner tout segment d'ADN qui contient une séquence qui s'exprime en ARN ou en protéine. Les séquences qui participent à la régulation de cette expression sont aussi censées faire partie du gène (lorsqu'elles sont connues). Cependant on appelle souvent gène la seule région traduite en protéine.
L'utilisation du terme de gène en biologie moléculaire est donc large et assez floue. Elle n'est de plus pas directement liée à l'existence d'un caractère observable.

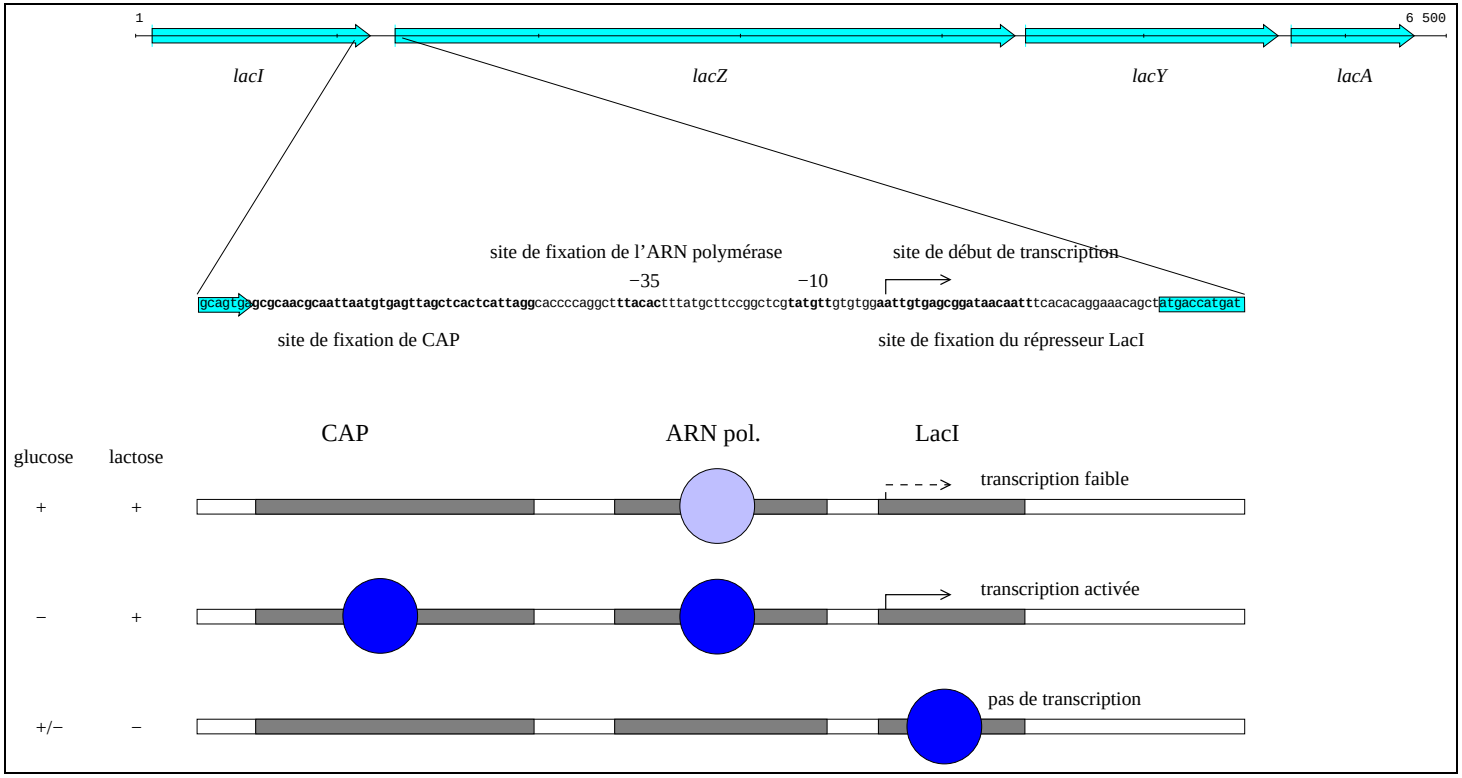

FIG. 1.8 – Régulation de la transcription de l'opéron lactose d'*E. coli*.

La comparaison des séquences révèle l'évolution

Les cellules vivantes descendent toutes d'un ancêtre commun et les séquences des gènes gardent la trace de cette évolution. À titre d'exemple, la séquence protéique du lysozyme que l'on trouve dans l'œuf de poule (présenté figure 1.4) et du lysozyme du ver à soie sont alignées ci-dessous.

```
43.220% identity (45.536% ungapped) in 118 aa overlap (1-118:1-112)

               10        20        30        40        50        60
6lyz_  KVFGRCELAAAMKRHGLDNYRGYSLGNWVCAAKFESNFNTQATNRNTDGSTDYGILQINS
       :.: :: :.  ...::...     . :::: .. ::. .:. :: : .:: ::::.:::.
1gd6A  KTFTRCGLVHELRKHGFEENL---MRNWVCLVEHESSRDTSKTNTNRNGSKDYGLFQIND
               10        20           30        40        50

               70        80        90       100       110       120
6lyz_  RWWCNDGRTPGSRNLCNIPCSALLSSDITASVNCAKKIVSDGNGMNAWVAWRNRCKGTDV
       :.::. : .::.   ::. :: ::..::: ...:::::: .  . ..:: .:.:.:.:.
1gd6A  RYWCSKGASPGKD--CNVKCSDLLTDDITKAAKCAKKIYKR-HRFDAWYGWKNHCQGSLP
        60        70          80        90        100       110


6lyz_  QAWIRGCRL

1gd6A  DISSC
```

Certaines séquences des ARN de structure sont très conservées et sont de plus présentes dans toutes les cellules vivantes. Elles permettent de tenter de reconstruire l'arbre phylogénétique qui retrace les relations de parenté entre tous les organismes. Leur étude a conduit au début des années 1970 à subdiviser les procaryotes en deux groupes : les eubactéries et les archées. D'un point de vue phylogénétique, ces deux grands groupes sont aussi différents l'un de l'autre qu'ils le sont des eucaryotes.

1.2 Séquençage et analyse des génomes

Ome is where the art is

...
The Investome—the complete sum of all the money invested in biotechnology companies. Investome researchers study the apparently random sequence of events and interactions that modulate events in the buy-sell cycle in biotechnology. Over the past two decades or so, a great deal of effort has been devoted to understanding why so much of the biotechnology investome is devoted to junk. One hypothesis is that biotechnology is sustained by pathways that link a gradient of credibility to the production of corporate energy and hype. As money from investors flows down that gradient (from barely conceivable to almost believable), flashes of light are emitted. Early in the cycle, these flashes lure more investors to other unlikely projects. Thus, the cycle is apparently self-sustaining. However, with time investors become immune to the attractant light and biotechnology has to adapt to new circumstances.
...

Éditorial de *Nat. Biotech.*, décembre 1999.

1.2.1 La biologie à « haut débit »

Au cours des années 90, de nouvelles technologies ont permis d'augmenter considérablement la quantité disponible de certains types de données. Ces technologies qualifiées, de façon assez évocatrice, de technologies à « haut débit » ou à « grande échelle » ont eu un profond impact sur la biologie moléculaire.

Aussi bien par son importance actuelle que d'un point de vue chronologique, **le séquençage de l'ADN** peut être considéré comme la première des technologies de la biologie à grande échelle. Il permet aujourd'hui d'accéder au génome complet d'un organisme vivant, c'est à dire à l'ensemble de son patrimoine génétique. Les principales autres technologies se sont développées, au moins en grande partie, dans le cadre de l'interprétation des séquences de génomes. Elles concernent essentiellement l'étude des ARNm et des protéines. S'inscrivant dans une démarche de connaissance globale de la chimie des organismes vivants, les technologies de la biologie à grande échelle ont entraîné une prolifération des termes en « -ome » dont les plus utilisés sont ceux de transcriptome et de protéome qui désignent respectivement l'ensemble des ARN transcrits et des protéines traduites. Il existe cependant une différence fondamentale entre le génome et les autres « omes ». En effet, le génome ne varie pratiquement pas dans un organisme vivant ; au contraire,

le transcriptome et le protéome changent, selon les sources d'énergies disponibles, le type de cellule considéré dans un organisme multicellulaire, ou au cours du cycle cellulaire, par exemple.

En dehors du séquençage de l'ADN, on peut citer parmi les plus importantes des technologies de la biologie à grande échelle :

Le séquençage des ARNm eucaryotes. Il s'agit souvent du séquençage d'EST (*Expressed Sequence Tag*). Les EST sont des séquences le plus souvent partielles provenant d'un séquençage unique (donc d'assez mauvaise qualité) de la séquence d'ADN complémentaire d'un ARNm. Ces séquences sont principalement utilisées pour identifier les séquences transcrites du génome et pour donner des indications sur les gènes transcrits dans les différents tissus d'un organisme. De plus, dans une certaine mesure, le séquençage d'EST peut donner des indications sur le niveaux d'expression des gènes. Toutefois, les ARNm transcrits en un petit nombre de copies ne sont en général pas séquencés. À coté du séquençage des EST, des projets de séquençage de bonne qualité d'ARNm complets sont aussi menés (voir par ex. [KSS+01]).

La technologie des lames de verres et puces à ADN. Là encore, l'objectif est d'identifier les gènes transcrits dans différentes conditions cellulaires (voir par ex. [SSZ+98]). La technologie consiste à fixer sur un petit support (une lame de verre ou une puce) de façon ordonnée (en tableau) un grand nombre de fragments d'ADN complémentaires des ARN supposés transcrits. La mesure de l'expression se fait ensuite en mettant le support en présence d'ARN marqués extraits de cellules. Les ARN s'hybrident alors avec les ADN complémentaires fixés sur le support, ce qui permet de repérer les gènes transcrits. On étudie ainsi par exemple simultanément l'expression des quelque 4 000 gènes d'une bactérie comme *B. subtilis* (voir par ex. [AYK+03]). Ces expériences posent encore souvent des problèmes de reproductibilité et seules les variations importantes de niveaux d'expressions peuvent être mises en évidence avec certitude.

L'analyse des protéines par spectrométrie de masse. Cette méthode n'est pas vraiment nouvelle mais a connu des améliorations. Couplée à une électrophorèse bidimensionnelle qui sépare les différentes protéines d'un extrait cellulaire, la spectrométrie de masse permet d'étudier directement la séquence de nombreuses protéines (voir par ex. [WWY01]). Si la séquence du génome est connue, la séquence exacte de la protéine peut être déterminée. De plus, la méthode permet d'accéder aux modifications chimiques des protéines après la traduction.

L'étude des interactions entre protéines. La méthode du double hybride permet de rechercher à grande échelle les interactions possibles entre protéines (voir par ex. [RSR+01]). L'interprétation des résultats nécessite beaucoup de précautions, notamment parce que de nom-

breuses interactions qui n'ont pas lieu dans la cellule peuvent être détectées.

Dans les années à venir, on peut probablement s'attendre à ce que d'autres technologies soient développées et prennent une grande importance. Cependant, l'étude des séquences d'ADN devrait conserver une place centrale : d'une part, l'ADN contient l'ensemble de l'information génétique et, d'autre part, ces séquences peuvent être connues de façon fiable et relativement peu onéreuse. À titre indicatif, le coût du séquençage de bonne qualité d'un génome microbien complet est aujourd'hui de l'ordre de $0,1$ \$ par base dans certains grands centres de séquençage [FEN+02], le coût d'une puce à ADN (non réutilisable) fabriquée en série (donc sur un organisme déjà étudié) est d'environ 1000 \$ [Til03]

Pour permettre de mieux appréhender l'ampleur et la rapidité de l'introduction de technologies à haut débit en biologie, la section suivante commente quelques temps forts de la chronologie du séquençage de l'ADN. Les conséquences de ces changements technologiques sont le sujet de la section 1.2.2.

Petite chronologie du séquençage des génomes

Le séquençage pendant la période pré-génomique. Après la découverte de la structure de l'ADN en 1953 et la reconnaissance de son rôle comme support de l'information génétique par l'ensemble de la communauté scientifique, les grands principes de l'expression de l'information génétique ont été élucidés durant les années 50 et 60. En dépit de la compréhension de ces mécanismes, l'information génétique est restée largement inaccessible jusqu'à la mise au point de techniques efficaces de séquençage de l'ADN à la fin des années 70, notamment par la méthode de Sanger [SNC77] qui est aujourd'hui la plus utilisée. Dans les années 80 le séquençage de quelques dizaines de milliers de bases était possible. Ainsi, dès 1982, la séquence des 48 502 paires de bases (bp) du génome d'un phage (virus de bactérie) de la bactérie *E.coli*, le phage λ, fut déterminée [SCH+82]. Dès cette époque, le projet du séquençage complet du génome (4.6 Mbp) de *E. coli*, la bactérie plus étudiée, et même celui du séquençage du génome humain étaient évoqués [BPB+97, Ols93, VAM+01, LLB+01].

Cependant il restait à réaliser un saut quantitatif de plusieurs ordres de grandeur entre le séquençage de quelques dizaines de milliers de paires de bases et le séquençage des plusieurs millions de paires de bases qui constituent un génome bactérien. Il s'agissait même d'un saut gigantesque si l'on considère le séquençage d'un génome de plusieurs milliards de paires de bases comme le génome humain. Ce saut a été réalisé au cours des années 90 essentiellement grâce au financement de grands projets, qui ont accéléré l'automatisation du séquençage, et à la mise en place de grandes infrastructures.

1988-1992 : Le lancement des grands projets de séquençage. Le *Human Genome Project* (HGP) prend forme en 1988 dans un rapport du *National Research Council* américain et son financement par l'administration américaine débute vraiment en 1990. L'objectif est d'aboutir au séquençage complet du génome humain en 2005. Cet échéancier est optimiste car la tâche est inaccessible aux technologies du début des années 90. Il rend compte de la nature volontariste de la politique de financement du HGP. Les objectifs de cette politique sont multiples. Sur le plan scientifique, on peut distinguer au moins deux niveaux d'objectifs [Ols93]. Le premier niveau est l'espoir d'importants progrès en biologie humaine et en médecine. Le séquençage du génome est notamment perçu comme le moyen de transférer, par la comparaison des séquences, les connaissances acquises sur d'autres espèces (des espèces modèles) vers la biologie humaine. Le second niveau d'objectif est plus général : c'est la volonté de favoriser le développement des technologies de séquençage de l'ADN.

Parallèlement, des projets de séquençage de génomes de micro-organismes modèles, tels que *Saccharomyces cerevisiae* (levure de bière) et *Bacillus subtilis* ont aussi débuté. Ces projets démarrent respectivement en 1989 et 1990 et sont menés dans le cadre de consortiums impliquant de nombreux laboratoires. Le TIGR (*The Institute for Genomic Research*), institution fondée en 1992 et financée conjointement par des fonds publics et privés, a aussi joué un rôle de premier plan dans l'évolution des conditions de séquençage.

1995-1998 : Les premiers génomes complets. En 1995, ces efforts aboutissent à la publication des premières séquences de génomes complets par le TIGR. Il s'agit de deux bactéries, pathogènes pour l'Homme, *Haemophilus influenzae* (1.8 Mbp) [FAW+95] et *Mycoplasma genitalium* (0,58 Mbp) [FGW+95]. Les séquences des micro-organismes modèles suivent en 1996 et 1997 : *E. coli*, *S. cerevisiae* (13 Mbp) et *B. subtilis* (4.2 Mbp). Le premier séquençage de génome d'un organisme multicellulaire, le ver nématode *Caenorhabditis elegans*, est terminé quant à lui en 1998 (97 Mbp) dans le cadre du HGP dans lequel il a été intégré en tant qu'organisme modèle.

Il est révélateur que le premier organisme « séquencé » (*H. influenzae*) ait été choisi par le TIGR en tant que génome bactérien « typique » et non pour sa spécificité (pathogénie ou autre). Il s'agissait en fait de démonter que, sans connaissance préalable ou presque, le génome de la plupart des bactéries peuvent être séquencées. La technique employée, dite du *shotgun*, n'est pas compliquée. Elle consiste à casser les molécules d'ADN en petits morceaux, puis à réaliser le séquençage de morceaux choisis aléatoirement et enfin d'assembler, grâce à leurs chevauchements, les séquences obtenues. La technique n'est pas non plus nouvelle, puisqu'elle a déjà permis le séquençage du phage λ en 1982. Mais c'est la preuve que, grâce à quelques améliorations, elle est applicable à des séquences beaucoup plus longues. La démonstration

peut sans doute être considérée comme le point de départ de l'accélération fulgurante de l'avancement des projets de séquençage et de l'augmentation du nombre de projets à partir de la fin des années 90.

1998-2001 : L'accélération du séquençage. En 1998, l'aboutissement du séquençage du génome humain pour 2005 semble encore incertain. Le HGP a fait beaucoup d'efforts de cartographie, notamment dans le but de faciliter l'assemblage final des séquences, mais n'a déterminé qu'environ 5% de la séquence. C'est à cette date que la société privée *Celera Genomics* est créée par J.C. Venter (déjà fondateur du TIGR). *Celera* se déclare concurrente du HGP pour le séquençage du génome humain en annoncant qu'elle compte obtenir en trois ans la séquence par *shotgun* à l'échelle du génome. En guise de préparation, les installations mises en place sont utilisées la première année pour le séquençage du premier insecte, la mouche du vinaigre *Drosophila melanogaster* (un autre organisme modèle).

En 1999, le HGP se lance lui aussi dans une phase de production massive de séquences par *shotgun* avec l'objectif d'obtenir pour 2001, soit en même temps que *Celera*, une première version de la séquence. En 2001, le HGP [LLB+01] et *Celera* [VAM+01] annoncent simultanément les premières versions de la séquence couvrant environ 95% du génome (mais à peu près toutes les régions où sont situés les gènes). Dix-huit mois environ ont suffis aux deux organisations pour réaliser la plus grande partie du séquençage du génome humain. Lors de sa publication, la séquence d'une trentaine de génomes de micro-organismes (majoritairement des bactéries pathogènes) était déjà publiée. Toutefois, la longueur du génome humain atteignait tout de même à 8 fois la longueur cumulée de l'ensemble des autres génomes séquencés.

La situation actuelle Fin août 2003, les génomes complets (ou presque) de 132 bactéries, 16 archées, 5 eucaryotes unicellulaires, 4 eucaryotes multicellulaires sont « annotés » et accessibles gratuitement dans les banques de données[1]. Le génome humain est presque complet et ceux de trois autres vertébrés sont bien avancés (la souris *Mus musculus*, le rat *Rattus norvegicus* et un poisson modèle *Danio rerio*). Sept plantes d'intérêt agroalimentaire sont aussi en cours de séquençage.

Comme on pouvait s'y attendre, le choix des organismes est très anthropocentrique. En plus du séquençage du génome humain, la plupart des organismes séquencés présentent un intérêt direct pour l'Homme : certains ont un intérêt scientifique immédiat comme les organismes modèles qui ont déjà été beaucoup étudiés ; d'autres présentent un intérêt médical, comme les pathogènes de l'Homme, ou encore un intérêt agroalimentaire, comme les bactéries lactiques, les animaux et plantes de la ferme ainsi que certains de leurs pathogènes ; d'autres, enfin, sont intéressants sur un plan technologique,

[1] http ://www.ncbi.nlm.nih.gov :80/entrez/

comme les micro-organismes vivant dans des milieux inhospitaliers (chaud, salé, ...) dont les molécules peuvent présenter des propriétés remarquables.

Les travaux actuels restent dans cette optique, notamment par le séquençage d'organismes très proches d'un point de vue évolutif. À titre d'exemple, les génomes complets de 4 souches de la bactérie *E. coli* ont déjà été publiés et au moins 2 autres sont en cours de séquençage. Par ailleurs, les États-Unis, l'Angleterre, le Japon, la Chine, le Canada ainsi que des entreprises privées, ont annoncé en octobre 2002 le financement de ce qui est présenté comme la seconde phase du séquençage du génome humain. L'objectif est l'étude de la variabilité du génome humain par un séquençage au moins partiel de nombreux individus.

À moyen terme, après le séquençage du laboratoire, de l'Homme, de l'hôpital et de la ferme, on peut penser que le spectre des organismes séquencés s'élargira considérablement. Cet élargissement devrait offrir de nouvelles perspectives, notamment pour l'étude des micro-organismes incroyablement divers qui peuplent la biosphère et qui ne sont pas, pour la plupart, cultivables en laboratoire à l'heure actuelle.

1.2.2 Analyse des génomes et biologie moléculaire *in silico*

Informatique et statistique pour l'analyse de génome. Bien avant l'essor de la biologie à grande échelle, des algorithmes étaient déjà nécessaires pour comparer les rares séquences disponibles. Ainsi, dès 1970, Needleman et Wunsch [NW70] décrivaient un algorithme de comparaison de deux séquences protéiques. C'est cependant au cours des années 90, avec l'avènement des technologies de la biologie à grande échelle (et des micro-ordinateurs) que l'utilisation de l'informatique s'est généralisée afin de stocker, intégrer et tenter d'interpréter les énormes quantités de données produites.

Parallèlement à l'informatique, les statistiques se sont révélées très utiles pour répondre à certaines questions, notamment pour quantifier l'incertitude liée à l'interprétation des données récoltées. Par exemple, quand peut on dire qu'une similitude trouvée en comparant une séquence à un ensemble de séquences n'est pas due au « hasard » ? En amont de ces problèmes, qui relèvent principalement du calcul de significativité, les statistiques ont aussi largement guidé la conception des méthodes d'extraction d'information. Par exemples, le choix de la mesure à étudier pour l'identification des gènes au sein d'une séquence nucléique, ou encore celui de la fonction de score à maximiser lors de l'alignement de deux séquences, utilisent des méthodes statistiques.

Les premières surprises de l'analyse des génomes. Le premier constat qui mérite d'être fait est qu'il n'y a pas eu de grosse surprise. Cependant, certains phénomènes se sont révélés inattendus par leur ampleur.

Ainsi chez les bactéries, la proportion de gènes de fonction inconnue a-t-elle surpris. Lors de l'annotation des génomes des deux principales bactéries modèles *E. coli* et *B. subtilis* en 1997, aucune fonction n'a pu être proposée pour respectivement 38% et 42% des protéines [BPB+97, KOM+97]. Malgré des milliers d'expériences réalisées depuis plus de 40 ans, seul environ un tiers des quelque 4 000 gènes avaient fait l'objet d'études expérimentales. Toutefois, pour une partie des autres gènes, une fonction plus ou moins précise a pu être proposée par la mise en évidence de similitudes de séquence avec des protéines de fonction connue. C'est le cas de 28% des protéines de *B. subtilis*.

Depuis l'annotation des bactéries modèles, la proportion des gènes de fonction inconnue a plutôt augmenté. Les nouvelles bactéries séquencées n'ont pas fait l'objet d'études expérimentales d'une ampleur comparable à celles menées sur les bactéries modèles. Ainsi chez ces bactéries, la proportion de gènes codant pour des protéines de fonction inconnue est de l'ordre de 40 à 50%. À titre d'exemple, lors de l'annotation du génome d'*Oceanobacillus iheyensis* [TTU02] une bactérie certes inconnue et vivant dans les fonds marins mais assez proche d'un point de vue phylogénétique de *B. subtilis*, aucune fonction n'a pu être proposée pour 44% des protéines. Parmi ces dernières, 70% présentaient des similitudes de séquence avec d'autres protéines de fonction inconnue.

À coté de cette relative surprise, la seconde surprise concerne l'importance des transferts génétiques horizontaux entre bactéries, c'est à dire l'intégration dans le génome d'ADN provenant d'autres bactéries. Sur un plan quantitatif, ceux ci concerneraient plus de 10% des protéines codées chez de nombreuses bactéries [MRV+91, OLG00]. D'un point de vue qualitatif, les transferts génétiques horizontaux sont apparus à l'origine de l'acquisition de fonctions essentielles à l'adaptation des bactéries à leur niche écologique. Ils concernent notamment les fonctions de pathogénicité, de symbiose ou de protection vis à vis de substances toxiques telles que les antibiotiques ou les métaux lourds. Les transferts horizontaux ont même été supposés responsables de l'organisation en opérons des gènes bactériens [Law99]. Enfin, les transferts horizontaux expliquent sans doute également l'existence d'une fraction importante des protéines n'ayant aucune similitude de séquence avec d'autres protéines.

Le séquençage du génome humain a également été à l'origine de surprises. Parmi celles-ci, le nombre relativement faible de gènes codant pour des protéines constitue certainement l'observation la plus surprenante : attendus autour de 100 000, ils ne seraient finalement que 30 000 à 40 000 soit environ 10 fois le nombre observé chez une bactérie comme *E. coli* ou *B. subtilis* [LLB+01]. Ce rapport faible contraste avec l'apparente différence de complexité des organismes considérés mais souligne sans doute la plus grande complexité du contrôle de l'expression de l'information génétique chez l'Homme. D'une part, il semble qu'au moins 40 à 60% des ARN transcrits

chez l'Homme, puissent être épissés de plusieurs façons différentes ; ainsi, dans de nombreux cas, un même gène peut coder plusieurs protéines différentes [ML02]. D'autre part, moins de 2% du génome humain est traduit en protéines, contre environ 85 à 90% chez les bactéries. Les fonctions associées au reste du génome sont mal comprises. De façon certaine, nombre de régions qui ne codent pas pour des protéines sont impliquées dans des mécanismes de régulation de l'expression génétique que ce soit au niveau de la transcription ou de la traduction. Cependant, le génome contient aussi probablement beaucoup de régions sans réelle « fonction » biologique comme les quelque 40% du génome constitués d'éléments répétés un grand nombre de fois.

Les enjeux de l'analyse des génomes La disponibilité du génome complet d'un organisme est d'un grand intérêt pratique. Par exemple :

- Si l'on parvient, par l'étude de la co-occurrence d'un marqueur génétique et d'une maladie, à localiser approximativement un gène impliqué dans cette maladie, la disponibilité de la séquence peut permettre d'identifier le gène.
- Si l'on dispose de deux souches de bactéries dont l'une est pathogène et l'autre non, la comparaison des génomes peut aider à identifier les gènes responsables du caractère pathogène.
- Si l'on s'intéresse à une fonction particulière pour laquelle des exemples de gènes sont connus, la disponibilité du génome complet peut permettre de constituer très rapidement un ensemble de gènes candidats.

Dans tous ces cas la séquence est d'une aide incomparable car elle permet de choisir rapidement les cibles pour des études expérimentales plus précises. Le séquençage d'un génome microbien est même aujourd'hui considéré comme un préalable quasi indispensable à des études de biologie moléculaire efficaces.

D'un point de vue beaucoup plus fondamental, un génome est une source exceptionnelle d'informations sur l'évolution des organismes vivants. En effet, un génome contient la trace de certains processus évolutifs, à travers les répétitions de séquences, les hétérogénéités de composition, ou la présence de gènes fossiles non fonctionnels. La comparaison de génomes constitue une source d'information encore plus riche car elle confère une vision globale des fondements génétiques des différences entre organismes.

Ces utilisations des séquences génomiques se situent aux extrémités d'un éventail allant de l'aspect purement utilitaire à la connaissance fondamentale. La connaissance de la fonction des séquences génomiques détermine évidemment en grande partie la puissance de ces utilisations. Ceci nous amène au cœur du problème de l'analyse des génomes : identifier la fonction des séquences disponibles. Le génome contenant toute l'information génétique, son séquençage constitue une première avancée vers une connaissance globale de la chimie des organismes vivants. Mais celle-ci doit être poursuivie par un

effort d'identification de la fonction des séquences. Des expériences à grande échelle sont menées à cet effet et comprennent notamment des analyses du transcriptome, ou de grands projets d'analyse fonctionnelle, comme celui visant à étudier chez *B. subtilis* [KEA+03] les phénotypes associés à la perte de gènes de fonction inconnue. L'analyse *in silico* des génomes joue également un rôle de premier plan, pour l'interprétation des résultats expérimentaux, mais aussi pour pousser le plus loin possible l'interprétation de la séquence elle-même.

Rechercher la fonction à partir de la séquence. Les questions sont nombreuses au niveau de la séquence d'ADN : où sont les régions qui codent pour des protéines ? Où sont les signaux ? Où sont les ARN non traduits ? Différentes approches pour la prédiction des régions codantes sont présentées en annexe A.

Une fois les régions codant les protéines déterminées, ou seulement prédites, la recherche de la fonction de ces protéines constitue l'un des principaux objectif de l'analyse *in silico* des séquences. Il s'agit d'un problème très large. En effet, on entend sous le terme de « fonction » d'une protéine à la fois la fonction moléculaire, la fonction cellulaire, et la fonction phénotypique (voir par exemple [BDDL+98]). De plus, même au niveau moléculaire, la fonction peut être décrite avec différents niveaux de résolution. Ainsi, à faible résolution, la fonction moléculaire de LacI est de se fixer à l'ADN ; à moyenne résolution, LacI est un répresseur de la transcription ; enfin, à haute résolution, LacI est le répresseur de la transcription de l'opéron lactose qui se fixe sur le site indiqué figure 1.8 et qui change de conformation lorsqu'il est lié au lactose modifiant ainsi sa capacité de fixation à l'ADN.

Actuellement, la seule méthode un peu systématique pour attribuer *in silico* une fonction à une protéine passe par la mise en évidence de similitudes avec des protéines de fonction connue. Cette approche est dite extrinsèque. À de rares exceptions près, deux protéines similaires au niveau de la séquence partagent certaines propriétés, au moins au niveau de leur fonction moléculaire.

Cependant, dans de nombreux cas, aucune similitude de séquence avec une protéine de fonction connue ne peut être trouvée. Dans ces cas, deux grandes directions sont possibles. Elles passent, soit par la mise au point de méthodes permettant de trouver des similitudes plus lointaines, soit par l'étude des propriétés intrinsèques de la séquence. Le contexte du gène sur la séquence peut aussi être une source d'information notamment chez les bactéries, où les gènes d'un même opéron participent souvent à une même fonction cellulaire.

Pour conclure, le fossé est immense entre la simple connaissance de la séquence figée et linéaire d'un génome et la compréhension de l'organisation dynamique et spatiale des phénomènes biologiques. Parmi les méthodes qui

peuvent aider dans cette tâche, les approches *in silico* ont un rôle à jouer d'autant plus important que la quantité de séquences connues augmente à un rythme que les méthodes expérimentales de détermination de la fonction ne suivent pas. D'autre part, mener le plus loin possible l'analyse *in silico* des séquences se justifie aussi car les séquences génomiques seront sans doute bientôt, et resteront probablement longtemps, les seules données moléculaires disponibles sur les micro-organismes non cultivables.

1.3 Modéliser les séquences

> ...
> Les Babyloniens sont peu spéculatifs. Ils acceptent les décisions du hasard, ils lui livrent la vie, l'espoir, la terreur panique, mais ils ne s'avisent pas d'interroger ses lois vertigineuses, et les sphères giratoires qui le révèlent n'éveillent pas leur curiosité. Cependant, la déclaration officieuse que j'ai rapportée inspira beaucoup de discussions de caractère juridico-mathématique. De l'une d'elles surgit la conjecture suivante : si la loterie est une intensification du hasard, une infusion périodique du chaos dans le cosmos, ne conviendrait-il pas que le hasard intervînt dans toutes les étapes du tirage et non pas dans une seule ?
> ...
>
> J.L. Borges, *Fictions – La loterie de Babylone* [2].
> Traduction Ibarra

La modélisation probabiliste des séquences consiste à voir celles-ci comme le résultat de tirages aléatoires : le modèle attribue à chaque séquence possible une probabilité d'apparition. À travers la définition de la probabilité d'apparition des différentes séquences, c'est en fait une description des séquences que le modèle (ou plutôt le modélisateur) propose.

Comment décrire les séquences à travers un modèle probabiliste ? Quelles sont les principales approches pour utiliser ces modèles en vue d'interpréter les séquences ? Quel est le sens de la description et de l'interprétation des séquences biologiques par ces modèles ? Voilà les grandes questions auxquelles cette section va essayer d'apporter quelques réponses. La section s'achève par une présentation de l'intérêt des modèles de chaînes de Markov cachées.

1.3.1 Exemples introductifs

Deux exemples sont présentés ici pour introduire l'utilisation des modèles probabilistes de séquences. Il s'agit dans les deux exemples de déterminer la fonction (ou nature) d'une séquence observée. Dans le premier exemple, l'objectif est de pouvoir dire si la séquence correspond ou non à un site de fixation d'une protéine sur l'ADN. Dans le second exemple, le problème consiste à déterminer si la séquence se situe dans une région qui code pour une protéine (on dira codante), et le cas échéant, quelle est sa phase de lecture. Dans ces deux cas, des jeux de séquences de natures connues, dits ensembles d'apprentissage, sont utilisés pour définir des modèles probabilistes correspondant aux différentes natures de séquences : « site » ou « non site », dans

[2] Jolie citation utilisée par Christian Robert en avant propos de la présentation des méthodes MCMC [Rob96]

gène	séquence promotrice
abrB-P2	aaacaaaatgattgacgattatt---ggaaaccttgttatgctatgaag
acsA	tattttaaaaattgagaagaata---tgaatatatactataataattgt
acuA	tataaaccattgtgaaaacgctt----tataatttggtattcttaaaga
addA	atagatcagattggtcattttcg---tcaacattcgataaaatatag
ald	ttaatcaaacaaagaatttcca---aaatatcaagctacactaaaaat
alkA	aagataacaaaatgagtaaagat----gattatgtgataaactaatttc
alsS	tcgaatcgatattggaggtcaat--ttccaaagagtgtatagtgaaactt
amyLY	actctgccaagttgtttgatag---agtgattgtgataatttaaatgt

TAB. 1.2 – Quelques séquences de promoteurs dépendant de Sigma A chez *B. subtilis* [Hel95]. Les boîtes -35 et -10 sont respectivement colorées en rouge et en vert. La fin de la séquence correspond au site d'initiation de la transcription. Les tirets sont introduits de façon à aligner les boîtes -35 et -10 des différentes séquences.

le premier exemple ; « intergénique » ou « codante en phase i », dans le deuxième exemple.

Recherche de sites

La table 1.2 montre une partie de la séquence d'ADN de quelques promoteurs de *B. subtilis*. Les séquences présentées sont connues expérimentalement pour être des sites de fixation de la sous-unité Sigma A de l'ARN polymérase. Ces sites sont classiquement décrits comme étant composés de deux boîtes nommées boîte -35 et boîte -10, en référence à leur distance par rapport au site d'initiation de la transcription. Les séquences de ces boîtes sont particulièrement importantes pour la fixation de l'ARN polymérase et peuvent être considérées comme des signaux positionnés sur la molécule d'ADN.

On cherche à déterminer si une séquence $x_1^6 = (x_1, x_2, \ldots, x_6)$ de six nucléotides, ne faisant pas partie des exemples connus, correspond ou non à la séquence d'une boîte -10 (que l'on appellera site ici). Les approches statistiques naturelles pour répondre à cette question se fondent sur la modélisation probabiliste de l'apparition des séquences correspondant à un site, et de celles qui n'y correspondent pas. La séquence observée x_1^6 est alors vue comme la réalisation d'un vecteur aléatoire X_1^6.

L'observation des quelques séquences présentées permet de remarquer rapidement que la fréquence de chacun des nucléotides diffère d'une position à l'autre au sein du site. Le modèle « site » le plus simple qui tienne compte de cette propriété considère que les nucléotides apparaissent indépendamment et selon une loi d'apparition différente à chaque position. Sous ce modèle dit hétérogène, la probabilité d'apparition de la séquence observée x_1^6 s'exprime

	a	**g**	**c**	**t**
pos. 1	0,03	0,01	0,06	0,90
pos. 2	0,95	0,01	0,00	0,04
pos. 3	0,22	0,04	0,13	0,61
pos. 4	0,80	0,06	0,03	0,11
pos. 5	0,69	0,06	0,17	0,08
pos. 6	0,04	0,00	0,02	0,94

TAB. 1.3 – Fréquences d'apparition des nucléotides aux différentes positions des boites -10 des promoteurs dépendant de Sigma A. Le calcul a été effectué à partir de 142 sites de fixation de Sigma A déterminés expérimentalement [Hel95].

comme le produit des probabilités d'apparition de chacun de ses nucléotides

$$P_s(X_1^6 = x_1^6) = b_1(x_1) \times b_2(x_2) \times \ldots \times b_6(x_6) \ ,$$

où $b_t(x)$ correspond à la probabilité de l'apparition du nucléotide x en position t ($b_t(x) = P(X_t = x)$). Si l'ensemble d'apprentissage est suffisamment grand, les paramètres b_t sont le plus souvent estimés par les fréquences empiriques d'apparition des nucléotides dans l'ensemble d'apprentissage. Ceci revient en fait à utiliser l'estimateur du maximum de vraisemblance des b_t sur l'ensemble d'apprentissage. Ces fréquences sont présentées table 1.3.

Pour les séquences qui ne correspondent pas à des sites, un modèle simple consiste à considérer les nucléotides comme apparaissant indépendamment les uns des autres avec une fréquence qui ne dépend pas de leur position dans la séquence :

$$P_{ns}(X_1^6 = x_1^6) = b_0(x_1) \times b_0(x_2) \times \ldots \times b_0(x_6) \ ,$$

où $b_0(x)$ correspond à la probabilité d'apparition du nucléotide x au sein des séquences qui ne sont pas des sites.

À travers le choix de ces deux modèles, on a défini deux mesures : l'une, $P_s(x_1^6)$ de la ressemblance entre la séquence observée x_1^6 et les sites connus ; l'autre, $P_{ns}(x_1^6)$ de la ressemblance entre x_1^6 et les séquences qui ne correspondent pas à des sites. Naturellement, plus $P_s(x_1^6)$ est grand, et $P_{ns}(x_1^6)$ petit, plus on est tenté de dire que x_1^6 correspond à la séquence d'un site. Il s'agit là d'un premier pas vers la prédiction du caractère « site » ou « non site » de la séquence. Les deux paragraphes suivants présentent deux approches de l'utilisation de ces modèles dans le but de prédire la nature de la séquence.

Point de vue 1. Le problème peut être vu comme le test de l'hypothèse H_0 = « cette séquence n'est pas la séquence d'un site » contre H_1 = « cette

séquence est la séquence d'un site ». Après le choix des modèles « site » et « non site », ces hypothèses sont reformulées en :

$$\begin{cases} H_0 = \text{« cette séquence a été générée par le modèle non site »} \\ H_1 = \text{« cette séquence a été générée par le modèle site »} \ . \end{cases}$$

Pour ce test, le lemme de Neyman-Pearson assure que la décision de rejeter H_0 doit être prise en fonction de la valeur du rapport de la probabilité d'apparition de la séquence sous le modèle « site » et sous le modèle « non site ». C'est à dire qu'il faut se fixer une règle de décision consistant à rejeter l'hypothèse H_0 dès que

$$\frac{P_s(X_1^6 = x_1^6)}{P_{ns}(X_1^6 = x_1^6)} > s(\alpha),$$

où s est un seuil choisi en fonction du niveau α souhaité pour le test (α correspond à la probabilité de rejeter H_0 lorsque H_0 est vraie). Cette décision est la meilleure dans le sens où aucun autre test de ce niveau (qui rejette aussi peu souvent H_0 à tort) ne sera plus puissant (ne pourra rejeter plus souvent l'hypothèse H_0 lorsque H_1 est vraie).

Point de vue 2. Le problème peut aussi être abordé sous un autre angle qui consiste à voir la variable aléatoire X_1^6 comme le résultat d'une procédure de tirage en deux étapes. La première étape est de choisir la nature « site » ou « non site » de la séquence qui sera générée selon des probabilités $P(\text{site}) = p$ et $P(\text{non site}) = 1 - p$. La seconde étape génère la séquence X_1^6 selon le modèle « site » ou « non site » conformément au résultat du premier tirage :

$$\begin{cases} P(x_1^6 \mid \text{site}) = P_s(x_1^6) \\ P(x_1^6 \mid \text{non site}) = P_{ns}(x_1^6) \ , \end{cases}$$

où l'on confond (à partir de maintenant) x et $X = x$ pour alléger les notations lorsqu'il n'y a pas d'ambiguïté. Dans ce cadre, la valeur de p est choisie de façon à rendre compte de la fréquence attendue des séquences qui correspondent à des sites parmi les séquences étudiées. Sous ce modèle, on peut calculer la probabilité des événements « site » et « non site » sachant la séquence observée x_1^6, à savoir $P(\text{site} \mid x_1^6)$ et $P(\text{non site} \mid x_1^6)$. Ces probabilités sont obtenues grâce à la formule de Bayes :

$$\begin{aligned} P(\text{site} \mid x_1^6) &= \frac{P(x_1^6 \mid \text{site})P(\text{site})}{P(x_1^6 \mid \text{site})P(\text{site}) + P(x_1^6 \mid \text{non site})P(\text{non site})} \\ P(\text{non site} \mid x_1^6) &= \frac{P(x_1^6 \mid \text{non site})P(\text{non site})}{P(x_1^6 \mid \text{site})P(\text{site}) + P(x_1^6 \mid \text{non site})P(\text{non site})} \ . \end{aligned}$$

L'expression du rapport de ces deux probabilités sépare d'une part la contribution de la séquence observée et d'autre part la contribution de la fréquence attendue des événements « site » et « non site » :

$$\frac{P(\text{site} \mid x_1^6)}{P(\text{non site} \mid x_1^6)} = \frac{P(x_1^6 \mid \text{site})}{P(x_1^6 \mid \text{non site})} \times \frac{P(\text{site})}{P(\text{non site})} \ .$$

On remarque que la même quantité est utilisée dans le test dans le point de vue 1 pour refléter les informations apportées par la séquence sur sa nature.

Ainsi, quel que soit le point de vue adopté, l'information apportée par la séquence s'exprime comme le rapport des probabilités d'apparition de la séquence sous le modèle « site » et « non site ». Étant donné que les modèles choisis considèrent les nucléotides comme apparaissant indépendamment les uns des autres, ce rapport se décompose en un produit de termes, dont chacun correspond à la participation d'une position de la séquence :

$$\frac{P(x_1^6 \mid \text{site})}{P(x_1^6 \mid \text{non site})} = \frac{b_1(x_1)}{b_0(x_1)} \times \frac{b_2(x_2)}{b_0(x_2)} \times \ldots \times \frac{b_6(x_6)}{b_0(x_6)} \ .$$

Le logarithme de ce rapport s'exprime comme la somme d'une contribution de chaque position, il est souvent appelé score. Dans l'approche test présentée dans le point de vue 1, la loi du rapport sous H_0, que l'on doit connaître pour calculer le niveau du test, est classiquement obtenue à travers la loi de son logarithme.

Recherche des régions qui codent pour des protéines

Comme le montre la table 1.4, les régions traduites en protéines ont des propriétés de composition différentes des régions intergéniques. La plupart des méthodes intrinsèques pour mettre en évidence les régions codantes utilisent largement ces propriétés.

Soit x_1^n une séquence dont on veut déterminer la nature codante ou intergénique. On propose pour cette séquence deux modélisations dont : l'une correspondant aux régions intergéniques, et l'autre aux régions codantes.

Modélisation des régions intergéniques. Sous le modèle qui tient compte de la composition en mono-nucléotides, la probabilité d'apparition de la séquence s'écrit :

$$P(x_1^n \mid \text{inter.}) = b_0(x_1) \times b_0(x_2) \times \ldots \times b_0(x_n) \ .$$

Mais la table 1.4 montre des dépendances entre les apparitions des nucléotides adjacents. Par exemple, la fréquence globale d'apparition d'un g est de $0,18$ mais varie de $0,15$ à $0,21$ selon le nucléotide qui le précède. Pour

intergénique				
	a	**g**	**c**	**t**
tot.	0,31	0,18	0,19	0,32
a	0,39	0,17	0,15	0,29
g	0,33	0,21	0,21	0,25
c	0,32	0,15	0,22	0,31
t	0,24	0,18	0,19	0,39

	codon pos. 1				codon pos. 2				codon pos. 3			
	a	**g**	**c**	**t**	**a**	**g**	**c**	**t**	**a**	**g**	**c**	**t**
tot.	0,30	0,34	0,19	0,17	0,33	0,15	0,21	0,31	0,27	0,23	0,21	0,29
a	0,36	0,33	0,13	0,18	0,37	0,12	0,18	0,33	0,36	0,19	0,17	0,28
g	0,30	0,29	0,27	0,14	0,37	0,20	0,23	0,20	0,25	0,21	0,34	0,20
c	0,33	0,34	0,17	0,16	0,33	0,14	0,20	0,33	0,32	0,27	0,17	0,24
t	0,22	0,38	0,19	0,21	0,20	0,11	0,24	0,45	0,15	0,27	0,22	0,36

TAB. 1.4 – Propriétés de composition des régions codantes et non codantes de *B. subtilis* calculées à partir des annotations du fichier GenBank. Au sein des régions codantes, la position dans le codon est prise en compte. La première ligne contient les fréquences des mono-nucléotides. Les quatre lignes suivantes correspondent aux fréquences des transitions entre nucléotides (par exemple, ligne a colonne g : la fréquence d'un g lorsqu'il est précédé d'un a).

tenir compte de ces dépendances, un modèle de chaîne de Markov peut être utilisé. Les paramètres de ce modèle sont de la forme :

$$\begin{cases} b(y;x) = P(X_t = y \mid X_{t-1} = x) & \text{la matrice de transition} \\ b(x) = P(X_1 = x) & \text{la loi initiale .} \end{cases}$$

La matrice de transition contient les probabilités d'apparition de chaque nucléotide sachant le nucléotide précédent. La probabilité d'apparition de la séquence s'écrit alors :

$$P(x_1^n \mid \text{inter.}) = b_i(x_1) \times b_i(x_2; x_1) \times \ldots b_i(x_n; x_{n-1}) \ .$$

Ce modèle ne prend en compte que la composition en di-nucléotides (deux séquences de même composition en di-nucléotides ont la même probabilité d'apparition). Pour prendre en compte la composition en mots de longueur $r+1$ on peut utiliser un modèle markovien d'ordre r dont la matrice de transition est de la forme $b(y;w) = P(X_t = y \mid X_{t-r}^{t-1} = w)$, où w est un mot de longueur r. Par extension, le modèle d'indépendance est parfois appelé modèle de chaîne de Markov d'ordre 0.

Modélisation des régions codantes. La table 1.4 montre que, dans les régions codantes, la fréquence d'apparition d'un nucléotide varie aussi selon

la position de ce nucléotide dans le codon. Il est bien sûr très intéressant, notamment en vue de retrouver la phase de lecture, de prendre en compte cette périodicité. La modélisation de la périodicité de composition des séquences codantes nécessite d'introduire une loi d'apparition des nucléotides différente pour chacune des trois positions des codons. Les paramètres de ce modèle se séparent en trois jeux b_{+1}, b_{+2}, et b_{+3}, où b_{+i} décrit la fréquence d'apparition des nucléotides en position $+i$ des codons. Sous ce modèle, dit périodique, la probabilité d'apparition de la séquence x_1^n s'écrit de trois façons différentes selon la phase $+1$, $+2$ ou $+3$ dans laquelle la séquence code :

$$\begin{aligned} P(x_1^n \mid +1) &= b_{+1}(x_1)b_{+2}(x_2)b_{+3}(x_3)b_{+1}(x_4)b_{+2}(x_5)b_{+3}(x_6)\dots \\ P(x_1^n \mid +2) &= b_{+2}(x_1)b_{+3}(x_2)b_{+1}(x_3)b_{+2}(x_4)b_{+3}(x_5)b_{+1}(x_6)\dots \\ P(x_1^n \mid +3) &= b_{+3}(x_1)b_{+1}(x_2)b_{+2}(x_3)b_{+3}(x_4)b_{+1}(x_5)b_{+2}(x_6)\dots \ . \end{aligned}$$

Pour les régions codantes, on utilise classiquement un modèle de Markov d'ordre au moins égal à 2 afin de prendre en compte l'absence de codon stop en phase : $b_{+3}(a;tg) = 0$, $b_{+3}(a;ta) = 0$, $b_{+3}(g;ta) = 0$. On modélise de la même façon les séquences codantes sur le brin complémentaire dont on notera les phases de lecture -1, -2 et -3.

Le logiciel GENMARK a été un des premiers à utiliser ces modèles [BM93]. Une séquence observée est supposée provenir soit d'une région intergénique (0), soit d'une région codante sur le brin étudié $(+1,+2,+3)$, soit d'une région codante sur le brin complémentaire $(-1,-2,-3)$. Après avoir introduit les fréquences attendues $P(i)$ de chacune de ces natures de séquences, la formule de Bayes permet d'évaluer localement la nature codante d'une séquence :

$$P(i \mid x_1^n) = \frac{P(x_1^n \mid i)P(i)}{\sum_{j=-3}^{3} P(x_1^n \mid j)P(j)} \ .$$

1.3.2 Les modèles probabilistes et leurs utilisations

Le sens de l'utilisation des modèles probabilistes pour les séquences biologiques

Les modèles. Puisque le « hasard » semble intervenir à toutes les échelles de la vie, que ce soit au niveau moléculaire, ou au cours de l'évolution, on pourrait considérer comme naturelle l'utilisation de modèles probabilistes des séquences biologiques. Cependant, une justification de ce type est un peu naïve. Elle sous-entend, en effet, que les modèles utilisés visent à reproduire un processus réel. Certains modèles, comme les modèles d'évolution « verticale » d'une position d'une séquence le long des branches d'un arbre phylogénétique, présentent peut-être assez bien cette caractéristique. Au contraire, les modèles qui nous intéressent ici visent plutôt à décrire l'enchaînement des nucléotides le long de la séquence, donc à réaliser une modélisation « horizontale » des séquences observées. Par ailleurs, les séquences sont généralement

considérées comme apparaissant de façon indépendante les unes des autres. Le plus souvent, ces modèles ne peuvent donc pas prétendre correspondre à un mécanisme réel d'apparition des séquences.

On préférera donc considérer le modèle probabiliste comme un moyen de résumer les propriétés des séquences étudiées, dont un intérêt est de permettre l'utilisation des approches statistiques. L'objectif de l'utilisation d'un modèle probabiliste est l'interprétation des séquences, c'est à dire essentiellement la détermination de la fonction d'une molécule ou d'une région de molécule à partir de sa séquence. Cette tâche est rendue possible par la modélisation de la séquence conditionnellement à la fonction.

D'un point de vue biologique, la relation entre la séquence d'une molécule et sa fonction (ou nature) paraît à la fois très complexe et très variable. Puisque les séquences contiennent l'information génétique, il est clair que la séquence est, plus ou moins directement et selon des mécanismes très variés, à l'origine de la fonction[3] d'une molécule. Cependant, si l'on prend en compte l'évolution, qui repose sur un double mécanisme de variation au niveau de la séquence et de sélection sur des critères de fonction, le rapport entre séquence et fonction est moins unidirectionnel qu'il n'y paraît. Cette sélection consiste, pour une large part, à éliminer les variations défavorables entraînant une perte de fonction. Or, la conservation d'une fonction ne nécessitant pas toujours le même degré de conservation de la séquence, une séquence évolue plus ou moins librement selon sa fonction. Dans le cas extrême de l'évolution neutraliste, la séquence évolue sans subir de pression de sélection. Enfin, la nature de la séquence elle-même (ex : nature *leading/lagging strand*, nature transcrit/non transcrit [Lob96] ou même organisme hôte de la séquence) peut avoir une influence directe sur les mutations qu'elle subit.

En conclusion, la relation séquence-fonction est très complexe. Dans ce contexte, il est assez remarquable que les méthodes statistiques puissent largement s'abstraire de ces considérations et proposer un cadre relativement homogène pour l'analyse des séquences. Ceci est rendu possible par le caractère très descriptif des modèles utilisés : quelle que soit la nature de la séquence modélisée, celle-ci est toujours vue comme une suite de lettres auxquelles il suffit d'attribuer des fréquences d'apparition.

L'interprétation de la séquence fondée sur les modèles. De même que les modèles probabilistes des séquences ne prétendent pas refléter la réalité biologique des processus d'apparition des séquences, les procédures statistiques de prédiction fondées sur ces modèles ne tentent pas de reproduire le mécanisme à l'origine de la relation séquence-fonction. Par exemple, une prédiction des régions codantes fondée sur les caractéristiques de composition de l'ADN ne prend pas en compte les mécanismes biologiques de l'expression

[3] On peut bien sûr faire valoir que la séquence ne suffit pas car une fonction biologique n'a de sens que dans un contexte.

de l'information génétique (choix des régions transcrites sur l'ADN, choix des régions traduites sur l'ARN). Elle prend par contre en considération les propriétés de composition des séquences protéiques et l'usage préférentiel de certains codons synonymes qui sont plutôt une conséquence de la traduction. De la même façon, l'utilisation de modèles probabilistes pour les sites de fixation d'une molécule ne tente pas de reproduire le mécanisme de fixation de la molécule à l'ADN. Toutefois, la modélisation des séquences apparaissant au sein des sites de fixation reflète en partie l'affinité de la molécule pour la séquence.

Le cas de la recherche de sites de fixation. *Pour se prononcer sur la nature « site » ou « non site » d'une séquence x_1^6, les approches statistiques invitent à examiner le rapport $P_s(x_1^6)/P_{ns}(x_1^6)$. Existe-t-il un lien entre ce rapport et le mécanisme physico-chimique de la fixation d'une molécule sur une séquence ?*

Cette question n'a probablement pas de légitimité d'un point de vue statistique, mais peut sûrement se poser lorsque l'interaction moléculaire est perçue comme un phénomène thermodynamique. De fait, il semble qu'il existe une certaine analogie entre les approches physico-chimiques et statistiques de détermination du caractère site ou non site d'une séquence (voir notamment [Sto90]). Considérons la fixation d'une molécule A à une molécule B :

$$A + B \rightleftharpoons AB \ .$$

S'il s'agit d'une réaction entre des molécules diluées en solution homogène, les concentrations $[A]$ et $[B]$ de molécules libres et $[AB]$ du complexe vérifient, à l'équilibre thermodynamique,

$$\Delta G^0 = -RT \ln \frac{[AB]}{[A][B]} \ ,$$

où ΔG^0 est la variation d'enthalpie libre standard associée à la fixation. Considérons maintenant la présence simultanée d'une multitude de types de molécules B caractérisées par des séquences x_1^6 différentes ($4^6 = 4\,096$). La relation d'équilibre précédente est vérifiée pour chacun des types $B_{x_1^6}$, donc

$$\forall x_1^6, \quad \Delta G^0_{x_1^6} = -RT \ln \frac{[AB_{x_1^6}]}{[A][B_{x_1^6}]} \ .$$

D'un point de vue physico-chimique, l'évaluation de $\Delta G^0_{x_1^6}$ est donc essentielle à la détermination des séquences x_1^6 qui correspondent à des sites de fixation de A. D'après les équations d'équilibre on a la relation de proportionnalité

$$\Delta G^0_{x_1^6} \propto \ln \frac{[AB_{x_1^6}]}{[B_{x_1^6}]} + cste \ .$$

Le lien entre les approches physico-chimiques et probabilistes peut alors être établi en introduisant les relations

$$\left\{ \begin{array}{lcl} [AB_{x_1^6}] & \propto & P_s(x_1^6) \\ [B_{x_1^6}] & \propto & P_{ns}(x_1^6) \ , \end{array} \right.$$

c'est à dire en supposant que $P_s(x_1^6)$ correspond à la proportion des complexes $AB_{x_1^6}$ parmi l'ensemble des complexes et que $P_{ns}(x_1^6)$ reflète la proportion des molécules $B_{x_1^6}$ parmi l'ensemble des molécules B libres. On obtient donc

$$\Delta G^0_{x_1^6} \ \propto \ \ln \frac{P_s(x_1^6)}{P_{ns}(x_1^6)} + cste' \ ,$$

exhibant l'analogie entre les approches physico-chimiques et statistiques. De plus, l'hypothèse d'indépendance des positions dans le modèle probabiliste permet d'écrire

$$\Delta G^0_{x_1^6} \ \propto \ \sum_{t=1}^{6} \ln \frac{b_t(x_t)}{b_0(x_t)} + cste' \ .$$

Elle correspond donc à l'hypothèse d'additivité des contributions énergétiques de chaque position dans l'approche physico-chimique.

Ce raisonnement est contestable à bien des points de vue. On soulignera surtout la naïveté de l'hypothèse principale : l'interprétation de la fréquence des séquences correspondant à des sites fonctionnels comme étant le reflet direct des proportions des différentes séquences engagées dans des complexes à l'équilibre thermodynamique. Ce raisonnement a cependant le mérite de montrer que les approches statistiques et mécanistes ne sont pas en complète contradiction sur le problème de la recherche des sites de fixation.

Différentes approches pour l'utilisation des modèles

Pour décrire brièvement les différents cadres d'utilisation des modèles probabilistes de séquence, il convient probablement de distinguer deux approches. Elles correspondent aux deux points de vue du premier exemple présenté : le test d'hypothèses et l'intégration des alternatives au sein d'un unique modèle.

Le test d'hypothèses. D'une manière générale un test est une procédure raisonnée de rejet d'une hypothèse H_0, dite hypothèse nulle. Celle-ci décrit, sous la forme d'un modèle probabiliste, ce que l'on ne cherche pas, autrement dit « ce qui est attendu si il n'y a rien d'exceptionnel ». La plupart du temps ce rejet n'a d'intérêt qu'au regard d'une caractéristique particulière des observations.

Dans ce cadre, un modèle correspondant à l'hypothèse nulle peut suffire. Citons l'exemple de l'identification des mots dont le nombre d'occurrences est exceptionnellement élevé (ou exceptionnellement faible) [RSW00] qui peut être l'indice d'une fonction biologique particulière. La caractéristique qui nous intéresse alors est le nombre d'occurrences du mot. Le modèle de séquence ne sert qu'à donner un sens à l'adjectif « exceptionnel » grâce à la description du nombre attendu d'occurrences. Quel que soit le modèle nul, il est bien évident qu'il ne correspond pas à la réalité des séquences observées : son rejet n'a donc un intérêt que si il pointe un mot particulier. Un second exemple est celui des procédures de comparaison de séquences comme les effectuent les logiciels BLAST [AMS+97] et FASTA [Pea96]. Elles ne reposent pas au premier abord sur un modèle probabiliste d'une séquence ressemblant à une autre. Le modèle nul considère toutes les séquences comme indépendantes les unes des autres et c'est une fonction de score, choisie pour croître avec le degré de ressemblance entre deux séquences, qui indique la direction dans laquelle il est intéressant de rejeter ce modèle nul.

La modélisation probabiliste de ce que l'on cherche n'est donc pas toujours nécessaire. Cependant, comme dans l'exemple de la recherche de sites par test d'hypothèses, elle constitue souvent un moyen puissant de construire une fonction de score permettant le rejet du modèle nul dans les cas intéressants. De fait, les fonctions de score définies par les matrices PAM [DSO78] et BLOSUM [HH92], couramment utilisées pour la comparaison des séquences protéiques, sont construites à partir d'un modèle probabiliste de la ressemblance de deux séquences.

Peut-on modéliser uniquement ce que l'on cherche ? La question naît de l'idée d'échanger le rôle des hypothèses, c'est à dire d'essayer de rejeter l'hypothèse « la séquence provient du modèle de ce que l'on cherche » sans définir ce que l'on ne cherche pas. Il est bien sûr préférable de modéliser l'hypothèse alternative. Cependant, cette question a l'intérêt de pointer l'asymétrie fondamentale de l'approche test. Un test tente de rejeter une hypothèse, éventuellement lorsqu'une autre hypothèse est vraisemblable, mais il s'agit toujours du rejet d'une hypothèse. Un problème peut donc se poser en pratique : le choix de l'hypothèse que l'on souhaite rejeter. Ce problème d'asymétrie est accru lorsque l'on considère plus de deux hypothèses. Un des intérêts de l'approche présentée ci-dessous est de rendre symétrique l'utilisation des différents modèles.

L'intégration des alternatives au sein d'un même modèle. Cette approche consiste à construire un « sur-modèle » englobant des modélisations qui correspondent à des natures de séquence différentes. Elle peut être vue comme une version bayésienne de l'approche test présentée juste avant. Pour cela, il est bien sûr nécessaire de modéliser l'apparition des séquences selon leur nature mais il faut encore réunir ces modèles en un unique modèle

à même de décrire toutes les natures de séquence. Ce modèle est classiquement construit en introduisant au moins une variable aléatoire non observée (cachée !) correspondant à la nature de la séquence. Dans les exemples introductifs cette variable correspond au caractère « site » - « non site » et à « codant en phase i » - « intergénique ». Ce cadre permet de faire reposer l'interprétation de la séquence sur le calcul de la probabilité des différentes natures conditionnellement à la séquence observée.

De façon intéressante, cette approche ouvre la voie à des analyses « non supervisées » des séquences. Le modèle global peut en effet être ajusté (estimation de ses paramètres, ou même sélection de modèle) directement sur un ensemble de données observées plutôt que sur des ensembles d'apprentissage (constitués de séquences de nature connue).

Une direction naturelle pour améliorer les performances des approches modélisant de ce que l'on cherche est de tenter d'améliorer cette modélisation. Dans cette optique, les modèles de chaînes de Markov cachées présentés dans la section suivante apparaissent particulièrement intéressants.

1.3.3 Vers les modèles de Markov cachés

Motivations

Les approches présentées dans les deux exemples de la section 1.3.1 permettent d'évaluer la nature sous-jacente d'un fragment de séquence donné. Pour utiliser ces approches lors de l'analyse de séquences entières, et donc beaucoup plus longue, une solution consiste à analyser séparément des fragments de séquence chevauchants, chaque fragment étant défini par la position d'une fenêtre glissant le long de la séquence.

Dans le premier exemple, il s'agit d'une fenêtre de six nucléotides puisque le modèle décrit la séquence de six nucléotides correspondant à la boîte -10 du site de fixation de l'ARN polymérase. Il est clair cependant que six nucléotides ne permettent pas d'identifier avec confiance le caractère « site » ou « non site » d'une séquence. Biologiquement, cette séquence correspondant à la boîte -10 ne suffit pas non plus à définir un site de fixation de l'ARN polymérase. Il existe en amont une deuxième séquence importante située correspondant à la boîte -35 qui interagit également avec l'ARN polymérase. Enfin, d'autres signaux peuvent intervenir ailleurs sur la séquence pour faciliter la fixation de l'ARN polymérase. Le site de fixation sur l'ADN de la protéine CAP intervenant dans la régulation de l'opéron lactose (page 16) en est un exemple. Comment prendre en compte la présence simultanée de ces différentes séquences pour identifier un site ? Dans le cadre de la modélisation probabiliste des séquences, la solution naturelle consiste à ne plus modéliser un seul site mais plusieurs sites, séparés par des séquences de longueurs plus ou moins variables.

Pour analyser une séquence grâce au modèle présenté dans le second exemple, le logiciel GENMARK utilise habituellement une fenêtre de 96 nucléotides. Pour expliquer ce choix, on peut déjà remarquer que le modèle a le défaut important de considérer la séquence comme étant soit entièrement codante, soit entièrement intergénique. L'utilisation d'une fenêtre courte rend ainsi le modèle plus réaliste, les fenêtres courtes enjambant moins les frontières entre régions codantes et intergéniques. De plus, il faut une fenêtre courte pour obtenir une information locale permettant d'identifier avec une relative précision les frontières codant - intergénique. Cependant la fenêtre doit aussi être assez longue pour contenir suffisamment d'information sur la nature de la séquence. Le choix d'une fenêtre de 96 nucléotides est donc le résultat d'un compromis délicat. Ce problème peut être résolu par une modélisation de l'alternance des régions codantes et non codantes le long de la séquence qui évite l'utilisation d'une fenêtre glissante.

Les limites des approches proposées dans ces deux exemples peuvent donc être repoussées en modélisant l'alternance le long de la séquence des régions de natures différentes : c'est précisément ce que permettent les modèles de chaînes de Markov cachées.

Introduction aux modèles de chaînes de Markov cachées

Les modèles de chaînes de Markov cachées (HMM pour *Hidden Markov Model*) sont une extension des modèles utilisés pour rendre compte de l'existence de séquences distinctes de natures différentes. Dans ces derniers on introduit une variable qui correspond à la nature de la séquence. Cette variable est cachée, au sens où elle n'est observée que par ses effets sur les propriétés de la séquence observée. Ce sont des modèles de mélanges : les natures des différentes séquences sont modélisées comme des variables aléatoires indépendantes les unes des autres. Or, comme on vient de l'évoquer, la nature d'un fragment de séquence dépend de la nature des fragments adjacents. C'est en prenant en compte ce contexte que l'information sur une position particulière peut être enrichie.

Les HMM modélisent ces dépendances sous la forme d'une chaîne de Markov, soit un modèle très simple qui décrit uniquement les probabilités de transition « d'une nature à une autre ». Les différentes natures de séquences correspondent aux états cachés du modèle. À chacun des états cachés est associée une loi d'apparition des lettres de la séquence, dite aussi loi d'émission. Celle-ci peut être choisie avec une grande liberté. Dans ce cadre, les modèles markoviens sont souvent d'un grand intérêt car ils permettent de faire dépendre des quelques nucléotides précédents la probabilité d'apparition d'un nucléotide (c'est à dire de prendre en compte la composition en mots plutôt qu'en lettres).

Généralement, seules certaines transitions entre états cachés sont autorisées. On peut les représenter sous la forme d'un graphe. La figure 1.9 montre

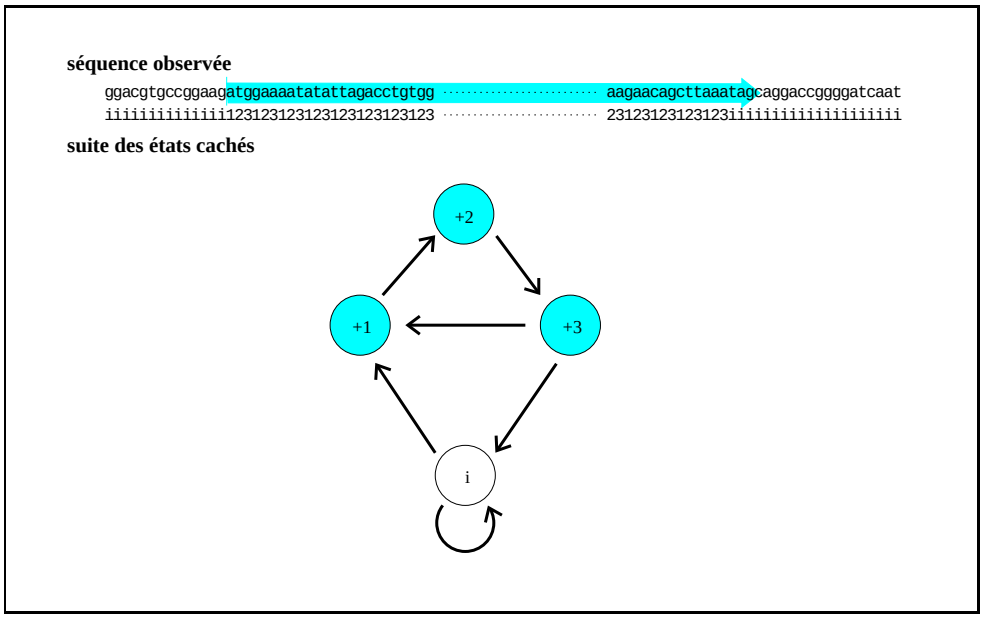

FIG. 1.9 – Graphe des états cachés d'un HMM simple modélisant l'alternance des régions codantes et intergéniques le long des séquences. Les cercles représentent les états cachés (i intergénique ; +1,+2 et +3 les différentes positions dans les codons), les flèches représentent les seules transitions autorisées.

un graphe correspondant à un modèle simple qui décrit l'alternance des régions codantes et intergéniques le long d'une séquence d'ADN. Le modèle peut évidemment être étendu pour tenir compte de la présence d'un codon start et d'un codon stop aux limites de la région codante, de la présence d'introns, et de l'existence de signaux aux bords des introns. Les HMM permettent de modéliser très librement l'alternance de textures et de signaux le long des séquences. Les modèles hétérogènes et périodiques présentés dans les exemples introductifs peuvent notamment être vus comme des enchaînements d'états cachés. Les HMM ne se contentent donc pas de modéliser l'apparition des séquences conditionnellement à leur nature mais modélisent aussi la nature elle-même de la séquence. Il peut paraître difficile de voir comme un événement aléatoire la position des régions codantes sur une séquence. Il faut néanmoins bien comprendre qu'il s'agit essentiellement de résumer des caractéristiques des séquences, comme la distribution des longueurs des régions codantes et intergéniques.

La figure 1.10 montre la structure d'un modèle qui prend en compte la présence simultanée de la boîte -10 et de la boîte -35 afin de prédire un site de fixation de l'ARN polymérase [JLK+01]. Chaque état, symbolisé par un cercle coloré, a sa propre loi d'émission des nucléotides. Ceci permet de rendre compte de la fréquence d'apparition des nucléotides associée à

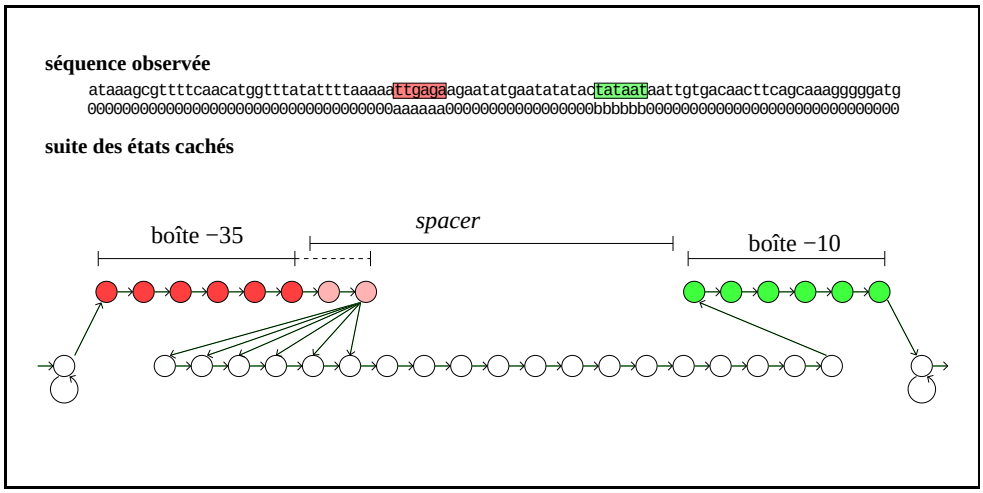

FIG. 1.10 – Structure du HMM utilisé par Jarmer *et al.* [JLK+01] pour modéliser le site de fixation de l'ARN polymérase chez *B. subtilis*.

chacune des positions au sein des boîtes. Au contraire, un même modèle de Markov d'ordre 1 d'émission des nucléotides (le « *background* ») est associé à tous les états représentés par des cercles vides. L'enchaînement de nombreux états cachés entre les deux boîtes permet de modéliser la séquence qui les sépare, appelée « *spacer* », dont la longueur est supposée comprise entre 16 et 21 bp. Deux états ont été ajoutés à droite des six positions classiquement considérées comme appartenant à la boîte -35, illustrant les difficultés que peut présenter le choix d'un modèle.

Les problèmes qui se posent, une fois la structure du modèle choisie, sont le choix des paramètres du modèle et la reconstruction de la suite des états cachés associée à une séquence observée. Quelques méthodes statistiques ont été développées pour faciliter le choix de la structure du modèle. Ce problème demeure néanmoins beaucoup plus difficile que celui de l'estimation des paramètres. Heureusement, la simplicité d'interprétation de la structure d'un HMM rend souvent assez facile la traduction des connaissances biologiques en une architecture de modèle (au moins approximative) appropriée à l'usage envisagé. On appellera ici architecture du modèle ce qui relève du choix de la forme du modèle : choix du nombre d'états cachés, des transitions autorisées et du modèle d'émission des observations.

La chapitre 2 débute par une description plus formelle des HMM et présente ensuite les principaux aspects mathématiques de leurs utilisations.

Chapitre 2

Aspects mathématiques de la mise en œuvre des chaînes de Markov cachées

L'objectif de ce chapitre est de décrire les principaux aspects mathématiques et algorithmiques de la mise en oeuvre des modèles de chaînes de Markov cachées. Il commence par une présentation du modèle HMM et de quelques-unes de ses extensions. La section 2.2 décrit ensuite les principales approches pour « reconstruire » le chemin caché à partir d'une séquence observée lorsque le modèle et ses paramètres sont connus. Une procédure de calcul de la probabilité d'apparition d'une séquence observée, c'est à dire la vraisemblance des paramètres, est présentée dans la section 2.3. L'efficacité de ces algorithmes dont le coût de calcul ne croît que linéairement avec la longueur de la séquence est une des raisons essentielles du succès des HMM. La section 2.4 présente les principales méthodes d'estimation des paramètres d'un HMM. Ces méthodes utilisent largement les algorithmes de reconstruction du chemin caché. Les travaux présentés dans les chapitres 3, 4 et 5 se fondent en grande partie sur ces méthodes d'estimation. Enfin, la section 2.5 présente l'approche statistique utilisée dans le chapitre 5 pour la sélection de modèle (le choix de l'architecture du modèle).

2.1 Le modèle de chaînes de Markov cachées

2.1.1 Définitions et notations

Le modèle de chaînes de Markov cachées modélise deux processus emboîtés [Rab89, MZ97]. Le processus observé correspond à la séquence (ou l'ensemble de séquences) observée. Le processus caché représente la structure sous-jacente de la séquence, que l'on cherchera à reconstruire.

Le processus caché

Le processus caché, noté $(S_t)_{t\in\{1,2,\ldots\}}$, correspond à la suite des états cachés. Les états cachés S_t prennent leurs valeurs dans un espace discret $\mathcal{S}$ de dimension finie $|\mathcal{S}|$. Ce processus est une chaîne de Markov homogène d'ordre 1 dont on notera a les paramètres

$$\begin{cases} \text{la matrice de transitions d'éléments définis par} \\ \qquad a(u,v) = P(S_{t+1} = v \mid S_t = u), \quad u,v \in \mathcal{S}^2, \quad \sum_{v\in\mathcal{S}} a(u,v) = 1 \\ \text{et la distribution initiale} \\ \qquad a(u) = P(S_1 = u), \quad u \in \mathcal{S}, \quad \sum_{u\in\mathcal{S}} a(u) = 1 \ . \end{cases}$$

Sous ce modèle, la probabilité d'apparition d'une suite d'états cachés de longueur n particulière $s_1^n = (s_1, \ldots, s_n)$ s'écrit

$$\begin{aligned} P(S_1^n = s_1^n) &= P(S_1 = s_1)P(S_2 = s_2 \mid S_1 = s_1)P(S_3 = s_3 \mid S_2 = s_2) \\ &\qquad \times \ldots \times P(S_n = s_n \mid S_{n-1} = s_{n-1}) \\ &= a(s_1) \prod_{t=1}^{n-1} a(s_t, s_{t+1}) \ . \end{aligned}$$

Le modèle de chaîne de Markov est l'un des plus simples que l'on puisse envisager, le processus ainsi modélisé n'a qu'une mémoire très limitée puisque

$$P(S_t^n = s_t^n \mid S_1^{t-1} = s_1^{t-1}) = P(S_t^n = s_t^n \mid S_{t-1} = s_{t-1}) \ .$$

L'information qu'apporte la connaissance du passé $(S_1^{t-1} = s_1^{t-1})$ sur le futur $(S_t^n = s_t^n)$ se résume donc uniquement à la connaissance du dernier état du passé $(S_{t-1} = s_{t-1})$. Symétriquement, cette propriété implique que toute l'information apportée par la connaissance du futur $(S_{t+1}^n = s_{t+1}^n)$ se résume à celle du premier état du futur $(S_{t+1} = s_{t+1})$:

$$\begin{aligned} &P(S_1^t = s_1^t \mid S_{t+1}^n = s_{t+1}^n) \\ &= \frac{P(S_1^t = s_1^t, S_{t+1} = s_{t+1}, S_{t+2}^n = s_{t+2}^n)}{P(S_{t+1} = s_{t+1}, S_{t+2}^n = s_{t+2}^n)} \\ &= \frac{P(S_1^t = s_1^t, S_{t+1} = s_{t+1})}{P(S_{t+1} = s_{t+1})} \times \frac{P(S_{t+2}^n = s_{t+2}^n \mid S_{t+1} = s_{t+1})}{P(S_{t+2}^n = s_{t+2}^n \mid S_{t+1} = s_{t+1})} \\ &= P(S_1^t = s_1^t \mid S_{t+1} = s_{t+1}) \ . \end{aligned}$$

L'efficacité de la mise en oeuvre des modèles de chaînes de Markov cachées (sections 2.2 et 2.3) est liée à ces propriétés d'indépendance conditionnelle.

Pour finir cette brève présentation du processus caché, notons que bien souvent on souhaite interdire certaines transitions entre états cachés. Il suffit alors de fixer à zéro les probabilités de ces transitions ($a(u,v)=0$). Lorsque la distribution initiale est différente de la loi stationnaire du processus caché, il est courant de se représenter l'existence d'un état « *begin* » correspondant à la position -1 de la séquence [DEKM98]. Enfin, si on désire modéliser la fin de la séquence, il est généralement naturel de la considérer comme l'atteinte d'un état caché particulier « *end* ».

Le processus observé

Le processus observé, noté $(X_t)_{t\in\{1,2,\ldots\}}$, correspond à la séquence observée. Dans le cadre de la modélisation de séquences de macromolécules biologiques, il est à valeurs discrètes dans un alphabet $\mathcal{X}$ de taille 4 pour les acides nucléiques et 20 pour les protéines. L'apparition, dite aussi émission, des observations peut être modélisée comme uniquement dépendante de l'état caché sous-jacent. La séquence observée X_1^n est alors modélisée conditionnellement à la suite des états cachés selon un modèle défini par des paramètres b de la forme

$$b_u(x) = P(X_t = x \mid S_t = u), \quad u \in \mathcal{S},\ x \in \mathcal{X}, \quad \sum_{x\in\mathcal{X}} b_u(x) = 1\ .$$

Cependant il peut être intéressant de prendre en compte aussi les quelques observations précédentes grâce à un modèle markovien d'émission des observations dont on notera encore b les paramètres :

$$\begin{cases} \text{pour } t > r_u, & b_u(x;w) = P(X_t = x \mid S_t = u, X_{t-r_u}^{t-1} = w) \\ & \qquad u \in \mathcal{S},\ x \in \mathcal{X},\ w \in \mathcal{X}^{r_u} \text{ avec } \sum_{x\in\mathcal{X}} b_u(x;w) = 1 \\ \text{pour } t \le r_u, & b_u(x;w) = P(X_t = x \mid S_t = u, X_1^{t-1} = w), \\ & \qquad u \in \mathcal{S},\ x \in \mathcal{X},\ w \in \mathcal{X}^{t-1} \text{ avec } \sum_{x\in\mathcal{X}} b_u(x;w) = 1\ , \end{cases}$$

où r_u correspond à l'ordre du modèle markovien d'apparition des observations conditionnellement à l'état caché u. Ce modèle d'émission des observations peut être étendu aux modèles markoviens à mémoire variable (où r_u est une fonction de w, voir par exemple [SDKW98, DHK+99]) ou à n'importe quelle autre loi d'émission qui dépend du contexte ayant précédé dans la séquence observée.

Plus généralement, le modèle d'émission des observations peut être choisi très librement en fonction des données traitées et de l'objectif de la modélisation. Il peut s'agir en particulier de données continues (comme dans [RCD93]) ou de données multivariées. La reconstruction du chemin caché nécessite uniquement de disposer d'une fonction de densité qui décrit la loi d'émission des

observations conditionnellement à l'état caché. Quant aux problèmes d'estimation que peut entraîner le choix d'un modèle complexe d'émission des observations, ils ne sont pas réellement spécifiques au cadre HMM.

Dans ce chapitre on supposera que le modèle d'émission des observations est un modèle markovien dont, pour simplifier les notations, l'ordre r_u sera considéré égal à r pour tous les états cachés. On parlera alors d'un modèle dit M1-Mr [Chu89, Mur98, BHW00].

2.1.2 Vraisemblance des données complètes et propriétés d'indépendance conditionnelle

L'ensemble des paramètres d'un HMM sera noté $\theta = (a, b)$. Le nombre de paramètres d'un HMM est rapidement très élevé. Pour un modèle M1-Mr, il est ainsi de $|\mathcal{S}|(|\mathcal{S}|-1) + |\mathcal{S}||\mathcal{X}|^{r-1}(|\mathcal{X}|-1)$, où $|\mathcal{S}|$ désigne la dimension de $\mathcal{S}$, sans compter les paramètres des distributions initiales des différents modèles markoviens.

Sous le modèle de paramètres $\theta = (a, b)$, la probabilité de l'événement $(S_1^n = s_1^n, X_1^n = x_1^n)$ s'écrit

$$\begin{aligned}
&P_\theta(X_1^n = x_1^n, S_1^n = s_1^n) \\
&\quad = \; P_\theta(S_1^n = s_1^n) P_\theta(X_1^n = x_1^n \mid S_1^n = s_1^n) \\
&\quad = \; P_\theta(S_1 = s_1) \prod_{t=2}^{n} P_\theta(S_t = s_t \mid S_{t-1} = s_{t-1}) \\
&\qquad\qquad \times \prod_{t=1}^{n} P_\theta(X_t = x_t \mid S_t = s_t, X_{t-r}^{t-1} = x_{t-r}^{t-1}) \\
&\quad = \; a(s_1) \prod_{t=1}^{n-1} a(s_t, s_{t+1}) \prod_{t=1}^{n} b_{s_t}(x_t; x_{t-r}^{t-1}) \; . \qquad (2.1)
\end{aligned}$$

Il s'agit de la vraisemblance de θ sur les données complètes (s_1^n, x_1^n). Il est possible de représenter la façon dont se factorise la vraisemblance des données complètes sous la forme d'un graphe orienté appelé DAG (*Directed Acyclic Graph*) [Lau96, Spi98]. Les nœuds du DAG correspondent aux variables (ici $S_1, \ldots, S_t, \ldots$ et $X_1, \ldots, X_t, \ldots$) et ses arcs symbolisent les factorisations : le terme $P(S_t = s_t \mid S_{t-1} = s_{t-1})$ correspond ainsi à un arc allant de S_{t-1} à S_t, le terme $P(X_t = x_t \mid S_t = s_t, X_{t-1} = x_{t-1})$ du modèle M1-M1 ($r = 1$) correspond à deux arcs allant respectivement de S_t et de X_{t-1} à X_t. La figure 2.1 montre les DAG correspondant aux modèles M1-M0 et M1-M1.

Certaines propriétés d'indépendance conditionnelle entre les différentes variables du modèle sont visibles dans le DAG. Il s'agit de propriétés orientées. Un nœud est notamment indépendant ($\perp$) de ses non descendants condi-

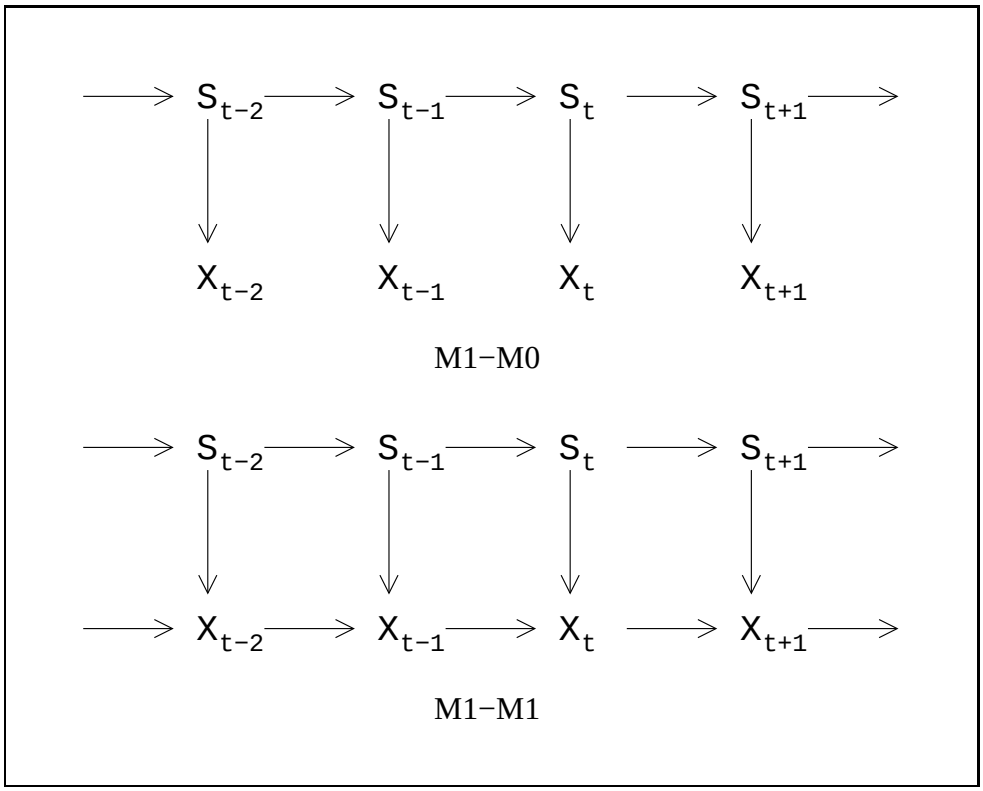

FIG. 2.1 – Les DAG des modèles M1-M0 et M1-M1.

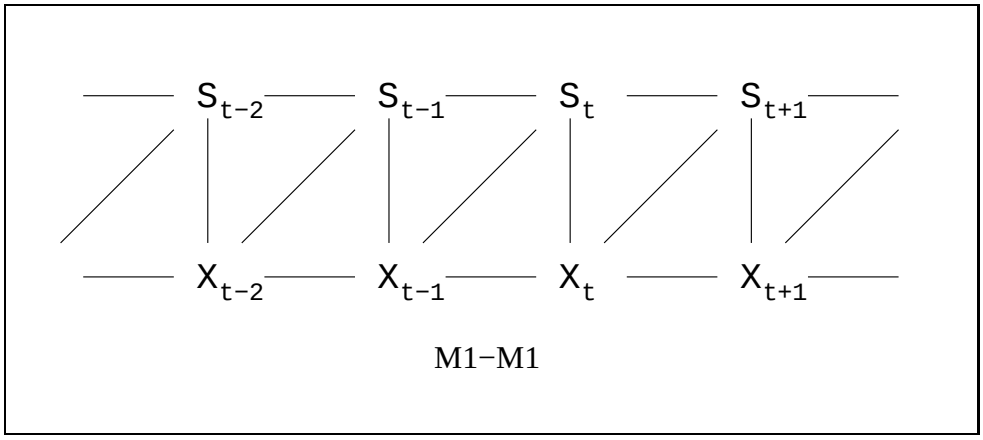

FIG. 2.2 – Le graphe moral du modèle M1-M1.

tionnellement (|) à ses voisins, d'où par exemple dans le modèle M1-M1

$$P(X_t = x_t \mid S_1^t = s_1^t, X_1^{t-1} = x_1^{t-1}) \quad = \quad P(X_t = x_t \mid S_t = s_t, X_{t-1} = x_{t-1}) \tag{2.2}$$

car

$$X_t \perp S_1^{t-1}, X_1^{t-2} \mid S_t, X_{t-1} \ .$$

De la même façon, on a

$$P(S_t = s_t \mid S_1^{t-1} = s_1^{t-1}, X_1^{t-1} = x_1^{t-1}) \quad = \quad P(S_t = s_t \mid S_{t-1} = s_{t-1}) \ , \tag{2.3}$$

car

$$S_t \perp S_1^{t-2}, X_1^{t-1} \mid S_{t-1} \ .$$

À partir du DAG, il est possible de construire le « graphe moral » (*moral graph*) [Lau96] qui permet de visualiser d'autres propriétés d'indépendance conditionnelle. Ce graphe est non orienté. Il s'obtient en ajoutant au DAG des arcs entre les couples de parents de chacun des nœuds du DAG (par « mariage ») et en supprimant l'orientation des arcs. Le graphe moral du modèle M1-M1 est représenté dans la figure 2.2.

Les propriétés d'indépendance visibles dans le graphe moral sont non orientées. Une de ces propriétés est que tout nœud est indépendant des autres conditionnellement à ses voisins. Plus généralement, deux ensembles de nœuds séparés par un troisième ensemble sont indépendants l'un de l'autre conditionnellement à ce troisième, d'où par exemple :

$$\begin{aligned} &P(S_{t-1} = s_{t-1} \mid S_t = s_t, X_1^n = x_1^n) = \\ &\qquad P(S_{t-1} = s_{t-1} \mid S_t = s_t, X_1^{t-1} = x_1^{t-1}) \end{aligned} \tag{2.4}$$

puisque

$$S_{t-1} \perp X_t^n \mid S_t, X_1^{t-1} \ .$$

2.1.3 Modèle semi-markovien caché et autres généralisations

Le cadre HMM restreint la modélisation du processus caché au modèle de chaîne de Markov. Cette restriction est à l'origine des propriétés d'indépendance conditionnelle du modèle de chaînes de Markov cachées présentées dans la section précédente. Comme on le verra dans les sections 2.2 et 2.3, elles permettent la reconstruction du chemin caché et le calcul de la vraisemblance pour un coût qui croît linéairement avec la longueur de la séquence. Au prix d'un accroissement (souvent) considérable du coût des calculs, il est possible d'utiliser d'autres modèles pour le processus caché. Parmi ces modèles alternatifs, le modèle semi-markovien est sans doute le plus utilisé en biologie, notamment pour la prédiction de gènes [KHRE96, BK97, LB98].

Dans un modèle markovien « classique » la durée du temps de séjour dans un état u est modélisée implicitement par une loi géométrique de paramètre $1 - a(u,u)$ (d'espérance $1/(1-a(u,u))$). En effet, la probabilité de rester dans l'état u un temps exactement égal à k est donnée par

$$\begin{aligned} & P(S_{t_0+1}^{t_0+k-1} = (u,\ldots,u), S_{t_0+k} \neq u \mid S_{t_0} = u, S_{t_0-1} \neq u) \\ &= P(S_{t_0+1}^{t_0+k-1} = u^{k-1}, S_{t_0+k} \neq u \mid S_{t_0} = u) \\ &= \Big(\prod_{t=t_0+1}^{t_0+k-1} P(S_t = u \mid S_{t-1} = u) \Big) P(S_{t_0+k} \neq u \mid S_{t_0+k-1} = u) \\ &= a(u,u)^k (1 - a(u,u)) \ . \end{aligned}$$

Le modèle implique notamment une distribution des temps de séjours décroissante. Il s'agit d'une limitation qui peut être en partie contournée en restant dans le cadre markovien « classique ». L'idée consiste à représenter les objets pour lesquels on souhaite modéliser une longueur non géométrique non pas par un unique état mais par plusieurs. Ainsi, le modèle représenté figure 1.9 impose une longueur divisible par trois aux régions codantes, mais qui reste quand même fondamentalement géométrique. Des distributions non géométriques peuvent aussi être obtenues (voir par exemple [DEKM98]) notamment parce que la somme de deux lois géométriques n'est pas une loi géométrique et qu'au prix de l'introduction d'un nombre d'états égal à la taille du support toutes les distributions discrètes à support fini peuvent être obtenues (avec un paramètre par longueur autorisée, donc sous forme non paramétrique !).

Le modèle semi-markovien [Cin69, Kul97], généralise le modèle de chaînes de Markov en permettant la modélisation explicite des durées des séjours dans les états, les transitions entre états aux instants de sauts restant markoviennes. Le processus semi-markovien proprement dit est

$$\underbrace{S_{T_1}, \ldots, S_{T_2-1}}, \underbrace{S_{T_2}, \ldots, S_{T_3-1}}, \underbrace{S_{T_3}, \ldots}\ ,$$

où les T_i sont les instants de sauts et

$$S_t = S_{T_i}, \quad \text{pour tout } t \text{ t.q. } T_i \leq t < T_{i+1} \ .$$

Il peut être défini comme l'emboîtement de deux processus : le processus des instants de sauts $\{T_i\}_{i=\{1,2,\ldots\}}$ à valeurs strictement croissantes dans $\mathbb{N}_+^\star$; et le processus des transitions entre états $\{S_{T_i}\}_{i=\{1,2,\ldots\}}$ défini aux instants de sauts et à valeurs dans $\mathcal{S}$ qui est une chaîne de Markov. Les paramètres (a,d)

d'un modèle de chaînes semi-markoviennes sont de la forme[1] :

$$\text{distribution initiale} \quad \begin{cases} T_1 = 1 \\ a(u) = P(S_{T_1} = u) \end{cases}$$

$$\text{puis} \quad \begin{cases} d_u(k) = P(T_{i+1} = t_i + k \mid T_i = t_i, S_{T_i} = u) \\ a(u,v) = P(S_{T_{i+1}} = v \mid S_{T_i} = u) \ , \end{cases}$$

où a correspond aux paramètres de la chaîne de Markov des transitions entre états définie aux instants de saut et d correspond aux lois de temps de séjour ($d_u(k)$ est la probabilité de rester un temps exactement égal à k dans l'état u). Cette paramétrisation est même un peu restrictive car il n'est guère plus compliqué de faire dépendre la loi du temps de séjour non seulement de l'état u sous-jacent mais aussi du prochain état v (on a alors d de la forme $d_{u,v}(k)$).

Dans un modèle semi-markovien caché (HSMM pour *Hidden Semi-Markov Model*[2]), le processus caché est une chaîne semi-markovienne [Rab89, Gué03]. Les états cachés sont donc modélisés comme apparaissant par plages dont la longueur est le temps de séjour. Dans ce cadre, l'émission des observations conditionnellement aux états cachés peut bien sûr être modélisée comme dans un HMM mais il est aussi assez naturel d'introduire des lois qui génèrent les observations par plages correspondant aux plages d'états cachés. Il suffit, en effet, que θ permette d'attribuer une valeur à la probabilité

$$P_\theta(X_t^{t+k-1} = x_t^{t+k-1} \mid X_1^{t-1}, T_i = t, S_{T_i} = u, T_{i+1} = t+k) \ .$$

Il est ainsi par exemple possible d'introduire des modèles d'émission périodiques pour la modélisation des séquences des régions traduites et des modèles hétérogènes pour celle des signaux (voir par exemple [BK97, LB98]). Les HSMM sont utilisés dans le chapitre 5 et leur description ne sera pas poursuivie dans ce chapitre. Le coût de la reconstruction du chemin caché d'un HSMM associé à une séquence observée croît théoriquement comme le carré de la longueur de la séquence (éventuellement plus rapidement lorsque certaines lois d'émission des observations sont utilisées). En pratique, ces modèles ne sont utilisés pour de longues séquences que lorsque des simplifications, ou même des approximations, sont possibles. On peut notamment citer la majoration des distributions de temps de séjour dont la loi n'est pas géométrique et, surtout, l'utilisation de pré-traitements pour trouver les positions possibles des changements d'états [Bur97].

Une autre généralisation des HMM est la modélisation de la séquence observée par un modèle de grammaire stochastique hors contexte. La structure cachée prend alors une forme d'arbre dont chaque noeud peut générer

[1] On suppose ici que la première position de la séquence correspond à l'entrée dans un état.

[2] Le modèle HSMM est parfois appelé aussi *Generalized HMM* ou *HMM with explicit state duration*.

deux lettres distantes sur la séquence. Ces modèles sont notamment utilisés pour tenir compte des dépendances à longue distance dues à l'appariement de nucléotides distants au sein des ARN de structure (voir par exemple [ED94, SBH+94, DEKM98]). Ils ne seront pas décrits plus en détail dans cette thèse. Signalons tout de même que, pour ces modèles, le coût de la reconstruction de la structure cachée croît comme le cube de la longueur de la séquence !

Pour finir de façon moins ambitieuse la description des généralisations concernant la modélisation du processus caché, notons qu'il est possible d'introduire des états qui n'émettent aucun symbole dans la séquence observée. L'intérêt de ces états « silencieux » réside principalement dans la diminution du nombre de transitions. La prise en compte de ces états dans les algorithmes de mise en oeuvre des HMM ne pose pas réellement de difficultés, elle est décrite notamment dans Durbin *et al.*, 1998 [DEKM98].

2.2 Reconstruction du chemin caché

Lorsque l'on connaît les paramètres $\theta = (a, b)$ d'un HMM, l'un des problèmes d'intérêt majeur est la reconstruction du chemin caché associé à une séquence observée. La difficulté de ce problème réside dans le nombre immensément grand de chemins s_1^n possibles. Ce nombre atteint $|\mathcal{S}|^n$ si toutes les transitions sont autorisées et rend donc définitivement impossible d'évaluer les chemins un à un. L'efficacité des algorithmes qui permettent d'aborder la reconstruction du chemin caché se fonde sur les propriétés d'indépendance conditionnelle (section 2.1.2).

L'algorithme de Viterbi et l'algorithme *forward-backward* sont les plus souvent utilisés (voir par exemple [Rab89, DEKM98]). Ils envisagent la reconstruction du chemin caché sous des angles différents. Après la présentation de ces deux algorithmes, cette section décrit aussi une procédure de tirage aléatoire du chemin caché conditionnellement à la séquence observée.

Pour simplifier les formules, on s'autorisera à omettre le nom des variables aléatoires dans les probabilités lorsque leur valeur a un nom explicite (par exemple on pourra écrire $P_\theta(S_1 = v, x_1)$ à la place de $P_\theta(S_1 = v, X_1 = x_1)$). De plus, on écrira x_{t-r}^t à la place de x_1^t lorsque $t \leq r$. Enfin, par souci de lisibilité, les justifications des formules sur lesquelles reposent les algorithmes de Viterbi et *forward-backward* sont présentées après les différentes étapes de chaque algorithme.

2.2.1 L'algorithme de Viterbi

L'algorithme de Viterbi permet de trouver le chemin caché $s_1^{n\star}$ le plus probable étant donnée la séquence observée et les paramètres $\theta = (a, b)$:

$$\begin{aligned} s_1^{n\star} &= \arg\max_{s_1^n} P_\theta(S_1^n = s_1^n \mid X_1^n = x_1^n) \\ &= \arg\max_{s_1^n} P_\theta(S_1^n = s_1^n, X_1^n = x_1^n) \ . \end{aligned}$$

Ces deux formulations sont équivalentes car $P_\theta(S_1^n = s_1^n \mid X_1^n = x_1^n) = P_\theta(S_1^n = s_1^n, X_1^n = x_1^n)/P_\theta(X_1^n = x_1^n)$. L'algorithme de Viterbi réalise cette tâche efficacement grâce à une récurrence « avant-arrière ».

La récurrence avant calcule, pour t croissant de 1 à n et pour tout $v \in \mathcal{S}$, la probabilité « du couple constitué de la séquence observée x_1^t et du chemin caché s_1^t le plus probable dont l'état en position t est v », soit $\max_{s_1^{t-1}} P_\theta(S_1^{t-1} = s_1^{t-1}, S_t = v, x_1^t)$.

La récurrence avant est initialisée pour $t = 1$ avec

$$P_\theta(S_1 = v, x_1) = b_v(x_1)a(v) \ .$$

Puis, pour t croissant de 2 à n, la formule de récurrence suivante est utilisée :

$$\begin{aligned} \max_{s_1^t} P_\theta(S_1^{t-1} = s_1^{t-1}, S_t = v, x_1^t) = \max_{s_{t-1}} \Big(& b_v(x_t; x_{t-r}^{t-1}) a(s_{t-1}, v) \\ & \times \max_{s_1^{t-2}} P_\theta(S_1^{t-2} = s_1^{t-2}, S_{t-1} = s_{t-1}, x_1^{t-1}) \Big) \ . \end{aligned} \tag{2.5}$$

En vue de la réalisation de la récurrence arrière, l'état s_{t-1} qui a permis d'obtenir $\max_{s_1^t} P_\theta(S_1^{t-1} = s_1^{t-1}, S_t = v, x_1^t)$ est mémorisé dans une variable que l'on notera $\mathrm{ptr}_t(v)$:

$$\begin{aligned} \mathrm{ptr}_t(v) = \arg\max_{s_{t-1}} \Big(& b_v(x_t; x_{t-r}^{t-1}) a(s_{t-1}, v) \\ & \times \max_{s_1^{t-2}} P_\theta(S_1^{t-2} = s_1^{t-2}, S_{t-1} = s_{t-1}, x_1^{t-1}) \Big) \end{aligned}$$

La récurrence arrière retrace $s_1^{n\star} = (s_1^\star, s_2^\star, \ldots, s_n^\star)$ en parcourant en arrière le meilleur chemin.

L'état $s_n^\star$ est

$$s_n^\star = \arg\max_v \Big(\max_{s_1^{n-1}} P_\theta(S_1^{n-1} = s_1^{n-1}, S_n = v, x_1^n) \Big) \ .$$

Puis, pour t décroissant de $n-1$ à 1, le chemin qui a abouti à $s_n^\star$ est retracé :

$$s_t^\star = \mathrm{ptr}_t(s_{t+1}^\star) \ .$$

La complexité de l'algorithme de Viterbi est proportionnelle à $m \times n$ où n est la longueur de la séquence et m le nombre de transitions autorisées entre les états cachés (t.q. $a(u,v) > 0$). D'un point de vue pratique, les probabilités $\max_{s_1^{t-1}} P_\theta(S_1^{t-1} = s_1^{t-1}, S_t = v, x_1^t)$ sont très petites. On peut par exemple travailler avec leur logarithme pour éviter les problèmes numériques.

Avec des modifications mineures, l'algorithme de Viterbi permet la recherche simultanée des k chemins les plus probables (voir par exemple [Zuk91]). Il faut alors calculer pendant la récurrence avant les k probabilités les plus élevées (la complexité est alors proportionnelle à $k \times n \times m$). Cependant l'intérêt de trouver les k meilleurs chemins est généralement très limité car ces chemins sont le plus souvent presque identiques. L'algorithme de Viterbi peut aussi être utilisé pour rechercher le chemin le plus probable sous certaines contraintes : si on cherche un chemin tel que $(s_t^\star, s_{t+1}^\star) \neq (u,v)$ il suffit de remplacer $a(u,v)$ par 0 lors de l'itération t de la récurrence avant. De cette façon il est notamment possible d'obtenir le chemin le plus probable qui diffère du meilleur à certaines positions choisies.

Quelle que soit la stratégie employée l'algorithme de Viterbi ne permet l'étude que d'un nombre très limité de chemins. De plus, chaque chemin reconstruit est obtenu sans information utilisable sur l'incertitude de la reconstruction, l'algorithme *forward-backward* aborde à sa manière cette question.

—*Justification de la relation de récurrence* 2.5 —

On a

$$\begin{aligned}
&P_\theta(x_1^t, S_t = v, S_1^{t-1} = s_1^{t-1}) \\
&\quad = P_\theta(x_t \mid x_1^{t-1}, S_t = v, S_1^{t-1} = s_1^{t-1}) P_\theta(S_t = v \mid x_1^{t-1}, S_1^{t-1} = s_1^{t-1}) \\
&\qquad\quad \times P_\theta(x_1^{t-1}, S_{t-1} = s_{t-1}, S_1^{t-2} = s_1^{t-2}) \\
&\quad = P_\theta(x_t \mid S_t = v, x_{t-r}^{t-1}) P_\theta(S_t = v \mid S_{t-1} = s_{t-1}) \qquad \text{(cf. Eq. 2.2 et 2.3)} \\
&\qquad\quad \times P_\theta(x_1^{t-1}, S_{t-1} = s_{t-1}, S_1^{t-2} = s_1^{t-2}) \\
&\quad = b(x_t; v, x_{t-r}^{t-1}) a(s_{t-1}, v) P_\theta(x_1^{t-1}, S_{t-1} = s_{t-1}, S_1^{t-2} = s_1^{t-2}) \ ,
\end{aligned}$$

d'où l'équation 2.5 :

$$\max_{s_1^t} P_\theta(x_1^t, S_t = v, S_1^{t-1} = s_1^{t-1}) =$$
$$\max_{s_{t-1}} \left(b_v(x_t; x_{t-r}^{t-1}) a(s_{t-1}, v) \times \max_{s_1^{t-2}} P_\theta(x_1^{t-1}, S_{t-1} = s_{t-1}, S_1^{t-2} = s_1^{t-2}) \right) .$$

2.2.2 L'algorithme *forward-backward*

L'algorithme *forward-backward* [BPSW70] permet de calculer, à chaque position t de la séquence, la probabilité de chaque couple d'états ($S_t =$

$u, S_{t+1} = v)$ et de chaque état $(S_t = u)$ conditionnellement à la séquence observée :

$$P_\theta(S_t = u, S_{t+1} = v \mid X_1^n = x_1^n) \text{ et } P_\theta(S_t = u \mid X_1^n = x_1^n),\ (u, v) \in \mathcal{S}^2 \ .$$

Comme l'algorithme de Viterbi, l'algorithme *forward-backward* réalise une récurrence « avant-arrière » d'où lui vient son nom.

La récurrence avant calcule, pour t croissant de 1 à n, les probabilités $P_\theta(S_t = v \mid x_1^{t-1})$ et $P_\theta(S_t = v \mid x_1^t)$ grâce à deux équations.

La première est l'équation « prédictive » :

$$\begin{aligned} P_\theta(S_1 = v) &= a(v) \quad \text{pour} \quad t = 1 \\ P_\theta(S_t = v \mid x_1^{t-1}) &= \sum_{u \in \mathcal{S}} a(u, v) P_\theta(S_{t-1} = u \mid x_1^{t-1}) \quad \text{pour} \quad t > 1 \ , \end{aligned}$$

la seconde est l'équation « de filtrage » :

$$P_\theta(S_t = v | x_1^t) = \frac{b_v(x_t; x_{t-r}^{t-1}) P_\theta(S_t = v \mid x_1^{t-1})}{\sum_{u \in \mathcal{S}} b_u(x_t; x_{t-r}^{t-1}) P_\theta(S_t = u \mid x_1^{t-1})} \ .$$

La récurrence arrière permet ensuite d'obtenir, pour t décroissant de $n-1$ à 1, les probabilités $P_\theta(S_t = u, S_{t+1} = v \mid x_1^n)$ et $P_\theta(S_t = u \mid x_1^n)$ en utilisant l'équation « de lissage » :

$$P_\theta(S_t = u, S_{t+1} = v \mid x_1^n) = \frac{P_\theta(S_t = u \mid x_1^t) a(u, v) P_\theta(S_{t+1} = v \mid x_1^n)}{P_\theta(S_{t+1} = v \mid x_1^t)} \ ,$$

d'où on déduit

$$P_\theta(S_t = u \mid x_1^n) = \sum_{v \in \mathcal{S}} P_\theta(S_t = u, S_{t+1} = v \mid x_1^n) \ .$$

L'algorithme *forward-backward* est parfois réalisé sous une forme différente (voir par exemple [Rab89, DEKM98]) dont la récurrence avant consiste à calculer $P_\theta(x_1^t, S_t = u)$ et la récurrence arrière $P_\theta(x_{t+1}^n \mid S_t = u, x_1^t)$. À partir de ces probabilités, on obtient aisément $P_\theta(S_t = u \mid x_1^n)$ et $P_\theta(S_t = u, S_{t+1} = v \mid x_1^n)$. Dans ce cas, il faut prévenir les problèmes numériques liés aux calculs de très petites probabilités pendant les récurrences avant et arrière, par exemple en travaillant avec les logarithmes.

Les probabilités $P_\theta(S_t = u \mid x_1^n)$ et $P_\theta(S_t = u, S_{t+1} = v \mid x_1^n)$ permettent d'obtenir la probabilité de n'importe quelle partie d'un chemin caché s_{t1}^{t2} conditionnellement à la séquence observée $P_\theta(S_{t1}^{t2} = s_{t1}^{t2} \mid x_1^n)$:

$$\begin{aligned} &P_\theta(S_{t1}^{t2} = s_{t1}^{t2} \mid x_1^n) = \\ &\qquad P_\theta(s_{t1}, s_{t1+1} \mid x_1^n) \times \frac{P_\theta(s_{t1+1}, s_{t1+2} \mid x_1^n)}{P_\theta(s_{t1+1} \mid x_1^n)} \times \ldots \times \frac{P_\theta(s_{t2-1}, s_{t2} \mid x_1^n)}{P_\theta(s_{t2-1} \mid x_1^n)} \ . \end{aligned}$$

Il faut enfin remarquer que lorsque certaines transitions entre états cachés sont interdites ($a(u,v) = 0$), les probabilités calculées grâce à l'algorithme *forward-backward* peuvent ne pas permettre de proposer facilement un unique chemin compatible avec la structure du modèle. En effet, $a\big(\max_u(P_\theta(S_t = u \mid x_1^n)), \max_u(P_\theta(S_{t+1} = u \mid x_1^n))\big)$ peut être égal à 0.

—*Justifications des équations de l'algorithme* forward-backward —

équation prédictive

$$\begin{aligned}
&P_\theta(S_t = v \mid x_1^{t-1}) \\
&\quad = \sum_{u\in\mathcal{S}} P_\theta(S_{t-1} = u, S_t = v \mid x_1^{t-1}) \\
&\quad = \sum_{u\in\mathcal{S}} P_\theta(S_t = v \mid S_{t-1} = u, x_1^{t-1}) P_\theta(S_{t-1} = u \mid x_1^{t-1}) \\
&\quad = \sum_{u\in\mathcal{S}} P_\theta(S_t = v \mid S_{t-1} = u) P_\theta(S_{t-1} = u \mid x_1^{t-1}) \qquad \text{(cf. Eq. 2.3)} \\
&\quad = \sum_{u\in\mathcal{S}} a(u,v) P_\theta(S_{t-1} = u \mid x_1^{t-1})
\end{aligned}$$

équation de filtrage

$$\begin{aligned}
P_\theta(S_t = v|x_1^t) &= P_\theta(S_t = v \mid x_t, x_1^{t-1}) \\
&= \frac{P_\theta(S_t = v, x_t \mid x_1^{t-1})}{P_\theta(x_t \mid x_1^{t-1})} \\
&= \frac{P_\theta(S_t = v, x_t \mid x_1^{t-1})}{\sum_{u\in\mathcal{S}} P_\theta(S_t = u, x_t \mid x_1^{t-1})} \\
&= \frac{P_\theta(x_t \mid S_t = v, x_1^{t-1}) P_\theta(S_t = v \mid x_1^{t-1})}{\sum_{u\in\mathcal{S}} P_\theta(x_t \mid S_t = u, x_1^{t-1}) P_\theta(S_t = u \mid x_1^{t-1})} \\
&= \frac{P_\theta(x_t \mid S_t = v, x_{t-r}^{t-1}) P_\theta(S_t = v \mid x_1^{t-1})}{\sum_{u\in\mathcal{S}} P_\theta(x_t \mid S_t = u, x_{t-r}^{t-1}) P_\theta(S_t = u \mid x_1^{t-1})} \qquad \text{(cf. Eq. 2.2)} \\
&= \frac{b_v(x_t; x_{t-r}^{t-1}) P_\theta(S_t = v \mid x_1^{t-1})}{\sum_{u\in\mathcal{S}} b_u(x_t; x_{t-r}^{t-1}) P_\theta(S_t = u \mid x_1^{t-1})}
\end{aligned}$$

équation de lissage

$$\begin{aligned}
&P_\theta(S_{t-1}=u, S_t=v \mid x_1^n)\\
&\quad= P_\theta(S_{t-1}=u \mid S_t=v, x_1^n) P_\theta(S_t=v \mid x_1^n)\\
&\quad= P_\theta(S_{t-1}=u \mid S_t=v, x_1^{t-1}) P_\theta(S_t=v \mid x_1^n) \qquad \text{(cf. Eq. 2.4)}\\
&\quad= \frac{P_\theta(S_{t-1}=u, S_t=v \mid x_1^{t-1}) P_\theta(S_t=v \mid x_1^n)}{P_\theta(S_t=v \mid x_1^{t-1})}\\
&\quad= \frac{P_\theta(S_{t-1}=u \mid x_1^{t-1}) P_\theta(S_t=v \mid S_{t-1}=u, x_1^{t-1}) P_\theta(S_t=v \mid x_1^n)}{P_\theta(S_t=v \mid x_1^{t-1})}\\
&\quad= \frac{P_\theta(S_{t-1}=u \mid x_1^{t-1}) a(u,v) P_\theta(S_t=v \mid x_1^n)}{P_\theta(S_t=v \mid x_1^{t-1})} \qquad \text{(cf. Eq. 2.3)}
\end{aligned}$$

2.2.3 Simulation du chemin caché

On décrit ici un algorithme permettant de tirer $\tilde{s}_1^n$ avec la probabilité $P_\theta(S_1^n = \tilde{s}_1^n \mid x_1^n)$, il pourra notamment servir pour l'estimation bayésienne sur les données incomplètes (voir par exemple [QT91, RCD93, Mur98]).

Le terme $P_\theta(S_n = v \mid x_1^n)$, obtenu à la fin de la récurrence avant de l'algorithme *forward-backward*, permet de tirer $\tilde{s}_n$. Les $\tilde{s}_t$ peuvent ensuite être tirés, pour t décroissant de $n-1$ à 1, selon les probabilités :

$$\begin{aligned}
&P_\theta(S_t=u \mid S_{t+1}^n = \tilde{s}_{t+1}^n, x_1^n)\\
&\quad= P_\theta(S_t=u \mid S_{t+1}=\tilde{s}_{t+1}, x_1^t)\\
&\quad= \frac{P_\theta(S_t=u, S_{t+1}=\tilde{s}_{t+1} \mid x_1^t)}{\sum_{v\in\mathcal{S}} P_\theta(S_t=v, S_{t+1}=\tilde{s}_{t+1} \mid x_1^t)}\\
&\quad= \frac{P_\theta(S_{t+1}=\tilde{s}_{t+1} \mid S_t=u) P_\theta(S_t=u \mid x_1^t)}{\sum_{v\in\mathcal{S}} P_\theta(S_{t+1}=\tilde{s}_{t+1} \mid S_t=u) P_\theta(S_t=u \mid x_1^t)}\\
&\quad= \frac{a(u,\tilde{s}_{t+1}) P_\theta(S_t=u \mid x_1^t)}{\sum_{v\in\mathcal{S}} a(u,\tilde{s}_{t+1}) P_\theta(S_t=u \mid x_1^t)} \ .
\end{aligned}$$

2.3 Calcul de la vraisemblance des données incomplètes

La probabilité $P_\theta(X_1^n)$ correspond à la vraisemblance des paramètres θ pour l'observation X_1^n. On l'appellera vraisemblance des données incomplètes pour la distinguer de $P_\theta(S_1^n, X_1^n)$. Il est souvent intéressant de la calculer, aussi bien pour contrôler la convergence d'un algorithme de maximisation de la vraisemblance que pour utiliser le modèle dans le cadre d'un test statistique. La vraisemblance peut s'écrire comme la somme, sur tous les chemins cachés possibles s_1^n, de la vraisemblance des données complètes :

$$P_\theta(X_1^n) = \sum_{s_1^n} P_\theta(X_1^n, S_1^n = s_1^n) \ .$$

Dans cette somme, chacun des termes est facile à évaluer (voir Eq. 2.1) mais, comme on l'a déjà évoqué, le nombre de termes est définitivement rédhibitoire.

Quelle que soit la stratégie employée pour calculer la vraisemblance des données incomplètes, ce calcul repose, comme la reconstruction du chemin caché, sur les propriétés d'indépendance conditionnelle au sein des HMM. Elle peut par exemple être calculée au cours de la récurrence avant de l'algorithme *forward-backward*. On a en effet :

$$\begin{aligned} P_\theta(X_1^t = x_1^t) &= \sum_{u\in\mathcal{S}} P_\theta(X_t = x_t, S_t = u, x_1^{t-1}) \\ &= \sum_{u\in\mathcal{S}} P_\theta(X_t = x_t \mid S_t = u, x_{t-r}^{t-1}) P_\theta(S_t = u, x_1^{t-1}) \\ &= P_\theta(X_1^{t-1} = x_1^{t-1}) \sum_{u\in\mathcal{S}} b_u(x_t; x_{t-r}^{t-1}) P_\theta(S_t = u \mid x_1^{t-1}) \ , \end{aligned}$$

où le terme $P_\theta(S_t = u \mid x_1^{t-1})$ est calculé pendant la récurrence avant grâce à l'équation prédictive. Ce calcul doit être fait avec les logarithmes car la vraisemblance est très petite :

$$\log P_\theta(X_1^n = x_1^n) = \sum_{t=1}^{n} \log\Big(\sum_{u\in\mathcal{S}} b_u(x_t; x_{t-r}^{t-1}) P_\theta(S_t = u \mid x_1^{t-1})\Big) \ .$$

Signalons que cette dernière formule rend possible le calcul récursif des dérivées partielles de la logvraisemblance et l'optimisation des paramètres du modèle au cours de la récurrence [Mev97].

2.4 Estimation des paramètres

L'estimation des paramètres est généralement une étape incontournable de l'utilisation d'un modèle probabiliste pour analyser des données. Un estimateur est une fonction des observations vers l'espace des paramètres choisie pour proposer un jeu de paramètres proche de celui supposé avoir généré les données observées. Avec les chaînes de Markov cachées, et plus généralement les modèles à variables cachées, le problème de l'estimation peut se poser de deux façons selon le type de données disponibles. On dispose parfois des données « complètes » (x_1^n, s_1^n). C'est à dire que la suite des états cachés est connue, par exemple grâce à des études expérimentales. Dans d'autres cas, seules les données « incomplètes » x_1^n sont disponibles. Évidemment le problème est plus difficile dans le second cas. Il est aussi plus intéressant car il offre la possibilité de mettre en évidence de propriétés des observations inattendues avant l'analyse. Enfin, dans certains cas, les données complètes ne sont connues que partiellement, l'estimation relève alors principalement des techniques utilisées sur les données incomplètes.

Les statistiques proposent deux cadres théoriques pour aborder l'estimation des paramètres d'un modèle. Dans le cadre « classique » (« fréquentiste » pour les statisticiens bayésiens), on cherche à construire une fonction des observations dont les propriétés sont étudiées pour différentes valeurs des paramètres réels. Dans ce cadre, l'estimateur du maximum de vraisemblance θ^{ML} défini sur les données complètes par

$$\theta^{\mathrm{ML}} = \arg\max_{\theta} P_\theta(X_1^n, S_1^n) \ ,$$

ou sur les données incomplètes par

$$\theta^{\mathrm{ML}} = \arg\max_{\theta} P_\theta(X_1^n) \ ,$$

est un des estimateurs les plus couramment utilisés car ses propriétés sont bonnes quand le jeu de données devient assez grand. Il est notamment consistant, c'est à dire que les paramètres estimés tendent vers les paramètres réels lorsque la quantité d'observations augmente suffisamment. On pourra consulter par exemple [BP66, lGM00] pour des résultats de consistance et de normalité asymptotique de l'estimateur du maximum de vraisemblance dans le cas discret (stationnaire et non stationnaire).

Le cadre bayésien aborde le problème sous un angle différent. Grâce à l'introduction d'une loi *a priori* sur les paramètres dont on notera $\pi(\theta)$ la densité[3], il permet d'étudier la distribution *a posteriori* des paramètres. On considère donc les paramètres comme des variables aléatoires, la loi *a priori* représentant l'information disponible *a priori* sur leur valeur. Les densités des lois *a posteriori* sont obtenues par la formule de Bayes. La densité de la loi *a posteriori* sur les données complètes $\pi(\theta \mid s_1^n, x_1^n)$ est

$$\begin{aligned} \pi(\theta \mid s_1^n, x_1^n) &= \frac{P_\theta(x_1^n, s_1^n)\pi(\theta)}{\int_\theta P_\theta(x_1^n, s_1^n)\pi(\theta)d\theta} \\ &\propto P_\theta(x_1^n, s_1^n)\pi(\theta) \ . \end{aligned} \tag{2.6}$$

De la même façon, la densité $\pi(\theta \mid x_1^n)$ de la loi *a posteriori* sur les données incomplètes est

$$\pi(\theta \mid x_1^n) = \frac{P_\theta(x_1^n)\pi(\theta)}{\int_\theta P_\theta(x_1^n)\pi(\theta)d\theta} \ .$$

Dans le cadre bayésien l'inférence repose sur la loi *a posteriori* qui est un compromis entre les connaissances *a priori* et les observations. La loi *a priori* peut cependant être non (ou très peu) informative. L'introduction d'une loi *a priori* sur les paramètres peut paraître discutable mais permet

[3]Conformément à un usage répandu, dans le cadre bayésien presque toutes les densités seront appelées π. Il s'agit selon le contexte de densités définies par rapport aux mesures de Lebesgue (pour les variables continues) ou de comptage (pour les variables discrètes).

de construire un cadre très séduisant pour aborder les problèmes statistiques (voir par exemple [Rob92]). À partir de cette distribution *a posteriori* on peut définir différents estimateurs dont les plus courants sont l'espérance de la distribution *a posteriori* et le maximum de la distribution *a posteriori* (MAP, Maximum *a posteriori*). Choisir l'espérance revient à minimiser une fonction de coût quadratique. Le MAP, quant à lui, peut être identique au maximum de vraisemblance pour certaines lois *a priori*. Il possède généralement des propriétés identiques à celles de l'estimateur du maximum de vraisemblance quand la quantité d'observations disponibles pour l'estimation augmente.

Dans cette section on va aborder successivement l'estimation sur les données complètes et l'estimation sur les données incomplètes. Dans chaque cas l'estimation par maximum de vraisemblance et l'estimation bayésienne seront présentées.

Pour alléger les formules, l'estimation sera supposée réalisée sur une unique séquence observée. Les estimateurs des paramètres des lois initiales des différents modèles markoviens sont dans ce cas peu intéressants et ne seront donc pas explicités. La modification des estimateurs pour l'estimation sur un ensemble de séquences modélisées comme des réalisations indépendantes d'un même HMM est directe. Des procédures d'estimation des paramètres des lois initiales, similaires à celles utilisées pour les autres paramètres, peuvent alors être utilisées.

2.4.1 Estimation sur les données complètes

Maximisation de la vraisemblance sur les données complètes

La vraisemblance sur les données complètes s'écrit (cf. Eq. 2.1)

$$\begin{aligned} P_\theta(X_1^n, S_1^n) &= a(S_1) \prod_{t=2}^{n} a(S_{t-1}, S_t) \prod_{t=1}^{n} b_{S_t}(X_t; X_{t-r}^{t-1}) \\ &= a(S_1) \prod_{t=1}^{r} b_{S_t}(X_t; X_1^{t-1}) \\ &\quad \times \prod_{u \in \mathcal{S}} \prod_{v \in \mathcal{S}} a(u,v)^{N_{uv}} \prod_{u \in \mathcal{S}} \prod_{w \in \mathcal{X}^r} \prod_{x \in \mathcal{X}} b_u(x; w)^{N_{wx;u}} \ , \end{aligned} \tag{2.7}$$

où le terme $a(S_1) \prod_{t=1}^{r} b_{S_t}(X_t; X_1^{t-1})$ correspond à la contribution des paramètres des lois initiales. Dans cette formule, N_{uv} désigne le comptage du nombre de transitions de u vers v dans S_1^n et $N_{wx;u}$ le nombre d'occurrences du mot wx (de longueur $r+1$) dans X_1^n dont la dernière lettre x est dans

l'état u :

$$\begin{aligned} N_{uv} &= \sum_{t=1}^{n-1} \mathbb{I}\{S_t = u, S_{t+1} = v\} \\ N_{wx;u} &= \sum_{t=r+1}^{n} \mathbb{I}\{X_{t-r}^t = wx, S_t = u\} \ , \end{aligned}$$

où $\mathbb{I}$ est la fonction indicatrice : $\mathbb{I}\{S_t = u, S_{t+1} = v\}$ vaut 1 si $S_t = u$ et $S_{t+1} = v$, 0 sinon.

On travaille en général avec le logarithme de la vraisemblance

$$\log P_\theta(X_1^n, S_1^n) = \log\left(a(S_1)\prod_{t=1}^{r} b_{S_t}(X_t; X_1^{t-1})\right) + \underbrace{\sum_{u\in\mathcal{S}}\sum_{v\in\mathcal{S}} N_{uv}\log a(u,v)}_{(1)} + \underbrace{\sum_{u\in\mathcal{S}}\sum_{w\in\mathcal{X}^r}\sum_{x\in\mathcal{X}} N_{wx;u}\log b_u(x;w)}_{(2)} .$$

L'estimateur du maximum de vraisemblance est obtenu par maximisation séparée de chacun des termes désignés par une accolade sous la contrainte $\sum_v a(u,v) = 1$ pour les termes (1) et $\sum_x b(x;u,w) = 1$ pour les termes (2). L'estimateur $\theta^{\mathrm{ML}} = (a^{\mathrm{ML}}, b^{\mathrm{ML}})$ est donc :

$$a^{\mathrm{ML}}(u,v) = \frac{N_{uv}}{\sum_{v'\in\mathcal{S}} N_{uv'}} \tag{2.8}$$

$$b_u^{\mathrm{ML}}(x;w) = \frac{N_{wx;u}}{\sum_{x'\in\mathcal{X}} N_{wx';u}} \ . \tag{2.9}$$

Il s'agit d'un estimateur très intuitif puisque, par exemple, la probabilité de transition de l'état u à l'état v $(a(u,v))$ est estimée par la fréquence empirique de v parmi les états apparaissant juste après un u.

Estimation bayésienne sur les données complètes

Loi *a priori.* Il faut tout d'abord introduire une loi *a priori* sur les paramètres θ. On choisit classiquement des lois de Dirichlet[4] (voir par exemple [RCD93, Mur97, BHW00]) aussi bien pour les probabilités de transition entre états cachés que pour les probabilités d'apparition des observations conditionnellement aux états cachés,.

Une loi de Dirichlet $D_q(\delta_1, \ldots, \delta_q)$, avec $\delta_i > 0$ pour $i \in \{1, \ldots q\}$, génère un vecteur $(y_1, \ldots, y_q)$ de longueur q dont les éléments sont à valeurs dans

[4] La loi de Dirichlet est une généralisation multidimensionnelle de la loi bêta.

$[0,1]$ et somment à 1. La densité de $D_q(\delta_1,\ldots,\delta_q)$ s'écrit :

$$\begin{aligned}\pi(y_1,y_2,\ldots,y_q) &= \frac{\Gamma(\sum_{i=1}^q \delta_i)}{\prod_{i=1}^q \Gamma(\delta_i)} \prod_{i=1}^q y_i^{\delta_i-1} \mathbb{I}_{\{\cap_{i=1,\ldots,q}(y_i\in[0,1]),\sum_{i=1}^q y_i=1\}} \\ &\propto \prod_{i=1}^q y_i^{\delta_i-1} \mathbb{I}_{\{\cap_{i=1,\ldots,q}(y_i\in[0,1]),\sum_{i=1}^q y_i=1\}} \ .\end{aligned}$$

Le choix des δ_i dépend de l'information *a priori* que l'on souhaite introduire. Lorsque tous les δ_i sont égaux à 1, la densité charge uniformément l'espace et la loi *a priori* peut être considérée comme non informative[5]. Par ailleurs, l'espérance de chacune des variables y_i est égale à $\delta_i/\sum_j \delta_j$ où $\sum_j \delta_j$ reflète la quantité d'information introduite *a priori*. Un grand intérêt de l'utilisation de la loi de Dirichlet comme loi *a priori* des paramètres des modèles considérés ici est que la loi *a posteriori* est alors comme la loi *a priori* une loi de Dirichlet (la loi Dirichlet est la loi conjuguée de la loi multinomiale).

Les lignes des matrices de transition des différents modèles markoviens sont modélisées par des lois de Dirichlet indépendantes, c'est à dire que pour chaque $u\in\mathcal{S}$ et $w\in\mathcal{X}^r$:

$$a(u,\bullet) \sim D_{|\mathcal{S}|}\big(\alpha(u,1),\ldots,\alpha(u,|\mathcal{S}|)\big) \tag{2.10}$$

$$b_u(\bullet;w) \sim D_{|\mathcal{X}|}\big(\beta_u(1;w),\ldots,\beta_u(|\mathcal{X}|;w)\big) \ , \tag{2.11}$$

où $a(u,\bullet)$ représente la ligne u de la matrice de transitions de la chaîne cachée et $b_u(\bullet;w)$ la ligne de la loi d'émission des observations dans l'état u associée au contexte w. La densité de la loi *a priori* de $\theta=(a,b)$ s'écrit donc

$$\begin{aligned}\pi(\theta) &\propto \prod_{u\in\mathcal{S}} \pi(a(u,\bullet)) \prod_{u\in\mathcal{S}}\prod_{w\in\mathcal{X}^r} \pi(b_u(\bullet;w)) \\ &\propto \prod_{u\in\mathcal{S}}\prod_{v\in\mathcal{S}} a(u,v)^{\alpha(u,v)-1} \prod_{u\in\mathcal{S}}\prod_{w\in\mathcal{X}^r}\prod_{x\in\mathcal{X}} b_u(x;w)^{\beta_u(x;w)-1} \ . \end{aligned} \tag{2.12}$$

Lorsque l'on souhaite fixer la valeur de certains paramètres, par exemple pour interdire la transition entre l'état caché u et l'état caché v ($a(u,v)=0$), il suffit de ne pas modéliser ces paramètres comme étant aléatoires. Les lois *a priori* (Eq. 2.10 et 2.11) ne concernent alors que les paramètres aléatoires.

Loi *a posteriori* et estimation sur les données complètes. La loi *a posteriori* des paramètres θ sur les données complètes est obtenue grâce à la

[5] La loi *a priori* non informative au sens de Jeffrey est celle dont les δ_i sont égaux à $1/2$.

formule de Bayes (Eq. 2.6) en utilisant les équations 2.7 et 2.12 :

$$\begin{aligned}
&\pi(\theta \mid x_1^n, s_1^n) \\
&\quad \propto \quad P_\theta(x_1^n, s_1^n)\pi(\theta) \\
&\quad \propto \quad \prod_u \prod_v a(u,v)^{N_{uv}} \prod_u \prod_w \prod_x b_u(x;w)^{N_{wx;u}} \\
&\qquad\qquad \times \prod_u \prod_v a(u,v)^{\alpha(u,v)-1} \prod_u \prod_w \prod_x b_u(x;w)^{\beta_u(x;w)-1} \\
&\quad \propto \quad \underbrace{\prod_u \prod_v a(u,v)^{N_{uv}+\alpha(u,v)-1}}_{(1)} \times \underbrace{\prod_u \prod_w \prod_x b_u(x;w)^{N_{wx;u}+\beta_u(x;w)-1}}_{(2)}.
\end{aligned}$$

On identifie là un produit de densités de lois de Dirichlet. Les lois *a posteriori* des différents paramètres sont donc des lois de Dirichlet indépendantes :

$$\begin{aligned}
a(u,\bullet) \mid x_1^n, s_1^n \quad &\sim \quad D_{|\mathcal{X}|}(N_{u1} + \alpha(u,1), \ldots, N_{u|\mathcal{S}|} + \alpha(u,|\mathcal{S}|)) \\
&\qquad \text{identifiée à partir des termes (1)} \qquad (2.13) \\
b_u(\bullet;w) \mid x_1^n, s_1^n \quad &\sim \quad D_{|\mathcal{X}|}(N_{w1;u} + \beta_u(1;w), \ldots, N_{w|\mathcal{X}|;u} + \beta_u(|\mathcal{X}|;w)) \\
&\qquad \text{identifiée à partir des termes (2) .} \qquad (2.14)
\end{aligned}$$

On remarque que les paramètres $\alpha(u,v)$ et $\beta_u(x;w)$ des lois *a priori* jouent un rôle similaire aux comptages N_{uv} et $N_{wx;u}$ dans la détermination des paramètres des lois *a posteriori*, ce qui leur confère une interprétation intuitive en terme de quantité d'information introduite *a priori*.

L'estimateur le plus couramment utilisé correspond à l'espérance de la loi *a posteriori* pour les différents paramètres, soit θ^{Bayes} :

$$\begin{aligned}
a^{\text{Bayes}}(u,v) &= \frac{N_{uv} + \alpha(u,v)}{\sum_v N_{uv} + \alpha(u,v)} \\
b_u^{\text{Bayes}}(x;w) &= \frac{N_{wx;u} + \beta(x;u,w)}{\sum_x N_{wx;u} + \beta(x;u,w)}
\end{aligned}$$

Par une maximisation similaire à celle utilisée dans le cadre de la maximisation de la vraisemblance, l'estimation par le maximum de la loi *a posteriori* conduit elle à θ^{MAP} :

$$\begin{aligned}
a^{\text{MAP}}(u,v) &= \frac{N_{uv} + \alpha(u,v) - 1}{\sum_v N_{uv} + \alpha(u,v) - 1} \\
b_u^{\text{MAP}}(x;w) &= \frac{N_{wx;u} + \beta(x;u,w) - 1}{\sum_x N_{wx;u} + \beta(x;u,w) - 1}.
\end{aligned}$$

L'estimateur du maximum *a posteriori* est donc identique à celui du maximum de vraisemblance lorsque tous les paramètres des lois de Dirichlet *a priori* sont pris égaux à 1.

Remarques sur l'estimation à partir des données complètes

Évidemment, les estimateurs sont toujours meilleurs lorsque l'on dispose de davantage de données pour l'estimation (pourvu que les données proviennent toutes d'un même modèle). Mais, lorsque la quantité de données disponible est insuffisante, l'estimateur du maximum de vraisemblance a un comportement nettement moins agréable que l'estimateur bayésien. Par exemple, un comptage nul conduit à estimer un paramètre à 0 alors que le jeu de données peut être d'une taille telle que l'observation de comptages nuls est attendue même si le paramètre est assez différent de 0. Une solution couramment employée est de rajouter des quantités aux comptages observés (par exemple 1 à tous les comptages), on parle alors de pseudo-comptages. Le cadre bayésien donne une justification à cette pratique : les quantités ajoutées aux comptages correspondent aux paramètres des lois de Dirichlet *a priori*.

Lorsque l'on dispose de plusieurs séquences modélisées comme des réalisations indépendantes d'un même HMM, on aboutit aisément à des estimateurs de forme identique à ceux présentés ci-dessus où les comptages sont effectués sur l'ensemble des séquences. Dans ce cas, les paramètres $a(u)$ et $b_u(x;w)$ avec $w \in \mathcal{X}^{r'}$ et $r' < r$, qui correspondent aux distributions initiales des modèles markoviens, peuvent être estimés selon une procédure identique à celle mise en œuvre pour les autres paramètres.

Enfin, lorsque le HMM est estimé sur les données complètes pour prédire ultérieurement la nature des séquences observées à l'aide d'une reconstruction du chemin caché, il peut être plus efficace d'utiliser un estimateur qui maximise directement un critère de discrimination des différentes natures de séquences. On peut par exemple chercher θ^{CML} (*Conditional Maximum Likelihood*) qui correspond à $\theta^{\mathrm{CML}} = \arg\max_\theta P_\theta(S_1^n \mid X_1^n)$. Cet estimateur est obtenu par optimisation numérique car à la différence des estimateurs présentés dans cette section il ne peut être obtenu directement [NM91, Kro94, Kro97].

2.4.2 Estimation sur les données incomplètes

Le problème de l'estimation sur les données incomplètes est nettement plus difficile que l'estimation sur les données complètes. Les estimateurs ne peuvent être exprimés sous une forme analytique simple. Ils sont obtenus grâce à des procédures numériques itératives qui utilisent la possibilité de reconstruire le chemin caché d'un HMM (ou de calculer la vraisemblance des données incomplètes) pour un coût de calcul raisonnable. On pourra par exemple consulter [Rab89, RCD93, Mur98, DEKM98].

Pour l'estimation par maximum de vraisemblance, l'algorithme EM est le plus utilisé. Pour l'estimation bayésienne, la loi *a posteriori* ne peut être obtenue analytiquement mais il est possible de l'échantillonner grâce à des

méthodes de Monte-Carlo par chaîne de Markov (MCMC).

L'algorithme EM

L'algorithme EM (*Expectation-Maximization*) est un algorithme itératif de maximisation locale de la vraisemblance des données incomplètes (ici $P_\theta(x_1^n)$) introduit par Dempster *et al.* (1977) [DLR77] dans les modèles à données manquantes (ici s_1^n). Il est aussi appelé algorithme de Baum-Welch [BPSW70] dans le cadre des HMM. À partir d'un jeu initial de paramètres $\theta^{(0)}$, chaque itération m permet d'obtenir un nouveau jeu de paramètres $\theta^{(m)}$ qui augmente la vraisemblance $P_{\theta^{(m)}}(x_1^n) > P_{\theta^{(m-1)}}(x_1^n)$. Une itération de l'algorithme est constituée d'une étape E qui correspond au calcul de l'espérance conditionnelle $Q(\theta \mid \theta^{(m-1)}) = E_{\theta^{(m-1)}}\big(\log P_\theta(x_1^n, s_1^n) \mid x_1^n\big)$ et d'une étape M qui consiste à proposer $\theta^{(m)}$ maximisant $Q(\theta \mid \theta^{(m-1)})$ en θ .

La justification de l'algorithme se fonde sur l'expression de la logvraisemblance des données incomplètes $\log P_\theta(x_1^n)$ comme une différence d'espérances conditionnelles. On a d'après la formule de Bayes :

$$\log P_\theta(x_1^n) \quad = \quad \log P_\theta(x_1^n, s_1^n) - \log P_\theta(s_1^n \mid x_1^n) \ .$$

D'où, en prenant l'espérance de chacun des termes sous $P_{\theta'}(s_1^n \mid x_1^n)$, on obtient

$$\begin{aligned} \log P_\theta(x_1^n) &= \sum_{s_1^n} \log P_\theta(x_1^n, s_1^n) P_{\theta'}(s_1^n \mid x_1^n) - \sum_{s_1^n} \log P_\theta(s_1^n \mid x_1^n) P_{\theta'}(s_1^n \mid x_1^n) \\ &= \underbrace{E_{\theta'}(\log P_\theta(x_1^n, s_1^n) \mid x_1^n)}_{Q(\theta|\theta')} - \underbrace{E_{\theta'}(\log P_\theta(s_1^n \mid x_1^n) \mid x_1^n)}_{H(\theta|\theta')} \ , \end{aligned}$$

car $E_{\theta'}(\log P_\theta(x_1^n) \mid x_1^n) = \log P_\theta(x_1^n)$.

Comme on l'a déjà évoqué l'étape E consiste à calculer $Q(\theta \mid \theta^{(m-1)})$ et l'étape M prend $\theta^{(m)} = \arg\max_\theta Q(\theta \mid \theta^{(m-1)})$. Ce choix de $\theta^{(m)}$ assure la croissance de la vraisemblance des données incomplètes. En effet,

$$\begin{aligned} &\log P_{\theta^{(m)}}(x_1^n) - \log P_{\theta^{(m-1)})}(x_1^n) \\ &\quad = \quad Q(\theta^{(m)} \mid \theta^{(m-1)}) - Q(\theta^{(m-1)} \mid \theta^{(m-1)}) \\ &\qquad\qquad -(H(\theta^{(m)} \mid \theta^{(m-1)}) - H(\theta^{(m-1)} \mid \theta^{(m-1)})) \\ &\quad \geq \quad 0 \ , \end{aligned}$$

puisque $Q(\theta^{(m)} \mid \theta^{(m-1)}) > Q(\theta^{(m-1)} \mid \theta^{(m-1)})$ par choix de $\theta^{(m)}$ et $H(\theta^{(m)} \mid \theta^{(m-1)}) < H(\theta^{(m-1)} \mid \theta^{(m-1)})$ d'après l'inégalité de Jensen.

Le calcul de $Q(\theta \mid \theta^{(m-1)})$ repose dans le cadre HMM sur l'algorithme

forward-backward. En effet,

$$\begin{aligned}
Q(\theta \mid \theta^{(m-1)}) &= E_{\theta^{(m-1)}}(\log P_\theta(x_1^n, s_1^n) \mid x_1^n) \\
&= E_{\theta^{(m-1)}}\Big(\log\big(a(s_1)\prod_{t=1}^{r} b_{s_t}(x_t; x_1^{t-1})\big) \\
&\qquad + \sum_{u,v} N_{uv} \log a(u,v) + \sum_{u,w,x} N_{wx;u} \log b_u(x;w) \;\Big|\; x_1^n \Big) \\
&= E_{\theta^{(m-1)}}(\log\big(a(s_1)\prod_{t=1}^{r} b_{s_t}(x_t; x_1^{t-1})\big) \mid x_1^n) \\
&\qquad + \sum_{u,v} \underbrace{E_{\theta^{(m-1)}}(N_{uv} \mid x_1^n)}_{n_{uv}^{(m-1)}} \log a(u,v) \\
&\qquad + \sum_{u,w,x} \underbrace{E_{\theta^{(m-1)}}(N_{wx;u} \mid x_1^n)}_{n_{wx;u}^{(m-1)}} \log b_u(x;w) \ ,
\end{aligned}$$

où $n_{uv}^{(m-1)}$ et $n_{wx;u}^{(m-1)}$ s'expriment en fonction des probabilités calculées grâce à l'algorithme *forward-backward* :

$$\begin{aligned}
n_{uv}^{(m-1)} &= \sum_t P_{\theta^{(m-1)}}(S_t = u, S_{t+1} = v \mid x_1^n) \\
n_{wx;u}^{(m-1)} &= \sum_t \mathbb{I}\{X_{t-r_u}^t = wx\} P_{\theta^{(m-1)}}(S_t = u \mid x_1^n) \ .
\end{aligned}$$

La nouvelle valeur $\theta^{(m)} = \arg\max_\theta Q(\theta \mid \theta^{(m-1)})$ des paramètres s'obtient par une maximisation similaire à celle de la vraisemblance des données complètes qui conduit à

$$a^{(m)}(u,v) = \frac{n_{uv}^{(m-1)}}{\sum_v n_{uv}^{(m-1)}} \tag{2.15}$$

$$b_u^{(m)}(x;w) = \frac{n_{wx;u}^{(m-1)}}{\sum_w n_{wx;u}^{(m-1)}} \ . \tag{2.16}$$

Ces formules sont similaires à celles de la maximisation de la vraisemblance sur les données complètes (Eq. 2.8 et 2.9) en remplaçant les comptages par leur espérance conditionnellement à x_1^n et aux paramètres $\theta^{(m-1)}$ (ce qui correspond à l'exploration de tous les chemins cachés possibles puisque le chemin est inconnu).

En résumé, l'algorithme EM alterne une reconstruction du chemin caché grâce à l'algorithme *forward-backward* avec une mise à jour des paramètres selon les équations 2.15 et 2.16. Il permet d'obtenir une suite $\theta^{(0)}, \theta^{(1)}, \ldots, \theta^{(m)}, \ldots$ assurant la croissance de la vraisemblance. Cette suite

converge vers le maximum de vraisemblance à condition que le point de départ $\theta^{(0)}$ en soit suffisamment proche (voir par exemple [BPSW70, Wu83, Mur97, EM02]). Si cette condition n'est pas satisfaite, la suite converge vers un maximum local (éventuellement vers un point selle). En pratique, deux problèmes se posent lors de l'utilisation de cet algorithme. Le premier est celui du choix d'un critère d'arrêt. Une solution simple et raisonnable consiste à arrêter l'algorithme à l'itération M telle que $\log P_{\theta^{(M)}}(x_1^n) - \log P_{\theta^{(M-1)}}(x_1^n) < \epsilon$ où ϵ est choisi à l'avance. Le second problème est sans doute plus sérieux : comment savoir si la valeur $\theta^{(M)}$ obtenue est proche du maximum global ? Il n'y a pas de réelle solution à ce problème et son ampleur dépend aussi bien du modèle que des données. Lorsque cela est possible, une première solution pragmatique est de choisir un point de départ que l'on pense pas trop éloigné du maximum global. Une autre solution qui peut être complémentaire de la première, consiste à réaliser plusieurs fois l'algorithme EM avec des initialisations différentes, par exemple aléatoires, puis à sélectionner après convergence les paramètres associés à la plus forte vraisemblance. L'observation de la dispersion des points de convergence permet alors aussi de se faire une idée de l'ampleur du problème posé par l'existence de maxima locaux.

Éstimation Bayésienne sur les données incomplètes par méthodes MCMC.

Comme on l'a déjà évoqué, la loi *a posteriori* de densité $\pi(\theta \mid x_1^n)$ ne peut être obtenue analytiquement. La clé de l'estimation bayésienne est l'échantillonnage de cette loi. En effet, si il était possible d'obtenir un échantillon indépendant $\theta^{(0)}, \theta^{(2)}, \ldots, \theta^{(m)}, \ldots$ de la loi *a posteriori*, on pourrait approcher l'espérance des différents paramètres $\theta = (\theta_i)_{i=1,\ldots,|\theta|}$ par méthode de Monte-Carlo

$$\theta_i^{\text{Bayes}} = \int_\theta \theta_i \pi(\theta \mid x_1^n) d\theta \tag{2.17}$$

$$\approx \frac{1}{M} \sum_{m=1}^{M} \theta_i^{(k)} \ , \tag{2.18}$$

et plus généralement n'importe quelle intégrale de la forme $\int_\theta f(\theta)\pi(\theta \mid x_1^n)d\theta$. Il n'est en réalité pas possible d'obtenir efficacement un tel échantillon indépendant. Cependant, les méthodes de Monte-Carlo par chaînes de Markov (MCMC) permettent de générer un échantillon de réalisations non indépendantes de la loi d'intérêt comme des réalisations de la loi stationnaire d'une chaîne de Markov. L'approximation 2.18 peut alors toujours être utilisée. On distingue classiquement deux grands types d'algorithmes MCMC : l'algorithme de Metropolis-Hastings et l'algorithme de Gibbs (voir par exemple [Has70, GG84, Tie94, WRGS96, Rob96] pour une présentation détaillée de ces méthodes).

Dans le cas des modèles à données incomplètes, l'idée essentielle est d'échantillonner non seulement les paramètres θ mais aussi les données manquantes (ici les s_1^n). On cherche alors à construire une chaîne de Markov dont la loi stationnaire est de densité $\pi(\theta, s_1^n \mid x_1^n)$ dont $\pi(\theta \mid x_1^n)$ est une marginale. Pour cela, on utilise un algorithme itératif. Chaque itération met à jour successivement les composantes θ et s_1^n, en les tirant (« à la Gibbs ») selon leur loi conditionnelle.
L'itération m de l'algorithme se décrit alors

étape 1 : mise à jour de θ en tirant $\theta^{(m)} \sim \theta \mid s_1^{n\,(m-1)}, x_1^n$,

étape 2 : mise à jour de s_1^n en tirant $s_1^{n\,(m)} \sim s_1^n \mid \theta^{(m)}, x_1^n$.

La loi de $\theta \mid s_1^n, x_1^n$ se décompose en lois de Dirichlet indépendantes pour les différents jeux de paramètres $a(u, \bullet)$ et $b_u(\bullet; w)$ (cf. estimation bayésienne sur les données complètes). Ces paramètres sont donc mis à jour indépendamment lors de l'étape 1 en les tirant selon les lois de Dirichlet données par les équations 2.13 et 2.14. La procédure de tirage du chemin caché selon la loi $s_1^n \mid \theta^{(m)}, x_1^n$ à l'étape 2 est, quant à elle, décrite dans la section 2.2.

L'échantillon $(\theta^{(0)}, s^{(0)}), (\theta^{(1)}, s^{(1)}), \ldots, (\theta^{(m)}, s^{(m)}), \ldots$ généré par l'algorithme MCMC, est une suite de réalisations non indépendantes de la loi de densité $\pi(\theta, s_1^n \mid x_1^n)$. En effet, la chaîne de Markov construite admet $\pi(\theta, s_1^n \mid x_1^n)$ comme loi invariante puisque chaque étape préserve par construction la loi d'intérêt $\pi(\theta, x_1^n \mid s_1^n)$. On vérifie par exemple que si (θ, s_1^n) est distribué selon la densité $\pi(\theta, s_1^n \mid x_1^n)$, alors le couple $\theta, \tilde{s}_1^n$ contenant le nouveau $\tilde{s}_1^n$ généré à l'étape **2** selon la densité $\pi(\tilde{s}_1^n \mid \theta, x_1^n)$ sera lui aussi distribué selon la densité $\pi(\theta, \tilde{s}_1^n \mid x_1^n)$:

$$\begin{aligned}\int_{s_1^n} \pi(\tilde{s}_1^n \mid \theta, x_1^n)\pi(\theta, s_1^n \mid x_1^n) ds_1^n &= \pi(\tilde{s}_1^n \mid x_1^n, \theta)\pi(\theta \mid x_1^n) \\ &= \pi(\tilde{s}_1^n, \theta \mid x_1^n) \ .\end{aligned}$$

De plus, cette loi invariante est unique et la chaîne de Markov converge pour (presque[6]) tout point de départ vers ce régime stationnaire. La condition de convergence est que la chaîne de Markov construite permette de passer de n'importe quel point de l'espace à n'importe quel autre point en un nombre fini d'itérations (ici une seule car $\pi(\theta \mid s_1^n, x_1^n) > 0$ et $\pi(s_1^n \mid \theta, x_1^n) > 0$). La méthode d'échantillonnage est en fait très générale et très souple. Elle peut se décliner en de nombreux algorithmes MCMC pour un même problème pourvu que chaque itération conserve la loi stationnaire et qu'un nombre fini d'itérations permette d'atteindre n'importe quel point de l'espace à partir de n'importe quel autre.

Il s'agit cependant de propriétés théoriques. Elles n'excluent pas que la convergence ne puisse être obtenue qu'après un temps très long et même « infini » d'un point de vue pratique. Notamment, il n'est pas rare que

[6]Des résultats de convergence plus fins ont été démontrés [RCD93], ils assurent la convergence pour tout point de départ et à vitesse géométrique.

l'algorithme semble avoir convergé mais n'échantillonne qu'un mode local de la loi *a posteriori*. La convergence rapide vers la bonne loi dépend du choix du point de départ mais aussi de l'ampleur des sauts que l'algorithme MCMC permet de réaliser lors des mises à jours. C'est pourquoi on a préféré ici l'algorithme détaillé ci-dessus à un algorithme quelque peu différent qui consiste à mettre à jour s_t position par position (voir par exemple [RCD93]) selon la loi conditionnelle $s_t \mid s_1^t, s_{t+1}^n, \theta, x_1^n$. Ces mises à jour sont moins coûteuses que la simulation de s_1^n selon $s_1^n \mid \theta, x_1^n$ mais sont aussi beaucoup plus conservatrices. Pour certaines architectures de modèles, cet algorithme MCMC ne permet tout simplement pas d'atteindre tout l'espace à partir d'un seul point de départ.

Les itérations $1, \ldots, m_0$ ne sont généralement pas prises en compte car elles sont souvent très dépendantes du point de départ et ne correspondent donc pas au comportement stationnaire de la chaîne de Markov. Le choix de m_0 et M devrait reposer sur un contrôle de la convergence. Une inspection visuelle de l'évolution des quantités échantillonnées au cours des itérations permet souvent un premier niveau de contrôle. Des solutions techniques pour contrôler la convergence existent aussi mais elles ne seront pas abordées dans cette thèse (voir par exemple [Rob95, WRGS96, Bro98, Rob98, MCC98]).

À partir de l'échantillon généré l'estimateur bayésien peut être approché par

$$\begin{aligned} a^{\text{Bayes}}(u,v) &\approx \frac{1}{M} \sum_{m=m_0+1}^{m_0+M} a^{(m)}(u,v) \\ b_u^{\text{Bayes}}(x;w) &\approx \frac{1}{M} \sum_{m=m_0+1}^{m_0+M} b_u^{(m)}(x;w). \end{aligned}$$

On peut aussi calculer, par exemple, la probabilité que l'état caché u soit associé à une position t de la séquence en prenant en compte l'incertitude sur l'estimation des paramètres :

$$\begin{aligned} P(S_t = u \mid x_1^n) &= \int_{\theta, s_1^n} \mathbb{I}\{s_t = u\} \pi(\theta, s_1^n \mid x_1^n) \, d\theta \, ds_1^n \\ &\approx \frac{1}{M} \sum_{m=m_0+1}^{m_0+M} \mathbb{I}\{s_t^{(m)} = u\} \ , \end{aligned}$$

ou grâce à une approximation plus précise dite « Rao-Blackwellisation » [GS90, LWK94] :

$$P(S_t = u \mid x_1^n) \approx \frac{1}{M} \sum_{i=m_0+1}^{m_0+M} P(S_t = u \mid \theta^{(m)}, x_1^n) \ .$$

Pour en finir momentanément avec les MCMC, notons que pour certains problèmes il est impossible de simuler une des composantes directement selon

sa loi conditionnellement à toutes les autres composantes. Des mises à jour dites « à la Metropolis-Hastings », fondées sur une procédure d'acceptation-rejet, peuvent être utilisées (voir page 77). D'un point de vue général, les méthodes MCMC jouent aujourd'hui un grand rôle dans l'essor des approches bayésiennes. Le développement de ces méthodes souvent très calculatoires a été favorisé par la disponibilité d'ordinateurs de plus en plus puissants.

Remarques sur l'estimation des paramètres sur les données incomplètes. L'estimation bayésienne par méthodes MCMC résout partiellement le problème de convergence vers des maxima locaux de l'algorithme EM. Cependant il ne s'agit pas d'un avantage réellement lié à l'approche bayésienne. En effet, les méthodes MCMC peuvent aussi s'appliquer au cadre de l'estimation par maximum de vraisemblance (voir par exemple [Mur97]). Une autre solution est d'introduire du « bruit » dans les premières étapes de l'algorithme EM (voir par exemple [KBM+94]).

Comme pour l'estimation sur les données complètes, l'intérêt pratique de l'approche bayésienne réside plutôt dans le « bon » comportement des estimateurs obtenus lorsque le nombre d'observations disponibles pour l'estimation est relativement faible. D'un point de vue plus général, l'étude de la loi *a posteriori* permet de quantifier et de tenir compte de l'incertitude existante sur l'estimation des paramètres. L'intérêt en est bien sûr plus grand lorsque le jeu de données est petit puisque l'incertitude est alors plus grande. Cependant, le post-traitement des informations générées par les algorithmes MCMC peut s'avérer lourd. Si l'objectif est uniquement d'introduire de l'information *a priori* dans l'estimation pour améliorer son comportement sur un jeu de données de taille réduite, le MAP peut être obtenu avec une procédure analogue à l'algorithme EM (voir par exemple [GL94]).

2.5 Sélection de modèle

Les algorithmes de choix du nombre d'états, des transitions autorisées et des lois d'émission des observations d'un HMM relèvent du domaine de la sélection de modèle. Les problèmes de choix de modèles sont classiquement distingués de ceux posés par l'estimation des paramètres dans lequel l'espace des paramètres est fixée (en particulier le nombre de paramètres).

Cette section débute par une présentation informelle de quelques aspects de la sélection de modèle dont l'objectif est principalement de positionner l'approche bayésienne présentée dans la section 2.5.2. La section 2.5.3 explique le principe des méthodes *Reversible Jump* MCMC qui seront utilisées dans cette thèse pour la mise en œuvre de cette approche bayésienne. Enfin, la section 2.5.4 présente l'utilisation des *Reversible Jump* MCMC pour le choix du nombre d'états cachés et l'ordre markovien d'émission des observations dans un HMM.

2.5.1 Le délicat problème de la sélection de modèle

Le principe de parcimonie

La difficulté de la sélection de modèle réside essentiellement dans le choix d'un critère d'évaluation des différents modèles. Ce choix doit bien sûr dépendre de l'application envisagée pour le modèle choisi. On remarque à ce propos que lors de la construction « à la main » d'un HMM, une partie du choix du modèle est dictée par l'utilisation que l'on compte en faire. Ici, le problème du choix d'un modèle va être abordé sous un angle technique qui consiste à chercher, pour un jeu d'observations D, le « bon modèle » $i \in \mathcal{M}$ où $\mathcal{M}$ est un ensemble fini de modèles (tous censés être potentiellement adaptés à l'utilisation envisagée) dont les paramètres sont à valeurs dans Θ_i. Une fois le problème posé en ces termes, la question du choix du critère reste presque entière. La principale source de difficultés réside dans la volonté de comparer des modèles qui permettent de décrire une « réalité » plus ou moins complexe au travers de paramètres dont le nombre diffère d'un modèle à l'autre.

Le principe qui guide toute sélection de modèle raisonnable est qu'une bonne description (« bon modèle ») de la réalité est une description à la fois fidèle et simple. Ce « principe de parcimonie » privilégie aux descriptions simples sur les descriptions complexes. Sans avoir besoin d'invoquer des considérations esthétiques ou épistémologiques, il est d'une grande importance pratique car la complexité inutile d'un modèle nuit à la pertinence de son utilisation. En effet, un modèle trop complexe peut être ajusté trop précisément aux données observées, le modèle ainsi ajusté décrit alors d'autres observations moins bien qu'un « bon modèle ». En langage plus statistique, la variance des estimateurs des paramètres est plus élevée que celle des paramètres d'un modèle plus simple : l'estimation des paramètres est trop dépendante d'un jeu d'observations particulier.

Quelques approches pour la sélection de modèle

La validation croisée et le critère AIC. Le critère de sélection peut se fonder directement sur la performance du modèle au regard de la tâche pour laquelle on souhaite l'utiliser. La sélection de modèle est alors « supervisée » : son objectif est défini comme la reproduction de résultats connus sur un jeu de données d'apprentissage. Il est dans ce cas assez simple de construire un critère d'évaluation, mais il est aussi crucial d'utiliser un jeu de données d'évaluation distinct de celui du jeu de données d'apprentissage. Un modèle trop complexe aura, en effet, tendance à être sur-ajusté aux données d'apprentissage (« à les apprendre par cœur »). Cette procédure est souvent qualifiée de validation croisée (*Cross validation*). Un de ses inconvénients est de diminuer la quantité de données effectivement disponible pour l'estimation des paramètres puisque des données doivent être conservées pour

l'évaluation. Une procédure classique, mais coûteuse en calcul, pour contourner cet inconvénient consiste à réaliser de nombreuses validations croisées en ne conservant à chaque fois qu'un petit jeu de données pour l'évaluation (éventuellement une seule).

Dans cette thèse le problème du choix automatique d'un modèle est abordé sous l'angle d'une sélection non supervisée. Le bon modèle est alors défini comme celui qui s'approche le plus de la réalité, dans la limite de la quantité de données disponibles pour son estimation. La validation croisée peut aussi être utilisée dans ce cadre, il suffit pour cela de choisir un critère de sélection comme la vraisemblance qui reflète la façon dont le modèle s'ajuste aux données d'évaluation (voir par exemple [vdLDK03]). On peut noter que le critère AIC (*Akaike Information Criterion*), souvent utilisé pour comparer des modèles, peut être interprété comme une approximation de l'espérance du résultat d'une telle validation croisée [Aka74, Sto77, BA98]. En effet, le critère AIC associé au modèle i défini par

$$\mathrm{AIC}_i \quad = \quad -2 \log \max_{\theta \in \Theta_i} P_\theta(D) + 2dim(\Theta_i) \ ,$$

est obtenu comme une approximation de

$$-2E_{D_1} E_{D_2} \log P_{\theta^{\mathrm{ML}}(D_1)}(D_2) \ ,$$

où D_1 et D_2 sont des jeux de données provenant de la réalité, et $\theta^{\mathrm{ML}}(D_1) = \arg\max_{\theta \in \Theta_i} P_\theta(D_1)$ est l'estimateur du maximum de vraisemblance obtenu pour le modèle considéré sur D_1. Les variables D_1 et D_2 peuvent donc être vues respectivement comme les données d'apprentissage et d'évaluation. Lorsque le vrai modèle est contenu dans $\mathcal{M}$ et que le jeu de données d'apprentissage devient assez grand, le modèle maximisant le critère AIC_i est souvent un peu plus grand que le vrai modèle. En d'autres termes le critère AIC, malgré sa justification essentiellement asymptotique, ne permet pas de construire un estimateur consistant du modèle.

Le test du rapport de vraisemblance. Le test du rapport de vraisemblance est classiquement utilisé pour l'étude de certaines familles de modèles emboîtés. Il permet de tester l'hypothèse nulle « les observations ont été générées par un modèle i » contre l'alternative « les observations ont été générées par un modèle j », où le modèle i est inclus dans le modèle j. Ce test se fonde sur la convergence, sous l'hypothèse nulle (le petit modèle i est le vrai), de la loi du logarithme du rapport $\max_{\theta \in \Theta_j} P_\theta(D)$ sur $\max_{\theta \in \Theta_i} P_\theta(D)$ vers une loi du χ^2 dont le nombre de degrés de liberté est $dim(\Theta_j) - dim(\Theta_i)$. Ce comportement asymptotique simple n'est valable que sous certaines hypothèses dont l'existence d'une écriture unique du modèle i comme une version contrainte du modèle j. Cette hypothèse est par exemple mise en défaut dans le cadre de l'étude du nombre d'états cachés d'un HMM : un modèle à

i états peut s'écrire dans un modèle à $i+1$ états soit comme la contrainte « la probabilité d'entrer dans un des états est nulle » soit comme la contrainte « deux états ont la même loi d'émission des observations » (voir par exemple [RTA98, GK00]).

Outre cette hypothèse restrictive nécessaire pour obtenir facilement le niveau du test, la construction d'un test entre deux hypothèses aussi simples n'est souvent pas une approche satisfaisante pour trouver un modèle parmi un ensemble de modèles candidats. Chaque test ne « compare » que deux modèles qui sont de plus traités de façon asymétrique. Il faut alors choisir un modèle à partir du résultat de nombreux tests non indépendants, de puissances différentes et potentiellement contradictoires : il n'existe souvent pas de méthode réellement satisfaisante pour choisir un modèle à partir des résultats.

La sélection de modèle vue comme un problème d'estimation. Choisir un modèle dans un ensemble peut être vu comme un problème d'estimation. Le paramètre à estimer est alors un indicateur du modèle supposé avoir généré les observations. Dans ce cadre, la construction d'estimateurs par pénalisation du maximum de la vraisemblance (comme le critère AIC) est très étudiée. Le problème est essentiellement abordé comme celui du choix de la pénalisation qui assure un bon comportement asymptotique de l'estimateur lorsque le vrai modèle appartient à l'ensemble des modèles considérés. La consistance des estimateurs est particulièrement recherchée : l'estimateur doit permettre de trouver le vrai modèle lorsqu'il appartient à l'ensemble et que le jeu de données devient suffisamment grand. Dans le cadre des modèles qui nous intéressent ici, on peut notamment citer les travaux sur l'estimation de l'ordre d'un modèle de Markov (voir par ex. [Fin90, CS00]) et du nombre d'états cachés d'un HMM (voir par ex. [Fin90, LP92, GB03]).

L'importance pratique du bon comportement asymptotique d'un estimateur du modèle sous l'hypothèse de l'existence du vrai modèle dans l'ensemble des modèles étudiés est difficile à appréhender. La consistance relève en effet d'une vision abstraite du problème de la sélection de modèle. On peut rapporter à ce propos une phrase de Burnham et Anderson (1998) [BA98] pour défendre l'utilisation du critère AIC : « *If an investigator knew that a true model existed and that it was in the set of candidate models, would not he know which one it was ?* ». Il faut bien reconnaître qu'en pratique la réalité n'appartient pas à l'ensemble des modèles considérés et les conditions ne sont pas réellement asymptotiques. Cependant, il semble clair que la consistance est plutôt désirable, pourvu que ce ne soit pas au prix d'un mauvais comportement en conditions non asymptotiques.

L'approche adoptée ici pour la sélection de modèle est bayésienne. Un des intérêts du cadre bayésien est de permettre la construction presque automatique d'un critère qui a une justification simultanément non asymptotique et

asymptotique. Cette approche, qui ne nécessite pas d'hypothèses sur l'emboîtement des modèles, présente aussi d'autres intérêts. On citera en particulier la possibilité d'une quantification de l'incertitude sur le choix du modèle et d'une utilisation des modèles en tenant compte de cette incertitude. À l'intérieur du cadre bayésien, ce travail a plus précisément porté sur l'utilisation de la méthode dite *Reversible Jump* MCMC qui autorise au moins dans certains cas l'étude de grandes familles de modèles. On peut enfin noter que le travail présenté dans le chapitre 5 sur la sélection de modèle dans le cadre bayésien, par méthodes *Reversible Jump* MCMC, prolonge de manière naturelle l'utilisation d'algorithmes MCMC pour la recherche de motifs où ils se sont déjà révélés performants.

2.5.2 L'approche bayésienne

Le « critère » bayésien

Supposons que l'on souhaite choisir un modèle dans un ensemble de modèles $\mathcal{M}$. Pour chaque modèle i on a déjà accepté l'idée de mettre une loi *a priori* $\pi_i(\theta)$ sur Θ_i l'espace des paramètres de chacun de ces modèles, où Θ_i est l'espace des paramètres associé au modèle i. Il semble naturel, dans le cadre bayésien, d'ajouter une loi *a priori* sur l'ensemble des modèles $i \in \mathcal{M}$: $\pi(i) = P(M = i)$. La formule de Bayes permet alors d'obtenir la probabilité *a posteriori* de chaque modèle conditionnellement aux observations D :

$$\begin{aligned} P(M = i \mid D) &= \frac{P(D \mid M = i)P(M = i)}{\sum_j P(D \mid M = j)P(M = j)} \\ &\propto P(D \mid M = i)P(M = i) \ . \end{aligned}$$

Dans cette formule, le terme $P(M = i)$ correspond à la probabilité *a priori* du modèle i et représente donc la contribution de l'*a priori* sur le choix du modèle i. Le terme $P(D \mid M = i)$ reflète quant à lui l'information apportée par l'observation D et correspond ainsi à la contribution de l'observation sur le choix du modèle i. Le terme $P(D \mid M = i)$ parfois appelé vraisemblance marginale joue donc un rôle fondamental. C'est une intégrale sur Θ_i :

$$P(D \mid M = i) = \int_{\theta \in \Theta_i} P_\theta(D)\pi_i(\theta)\, d\theta \ .$$

Contrairement à $\max_{\theta \in \Theta_i} P_\theta(D)$ (le maximum de la vraisemblance dans le modèle i), la vraisemblance marginale $P(D \mid M = i)$ n'a pas une tendance naturelle à augmenter lorsque la dimension de Θ_i augmente. En effet, Θ_i a bien tendance à contenir des valeurs de θ qui donnent une plus forte probabilité d'apparition aux données, mais ces valeurs sont aussi plongées dans un espace plus grand, elles sont donc moins chargées par $\pi_i(\theta)$. La vraisemblance marginale $P(D \mid M = i)$ est plus où moins grande en fonction de la

façon dont le modèle i est adapté aux observations. Le rapport entre la vraisemblance marginale du « vrai » modèle, si il appartient à $\mathcal{M}$, et celle des autres modèles croît avec la quantité d'observations. Au contraire, le terme $P(M = i)$ a une influence qui ne dépend pas de la quantité d'observations. Il est important de comprendre qu'au moins quand D est grand et $\mathcal{M}$ de dimension finie, ce n'est pas la loi *a priori* sur les modèles qui incarne le principe de parcimonie : c'est la dimension elle-même des modèles qui à travers la vraisemblance marginale pénalise les grands modèles.

L'approche bayésienne permet de construire facilement un estimateur $\hat{i} = \arg\max_{i \in \mathcal{M}} P(M = i \mid D)$. De plus la probabilité $P(M = i \mid D)$ a une interprétation directe en terme d'incertitude sur le choix d'un modèle lorsque les conditions ne sont pas asymptotiques.

La vraisemblance marginale $P(D \mid M = i)$ peut être vue comme le « critère » sur lequel se fonde la sélection bayésienne de modèle puisque c'est elle qui prend en compte l'information apportée par les observations. En particulier, elle est directement proportionnelle à $P(M = i \mid D)$ lorsque la loi *a priori* sur les modèles $\pi(i)$ est uniforme. Le rapport de deux vraisemblances marginales permet la comparaison de deux modèles, il est connu sous le nom de facteur de Bayes (voir en particulier [KR95]). Avant d'aborder le problème du calcul de $P(D \mid M = i)$ et de $P(M = i \mid D)$, on peut signaler l'existence d'approches bayésiennes alternatives pour aborder le problème de la sélection de modèle (voir par exemple [Gel96]).

Calcul analytique de la vraisemblance marginale

Le calcul direct de la vraisemblance marginale est rarement possible dans les cas intéressants. En pratique, il nécessite un modèle simple (par exemple sans variables cachées) et une loi *a priori* conjuguée. Un de ces cas concerne le choix de l'ordre d'un modèle de chaîne de Markov [FT99], on va le décrire brièvement.

Comme on l'a déjà évoqué à propos du choix de la loi *a priori* des paramètres b_u des modèles d'émission des observations conditionnellement aux états cachés, la loi *a priori* conjuguée d'un modèle de chaîne de Markov d'ordre r dans $\mathcal{X}$ est constituée de lois de Dirichlet indépendantes pour chaque contexte $w \in \mathcal{X}^r$. On se place ici dans un modèle de Markov (non caché) dont on notera b la matrice de transitions[7], la loi initiale étant fixée et uniforme. La densité de la loi *a priori* est

$$\pi_r(b) \quad = \quad \prod_{w \in \mathcal{X}^r} \left(\frac{\Gamma(\sum_{x \in \mathcal{X}} \beta(x;w))}{\prod_{x \in \mathcal{X}} \Gamma(\beta(x;w))} \prod_{x \in \mathcal{X}} b(x;w)^{\beta(x;w)-1} \right) \mathbb{I}_{\{b \in \Theta_r\}} \ ,$$

où les $\beta(x;w)$ sont les paramètres des lois *a priori* (de Dirichlet). La vrai-

[7] à ne pas confondre avec le b utilisé pour les HMM qui désigne l'ensemble des paramètres des lois d'émission du HMM

semblance des observations $P_b(x_1^n)$ s'écrit

$$P_b(x_1^n) \quad = \quad \frac{1}{|\mathcal{X}|^r} \prod_{w \in \mathcal{X}^r} \prod_{x \in \mathcal{X}} b(x;w)^{N_{wx}} \ , \quad b \in \Theta_r \ ,$$

où le terme $1/|\mathcal{X}|^r$ correspond à la probabilité d'apparition des r premiers symboles de la séquence x_1^n sous la loi uniforme. La vraisemblance marginale est alors

$$\begin{aligned} P(x_1^n \mid M = r) &= \int_{b \in \Theta_r} P_b(x_1^n) \pi_r(b) db \\ &= \prod_{w \in \mathcal{X}^r} \left(\frac{\Gamma(\sum_{x \in \mathcal{X}} \beta(x;w))}{\prod_{x \in \mathcal{X}} \Gamma(\beta(x;w))} \right) \frac{1}{|\mathcal{X}|^r} \int_{b \in \Theta_r} \prod_{w \in \mathcal{X}^r} \prod_{x \in \mathcal{X}} b(x;w)^{N_{wx} + \beta(x;w) - 1} db \\ &= \prod_{w \in \mathcal{X}^r} \left(\frac{\Gamma(\sum_{x \in \mathcal{X}} \beta(x;w))}{\prod_{x \in \mathcal{X}} \Gamma(\beta(x;w))} \right) \frac{1}{|\mathcal{X}|^r} \prod_{w \in \mathcal{X}^r} \left(\frac{\prod_{x \in \mathcal{X}} \Gamma(\beta(x;w) + N_{wx})}{\Gamma(\sum_{x \in \mathcal{X}} \beta(x;w) + N_{wx})} \right) . \end{aligned} \tag{2.19}$$

On peut donc obtenir analytiquement $P(M = r \mid x_1^n)$ en se donnant une loi *a priori* $\pi(r) = P(M = r)$ sur les différents modèles. La même approche permettrait de choisir un modèle de chaînes de Markov à ordre variable (qui dépend de w), et même éventuellement un modèle de contextes « à trous » (arbre de contextes). Dans ce dernier cas, étant donnés le nombre de modèles possibles et la structure plus complexe de l'emboîtement des différents modèles, il serait probablement intéressant d'intégrer le calcul de la vraisemblance marginale dans une méthode d'échantillonnage de $P(M = i \mid x_1^n)$.

Csiszar et Shields (2000) [CS00] montrent un résultat spectaculaire concernant l'estimation de l'ordre d'un modèle markovien par cette approche. Ils étudient le comportement de l'estimateur bayésien $\hat{r} = \arg\max_{r \in \mathcal{M}} P(M = r \mid x_1^n)$ avec une loi *a priori* sur les paramètres qui correspond à $\beta = 1/2$ et une loi *a priori* sur les modèles de la forme $P(M = r) \propto r^{-2} \mathbb{I}_{\{r \geq 0\}}$ (l'ordre n'est pas borné). Sous cette loi *a priori*, ils montrent que l'ordre estimé tend vers $+\infty$ lorsque le vrai modèle est d'ordre $r = 0$ et que les b_0 sont égaux à $1/|\mathcal{X}|$ (loi uniforme sur $\mathcal{X}$). Csiszar et Shields remarquent que ce comportement disparaît en ajoutant ce modèle « d'ordre 00 » dans $\mathcal{M}$, c'est à dire lorsque le vrai modèle n'est plus négligeable sous la loi *a priori*.

Calcul approché de la vraisemblance marginale

Le développement de méthodes pour le calcul approché de la vraisemblance marginale par MCMC a fait l'objet de nombreux travaux (voir par exemple [KR95, Chi95, Raf96]). La présentation de deux exemples ci-dessous est uniquement destinée à donner une première idée de ces approches.

Au premier abord il semble simple d'approcher $P(D \mid M = i)$ par une méthode de Monte-Carlo. Il suffit de simuler $\theta^{(0)}, \theta^{(1)}, \ldots, \theta^{(M)}$ sous la loi *a priori* puis d'utiliser l'approximation

$$\begin{aligned} P(D \mid M = i) &= \int_{\theta \in \Theta_i} P_\theta(D) \pi_i(\theta) d\theta \\ &\approx \frac{1}{M} \sum_{m=1}^{M} P_{\theta^{(m)}}(D) \ . \end{aligned}$$

Cependant, la convergence de cette approximation est très lente car la plupart des $\theta^{(m)}$ sont associés à une vraisemblance $P_{\theta^{(m)}}(D)$ très faible (d'autant plus que l'espace est de dimension élevée). L'approximation est donc sensible aux quelques valeurs associées à une vraisemblance élevée.

Pour obtenir une approximation de $P(D \mid M = i)$ un peu plus stable en pratique, une idée consiste à mieux explorer les régions de l'espace où les θ sont associés à une vraisemblance élevée en utilisant un échantillon $\theta^{(0)}, \theta^{(1)}, \ldots, \theta^{(M)}$ de la loi *a posteriori*. Comme on l'a déjà évoqué, les algorithmes MCMC permettent de simuler de tels échantillons. La vraisemblance marginale peut alors être approchée grâce à

$$P(D \mid M = i) \approx \left(\frac{1}{M} \sum_{m=1}^{M} \frac{1}{P_{\theta^{(m)}}(D)} \right)^{-1} ,$$

puisque

$$\begin{aligned} \frac{1}{P(D \mid M = i)} &= \int_{\theta \in \Theta_i} \frac{1}{P(D \mid M = i)} \pi_i(\theta) d\theta \\ &= \int_{\theta \in \Theta_i} \frac{1}{P_\theta(D)} \pi(\theta \mid D, M = i) d\theta \ . \end{aligned}$$

Notons enfin que le critère BIC (*Bayesian Information Criterion*), défini par

$$\mathrm{BIC}_i = -2 \log \max_{\theta \in \Theta_i} P_\theta(D) + dim(\Theta_i) \times \log n \ ,$$

où n est la taille de l'échantillon D, est lié à la vraisemblance marginale [Sch78, KW95]. En effet, $\mathrm{BIC}_j - \mathrm{BIC}_i$ est une approximation asymptotique de $-2 \times \big(\log P(D \mid M = j) - \log P(D \mid M = i) \big)$. La qualité de cette approximation dépend évidemment de la quantité d'observations et des modèles considérés mais elle ne nécessite pas de simulations (ni même de choix d'une loi *a priori* sur les paramètres).

En dehors des difficultés de calcul, la principale limite de l'approche visant à calculer la vraisemblance marginale est qu'elle nécessite son évaluation

pour de nombreux modèles lorsque $\mathcal{M}$ est grand. En revanche, la simulation de la loi *a posteriori* sur les modèles par des méthodes MCMC permet d'approcher directement $P(M = i \mid D)$ en s'affranchissant en partie de ce problème. En effet, l'algorithme se concentre (si il est efficace) sur l'échantillonnage des modèles associés à une probabilité *a posteriori* élevée.

2.5.3 Échantillonnage par *Reversible Jump* MCMC

La méthode *Reversible Jump* MCMC [Gre95, RG97] pour étudier la loi *a posteriori* sur les différents modèles $P(M = i \mid D)$, consiste à échantillonner la loi *a posteriori* jointe sur les modèles et leurs paramètres $\pi(i, \theta \mid D)$ avec $(i, \theta) \in \bigcup_{i \in \mathcal{M}} (\{i\} \times \Theta_i)$ dont $P(M = i \mid D)$ est une marginale.

Les algorithmes *Reversible Jump* MCMC se distinguent des autres approches proposées pour cette tâche essentiellement parce qu'ils sont une généralisation directe de l'algorithme de Metropolis-Hastings permettant les sauts entre les différents espaces Θ_i. Parmi les approches alternatives, on peut citer, d'une part, celle de Carlin et Chib (1995) [CC95] que l'on peut pratiquement voir comme un cas particulier de *Reversible Jump* MCMC, d'autre part, les approches d'échantillonnage en temps continu (voir par exemple [PS96, Ste00]) dont le comportement est très proche des *Reversible Jump* MCMC [CRR03].

Avant d'aborder les *Reversible Jump* MCMC, cette section débute logiquement par une présentation de l'algorithme de Metropolis-Hastings dans sa version « classique ».

L'algorithme de Metropolis-Hastings

En dimension fixée, l'algorithme de Metropolis-Hastings [Has70, Tie94] permet d'échantillonner une loi de densité $\pi(y)$ sur $\mathcal{Y}$ en construisant une chaîne de Markov dont la loi stationnaire est la loi d'intérêt π. Une itération de l'algorithme se décompose en

- étant donné $y^{(m-1)} = y$ simuler $\tilde{y}$ selon une loi instrumentale de densité $q_y(\tilde{y})$
- prendre $y^{(m)} = \tilde{y}$ avec la probabilité $\alpha(y, \tilde{y})$ sinon prendre $y^{(m)} = y$ (on a alors $y^{(m)} = y^{(m-1)}$). La probabilité d'acceptation $\alpha(y, \tilde{y})$ est donnée par

$$\alpha(y, \tilde{y}) \quad = \quad \min \left\{ 1; \frac{\pi(\tilde{y}) q_{\tilde{y}}(y)}{\pi(y) q_y(\tilde{y})} \right\} .$$

Sous des hypothèses très faibles sur la loi instrumentale[8], la chaîne de Markov ainsi construite admet la loi d'intérêt π comme loi stationnaire. L'efficacité de

[8] Il faut essentiellement que l'algorithme puisse atteindre tout le support de la loi d'intérêt.

l'algorithme dépend cependant beaucoup de l'adéquation entre la loi instrumentale et les caractéristiques de la loi à simuler. On remarque en particulier le rôle que joue $\pi(\tilde{y})/\pi(y)$ dans la probabilité d'acceptation des propositions $\tilde{y}$ de la loi instrumentale.

Un intérêt majeur de l'algorithme de Metropolis-Hastings est de ne nécessiter la connaissance de la densité de la loi à simuler qu'à une constante près. En effet, elle n'apparaît dans la probabilité d'acceptation qu'à travers le rapport $\pi(\tilde{y})/\pi(y)$. L'algorithme est donc particulièrement bien adapté à l'échantillonnage des lois *a posteriori* dont la densité est souvent facile à évaluer à une constante près puisque : $\pi(\theta \mid D) \propto \pi(\theta)P_\theta(D)$, où $\pi(\theta)$ est la densité de la loi *a priori* et $P_\theta(D)$ la vraisemblance.

Montrons que l'algorithme de Metropolis-Hastings préserve la loi d'intérêt. Le choix de α assure en effet la réversibilité du passage de tout ensemble A à tout ensemble B sous la loi d'intérêt. Il s'agit d'une condition suffisante connue sous le nom de « *detailed balance* », elle est vérifiée si

$$\int_{(y,\tilde{y})\in A\times B} \pi(y)p_y(\tilde{y})dyd\tilde{y} \;=\; \int_{(y,\tilde{y})\in A\times B} \pi(\tilde{y})p_{\tilde{y}}(y)dyd\tilde{y}\ , \qquad (2.20)$$

où $p_y(\tilde{y})$ est la densité du noyau de transition de la chaîne de Markov créée par l'algorithme. Le terme de gauche s'exprime

$$\begin{aligned}\int_{(y,\tilde{y})\in A\times B} \pi(y)p_y(\tilde{y})dyd\tilde{y} \;&=\; \int_{(y,\tilde{y})\in A\times B} \pi(y)q_y(\tilde{y})\alpha(y,\tilde{y})dyd\tilde{y} \\ &\quad + \underbrace{\int_{y\in A\cap B} \pi(y)r(y)dy}\ ,\end{aligned}$$

où le terme indiqué par une accolade correspond à la contribution des cas où la proposition est rejetée, $r(y)$ étant la probabilité de rester en y (rejet de la proposition) :

$$r(y) \;=\; \int_{\tilde{y}\in y} (1-\alpha(y,\tilde{y}))d\tilde{y}\ .$$

Ce terme est identique dans le membre de gauche et le membre de droite de la *detailed balance* (Eq. 2.20) qui devient donc :

$$\int_{(y,\tilde{y})\in A\times B} \pi(y)q_y(\tilde{y})\alpha(y,\tilde{y})dyd\tilde{y} = \int_{(y,\tilde{y})\in A\times B} \pi(\tilde{y})q_{\tilde{y}}(y)\alpha(\tilde{y},y)dyd\tilde{y}\ . \qquad (2.21)$$

On voit facilement que le choix de $\alpha(y,\tilde{y})$ (Eq. 2.20) assure cette égalité puisque :

$$\pi(y)q_y(\tilde{y})\alpha(y,\tilde{y}) \;=\; \pi(\tilde{y})q_{\tilde{y}}(y)\alpha(\tilde{y},y)\ .$$

L'échantillonnage d'une loi a *posteriori* dans un modèle de dimension élevée repose rarement sur un algorithme MCMC constitué d'une unique étape correspondant à l'algorithme de Metropolis-Hastings présenté ci-dessus et permettant à elle seule de visiter tout Θ. Comme l'algorithme MCMC constitué de deux étapes « à la Gibbs » présenté page 67, les algorithmes utilisés en pratique contiennent généralement plusieurs étapes. Chaque étape ne met à jour qu'une partie des paramètres en préservant leur loi conditionnelle aux autres paramètres. Dans un tel algorithme, des étapes « à la Metropolis-Hastings » et des étapes « à la Gibbs » peuvent coexister. Une étape « à la Gibbs » consiste à tirer directement selon la loi conditionnelle, elle peut être interprétée comme un cas particulier d'étape « à la Metropolis-Hastings » dont la probabilité d'acceptation vaut 1 (la loi « instrumentale » étant la loi conditionnelle).

L'algorithme *Reversible Jump* MCMC

L'algorithme *Reversible Jump* MCMC proposé par Green (1995) [Gre95] généralise l'algorithme de Metropolis-Hastings en permettant l'échantillonnage d'une loi $\pi(i, y)$ sur un espace $\bigcup_{i\in\mathcal{I}}(\{i\} \times \mathcal{Y}_i)$ où les $\mathcal{Y}_i$ peuvent être de dimensions différentes. On supposera dans un premier temps que les $\mathcal{Y}_i$ sont des espaces continus ($\subset \mathbb{R}^{dim(\mathcal{Y}_i)}$). L'échantillonnage est rendu possible grâce à l'introduction, dans l'algorithme MCMC, d'étapes qui permettent des sauts entre les différents espaces $\mathcal{Y}_i$ (en général de dimensions voisines). Les sauts sont réalisés grâce à des paires de mouvements $(m_{i\to j}, m_{j\to i})$ réversibles, c'est à dire vérifiant la « detailed balance » sous la loi d'intérêt. On va maintenant présenter la façon dont ces paires de mouvements réversibles, qui constituent les étapes « à la *Reversible Jump* Metropolis-Hastings », sont construites.

Les mouvements $m_{i\to j}$ et $m_{j\to i}$ sont respectivement associés au passage de $\mathcal{Y}_i$ à $\mathcal{Y}_j$ et de $\mathcal{Y}_j$ à $\mathcal{Y}_i$. Sans perte de généralité, supposons $dim(\mathcal{Y}_j) \geq dim(\mathcal{Y}_i)$.

Pour permettre le passage de $\mathcal{Y}_i$ à $\mathcal{Y}_j$ associé au mouvement $m_{i\to j}$, Green propose de commencer par tirer un vecteur ω de dimension telle que $dim(\omega) \geq dim(\mathcal{Y}_j) - dim(\mathcal{Y}_i)$ selon une loi instrumentale de densité $g_{y,m_{i\to j}}(\omega)$. Le point $\tilde{y}$ proposé dans l'espace $\mathcal{Y}_j$ est ensuite choisi de façon déterministe : $\tilde{y} = h_{m_{i\to j}}(y, \omega)$.

De même, le mouvement inverse $m_{j\to i}$ de $\tilde{y} \in \mathcal{Y}_j$ à $y \in \mathcal{Y}_i$ consiste à tirer un vecteur $\tilde{\omega}$ tel que $dim(y) + dim(\omega) = dim(\tilde{y}) + dim(\tilde{\omega})$ puis à proposer $y = h_{m_{j\to i}}(\tilde{y}, \tilde{\omega})$.

La condition $dim(y) + dim(\omega) = dim(\tilde{y}) + dim(\tilde{\omega})$ est nécessaire pour résoudre les problèmes posés par le changement de dimension. Il faut de plus qu'il existe une bijection différentiable f entre (y, ω) et $(\tilde{y}, \tilde{\omega})$: $(\tilde{y}, \tilde{\omega}) = f(y, \omega)$.

Cette procédure de proposition des sauts permet de calculer le taux d'ac-

ceptation qui assure l'équilibre de la *detailed balance*. Il suffit de reprendre l'équation 2.21 avec, maintenant, des intégrales de dimension $dim(\theta)+dim(\omega) = dim(\tilde{y}) + dim(\tilde{\omega})$ et A et B respectivement deux sous ensembles mesurables de $\mathcal{Y}_i$ et de $\mathcal{Y}_j$:

$$\begin{aligned}&\int_{(y,\tilde{y})\in A\times B} \pi(i,y) g_{i,y}(m_{i\to j}) g_{y,m_{i\to j}}(\omega)\alpha(y,\tilde{y})dyd\omega \\ &= \int_{(y,\tilde{y})\in A\times B} \pi(j,\tilde{y}) g_{j,\tilde{y}}(m_{j\to i}) g_{\tilde{y},m_{j\to i}}(\tilde{\omega})\alpha(\tilde{y},y)d\tilde{y}d\tilde{\omega}\ ,\end{aligned}$$

où $g_{i,y}(m_{i\to j})g_{y,m_{i\to j}}(\omega)$ est la densité de la loi instrumentale qui comprend le choix du mouvement $m_{i\to j}$ et du vecteur ω. La bijection f autorise le changement de variable (par exemple dans le terme de droite) :

$$\begin{aligned}&\int_{(y,\tilde{y})\in A\times B} \pi(i,y) g_{i,y}(m_{i\to j}) g_{y,m_{i\to j}}(\omega)\alpha(y,\tilde{y})dyd\omega \\ &= \int_{(y,\tilde{y})\in A\times B} \pi(j,\tilde{y}) g_{j,\tilde{y}}(m_{j\to i}) g_{\tilde{y},m_{j\to i}}(\tilde{\omega})\alpha(\tilde{y},y)\left|\frac{\partial f(y,\omega)}{\partial y\partial\omega}\right| dyd\omega\ ,\end{aligned}$$

où le terme

$$\left|\frac{\partial f(y,\omega)}{\partial y\partial\omega}\right|$$

est le jacobien associé au changement de variables. Il est alors aisé de déterminer le taux d'acceptation qui équilibre la *detailed balance* :

$$\left\{\begin{array}{rcl}\alpha(y,\tilde{y}) &=& min(1;A^+)\\ \alpha(\tilde{y},y) &=& min(1;1/A^+)\ ,\end{array}\right.$$

où

$$A^+ = \frac{\pi(j,\tilde{y})}{\pi(i,y)} \times \frac{g_{j,\tilde{y}}(m_{j\to i})g_{\tilde{y},m_{j\to i}}(\tilde{\omega})}{g_{i,y}(m_{i\to j})g_{y,m_{i\to j}}(\omega)} \times \left|\frac{\partial f(y,\omega)}{\partial y\partial\omega}\right|\ .$$

Dans le cas qui nous intéresse, la loi à échantillonner est du type $\pi(i,y,z)$ définie sur un espace $\bigcup_{i\in\mathcal{I}}(\{i\}\times\mathcal{Y}_i\times\mathcal{Z}_i)$, où $\mathcal{Y}_i$ est un espace continu et $\mathcal{Z}_i$ un espace discret. Le mouvement $m_{i\to j}$ consiste alors à proposer $(j,\tilde{y},\tilde{z})$ à partir de (i,y,z), selon une procédure identique à celle décrite ci-dessus en utilisant une loi instrumentale de densité $g_{y,z,m_{i\to j}}(\omega,\tilde{z})$. Suivant un raisonnement analogue au cas purement continu, la probabilité d'acceptation s'écrit alors

$$\left\{\begin{array}{rcl}\alpha\big((y,z),(\tilde{y},\tilde{z})\big) &=& min(1;A^+)\\ \alpha\big((\tilde{y},\tilde{z}),(y,z)\big) &=& min(1;1/A^+)\ ,\end{array}\right.$$

où

$$A^+ = \frac{\pi(j,\tilde{y},\tilde{z})}{\pi(i,y,z)} \times \frac{g_{j,\tilde{y},\tilde{z}}(m_{j\to i})g_{\tilde{y},\tilde{z},m_{j\to i}}(\tilde{\omega},z)}{g_{i,y,z}(m_{i\to j})g_{y,z,m_{i\to j}}(\omega,\tilde{z})} \times \left|\frac{\partial f(y,\omega)}{\partial y\partial\omega}\right|\ . \quad (2.22)$$

2.5.4 *Reversible Jump* MCMC pour les modèles de chaînes de Markov cachées

Pour utiliser l'approche bayésienne de sélection de modèle sur des HMM, il suffit de définir une loi *a priori* de densité[9] $\pi(i)$ sur l'ensemble $\mathcal{M}$ des différentes architectures de HMM. La loi *a priori* $\pi_i(\theta)$ $(= \pi(\theta \mid i))$ sur les paramètres $\theta = (a, b)$ des différents modèles de $\mathcal{M}$ est quant à elle choisie comme pour l'estimation des paramètres à architecture fixée.

Les méthodes *Reversible Jump* MCMC peuvent alors permettre d'échantillonner la loi *a posteriori* $\pi(i, \theta, s_1^n \mid x_1^n)$ dont $\pi(i, \theta \mid x_1^n)$ et $\pi(i \mid x_1^n)$ sont des marginales. Cet échantillonnage est rendu possible en introduisant, dans l'algorithme MCMC utilisé pour l'estimation à architecture fixée, des étapes « à la *Reversible Jump* Metropolis-Hastings » qui permettent de modifier pas à pas l'architecture du HMM.

On va présenter ici des algorithmes MCMC pour l'estimation du nombre d'états cachés et de l'ordre des modèles markoviens d'émission des observations. En dehors de l'intérêt propre de ces problèmes de sélection de modèle, un des objectifs de cette présentation est de discuter des conditions pour que l'échantillonnage par *Reversible Jump* MCMC soit efficace.

D'autres approches pour l'adaptation progressive de l'architecture d'un HMM

L'utilisation de méthodes *Reversible Jump* MCMC pour estimer l'architecture d'un HMM s'inscrit, en partie, dans le prolongement d'autres procédures d'adaptation pas à pas de l'architecture d'un HMM dont on va juste citer trois exemples. Le dernier nous concerne particulièrement car il se situe dans un cadre bayésien de sélection de modèle.

Krogh *et al.* (1994) [KBM+94] proposent une heuristique de « chirurgie de modèle » (*model surgery*) pour modifier progressivement un HMM dédié à l'alignement multiple de séquences protéiques. Sans entrer dans les détails, un tel HMM a une structure périodique linéaire dont le module élémentaire est constitué de trois états « *match* », « *insert* » et « *delete* » (ou deux selon le point de vue, l'état *delete* étant silencieux). L'heuristique consiste à supprimer ou insérer des modules après convergence de l'algorithme EM, selon la fraction des séquences dont le chemin caché visite les différents états d'un module. Un module est supprimé si son état *match* est visité par moins de la moitié des chemins cachés. Au contraire, un ou plusieurs nouveaux modules sont insérés si l'état *insert* est visité par plus de la moitié des chemins cachés. Cet algorithme semble donner des résultats satisfaisants. Étant donnée l'architecture du modèle et son interprétation biologique, il semble astucieux et de bon sens.

[9] Il suffit de connaître la densité de cette loi à un facteur près.

Freitag et McCallum (2000) [FM00] construisent un HMM pour extraire de l'information textuelle (les états émettent des mots) en optimisant pas à pas l'architecture du HMM. Sept types d'opérations d'ajout d'états permettent d'atteindre un ensemble de modèles dont les architectures sont adaptées à l'utilisation du HMM. Le type d'opération détermine la façon dont les nouveaux états sont connectés aux états préexistants. Chaque étape de l'algorithme d'optimisation consiste à tenter toutes les opérations possibles, puis à sélectionner la meilleure architecture résultante par validation croisée supervisée.

Stolcke (1994) [Sto94] cherche à trouver, par fusion d'états (*merging*), l'architecture d'un HMM modélisant les différentes prononciations d'un même mot. Ces prononciations sont « écrites » dans un alphabet de 64 phonèmes. Cet algorithme nous concerne particulièrement car il opère dans le cadre bayésien. En effet, les architectures sont évaluées grâce à une approximation de leur probabilité *a posteriori* sur un échantillon des différentes prononciations d'un même mot. On va donc présenter rapidement le type d'architectures explorées et la façon dont Stolcke propose de contourner la difficulté du calcul de la vraisemblance marginale. Les principales caractéristiques de ces architectures sont : chaque état n'émet qu'un sous-ensemble de phonèmes, le nombre de transitions par état est faible comparé au nombre total d'états, et enfin, le graphe des transitions autorisées est acyclique (exceptées les transitions d'un état caché vers lui même). On omettra ici les détails de la loi *a priori* sur les architectures.

L'architecture du HMM de départ est « maximale » au sens où chaque prononciation est modélisée par une suite particulière d'états cachés. Ces états sont connectés par des transitions de probabilité 1 et n'émettent que l'unique phonème qui leur correspond dans cette prononciation. L'algorithme modifie ensuite pas à pas l'architecture par fusion de couples d'états. L'état qui résulte d'une fusion hérite des transitions impliquant les deux états parents. En simplifiant un peu, toutes les fusions sont évaluées à chaque itération de l'algorithme et la fusion à l'origine de la plus importante augmentation de la probabilité *a posteriori* du modèle est finalement retenue.

La clé de l'efficacité relative de l'algorithme est l'approximation de la vraisemblance marginale. La « vraie » vraisemblance marginale d'un HMM est une intégrale sur les paramètres et l'ensemble des chemins possibles :

$$P(x_1^n \mid M = i) \quad = \quad \int_{\theta \in \Theta_i} \int_{s_1^n} P_\theta(x_1^n, s_1^n) \pi_i(\theta) ds_1^n d\theta \ .$$

Cette intégrale ne peut être calculée. Stolcke propose de l'approcher par une intégrale ne concernant que les paramètres qui peut être calculée analytiquement selon une procédure analogue à celle qui a permis d'obtenir l'équation 2.19 :

$$P(x_1^n \mid M = i) \quad \approx \quad \int_{\theta \in \Theta_i} P_\theta(x_1^n, s_1^{n\star}) \pi_i(\theta) d\theta \ ,$$

où $s_1^{n\star}$ est le « chemin caché de Viterbi ». Moralement, ce chemin est associé à une estimation de θ mais Stolcke la contourne : le « chemin caché de Viterbi » est hérité lors de la fusion de deux états. Stolcke justifie les approximations en soutenant que les architectures considérées contraignent fortement le chemin caché. Ces approximations reviennent, en fait, à remplacer l'objectif de maximisation de $P(M = i \mid x_1^n)$ en i par celui de maximisation de $P(M = i, s_1^n \mid x_1^n)$ en (i, s_1^n).

L'utilisation des *Reversible Jump* MCMC va être présentée ici sur des architectures de HMM beaucoup plus simples que celles étudiées par Stolcke. On verra tout de même comment un échantillonnage par *Reversible Jump* MCMC pourrait, au moins en théorie, permettre d'éviter les approximations utilisées par Stolcke.

Estimation du nombre d'états cachés d'un HMM dont les lois d'émission sont des $\mathcal{N}(0, \sigma_u)$

Une des premières applications des méthodes *Reversible Jump* MCMC a concerné l'estimation du nombre de composantes dans un mélange de lois normales par Richardson et Green (1997) [RG97]. L'extension[10] à l'estimation du nombre d'états cachés d'un HMM a ensuite été proposée par Robert *et al.* (2000) [RRT00]. Dans ces modèles HMM, toutes les transitions entre états cachés sont autorisées. Les deux travaux considèrent des modèles d'émission des variables observées dans $\mathbb{R}$: une loi $X_t \mid S_t = u \sim \mathcal{N}(\mu_u, \sigma_u)$ pour la composante u du mélange ; $X_t \mid S_t = u \sim \mathcal{N}(0, \sigma_u)$ pour l'état caché u du HMM.

En plus d'étapes de mise à jour des paramètres en dimension fixée, deux paires de mouvements « à la *Reversible Jump* Metropolis-Hastings » sont introduites pour permettre des passages du modèle à i composantes au modèle à $i+1$ composantes (états cachés pour les HMM) : la paire *split/merge* et la paire *birth/death* d'une composante u vide (t.q $\sum_t \mathbb{I}_{\{s_t=u\}} = 0$). On va rapidement présenter ici les principales caractéristiques de ces deux paires de mouvements dans le cas HMM.

Un objectif commun à ces deux paires de mouvements est de tenter des sauts entre des valeurs de paramètres suffisamment « proches » (mais dans des espaces différents) pour que les sauts proposés soient acceptés avec une fréquence relativement élevée. L'idée sous-jacente est que si la densité $\pi(i, \theta, s_1^n \mid x_1^n)$, où $\theta \in \Theta_i$ et $s_1^n \in \mathcal{S}_i^n$, est élevée, il y a alors des chances pour que la densité $\pi(i+1, \tilde{\theta}, \tilde{s}_1^n \mid x_1^n)$, où $\tilde{\theta} \in \Theta_{i+1}$ et $\tilde{s}_1^n \in \mathcal{S}_{i+1}^n$, le soit aussi (et vice-versa) et donc pour que le changement de dimension soit accepté.

La paire *split/merge*. Le mouvement *split* permet le saut du modèle à i états cachés au modèle à $i+1$ états cachés en « divisant » un des états cachés

[10] Le modèle de mélange peut être vu comme un cas « dégénéré » de HMM où les états cachés se succèdent de manière indépendante.

$u^\star$ ($u^\star \in \mathcal{S}_i$), choisi aléatoirement, en deux états u_1 et u_2 ($(u_1, u_2) \in \mathcal{S}_{i+1}^2$). Le vecteur ω (cf. Eq. 2.22) tiré pour permettre le saut est de dimension minimale : $2i$ éléments pour la transformation de la matrice a en $\tilde{a}$ plus 1 élément pour la transformation des paramètres b ($= (\sigma_u)_{u \in \mathcal{S}}$) des lois d'émission en $\tilde{b}$. Le mouvement est construit de manière à préserver certaines propriétés importantes du processus caché et du processus observé : la probabilité de $u \neq u^\star$ sous la loi stationnaire du processus caché ; les lois d'émission associées aux états $u \neq u^\star$; et le moment d'ordre 2 du processus observé (le moment d'ordre 1 étant nul puisque $X_t \mid S_t = u \sim \mathcal{N}(0, \sigma_u)$). Après avoir ainsi choisi les nouveaux paramètres $\tilde{\theta} = (\tilde{a}, \tilde{b})$, un nouveau chemin caché $\tilde{s}_1^n$ est proposé par affectation à u_1 ou à u_2 des positions anciennement affectées à $u^\star$ (t.q. $s_t = u^\star$). Cette affectation est réalisée par tirage selon une technique proche de celle présentée dans la section 2.2.3. Les détails de cette procédure de *split* sont assez techniques et ne seront pas décrits ici.

Le mouvement *merge* permet le saut exactement inverse. Une fois $(u_1, u_2) \in \mathcal{S}_{i+1}^2$ choisis, la procédure est déterministe : $dim(\tilde{\omega}) = 0$ et les états anciennement affectés à u_1 et à u_2 sont affectés à $u^\star$.

La paire *birth/death* d'un état vide. Elle est techniquement plus simple que la paire *split/merge*. Le mouvement *birth* ajoute un nouvel état caché au HMM sans modifier les lois d'émission associées aux états existants, sans modifier s_1^n (le nouvel état est donc vide) et avec des modifications minimales des probabilités de transitions entre les états existants. Comme le mouvement *split*, le mouvement *birth* nécessite le tirage d'un vecteur ω de dimension $2i + 1$. Le mouvement *death* consiste à choisir un état vide (si il en existe un) et à réaliser l'opération inverse.

On ne décrira pas non plus le détail de cette paire de mouvements car la paire de mouvements *birth/death* d'un état éventuellement non vide est décrite ci-dessous dans le cadre de la modélisation de séquences d'ADN.

Les performances de l'algorithme MCMC pour échantillonner la loi *a posteriori* du nombre d'état cachés dans la famille de HMM considérée par Robert *et al.* semblent relativement satisfaisantes. L'échantillonnage est permis grâce à des taux d'acceptation des mouvements *split/merge* respectivement de $4,4$ %, $1,4$ % et $0,26$ % sur 3 séquences de natures différentes et de longueurs respectives 1 700, 500 et 2 700. Les mouvements de la paire *birth/death* d'un état vide ne sont quant à eux pratiquement jamais acceptés (taux $< 3 \times 10^{-5}$).

Estimation du nombre d'états cachés d'un HMM dédié à l'étude des hétérogénéités de l'ADN

Boys et Henderson (2002) [BH02a] proposent d'appliquer la même méthodologie pour l'estimation du nombre d'états d'un HMM dédié à la mo-

délisation des séquences d'ADN. Dans ce HMM toutes les transitions entre états cachés sont autorisées et le modèle est du type M1-M0 (le modèle markovien d'émission des nucléotides est d'ordre 0). Comme on le verra dans le chapitre 3, ces architectures permettent la segmentation d'une séquence d'ADN en régions de composition homogène.

Contrairement à Robert *et al.*, Boys et Henderson obtiennent un taux d'acceptation satisfaisant pour les mouvements de la paire *birth/death* d'un état vide et extrêmement faible pour ceux de la paire *split/merge*. Ces taux sont respectivement de 11% et de 2×10^{-5} sur la séquence du phage λ (48 502 bp). Ils introduisent aussi une paire *birth/death* d'un état éventuellement non vide (on dira simplement *birth/death*) dont les mouvements sont acceptés à un taux intermédiaire de $3,7$ %. Pour comprendre ces résultats, nous avons implémenté la paire de mouvements *birth/death*.

La loi *a priori*. Boys et Henderson utilisent une loi de Poisson tronquée pour la modélisation *a priori* du nombre d'états cachés :

$$P(M=i) \propto \frac{\lambda^i}{i!}\mathbb{I}_{\{1\leq i\leq i_{max}\}} \ ,$$

où $\lambda = 3$ et $i_{max} = 9$. Les lois *a priori* sur les paramètres $\theta = (a,b) \in \Theta_i$ sont des lois de Dirichlet indépendantes (cf. Eq. 2.11 et 2.10) :

$$\begin{aligned} a(u,\bullet) \mid i &\sim D_i\big(\alpha_i(u,1),\ldots,\alpha_i(u,i)\big) \\ b_u(\bullet) &\sim D_4\big(\beta_u(1),\ldots,\beta_u(4)\big) \ , \end{aligned}$$

où $\alpha_i(u,u) = 0,999 \times info/i$, $\alpha_i(u,v) = 0,001 \times 1/(i-1) \times info/i$ pour $u \neq v$, info $= 1000 \times i_{max}$ et $\beta_u(x) = 1$. La loi *a priori* sur b est donc non informative. Au contraire, la loi *a priori* sur a charge la diagonale, elle est donc informative. Son « poids » correspond à celui de l'observation d'une suite d'états cachés de longueur $info$ où les durées de séjour dans chaque état sont d'une longueur moyenne de 1000 nucléotides ($info = \sum_{u,v}\alpha_i(u,v)$).

La paire *birth/death*. Le mouvement *birth* ajoute un état dans un HMM à i états. On notera $u^\star$ l'indice de ce nouvel état dans $\mathcal{S}_{i+1} = \{1,\ldots,i+1\}$. Les nouveaux paramètres $\tilde{\theta} = (\tilde{a},\tilde{b})$ qui « concernent » l'état caché $u^\star$ sont tirés selon leur loi *a priori* :

$$\begin{aligned} \tilde{a}(u^\star,\bullet) &\sim D_{i+1}\big(\alpha_{i+1}(u,1),\ldots,\alpha_{i+1}(u,i+1)\big) \\ \tilde{a}(u,u^\star) &\sim Be\big(\alpha_{i+1}(u,u^\star),\sum_{v\neq u^\star}\alpha_{i+1}(u,v)\big) \ , \quad u \neq u^\star \\ \tilde{b}_{u^\star}(\bullet) &\sim D_4\big(\beta_u(1),\ldots,\beta_u(4)\big) \ , \end{aligned}$$

où $Be\big(\alpha_{i+1}(u,u^\star),\sum_{v\neq u^\star}\alpha_{i+1}(u,v)\big)$ est la loi bêta marginale en $u^\star$ de la loi *a priori* sur $\tilde{a}(u,\bullet)$. Ces tirages correspondent au choix de ω ($2i+3$ paramètres

libres). Les paramètres qui ne « concernent » pas $u^\star$ sont, dans la mesure du possible, laissés inchangés :

$$\begin{aligned} \tilde{b}_u(\bullet) &= b_{pa(u)}(\bullet), \quad u \neq u^\star \\ \tilde{a}(u,v) &= a(pa(u),pa(v)) \times (1-\tilde{a}(u,u^\star)), \quad u \neq u^\star, v \neq u^\star \end{aligned} \tag{2.23}$$

où $pa(u)$ désigne l'indice de u dans le modèle d'origine et $(1-\tilde{a}(u,u^\star))$ est le facteur de renormalisation de la ligne $a(u,\bullet)$. Le nouveau chemin caché $\tilde{s}_1^n$ associé à $\tilde{\theta}$ est finalement tiré selon sa loi conditionnelle $\pi(\tilde{s}_1^n \mid i+1, \tilde{\theta}, x_1^n)$ grâce à l'algorithme présenté dans la section 2.2.3.

Le mouvement *death* permet le mouvement inverse de passage du modèle à $i+1$ états vers le modèle à i états. L'état $u^\star$ est choisi au hasard puis les nouveaux paramètres θ sont obtenus à partir de $\tilde{\theta}$ en supprimant les paramètres qui concernent l'état $u^\star$ et en renormalisant les lignes de la matrice $\tilde{a}$. Le nouveau chemin caché s_1^n est finalement tiré selon $\pi(s_1^n \mid i, \theta, x_1^n)$.

Les probabilités $g_i(m_{i\to i+1})$ et $g_i(m_{i\to i-1})$ qui correspondent respectivement à la probabilité de tenter un mouvement *birth* et à celle de tenter un mouvement *death* sont choisies comme : $g_i(m_{i\to i+1}) + g_i(m_{i\to i-1}) = 1$ avec $g_i(m_{i\to i+1}) = 1$ si $i = 1$; $g_i(m_{i\to i+1}) = 0$ si $i = i_{max}$; $g_i(m_{i\to i+1}) = 1/2$ sinon.

La probabilité d'acceptation d'un mouvement de cette paire est obtenue grâce à l'équation 2.22, où $j = i+1$, y est identifié à θ, z à s_1^n, ω à $\big(\tilde{a}(u^\star,\bullet), (\tilde{a}(u,u^\star))_{u\neq u^\star}, \tilde{b}_{u^\star}(\bullet)\big)$ et $\tilde{\omega}$ à $\varnothing$:

$$\begin{aligned} A^+ &= \frac{\pi(i+1,\tilde{\theta},\tilde{s}_1^n \mid x_1^n)}{\pi(i,\theta,s_1^n \mid x_1^n)} \\ &\times \frac{g_{i+1}(m_{i+1\to i}) g_{m_{i+1\to i}}(s_1^n)}{g_i(m_{i\to i+1}) g_{m_{i\to i+1}}(\tilde{a}(u^\star,\bullet),(\tilde{a}(u,u^\star))_{u\neq u^\star},\tilde{b}_{u^\star}(\bullet),\tilde{s}_1^n)} \\ &\times \prod_{u\neq u^\star} (1-\tilde{a}(u,u^\star))^{i-1} \ , \end{aligned}$$

où $\prod_{u\neq u^\star}(1-\tilde{a}(u,u^\star))^{i-1}$ est le jacobien[11] associé à la transformation 2.23. A^+ se récrit

$$\begin{aligned} A^+ &= \frac{\pi(i+1,\tilde{\theta},\tilde{s}_1^n,x_1^n)}{\pi(i,\theta,s_1^n,x_1^n)} \\ &\times \frac{g_{i+1}(m_{i+1\to i})\pi(s_1^n \mid i,\theta,x_1^n)}{g_i(m_{i\to i+1}) g_{m_{i\to i+1}}(\tilde{a}(u^\star,\bullet),(\tilde{a}(u,u^\star))_{u\neq u^\star},\tilde{b}_{u^\star}(\bullet))\pi(\tilde{s}_1^n \mid i+1,\tilde{\theta},x_1^n)} \\ &\times \prod_{u\neq u^\star} (1-\tilde{a}(u,u^\star))^{i-1} \ . \end{aligned}$$

[11] Le jacobien n'est pas égal à $\prod_{u\neq u^\star}(1-\tilde{a}(u,u^\star))^i$ car il n'y a que i paramètres libres par ligne de la matrice $\tilde{a}$ (voir [RG98]).

Boys et Henderson remarquent que s_1^n et $\tilde{s}_1^n$ n'interviennent pas dans A^+ puisque

$$\frac{\pi(i,\theta,s_1^n,x_1^n)}{\pi(s_1^n \mid i,\theta,x_1^n)} = \pi(i,\theta,x_1^n) \ .$$

Ce qui permet d'obtenir

$$\begin{aligned} A^+ \quad = \quad & \frac{\pi(i+1,\tilde{\theta},x_1^n)}{\pi(i,\theta,x_1^n)} \\ & \times \frac{g_{i+1}(m_{i+1\to i})}{g_i(m_{i\to i+1})g_{m_{i\to i+1}}(\tilde{a}(u^\star,\bullet),(\tilde{a}(u,u^\star))_{u\neq u^\star},,\tilde{b}_{u^\star}(\bullet))} \\ & \times \prod_{u\neq u^\star}(1-\tilde{a}(u,u^\star))^{i-1} \ . \end{aligned}$$

Le terme $\pi(i,\theta,x_1^n)$ s'écrit aussi $\pi(i,\theta)P_\theta(x_1^n \mid i)$ où $P_\theta(x_1^n \mid i)$ correspond à la vraisemblance des données incomplètes (section 2.3).

La paire de mouvements *birth/death* d'un état vide est essentiellement identique sauf que les états ne sont pas réaffectés lors des mouvements. Seuls des états vides peuvent donc être créés ou supprimés.

Une surprenante sensibilité du comportement de l'algorithme envers la loi *a priori*. Les résultats présentés ici ont été obtenus avec l'algorithme MCMC constitué des étapes de mises à jour de θ et de s_1^n présentées page 67 et d'une étape qui consiste à tenter un mouvement de la paire *birth/death*. Une itération de l'algorithme consiste à :

- mettre à jour $\theta = (a,b)$ en dimension fixée,
- mettre à jour s_1^n,
- tenter un changement du nombre d'états cachés à travers la paire de mouvements *birth/death* présentée ci-dessus.

La séquence étudiée est celle du phage λ.

Les valeurs $i^{(0)}, i^{(1)}, \ldots, i^{(M)}$ obtenues au cours de 50 000 itérations de l'algorithme sont présentées figure 2.3a. La distribution *a posteriori* du nombre d'états, estimée sur 40 000 itérations, est présentée figure 2.3b. Ces résultats sont globalement en accord avec ceux de Boys et Henderson. Le taux d'acceptation des mouvements *birth/death* est relativement satisfaisant $(2,7\%)$. L'algorithme semble échantillonner correctement la loi *a posteriori* $\pi(i \mid x_1^n)$. Le nombre d'états le plus probable *a posteriori* est 7.

La figure 2.3c montre le comportement du même algorithme avec la loi *a priori* sur a utilisée par Robert et al. où tous les α sont pris égaux à 1. Avec cette loi non informative, aucun changement de dimension n'est accepté après les premières itérations et le nombre d'états sur lequel se stabilise l'algorithme diffère d'une exécution à l'autre (notamment en fonction du nombre d'états initial). D'un point de vue pratique, l'algorithme ne permet donc pas d'échantillonner la loi *a posteriori*.

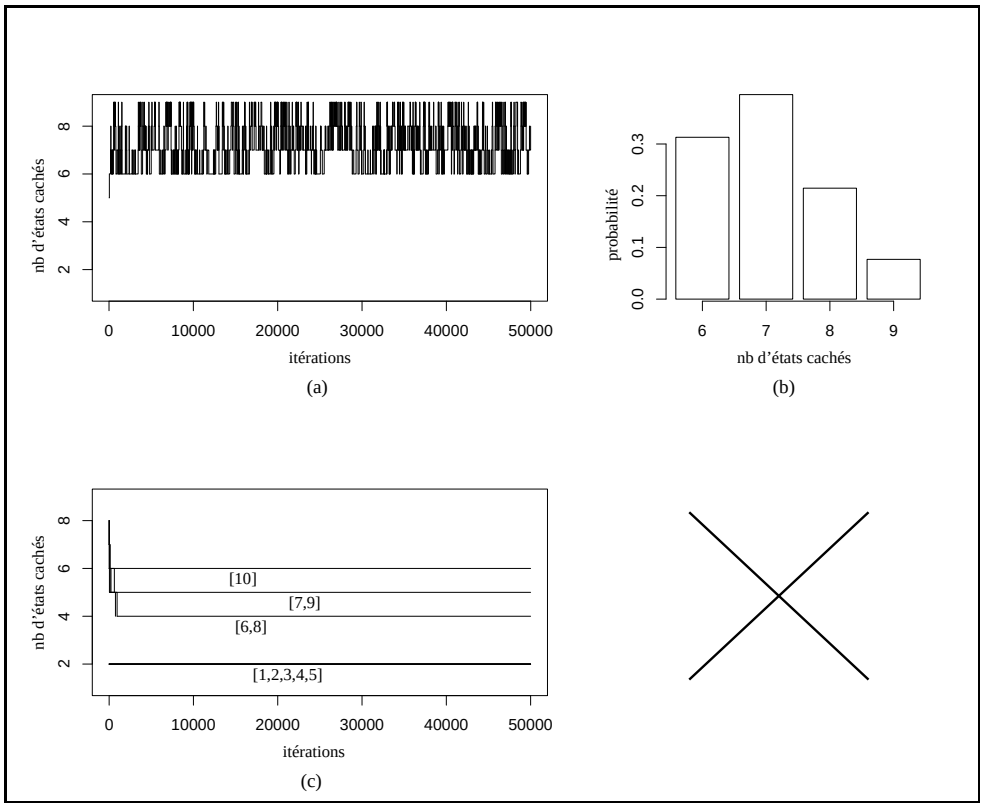

FIG. 2.3 – Estimation du nombre d'états cachés d'un HMM M1-M0 sur la séquence du phage λ (48 502 bp). **(a)** Échantillonnage du nombre d'états cachés au cours de 50 000 itérations de l'algorithme MCMC avec la loi *a priori* utilisée par Boys et Henderson. **(b)** Distribution *a posteriori* du nombre d'états cachés estimée sur les 40 000 dernières itérations présentée en (a). **(c)** Échantillonnage du nombre d'états cachés (comme (a)) mais avec une loi *a priori* uniforme sur la matrice de transition de la chaîne cachée, l'algorithme a été d'abord exécuté 5 fois [1-5] à partir d'un nombre d'états cachés initial de 2 et 5 fois [6-10] à partir d'un nombre d'états cachés initial de 8.

Le comportement décrit par Boys et Henderson apparaît donc très lié au choix d'une loi *a priori* informative particulière pour la matrice a de transition de la chaîne cachée. L'influence de cette loi *a priori* informative peut être compris, au moins en partie, en observant qu'elle entraîne une probabilité *a posteriori* non négligeable pour qu'un des états du HMM soit vide (puisqu'elle suppose *a priori* des séjours longs dans les états). Ainsi, au moins un des états est vide lors de $5,9\%$ des 50 000 itérations de l'échantillonnage présenté figure 2.3a. Au contraire, il n'y a jamais un état vide lorsque la loi *a priori* non informative est utilisée. Cette observation explique aussi probablement la différence entre le taux d'acceptation élevé des mouvements de *birth/death* d'un état vide décrit par Boys et Henderson et le taux extrêmement faible obtenu par Robert *et al.*. Enfin, il semble légitime de s'interroger sur le sens à donner à une estimation qui tient compte du fait qu'« étant donnée nos informations *a priori* », il est relativement probable que « des états existent mais n'aient pas été visités sur cette séquence ». Pourquoi ne pas alors supposer directement qu'il en existe beaucoup plus que $i_{max} = 9$!

Malgré quelques tentatives supplémentaires, nous n'avons par réussi à échantillonner la loi *a posteriori* associée à une loi *a priori* non informative sur la matrice a. Nous interprétons cet échec comme dû à l'ampleur des sauts tentés qui nécessitent l'introduction « à l'aveugle » de $2i+3$ nouveaux paramètres. Comme nous allons le voir maintenant, des sauts de grande ampleur peuvent tout de même être réalisés si il est possible d'avoir une idée de la « direction » dans laquelle proposer les nouveaux paramètres.

Estimation de l'ordre des modèles markoviens d'émission des observations

Le problème est ici le choix de l'ordre r_u du modèle markovien d'émission des observations associé à chaque état caché. Ce problème est plus difficile que celui de l'estimation de l'ordre d'une chaîne de Markov non cachée. En effet, la segmentation en états cachés est inconnue et peut dépendre de l'ordre des modèles d'émission associée aux différents états. D'autre part, le modèle recherché appartient à un ensemble éventuellement grand puisque toutes les combinaisons d'ordres sont envisagées. On va cependant voir que les méthodes *Reversible Jump* MCMC permettent, dans ce cas, une sélection efficace du modèle.

Le vecteur $r = (r_1, \ldots, r_{|\mathcal{S}|})$, où r_u est l'ordre associé à l'état u, est modélisé par des lois *a priori* indépendantes et uniformes :

$$r_u \sim \mathcal{U}_{[r_{min}, r_{max}]}, \quad u \in \mathcal{S} \ .$$

Les lois *a priori* sur les paramètres a et b sont données par les équations 2.11 et 2.10 où tous les α et tous les β sont pris égaux à 1.

L'algorithme proposé contient les deux étapes « à la Gibbs » de mise à jour de θ et de s_1^n en dimension fixée (cf. page 67), plus une étape « à la

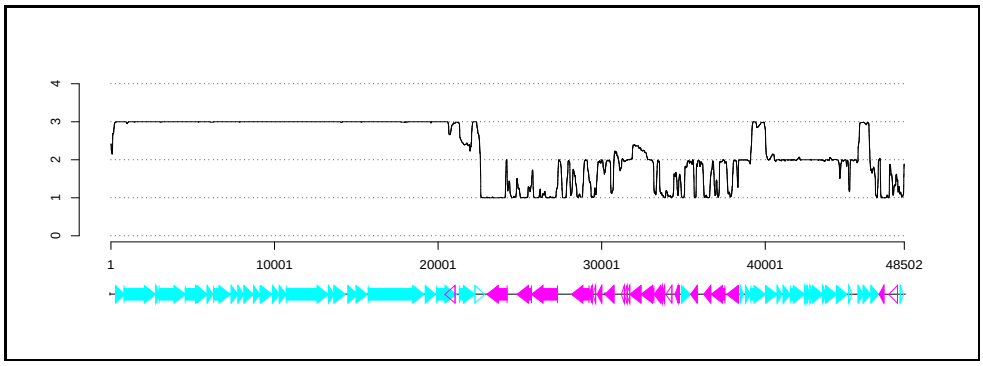

FIG. 2.4 – Estimation de l'ordre markovien d'émission des observations dans un HMM. Représentation de l'espérance de l'ordre de l'état caché associé à chaque position t ($\frac{1}{M}\sum_{m=1}^{M} r_{s_t^{(m)}}$) de la séquence du phage λ. La position et le sens des gènes (séquences codantes) sont représentés par des flèches en dessous du graphe.

Reversible Jump Metropolis-Hastings » qui permet de modifier l'ordre d'un des états cachés.

La procédure de changement d'ordre consiste à tirer au hasard un état $u^\star$, puis à choisir entre une tentative d'augmentation ou de diminution de son ordre : $\tilde{r}_{u^\star} = r_{u^\star} + 1$ ou $\tilde{r}_{u^\star} = r_{u^\star} - 1$. Les nouveaux paramètres d'émission $\tilde{b}_{u^\star}(\bullet, w)$, où $w \in \mathcal{X}^{\tilde{r}_{u^\star}}$ sont ensuite tirés selon leur loi conditionnelle aux autres paramètres $\pi(\tilde{b}_{u^\star} \mid r_{u^\star}, s_1^n, x_1^n)$, comme dans l'étape de mise à jour de b en dimension fixée. Tous les autres paramètres sont laissés inchangés : $\tilde{s}_1^n = s_1^n$, $\tilde{a} = a$, $\tilde{b}_u = b_u$ et $\tilde{r}_u = r_u$ pour $u \neq u^\star$.

Le probabilité d'acceptation de ce mouvement de changement d'ordre est $\min(1, A)$, où A est donné par l'équation 2.22 en identifiant ω à $\tilde{b}_{u^\star}$ et $\tilde{\omega}$ à $b_{u^s tar}$:

$$A = \frac{\pi(\tilde{r}, \tilde{\theta}, s_1^n \mid x_1^n)}{\pi(r, \theta, s_1^n \mid x_1^n)} \times \frac{g_{\tilde{r}_{u^\star}}(m_{\tilde{r}_{u^\star} \to r_{u^\star}}) \pi(b_{u^\star} \mid r_{u^\star}, s_1^n, x_1^n)}{g_{r_{u^\star}}(m_{r_{u^\star} \to \tilde{r}_{u^\star}}) \pi(\tilde{b}_{u^\star} \mid \tilde{r}_{u^\star}, s_1^n, x_1^n)} \times 1 \ .$$

On peut récrire cette équation

$$A = \frac{\pi(\tilde{r}_{u^\star} \mid s_1^n, x_1^n)}{\pi(r_{u^\star} \mid s_1^n, x_1^n)} \times \frac{g_{\tilde{r}_{u^\star}}(m_{\tilde{r}_{u^\star} \to r_{u^\star}})}{g_{r_{u^\star}}(m_{r_{u^\star} \to \tilde{r}_{u^\star}})} \ ,$$

où il est clair que $b_{u^\star}$ et $\tilde{b}_{u^\star}$ n'interviennent pas dans l'acceptation. Le mouvement peut donc être accepté ou refusé avant le tirage de $\tilde{b}_{u^\star}$. Cette formule

se récrit

$$\begin{aligned} A &= \frac{\pi(\tilde{r}_{u^\star}, s_1^n, x_1^n)}{\pi(r_{u^\star}, s_1^n, x_1^n)} \times \frac{g_{\tilde{r}_{u^\star}}(m_{\tilde{r}_{u^\star} \to r_{u^\star}})}{g_{r_{u^\star}}(m_{r_{u^\star} \to \tilde{r}_{u^\star}})} \\ &= \frac{\int_{b_{u^\star}} \prod_{x \in \mathcal{X}} \prod_{w \in \mathcal{X}^{\tilde{r}_{u^\star}}} b_{u^\star}^{N_{wx;u^\star}} \pi(b_{u^\star} \mid \tilde{r}_{u^\star}) db_{u^\star}}{\int_{b_{u^\star}} \prod_{x \in \mathcal{X}} \prod_{w \in \mathcal{X}^{r_{u^\star}}} b_{u^\star}^{N_{wx;u^\star}} \pi(b_{u^\star} \mid r_{u^\star}) db_{u^\star}} \times \frac{g_{\tilde{r}_{u^\star}}(m_{\tilde{r}_{u^\star} \to r_{u^\star}})}{g_{r_{u^\star}}(m_{r_{u^\star} \to \tilde{r}_{u^\star}})} , \end{aligned}$$

A est donc obtenu par une intégration du numérateur et du dénominateur comme pour le calcul de la vraisemblance marginale dans le modèle markovien (cf. Eq. 2.19).

Cette étape de changement d'ordre est un cas particulier d'algorithme *Reversible Jump* MCMC. Elle peut être vue comme une étape « à la Metropolis-Hastings » classique d'échantillonnage de $\pi(r_{u^\star} \mid s_1^n, x_1^n)$ puisque $b_{r_{u^\star}}$ et $\tilde{b}_{r_{u^\star}}$ n'interviennent pas dans la probabilité d'acceptation. On considérera cependant ici qu'il s'agit d'un algorithme *Reversible Jump* MCMC car on échantillonne finalement $\pi(r_{u^\star}, b_{u^\star} \mid s_1^n, x_1^n)$ où la dimension de $b_{u^\star}$ varie en fonction de la valeur de $r_{u^\star}$.

Contrairement à l'algorithme d'estimation du nombre d'états cachés, cet algorithme se comporte bien. Aucun saut n'est accepté après quelques itérations car le meilleur modèle est atteint et l'incertitude *a posteriori* est extrêmement faible sur le choix du modèle. Les problèmes de convergence vers des modes locaux de la loi *a posteriori* rencontrés en dimension fixée peuvent, tout de même, toujours survenir.

Sur la séquence du phage λ avec un modèle à 3 états, les ordres estimés sont 1, 2 et 3 (avec une probabilité *a posteriori* estimée à 1). La figure 2.4 représente la valeur de l'espérance de l'ordre de l'état caché associé à la position t ($\frac{1}{M}\sum_{m=1}^{M} r_{s_t^{(m)}}$) le long de la séquence.

Boys et Henderson (2002) [BH02b] ont construit un algorithme proche pour choisir l'ordre r d'un modèle M1-Mr. Dans un modèle M1-Mr, l'ordre est supposé identique pour tous les états cachés mais l'algorithme pourrait aussi bien s'appliquer sans cette contrainte. L'algorithme contient une étape de changement d'ordre « à la Gibbs » qui consiste à proposer le nouvel ordre $\tilde{r}$ selon la loi r conditionnellement à s_1^n et x_1^n. Avec les notations utilisées ici, on a $g_{r,s_1^n,x_1^n}(m_{r \to \tilde{r}}) = \pi(\tilde{r} \mid s_1^n, x_1^n)$. Cette loi peut être obtenue analytiquement d'une manière analogue à celle utilisée pour obtenir l'équation 2.19. La probabilité d'acceptation de ce mouvement est de 1.

Ces deux algorithmes sont très proches et d'efficacité certainement comparables.

Le chapitre 5 présente une mise en œuvre des méthodes *Reversible Jump* MCMC pour l'estimation de certaines propriétés d'un modèle semi-markovien

caché dédié à la recherche des sites de fixation de l'ARN polymérase chez les bactéries. Les questions de choix de modèle abordées dans le chapitre 5 pourraient se poser, et être traitées, de façon essentiellement similaire dans un cadre HMM. Ainsi, l'estimation de l'ordre d'un modèle markovien d'émission utilise une étape identique à celle présentée ci-dessus. Quant aux choix de la longueur des « boîtes » et du support de la distribution de la distance qui les sépare (cf. page 29), ils peuvent être interprétés comme le choix du nombre de certains états dans un HMM très structuré (cf. page 42), c'est à dire où la plupart des transitions entre états sont interdites. Ce problème diffère de celui de l'estimation du nombre d'états cachés présenté ci-dessus. Non seulement, très peu (voir aucun) de paramètres de transitions entre états cachés sont affectés par les changements de dimension. Mais surtout, une fois le type de saut choisi, il est possible d'avoir une idée assez précise du chemin caché sous le nouveau modèle. Les nouvelles valeurs à proposer pour la plupart des paramètres (a et b) peuvent donc être « bien choisies ».

On retrouverait cette possibilité dans le premier [KBM+94] et le troisième [Sto94] des exemples de modification pas à pas de l'architecture d'un HMM présenté au début de la section 2.5.4.

Chapitre 3

Mining *Bacillus subtilis* chromosome heterogeneities using hidden Markov models

Pierre Nicolas, Laurent Bize, Florence Muri, Mark Hoebeke, François Rodolphe, S. Dusko Ehrlich, Bernard Prum et Philippe Bessières

Abstract : We present the use of a new statistical segmentation method on the *Bacillus subtilis* chromosome sequence. Maximum likelihood parameter estimation of a hidden Markov model (HMM), based on the Expectation-Maximization algorithm (EM), enables one to segment the DNA sequence according to its local composition. This approach is not based on sliding windows ; it enables different compositional classes to be separated without prior knowledge of their content, size and localization. We compared these compositional classes, obtained from the sequence, with the annotated DNA physical map, sequence homologies, and repeat regions. The first heterogeneity revealed discriminates between the two coding strands and the non coding regions. Other main heterogeneities arise : some are related to horizontal gene transfer, some to t-enriched composition of hydrophobic protein coding strands, and others to the codon usage fitness of highly expressed genes. Concerning potential and established gene transfers, we found nine of the ten known prophages, plus fourteen new regions of atypical composition. Some of them are surrounded by repeats, most of their genes have unknown function or possess homology to genes involved in secondary catabolism, metal and antibiotic resistance. Surprisingly, we notice that all of these detected regions are $a + t$-richer than the host genome, raising the question of their remote sources.

3.1 Introduction

Numerous factors are known to affect statistical composition of chromosome DNA sequences, such as constraints related to coding properties [Sta94], gene transfers [LO98], and statistical biases related to replication [Lob96]. Horizontal gene transfer between bacteria species [Syv94], often due to mobile elements, is now recognized to play an important role in the acquisition of adaptive traits : such as pathogenicity [HBOMT97, GO97, HK00], resistance to antibiotics [Dav94] or heavy metals, such as mercury [OBSR97, LHS99], or arsenic [CJRS94]. Horizontal transfer has been shown to occur in a wide variety of ecosystems [Dav99], raising questions about the consequences of dispersion of genetic constructions from genetically modified organisms. More generally, horizontal transfer is considered as a driving force of bacteria evolution [Law99, OLG00, dlCD00]. Bacteria are known to integrate prophages [KOM$^+$97], and to have other ways to integrate foreign DNA sequences [LMS$^+$95, MRD99, RDV99a]. These transfers can correspond to DNA segments, which have different statistical properties from those of the host. A classical method of horizontal transfer detection, based on codon usage frequencies, has been introduced by Médigue [MRV$^+$91], and some other approaches have been recently reviewed in Karlin [Kar01].

Hidden Markov models (HMM) are good statistical tools for the analysis of this heterogeneity [Chu89, Mur97, Mur98, BMS$^+$99, PG99, BHW00]. We applied them to the *Bacillus subtilis* chromosome (4.2 Mbp long). In these models, one assumes that a DNA sequence is made up of successive segments, each one belonging to one of a finite number q of types.

Other statistical models could be used for sequence segmentation [BM98], in particular change point models [ORR$^+$99, LL99]. In these models there are no segment types, each segment of the sequence has its own set of composition parameters. In our study, we consider these models to be less realistic because we typically expect the same composition to be found in different segments of the chromosome.

In HMMs, each type of segment is characterized by its own statistical oligonucleotide composition, and the succession of types along the sequence is represented by an unobservable q-state Markov chain (the hidden chain). The aim is first to reconstruct these segments from the DNA sequence, and characterize the identified segment types ; then, to find correlations between segment types and biological DNA features, such as horizontal transfers.

3.2 Materials and methods

3.2.1 Hidden Markov Models

A DNA sequence can be represented by a finite series $x_1, \cdots, x_n$, each base x_t being taken from the alphabet $\mathcal{X} = \{a, c, g, t\}$. Hidden Markov mo-

dels are characterized by two processes (see for instance Rabiner, 1989). The hidden state process $s_1^n = (s_1, \cdots, s_n)$, such that $s_t \in \mathcal{S} = \{1, \cdots, q\}$, which in our set-up governs the succession of the segment types along the sequence, and the observed process $x_1^n = (x_1, \cdots, x_n)$ which corresponds to the observed DNA sequence. Hidden states are generated according to a homogeneous first order Markov chain (*M1*) with transition probabilities $a(u, v) = P(s_{t+1} = v \mid s_t = u)$, $u, v \in \mathcal{S}$. Conditional on the hidden process $s_1^n = (s_1, \cdots, s_n)$, the observed process $x_1^n = (x_1, \cdots, x_n)$ is a heterogeneous Markov chain : base x_t appears in the sequence with a probability distribution which depends on the actual hidden state s_t, as well as on previous bases $x_{t-r}, \ldots, x_{t-1}$. Higher order Markovian dependencies will not be considered for the hidden chain, as we expect to identify large compositional segments, but the number q of hidden states will vary ; similarly the order and other structural features of the observed chain will also vary. Thus, according to the characteristics of these Markov chains, several models of interest can be constructed.

The *M1-M0* model assumes that, conditional on the hidden state s_t, nucleotides x_t are drawn independently with a probability $b_u(x) = P(x_t = x \mid s_t = u)$, $x \in \mathcal{X}$, $u \in \mathcal{S}$. Hence, this model takes into account the local base composition of the sequence, and corresponds to the classical hidden Markov model described in the literature. More generally, the *M1-Mr* model assumes, conditional on the actual hidden state, a r order Markovian dependence between observations, with a transition probability $b_u(x; w) = P(x_t = x \mid s_t = u, x_{t-r}^{t-1} = w)$ for a nucleotide context w of length r. This model, introduced by Churchill [Chu89], accounts for the local $r+1$-nucleotide frequencies of the DNA sequence. We denote the whole set of model parameters by θ *i.e.* transition probabilities between states and between bases.

Given the sequence x_1^n, we consider the maximum likelihood estimator of θ. Consistency and normality results, that justify the maximum likelihood approach, were proved in the *M1-M0* model by Baum and Petrie [BP66], and extended to the *M1-Mr* model by Muri [Mur97]. Several methods exist for estimating the parameters of hidden Markov models, including stochastic likelihood maximization algorithm and bayesian estimation (see [Rab89, QT90, RCD93, Mur97, Mur98, DEKM98, BB98]). All these methods do not require any learning set of pre-segmented sequences to estimate θ. They only require specification of the model structure (number of states, q, and order of the model, r). In our study we choose not to introduce prior information, and choose the EM algorithm to maximize likelihood $P_\theta(x_1^n)$ [BMS+99], which proved to be one of the most effective.

Hidden states are missing data, and the likelihood is a sum over all hidden state paths $P_\theta(x_1^n) = \sum_{s_1^n \in S^n} P_\theta(x_1^n, s_1^n)$, which makes it not directly tractable. The EM algorithm is useful in many estimation problems involving missing data, including HMM. It is an iterative procedure that alternates two steps, see [Chu89, Rab89, DEKM98] for detailed description of

the HMM case and [DLR77] for mathematical proof of the convergence toward the maximum likelihood estimator. Given the current value $\theta^{(m-1)}$, the expectation $E_{\theta^{(m-1)}}(\log P_\theta(x_1^n, s_1^n) \mid x_1^n)$ is computed during the E-step and maximized over θ during the M-step. In the HMM context, the E-step consists of computing the probability of two consecutive hidden states $P_{\theta^{(m-1)}}(s_t = u, s_{t+1} = v \mid x_1^n)$ from which follows $P_{\theta^{(m-1)}}(s_t = v \mid x_1^n)$. These probabilities are computed using the Baum-Welch forward-backward recurrence. A new value $\theta^{(m)}$ is obtained in the M-step which increases the likelihood :

$$a^{(m}(u,v) \quad = \quad \frac{\sum_{t=1}^{n-1} P_{\theta^{(m-1)}}(s_t = u, s_{t+1} = v \mid x_1^n)}{\sum_{t=1}^{n-1} P_{\theta^{(m-1)}}(s_t = u \mid x_1^n)}$$

$$b_v^{(m)}(x;w) \quad = \quad \frac{\sum_{t=r+1}^{n} P_{\theta^{(m-1)}}(s_t = v \mid x_1^n) 1_{\{x_t = x, x_{t-r}^{t-1} = w\}}}{\sum_{t=r+1}^{n} P_{\theta^{(m-1)}}(s_t = v \mid x_1^n) 1_{\{x_{t-r}^{t-1} = w\}}}$$

where $1_{\{...\}}$ is equal to 1 if the sentence between the brackets is true and 0 otherwise. E and M steps are alternated until we come to an iteration M for which numerical convergence is reached.

Every limit point of a sequence $(\theta^{(m)})_{m \geq 0}$, generated by EM, satisfies the log-likelihood equations, and $(\theta^{(m)})_{m \geq 0}$ converges towards the maximum likelihood estimator, if the starting point $\theta^{(0)}$ is not too far from the true value of the parameter θ (see [Mur97]). This is why we run EM with multiple random initializations, and then select the final result presenting the highest likelihood. Computational cost of the algorithm is proportional to sequence length and to the square of the number of states. Obviously cost also grows proportionally with the number of required iterations which depends on the smoothness of the likelihood landscape. Memory requirement is proportional to sequence length and the number of states, but approximations of the E-step could be done to bypass this problem ; they were implemented but not used here.

In order to identify homogeneous segments in x_1^n, probabilities of each hidden state were computed at each position $P_{\theta^{(M)}}(s_t = u \mid x_1^n)$, using the forward-backward recurrence with the maximum likelihood estimator $\theta^{(M)}$. We did not use the popular Viterbi algorithm [KMH94, BK97, HSF97, LB98] which consists, given the sequence, in computing the most probable path of the hidden states. In the case of our poorly structured model Viterbi reconstruction is less informative than the one obtained by forward-backward recurrence. Nevertheless, for results interpretation and discussion, we will need to recognize segments. Hence contiguous positions having v as the most probable hidden state (i.e. where v maximizes $P_{\theta^{(M)}}(s_t = u \mid x_1^n)$) are identified as a homogeneous segment of class v. In the *M1-Mr* model, all

the homogeneous segments of type v are characterized by the same $r+1$-nucleotide composition $b_v^{(M)}(x;w)$.

3.2.2 Processing the chromosome

Results were obtained with the software RHOM (Research of HOMogeneous regions in DNA sequences), C++ sources are freely available for UNIX/Linux, at http ://www-mig.jouy.inra.fr/ssb/rhom/. RHOM implements the algorithms needed to estimate the parameters of a *M1-Mk* hidden Markov model and to produce a segmentation in the way presented in the previous section. Concerning the model, the user only chooses the number of hidden states and the length of the oligonucleotides taken into account. Different model orders, $0 \leq r \leq 3$, and different hidden state numbers, $2 \leq q \leq 8$, were used. All models were fitted to the whole sequence of the *B. subtilis* chromosome (4.2 Mbp) through likelihood maximization. To give an idea of the computational cost, processing the chromosome according to the 5 states *M1-Mk* model with 25 different start points, requires around 24 h and 550 MB of active memory on a SUN-SPARC 400MHz.

RHOM produces a graphical display of the estimated hidden state probabilities for all sequence positions. In our case, we relied on a more sophisticated graphical presentation of the results : chromosome contigs were viewed as "featured DNA physical maps", using appropriate graphical symbols for existing annotations of the sequence. Hidden state probabilities for each position were superimposed on the map, enabling a precise interpretation. Sequences and annotations were taken from MICADO (http ://locus.jouy.inra.fr/micado), a relational database dedicated to microbial genomes [BSB97], containing a translation of EMBL/GenBank sequence records.

Segment borders found with RHOM were compared with gene annotation coordinates. The different segment types were compared to previously described codon usage classes of *B. subtilis* coding sequences [MRD99]. As previously found in *Escherichia coli* [MRV+91], codon usage classes of *B. subtilis* are linked to biological characteristics of the genes : class I contains the majority of the genes, class II is enriched with genes that belong to translational processes, intermediate metabolism, and other highly expressed genes, and finally, class III corresponds to genes with properties of horizontally transferred sequences. Attempts to correlate RHOM heterogeneities with finer functional classes of genes were made using the metabolic classification of *B. subtilis* gene products, given in the publication of the complete genome [KOM+97], and maintained on the SubtiList WWW server (http ://genolist.pasteur.fr/SubtiList/). For this purpose, each gene was given the type of the homogeneous segment spanning the largest part of it, usually all the gene, or a high proportion of it.

Another kind of heterogeneity we have been looking for is related to gene

transfer. In our opinion, a strong assumption that a DNA sequence segment arises from a horizontal gene transfer relies on the simultaneous occurence of three features :

1. It has a singular oligonucleotide composition, compared to the context of the *B. subtilis* chromosome.
2. It bears genes with functions known to be transferred between bacterial species, such as pathogenicity and resistance factors.
3. It is surrounded by repeated sequences, or large intergenic regions (the "grey holes"), revealing probable chromosomal rearrangements.

For all atypical composition segments revealed by the statistical analysis, homologies for genes with an unknown function were systematically searched for. This was done, by protein homology searching against the *nr* non-redundant protein database at the NCBI using BLAST [AMS+97]. In this publication we report only highly significant similarities (i.e. when the expectation value is less than 1.10^{-10}). To detect repeated sequences, we produced dot plots of the segments, and compared them to repetitions revealed by systematic search [RDV99a].

3.3 Results

3.3.1 Program behavior

An interesting segmentation of the *B. subtilis* chromosome is obtained with the *M1-M2* model. Segments are long and coincide with genes or groups of them. In contrast, *M1-M0* and *M1-M1* models give very short segments of a few base pairs, which do not appear related to biological features. Thus, hidden state probabilities plotted along the sequence, give intermingled profiles. *M1-M2* and higher order Markov models integrate short range heterogeneities in each segment type, so that the hidden state chain can fit long range heterogeneities. Therefore the *M1-M2* model is a good choice to perform chromosome segmentation. Higher order models do not seem to significantly modify the results, while the number of parameters increases geometrically.

With the two state *M1-M2* model, hidden states fit gene orientation. With the three state *M1-M2* model, we typically get two states matching gene orientations (sensitivity : 86.48%, specificity : 90.60% at the nucleotide level), and the third one matching intergenic regions. Such a strong observation may be an indication of the appropriateness of a three state model for segmenting the chromosome. In term of oligonucleotide composition, coding strands are a+g-rich whereas intergenic regions are a+t-rich. Actually, intergenic regions, computed according to GenBank annotation, have an $a+t$-content of 63.2 %, and coding strands have an $a+g$ content of 54.0 %, in comparison to 56.4 % $a+t$-content and 49.9 % $a+g$-content for the whole chromosome. In this context, some genes, systematically found

associated to the state matching the intergenic regions, appear as atypical. More generally, genes were assigned to the class corresponding to their main hidden state in terms of base pairs.

3.3.2 Searching for gene transfers

In its search for three states RHOM was able to detect a first level of heterogeneity revealing previously identified prophages, and other DNA segments potentially arising from horizontal transfer, both containing genes that we call atypical (for details, see below). Atypical genes belong to the so-called $a+t$-rich type ($a+t$-content 66.0 %) which also contains most of the intergenic regions. These genes have a highly heterogeneous distribution along the chromosome, with a peak at the replication terminus, as shown in Figure 5-a.

These $a+t$-rich regions, contain 539 genes, of which 68 % have unknown functions, compared to 42 % for the complete genome. Here, the term unknown function means genes similar to unknown proteins or without similarities, taken from the functional classification of the bacterium. Genes without any similarity represent 56%, in comparison to 26 % for the complete genome. A cross comparison with the codon usage classification reveals that 80 % of the genes in the $a+t$-rich type belong to class III (specificity), while genes of class III represent 13 % of the *B. subtilis* genes. On the other hand, 81 % of these genes belong to the atypical state (sensitivity). Thus, there is a strong correspondance between our atypical regions and the codon usage class III genes.

Prophage detection Literature reports ten prophages integrated into the *B. subtilis* chromosome. Seven are putative, or prophage-like sequences *P1*–*P7* [KOM$^+$97], since their identification is only based on $a+t$-composition, and all these "prophages" are $a+t$-rich. There is experimental evidence for the three other prophages : *PBSX* [WDDM90], *skin* [TMS$^+$95], and *SP*β [ZKRH77]. In this context, the ability of the program to detect experimentally identified prophages and prophage-like sequences provides a biological validation of our approach.

The RHOM software was able to detect all these prophages, except *PBSX*. The latter is not detectable by RHOM because its content is too close to the local *B. subtilis* DNA composition, nor is it detectable by simple $a+t$-content analysis. At least two distinct explanations may be provided for the non-detection of *PBSX*, the first being that it was integrated a long time ago, resulting in the adaptation of its DNA composition to the host context ; the second being that its DNA composition was originally close to that of *B. subtilis*.

Our detection of *SP*β and of the three prophage-like sequences *P1*, *P3* and *P7*, is in keeping with the literature. For the remaining prophage-like

Functions	*HMM*	*Repeats*
P1 "prophage"	202 - 220	202 - 213
P2 "prophage"	529 - 570	555 - 567
see Table 2	570 - 600	—
P3 "prophage"	651 - 664	—
site-specific recombinase	738 - 747	—
multidrug-efflux transporter	818 - 822	—
—	1124 - 1130	—
P4 "prophage"	1262 - 1270	—
PBSX prophage (1320 - 1348)	—	—
—	1397 - 1399	1385 - 1424
—	1442 - 1447	—
—	1478 - 1482	—
P5 "prophage"	1879 - 1891	—
—	2038 - 2041	—
P6 "prophage"	2046 - 2073	2050 - 2060
SPβ prophage	2151 - 2286	—
skin prophage	2652 - 2701	2654 - 2701
P7 "prophage"	2707 - 2756	2725 - 2735
competence	3253 - 3257	—
arsenic resistance regul.	3463 - 3467	3462 - 3469
—	—	3608 - 3634
cell wall synthesis	3658 - 3685	3665 - 3672
ABC transporter	4123 - 4134	—
ABC transporter	4171 - 4176	4170 - 4176
streptothricin, tetracycline, mercury regul.	4184 - 4190	4189 - 4190

TAB. 3.1 – Coordinates (kbp) of potential horizontal transfer regions on the chromosome of *B. subtilis*. The *Functions* column indicates, either prophage and prophage-like elements, as mentioned in [KOM$^+$97], or identified functions and homologies. The *Repeats* column provides the positions of long repeats described by Rocha *et al.*[RDV99a].

sequences, the detection works, although some boundaries are located differently, or some of their genes remain in the types associated to *B. subtilis*. These differences are probably related to a better segmentation accuracy using HMM models than that produced by 10 kbp sliding windows with a 5 kbp step used in the calculation of the $a+t$-content [KOM$^+$97].

New detection of gene transfers In a similar manner to prophage detection, we found fourteen DNA segments identified as $a+t$-rich, and thus potentially arising from horizontal transfer. These segments are presented in Table 3.1 together with the prophages. In addition to this compositional observation, some other signs exist which strengthen our conclusions.

As reported in Table 3.1, there is a high correlation between detected segment locations and repeats described in Rocha *et al.* [RDV99a]. In particular, only one of these repeats was found not to be associated to the detection of some atypical genes. For example, the 3463-3467 kbp segment (Figure 3.1) is flanked on both sides by long direct repeats which probably

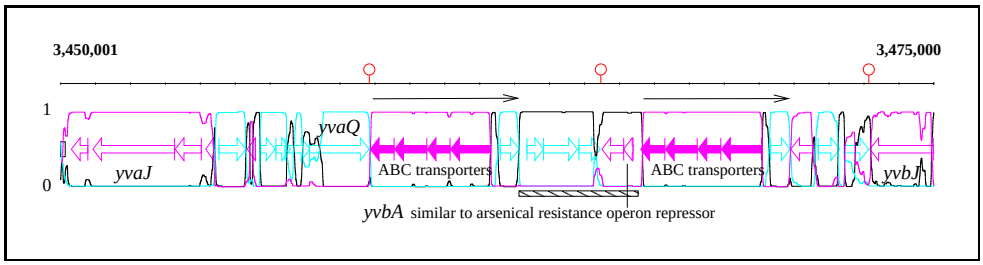

FIG. 3.1 – Detection using a three state *M1-M2* model of an atypical segment (3463-3467 kbp, underlined) surrounded by ABC transporter gene duplication (thin black arrows). Segment reconstruction on 25 kbp is shown. At each position, probabilities $P_{\theta(M)}(s_t = u \mid x_1^n)$, $u = 1, 2, 3$ (colour curves) are plotted on the DNA featured physical map. Filled arrows represent genes of known function, empty ones, those of unknown function, and red hairpins represent transcriptional terminators. The magenta state matches genes on the (+) strand whereas cyan denotes genes on the (-) strand. The black state ($a + t$-rich) fits either intergenic regions or atypical genes.

signal a chromosomal rearrangement. These duplicated fragments are larger than the transferred segment and contain five genes, four of which belong to the ABC transporter family. In this case, the putative transfer contains an unknown gene similar to an arsenical resistance operon repressor (*yvbA*). Figure 3.2 shows another kind of repetition, associated with the 4184-4190 kbp region of the $a + t$-rich type. These sequences are short, aproximately 100 bp long, and repeated four times. These repeats are regularly spaced, but not correlated with gene borders, and therefore, do not present the characteristics of an integron. Nevertheless, the four repeats surround four resistance related genes, two of them are experimentally proven to confer tetracycline resistance (*tetB*, *tetL*), and the other two are similar respectively to streptothrycin acetyl-transferase (*yyaR*), and to the mercuric resistance operon regulator (*yyaN*).

The 818-822, 1442-1447, and 4171-4176 kbp segments are surrounded by intergenic regions larger than usual, compared to sizes expected in bacterial chromosomes. Another segment located at 3658-3685 kbp contains "grey holes". This segment, including teichoic acid metabolism genes, is described in the literature as potentially arising from horizontal gene transfer [LMS+95] but the segment detected by the program is larger because it encompasses other genes also coding enzymes involved in cell wall synthesis.

The occurrence of genes having imprecisely known function in these $a+t$-rich regions, whose homologies are related to resistance functions, reinforces the hypothesis of gene transfer events. In addition to those previously mentioned in the 3463-3467 and 4184-4190 segments, we found a homology to a

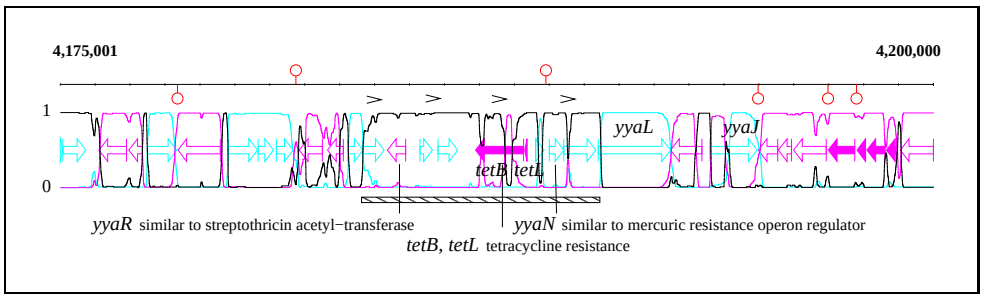

FIG. 3.2 – Detection of an atypical segment (4184-4190 kbp) using a three state *M1-M2* model, containing the tetracyclin resistance, and four direct repeats each approximately 100 bp in length, shown by small black brackets.

Genes	*Homologies*
ydeL	transcriptional regulator MocR (GntR family)
ydeP	cinnamoyl ester hydrolase
ydeQ	general stress protein 14 of *B. subtilis*
ydeR	antibiotic resistance translocase
ydeS	transcriptional regulator (TetR/AcrR family)
ydeT	transcriptional regulator (ArsR family)
ydfA	arsenical pump membrane protein
ydfB	antibiotic resistance protein
ydfC	permease
ydfD	transcriptional regulator MocR (GntR family)
ydfF	transcriptional regulator (ArsR family)
ydfH	nitrates/nitrites sensor protein
ydfI	nitrates/nitrites sensor protein
ydfJ	antibiotic transport-associated protein
ydfK	putative transport protein
nap	naproxen carboxylesterase (experimental evidence)
ydfL	multidrug-efflux transcriptional regulator
ydfM	cation efflux system cobalt-zinc-cadmium
ydfN	nitroreductase
ydfO	ABC transporter
ydfQ	thioredoxin

TAB. 3.2 – Homologies found in the 30 kbp long atypical 570-600 kbp region, downstream of the P2 prophage. Homologies have been found using BLAST against the non-redundant protein database at the NCBI and only highly significant homologies are reported (with E-value $< 10^{-10}$).

multidrug-efflux transporter in 818-822 (*yfmI*), and many other significant homologies in the largest newly detected 570-600 region. Remarkably, this 30 kbp segment, adjacent to the P2 "prophage", bears genes with numerous homologies, either related to resistance functions, or to *mocR*, the rhizopine catabolism regulator of *Sinorhizobium meliloti*. Rhizopine is a compound found in root nodules resulting from plant-bacteria symbiosis. All homologies of this segment are reported in Table 3.2.

Finally, a new potential mobile element was found, located between 738 and 747 kbp. Genes *yefB* and *yefC*, belonging to this segment, are homologous to site-specific recombinases. Moreover, *yeeA* shows similarity with a DNA modification methyltransferase suggesting the presence of a restriction-modification system. These systems are known to be often horizontally transferred [KNKTU99].

3.3.3 Heterogeneities and functional classes

After identifying the coding strand of genes and atypical segments related to horizontal transfer as being the main heterogeneities, our aim was to find additional ones that can be linked to significant biological features. For example, can we find other kinds of horizontal transfer that are not $a+t$-rich ? We thus fitted hidden Markov models with more than three states to the chromosome. As a consequence of incrementally adding one more state, the heterogeneity detected is not completely reorganised, but on the contrary, refined : one of the previously identified states is split. This enables us to present the results in a tree structure as shown in Figure 3.3. In the following section, we describe the new hidden states in the order in which they appear.

Hydrophobic proteins With the introduction of the fourth and fifth hidden states into the model, we distinguish a minority class of coding sequences, borne by both DNA strands of the chromosome. Compared to the usual $a+g$-rich coding strand composition, this new class is t enriched (31.9 % *versus* 25.7 % computed from annotations for the coding strand), and a depleted (21.8 % *versus* 29.9 %), in fact it appears as 56.4 % $t+g$-rich. When compared with the functional classification, these two hidden states are found to be strongly enriched in genes annotated as coding for Transport/Binding Proteins (TBP). These TBP coding genes represent 9.5 % of *B. subtilis* genes, 58 % of which belong to the minority class of coding sequences, while TBP genes constitute 36 % of the minority class.

The TBP category contains a lot of hydrophobic membrane proteins. Therefore, we looked for amino acid biased composition. Actually, we noticed that proteins belonging to this minority state present differences, compared to the majority state. Their amino acid composition is enriched in : Phenylalanine (frequency 0.072 *vs.* 0.039) for which codons, when ordered by *B. subtilis* preference [MRD99], are *ttt*, *ttc*; Isoleucine (0.096 *vs.* 0.068) *att*, *atc*,

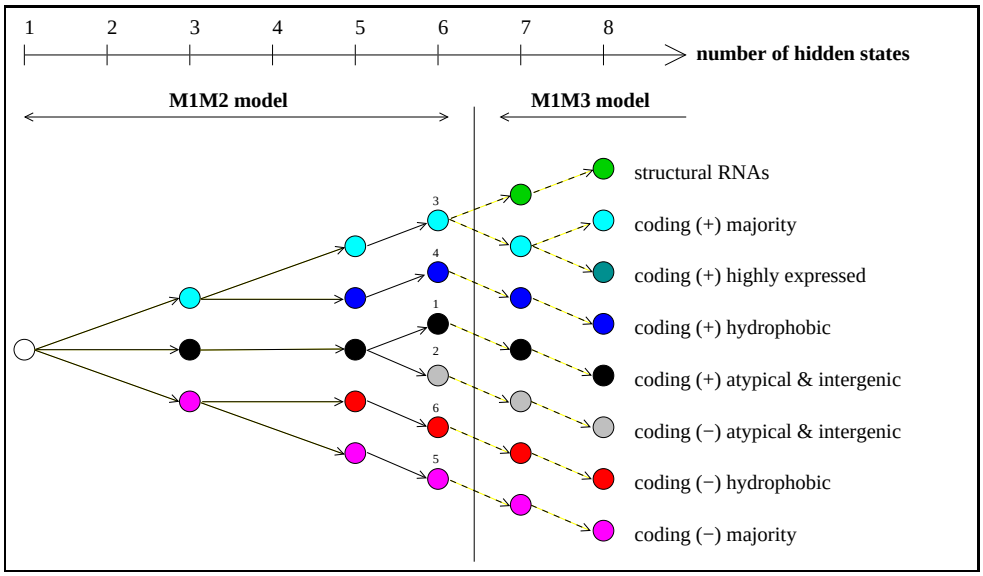

FIG. 3.3 – Tree representation of state subdivisions as a function of the number of hidden states. Numeric labels for the six state HMM correspond to those used in Figures 3.4 and 3.5. The symbols (+) and (-) indicate the coding strand.

ata; and Leucine (0.13 *vs.* 0.091) *ctg*, *ctt*, *tta*, *ttg*, *ctc*, *cta*. Their composition is simultaneously depleted in : Glutamate (0.031 *vs.* 0.080) *gaa*, *gag*; Aspartate (0.025 *vs.* 0.057) *gat*, *gac*; and Lysine (0.045 *vs.* 0.073) *aaa*, *aag*. These amino acid biases of the minority class respectively correspond to a t nucleotide/hydrophobic amino acid enrichment, and an a nucleotide/charged amino acid depletion. Thus these two states appear to be associated with genes which code hydrophobic proteins.

A sixth state leads to the separation of atypical $a+t$-rich coding sequences according to their transcriptional direction. Thus sensitivity of coding sense detection reaches 98.35 % (previously 86.48 %) but specificity drops to 85.35% since intergenic regions are counted together with atypical coding sequences. We investigated the trinucleotide composition of the segments obtained with a six state *M1-M2* model, by principal component analysis (see Figure 3.4). The first three axes explain 77 % of the total inertia. The first axis divides the cloud according to coding sense, the second axis distinguish $a+t$-rich segments (atypical coding and intergenic) from the others, while the third axis separates hydrophic coding sequences. The display of the distribution of the segments according to their associated type along the chromosome (Figure 3.5) reports the asymmetrical distribution of coding sequences between leading and lagging strands even for the atypical $a+t$-rich segments.

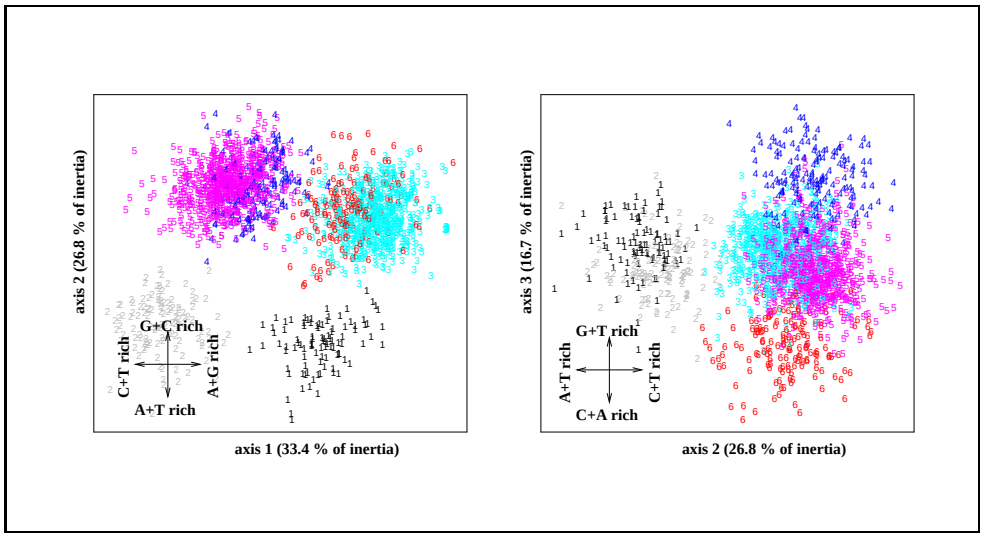

FIG. 3.4 – Principal Component Analysis on trinucleotide composition of segments (*M1-M2*, 6 states). The $a+t$-rich intergenic and atypical coding (+) and (-) senses are labelled 1 and 2. Labels 3 and 5 correspond to the (+) and (-) majority coding, while 4 and 6 are associated to the (+) and (-) hydrophobic coding states. The crosses display main compositional trends of the principal axes. Hidden state mononucleotide compositions are : 1-(a : 0.36, t : 0.28, c : 0.15, g : 0.21) ; 2-(a : 0.30, t : 0.37, c : 0.20, g : 0.14) ; 3-(a : 0.30, t : 0.24, c : 0.21, g : 0.25) ; 4-(a : 0.22, t : 0.32, c : 0.22, g : 0.24) ; 5-(a : 0.24, t : 0.30, c : 0.25, g : 0.21) ; 6-(a : 0.32, t : 0.22, c : 0.24, g : 0.22).

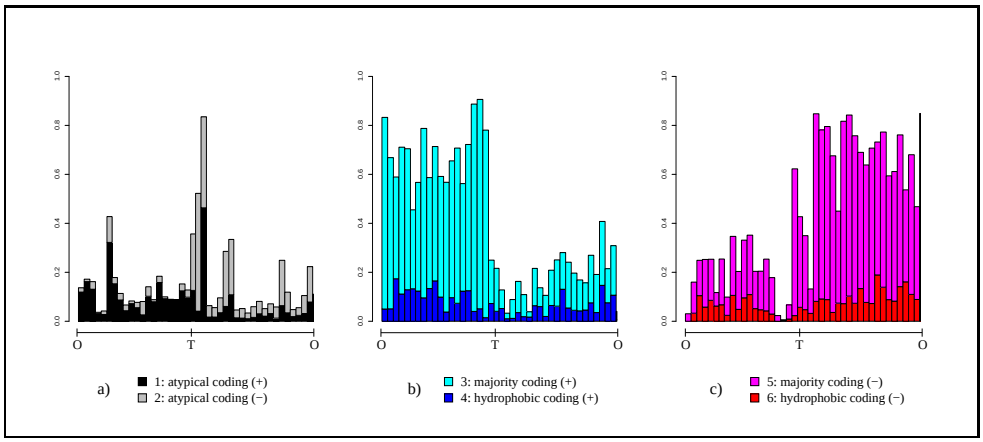

FIG. 3.5 – State repartition along the chromosome. Proportion of coding sequences of each hidden state in a sliding window of 100 kbp, *B. subtilis* chromosome is 4,215 kbp long, O is the replication origin, T the replication terminus at 2,017 kbp. b) and c) show asymmetrical repartition of coding sequence preferentially located on leading strand. This is already true for atypical coding sequences displayed in a), which are moreover highly concentrated near the replication terminus.

RNA genes Introducing a seventh hidden state requires increasing the model order from *M1-M2* to *M1-M3*. Otherwise, two hidden states remain melted, giving very short segments and intermingled probability profiles, as described in the program behavior section for *M1-M0* and *M1-M1* models. The newly identified composition type corresponds to structural RNA genes. In terms of base pairs, these genes cover 1.28 % of the chromosome. We identified structural RNA genes with a sensitivity and a specificity which, at the nucleotide level, are respectively 96.6 % and 90.8 %. At the gene level, we found all rRNA genes and 78 out of the 88 annotated tRNA genes. Thus the correspondance of this compositional type to structural RNA is highly accurate.

Highly expressed genes An eighth state extracts a subcategory from majority coding sequences from the (+) strand (the one given in the sequence file, in opposition to its reverse complement). This subcategory contains most of the codon usage class II genes, which have been characterized as highly expressed. Genes of class II on the (+) strand cover 2.5 % of the chromosome, 92 % are found in this new state, but they only represent 44 % of the subcategory which might interestingly extend class II. It is well known that such highly expressed proteins exhibit greater codon biases than others [GG82], producing a different statistical composition from the

remaining genes in bacteria (see [Mak96] for a review on high-level expression strategies in *E. coli*). They are especially concentrated within the 118-156 kbp region close to the replication origin, where most of them are related to transcription and translation : ribosomal proteins, RNA polymerase α and β subunits, translation initiation and elongation factors.

The highly expressed genes in this region were already detected with a two state *M1-M2* model fitted to the 100-200 kbp segment of the chromosome, while we had to go up to eight states when running on the complete chromosome. This exemplifies the interest and complementarity of using both local and complete genome analyses.

3.4 Discussion

3.4.1 General trends of nucleotide compositions

To summarize, general trends are : an $a+t$-enrichment for intergenic regions and those we identified as resulting from gene transfer ; a coding strand enrichment in $a+g$ (purine) for the majority class of genes, as opposed to a coding strand enriched in $t+g$ (keto) for a minority class of genes corresponding to those coding hydrophobic proteins. The trends are mixed, for instance, we were able to distinguish transcriptional orientation of $a+t$-rich horizontally transferred genes, according to their $a+g$-content.

The gc-skew $(N_g - N_c)/(N_g + N_c)$ and the at-skew are positive on the replication leading strand in this species, as previously described in [Lob96]. These skews are linked to a+g-enrichment of coding strands, due to a high preference in *B. subtilis* for encoding of proteins on the leading strand (Figure 5-b,c). These compositional biases violate the second Chargaff rule. This empirical statement assumes that the relations $N_a = N_t$ and $N_g = N_c$ are not only valid on double stranded DNA but also on each of the strands, if the sequence is long enough. Thus, two violation levels are clearly observed due to asymmetrical evolutionary pressure : the first is related to transcription (kilobase scale), the second to replication (megabase scale). Whereas some work noticed that results of asymmetric pressure related to the replication is observable (in particular [RDV99b, TC00]), our study does not notice asymmetric bias which could not be due to the transcription probably because of its low magnitude. Hypotheses explaining strand asymmetry compositions as results of mutation or selection pressures have been extensively discussed by Frank and Lobry [FL99].

Whereas the positive gc-skew of the leading strand is a general characteristic of bacterial genomes, a positive at-skew is more exceptional, and depends on the codon base position referred to [Lob96, MWD98]. Moreover, the preferred choice of a versus t at the third codon position is not the same for all the encoded amino acids [Sue99].

3.4.2 Gene transfers

Concordance between atypical nucleotide composition, occurence of repetitions, and presence of genes related to resistances or unknown functions, makes the horizontally transferred origin of the detected regions very likely. Overrepresentation of genes of unknown function within these regions could then be due to adaptive characters of these genes, which are only expressed in natural soil surroundings of the bacteria, and not required by them in the laboratory.

All prophages, plus fourteen other segments we identified, are $a+t$-rich in comparison to the chromosome composition. We cannot explain this general property, as we expected nucleotide composition heterogeneity among these horizontal transfers due to their supposed diverse origins. This is intriguing, particulary since $a+t$-enrichment of horizontal transfers seems common in genomes presenting compositional heterogeneity, including *E. coli*. However, we know of some cases where horizontal transfer is $g+c$-richer than the host genome, see for example the mu-like prophage integrated in the *Haemophilus influenzae* genome [FAW+95] or the *E. coli yagH* gene [GVPR99].

The widespread hypothesis is to explain the compositional heterogeneity of a genome as a snapshot of transferred DNA fragments progressively adapting to the host composition [LO98]. These transfers having originally a distinct nucleotide composition related to the source organism. Assuming this assumption is correct, horizontal transfers should generally come from source organisms which are $a+t$-richer than the host.

According to the analysis results on *E. coli* and *Salmonella* sequences of Syvanen [Syv94], an alternative interpretation may be suggested, whereby horizontal transfers essentially occur among strains of the same or close species. In this case, we can imagine a pool of adaptive genes that are shared by recurrent transfers. This may explain the relative homogeneity in $a+t$-richness of the detected fragments, because their shared nucleotide content could then be due to the same evolutionary pressure. Possible origins of the $a+t$-composition bias have been extensively discussed from different angles : not only in relation to gene transfers [Syv94], but also in the interpretation of the interspecific $a+t$-content differences [GL97], of the isochore existence among vertebrates [Ber00], or even of bacterial intraspecific $a+t$-content heterogeneity [Sue99], the latter relating to the context of our study.

3.4.3 Protein hydrophobicity

The $t+g$-richness of hydrophobic proteins appears as one of the main heterogeneity factors at the DNA composition level, coming just after heterogeneities derived from coding properties (coding/non coding, and the transcription direction), and atypical $a+t$-richness related to gene transfers. This is due to the preferential occurrence of t in the second codon position

of hydrophobic amino acids.

Influence of amino acid hydrophobicity at the nucleotide composition level has been previously considered [FL99], because global hydrophobicity has been shown as a main factor for protein variation in amino acid content [LG94]. We were surprised about the importance of this phenomenon on nucleotide composition, even without introducing any frame consideration. It could be interesting to take this heterogeneity into account to improve gene detection based on Markov models, as is done for genes of codon usage class III[BMK+95, LB98].

3.5 Perspectives

Fitting parameters of a hidden Markov model to the oligonucleotide composition of the chromosome, by likelihood maximization through the EM algorithm, leads to a segmentation correlated to biological features of the DNA sequence. It is somewhat remarkable that such structures could be identified without having to specify a window length, or any learning set. From here, the mathematical challenge would be to choose adequate selection criteria of the Markov order and of the number of states that define the model structure.

Initially considered as a tool for detecting horizontal gene transfers, this approach enables one to reveal many more heterogeneities, mainly linked to characteristics of coding sequences. On the basis of these results, it would be interesting to extend the model to phased sequences using a hidden Markov model which changes the hidden state periodically according to the codon position. These models allows for a more realistic representation of coding sequences. One promising feature of using such a model is to enable the combination of gene detection and heterogeneity description, in a similar manner to [BLB01]. In addition, focusing the analysis on the heterogeneity of intergenic regions appears to be very promising. To summarize, our general goal is to produce a finer integrated description of the structure of the chromosome, in terms of the statistical composition of biological features.

Chapitre 4

Accurate bacterial gene finding with self-training hidden Markov models : prediction of very short genes within the *Bacillus/Clostridium* group

Pierre Nicolas, François Rodolphe, Florence Muri, Anne-Sophie Tocquet, Mark Hoebeke, Kevin Bryson, Bernard Prum and Philippe Bessières

Abstract : We present a new, freely available, hidden Markov model based program for bacterial gene finding called SHOW. It performs rigorous parameter estimation without a training set of known genes. The following features are novel : modelling within gene change of composition, probabilistic scoring of every prediction, and prediction of multiple translation start sites if necessary. Results are in keeping with the best programs. We focus on one of the main remaining problem concerning bacterial CDS finding, the detection of short genes. We show that currently, their annotation is essentially a matter of policy choice, because their short sequences make their prediction uncertain. Thus, we intentionally use a moderate level of specificity to predict a set of candidate genes shorter than fifty amino-acids in the *Bacillus/Clostridium* group. Pairwise comparisons at the amino-acid level, followed by inspection of within codon mutations at the DNA level, allows us to identify new probable short genes and pseudogenes. In *Bacillus subtilis*, we predict thirty biologically unknown probable short genes, whereas the number of biologically known ones is about twenty. These results may be an important step toward a better understanding of the strange world of short protein coding genes.

4.1 Introduction

Two separate approaches are usually distinguished for the detection of coding sequences *in silico*. Extrinsic approaches bring together algorithms which are based on the use of multiple related sequences. Despite the number of genomes that have yet been sequenced, even in bacteria these methods face the problem that an important number of genes are without known homologs. On the contrary, intrinsic approaches only use the sequence under study as a source of information. The simplest example consists of looking for open reading frames (ORFs) of significant length. More sophisticated methods are mainly based on local sequence composition. Among algorithms based on nucleotide composition, it is worth distinguishing between algorithms using sliding windows such as GenMark [BM93], algorithms based on open reading frame (ORF) classification such as Glimmer [SDKW98], and algorithms based on hidden Markov models (HMMs) such as EcoParse [KMH94] or GeneMark.hmm [LB98]. Algorithms using HMMs are among the most effective, probably because HMMs allow the most realistic sequence modelling. Sliding window based algorithms face problems when trying to predict accurate gene limits. ORF classification algorithms have problems deciding whether ORFs are coding or not, which is not an easy statistical decision because a non-coding ORF could be a mixture of intergenic regions and coding sequences in other reading frames.

During the 90's, the advent of the sequencing projects made necessary the development of automatic gene detection algorithms. With the rapidly growing number of microbial genomes now sequenced, there is still a need for more sensitive and accurate gene prediction methods. In numerous cases, genome sequencing of an organism is preliminary to effective molecular biology. Hence annotation has become a routine task often carried out by teams with no gene detection specialists. Because each genome presents peculiar characteristics which greatly influence gene detection, a principle one being the variable g+c-content, self-learning algorithms will be useful. Numerous approaches have already been developed and implemented to perform this task. The automatic construction of training sets to estimate parameters for coding region composition has been carried out by selecting long ORFs [SDKW98], and regions presenting similarities with known proteins [FMMG98, BO99]. Clustering methods were applied to derive multiple models which take into account composition diversity of genes within a genome [BMK+95, HB98]. A heuristic approach to choose the parameters of coding and non-coding composition models from the global g+c-content of the genome has also been proposed [BB99]. Self-training of the composition models by iterative use of a gene detection program with refinement of the model at each iteration was used within GeneMark [HIHB97] and GeneMark.hmm (GeneMarkS program) [BLB01].

We present a HMM based approach including rigorous statistical maxi-

mum likelihood estimation of all the parameters of the model, without the need for a training set of known genes. An interface has been implemented which permits sophisticated model setting. The model setting facilities combined with automatic learning of parameters enable the user to adopt an interactive approach, and to refine the HMM for gene detection. Other novel features include computation of theoretical probabilities that a predicted gene is a "true positive" (in a similar manner to GenScan) and segmentation based on the forward-backward algorithm instead of the Viterbi algorithm. Choosing the prediction threshold for this probability allows one to tune the sensitivity/specificity ratio.

Besemer *et al.* [BLB01] emphasized the importance of correctly predicting the start position of the coding sequence (CDS) during annotation. Here, in a similar manner, we show that annotation of short genes remains largely a matter of annotation policy. Little is known about short genes, but some are known to have a key role (as reviewed in a special issue of *Peptides* journal [BK01]), particularly in regulation processes. Their prediction is made particularly difficult since little compositional information is available due to their short lengths. Even experimentally, they can be harder to detect than other genes [BHB97]. On the basis of the low number of similarities found between annotated short genes, Skovgaard *et al.* [SJB+01] concluded that short genes have been over-annotated in many genomes. Of course similarity at the amino-acid level between putative short genes does not prove the coding nature of the sequences — protein sequence similarity can simply be due to DNA sequence similarity, all the more likely if the sequences compared are evolutionary close. That is why Ochman [Och02] has proposed a criterion based on the ratio between the number of synonymous and non synonymous mutations along an alignment to distinguish true short genes by way of comparative genomics. However, this criterion is difficult to use properly [Law03]. Moreover, this method depends on a first annotation of the short genes in one of the genomes.

Our approach permits adjustment of the ratio between sensitivity and selectivity, which enables us to propose lots of candidate short genes. By systematic comparison of these intrinsic based predictions performed over multiple genomes, we try to confirm some of them in the second part of the paper. We present an easy to use criterion which reveals evolutionary preservation pressure at the protein level. Here we focus on genes shorter than fifty amino-acids, and on genomes of gram-positive bacteria belonging to the *Bacillus/Clostridium* group. The choice of this phylogenetical group takes into account the fact that gram-positive bacteria are known to use mRNA encoded peptides as pheromones.

Bacteria	**RefSeq**	**Publication**	**g+c-content**	**Length**	**nb of CDS**	**$\leq$ 100 aa**	**$\leq$ 50 aa**		**Upstream ATG**	
E. coli K12	NC_000913	1997	50.8%	4.6 Mbp	4279	392	42	0.98%	289	6.75%
EcoGene dataset					839	58	3	0.36%	151	18.0%
B. subtilis	NC_000964	1997	42.1%	4.2 Mbp	4112	497	58	1.41%	787	19.1%
B. halodurans	NC_002570	2000	43.4%	4.2 Mbp	4066	524	123	3.03%	888	21.8%
L. innocua	NC_003212	2001	37.4%	3.0 Mbp	2981	301	13	0.44%	462	15.5%
L. monocytogenes	NC_003210	2001	38.0%	2.9 Mbp	2855	245	8	0.28%	443	15.5%
S. pneumoniae TIGR4	NC_003028	2001	39.7%	2.2 Mbp	2094	399	156	7.45%	196	9.36%
S. pneumoniae R6	NC_003098	2001	39.7%	2.0 Mbp	2043	330	35	1.71%	48	2.35%
S. pyogenes	NC_002737	2001	38.5%	1.9 Mbp	1697	192	15	0.88%	160	9.43%
L. lactis	NC_002662	2001	35.3%	2.4 Mbp	2267	257	13	0.49%	184	8.12%
C. perfringens	NC_003366	2001	28.6%	3.0 Mbp	2660	253	15	0.56%	262	9.85%
C. acetobutylicum	NC_003030	2001	30.9%	3.9 Mbp	3672	415	28	0.76%	451	12.3%
S. aureus Mu50	NC_002758	2001	32.9%	2.9 Mbp	2714	359	56	2.06%	419	15.4%
S. aureus N135	NC_002745	2001	32.8%	2.8 Mbp	2594	308	41	1.58%	413	15.9%
M. tuberculosis H37Rv	NC_000962	1998	65.6%	4.4 Mbp	3927	263	8	0.20%	700	17.8%
M. tuberculosis CDC1551	NC_002755	2001	65.6%	4.4 Mbp	4187	568	147	3.51%	570	13.6%

TAB. 4.1 – Some characteristics of the genome sequences and annotations included in this study. Column "RefSeq" refers to the RefSeq accession numbers of the Genbank sequence files, column "Publication" contains the publication year of the paper describing the annotation, column "Upstream ATG" contains the proportion of the annotated CDSs having an in frame ATG upstream of their annotated start position.

4.2 Materials and methods

4.2.1 Biological sequences and annotations

Data set We based our study on 14 annotated genomes of gram-positive bacteria completed before August 2002, and on the genome of *Escherichia coli* K12.

In order to build-up a study set of genome sequences related to the gram-positive model *Bacillus subtilis*, we focused on the *Bacillus/Clostridium* group. This group, also known as the low-g+c gram-positive group, represents 12 of the 15 genomes included in our analysis : *B. subtilis* [KOM+97], *Bacillus halodurans* [TNT+00], *Listeria innocua* [GFB+01], *Listeria monocytogenes* [GFB+01], *Streptococcus pneumoniae* TIGR4 [TNP+01], *Streptococcus pneumoniae* R6 [HAA+01], *Streptococcus pyogenes* M1 GAS [FMA+01], *Lactococcus lactis* [BWM+01], *Clostridium perfringens* [SOH+02], *Clostridium acetobutylicum* [NBO+01], *Staphylococcus aureus* Mu50 [KOU+01] and *Staphylococcus aureus* N135 [KOU+01]. The sequenced genomes of *Mycoplasma* species were not included because of their atypical features compared to other genomes of *Bacillus/Clostridium* group, including their small size and alternative genetic code.

Our analysis included the model bacteria *E. coli* strain K12 [BPB+97] since it has the greatest amount of experimental data available. In particular, N-terminal sequences of 844 proteins have been experimentally determined, and are easily available through the EcoGene database interface [Rud00], providing their CDS coordinates on the genome sequence of K12. According to coordinates on the genome sequence, five of the CDSs do not have lengths wich are multiples of three. Because of sequencing errors or natural frameshifts, these 5 CDSs do not correspond to single ORFs on the genome sequence [Rud00]. In this study, we use only the remaining 839 proteins from the EcoGene dataset which correspond to single ORFs.

Genome sequences of two *Mycobacterium tuberculosis* strains H37Rv [CBP+98], and CDC1551 [FAE+02] were included in our analysis. They are interesting because their g+c-rich composition makes long non coding ORFs more frequent, and therefore gene detection harder.

Annotations policies Table 4.1 summarizes some information about the set of sequences and annotations used in this study. In order to describe annotations, this table looks at two different aspects of bacterial genome sequence annotation.

As emphasized by Besemer *et al.* [BLB01], the number of CDSs which have no in frame ATG between their annotated start and the next upstream in frame stop codon reveals differences between annotation policies. Their reasoning takes into account the main start codon (ATG) and the 3 stop codons (TGA, TAG, TAA). They assume equal probabilities of occurrence

of these and hence there should be 1/4 probability of finding an ATG when looking for the first in frame ATG or stop codon upstream of a true start codon. Thus, the number of genes having an ATG upstream of their annotated start, is expected to be close to 25%. This estimation, although based on simple assumptions, could be considered as a reference value, and justifies examining this quantity when comparing annotations. Table 4.1 shows a great variation between genomes in the proportion of genes having an ATG upstream of their annotated start. This varies from 2.35% for *S. pneumoniae* R6 to 21.8% for *B. halodurans*. Some of these variations are clearly due to annotation policy — for instance this proportion is 9.35% for *S. pneumoniae* TIGR4, much more than for *S. pneumoniae* R6. Interestingly, the proportion is 18.0% in the verified N-terminal EcoGene dataset. This value, compared to 6.75% in GenBank annotations, points out a bias in the GenBank annotations of *E. coli* towards CDSs which are too long.

Another divergence between annotation policies concerns short genes, and the proportion they represent in the set of all annotated genes. Table 4.1 shows that the proportion of genes shorter than fifty amino-acids varies from 0.20% in *M. tuberculosis* H37Rv to 7.45% in *S. pneumoniae* TIGR4. As is the case for the ATG statistics given above, some of these differences can only be explained by differences in the annotation policies. For instance, the proportion of genes shorter than 50 aa is more than 4 times higher in *S. pneumoniae* TIGR4 than in *S. pneumoniae* R6, and more than 17 times higher in the *M. tuberculosis* CDC1551 than in the *M. tuberculosis* H37Rv genome.

4.2.2 HMMs for gene detection

How sequences are modelled by HMMs Hidden Markov Models have numerous applications in very different fields, but our aim here is only to present them in connection with gene detection.

A DNA sequence is viewed as a series of successive segments, each one belonging to one out of a finite set of categories. These categories will represent, for instance, coding sequences, RNAs, intergenic sequences, etc. Each of these segment categories has its own statistical composition in terms of oligonucleotides. This structure is represented in HMMs by two different processes. The first process is the hidden process. It consists of a series of labels called hidden states. A run of the same hidden state indicates a segment belonging to the corresponding category. This process is unknown, and its reconstruction is the main objective. The second process is the observed process, consisting of the observed nucleotide sequence. At each position in the sequence, the probabilities of the four nucleotides occurring depends upon both the previous observed sequence and also the current hidden state. Hence, oligonucleotide segment composition depends on the segment category.

To be more precise, we must give a mathematical definition of these pro-

cesses. The hidden process is always a first order Markov chain on the set of hidden states. The fact it is a Markov chain introduces a correlation between successive hidden states, and enables one to vary mean segment lengths. The observed process is, conditionally on the hidden one, a heterogeneous Markov chain, whose characteristics (order, transition probabilities) depend on the hidden state. Hence, local composition depends on segment category and constitutes the information which is used to guess the original segmentation.

Parameters of a HMM (later considered as coordinates of a parameter vector θ) are the transition probabilities between hidden states a, and the observation emission probabilities given the current hidden state b. Let the hidden state at position t be denoted S_t, the observed nucleotide at position t be denoted by X_t, and the sequence of nucleotides from position $t1$ to $t2$ be denoted X_{t1}^{t2}. Then a contains the probabilities $a(u,v) = P(S_{t+1} = v \mid S_t = u)$ for each couple of hidden states (u,v). b contains $b_u(x;w) = P(X_t = x \mid S_t = u, X_{t-r_u}^{t-1} = w)$ for each hidden state u, nucleotide x, and nucleotide context w of length r_u, which is the Markov-order associated to state u.

Gene detection based on HMMs HMMs are very well suited for texture description. But, to be reliable, gene detection must also take signals into account. Signals and textures are represented in the HMMs we construct by series of hidden states connected by a graph of admissible transitions. We will now describe how we model the bacterial chromosome sequence, and explain how parameters are estimated.

Gene detection model settings Figure 4.1 represents the main components of the model.

Intergenic sequences are modelled using a single back looped hidden state, emitting the observed DNA sequence according to a second order Markov chain.

Coding sequences have a three-periodic composition "core", represented by a cycle of three hidden states, corresponding to the three positions within a codon. Each of these hidden states emits nucleotides according to a second order Markov chain. In frame stop codons are prevented by a zero probability of emitting a stop codon in the third state of the cycle. In order to ensure that start and stop codons delimit CDSs, appropriate sub-models are added upstream and downstream of the core of the CDS.

Heterogeneities of coding sequence compositions have been shown to be related to expression gene level [GG82] and to horizontal transfers [MRV+91]. Taking into account the atypical a+t-rich composition of some horizontally transferred genes has been shown to greatly improve gene detection sensitivity [BMK+95]. Protein hydrophobicity [LG94] has also been shown to be one of the most important factors of the chromosome heterogeneity in terms of nucleotide composition of coding sequences [NBM+02]. In order to

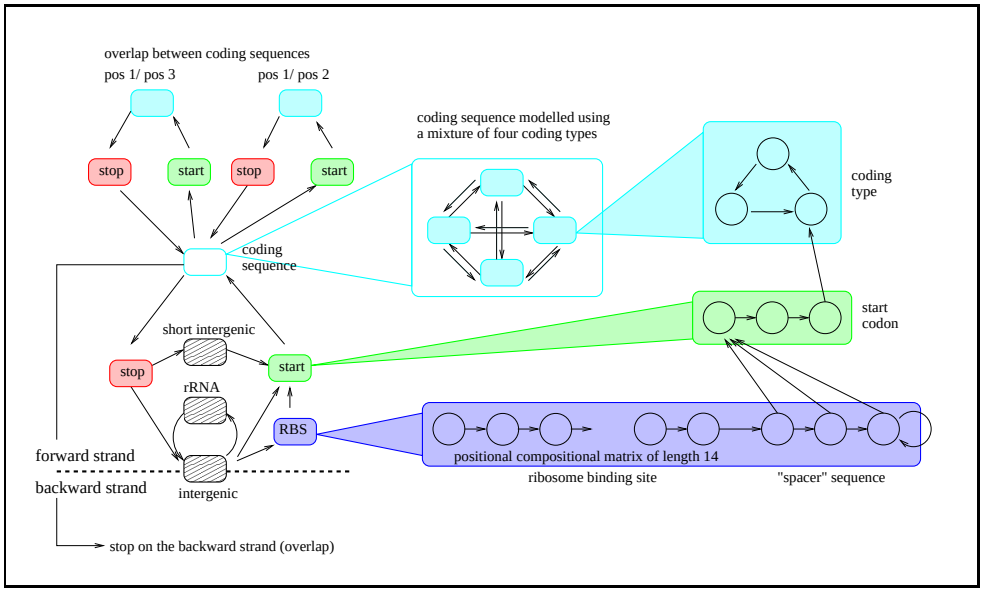

FIG. 4.1 – Graph of the main features of the HMM that we use for gene detection. The left part of this graph displays the biological features which are modelled (rounded rectangles). The right part displays details about how some of these features are modelled, where circles represent the actual individual hidden states of the HMM.

take these features into account, different sub-models corresponding to distinct coding types are reachable downstream from a start codon. Moreover, composition type can change within genes. This is more general than using disjoint gene types, and also more realistic, in particular with regard to the existence of hydrophobic regions in proteins.

Overlaps between CDSs are an important feature of the bacterial genome organisation (see [LB98] for the distribution of CDS overlap lengths). Our model distinguishes the very frequent short overlaps of 1, 2 or 4 nucleotides from the rare longer overlaps. Composition of long overlaps is modelled in the same way as non overlapping CDSs. Because not so much data are available in a single genome for their composition estimation, we use hidden states of Markov order 1. In order to prevent in frame stop codons we used a pseudo-second order model, corresponding to a Markov model of order 1 conditional on the absence of stop codons.

Taking into account the ribosome binding site (RBS) position has been shown to improve precise start site prediction [LB98, BLB01, SESS01], this might also help to predict short genes where CDS composition does not provide sufficient information. In our model, RBS sequences are modelled by a positional compositional matrix of Markov order 1 and length 14. The RBS is followed by a "spacer" sequence of minimal length 1.

Structural RNA composition is different from intergenic composition. In particular, it has been shown to be related to environmental conditions for the organism, such as temperature [GL97]. In order to prevent prediction of CDSs within rRNA genes, and to improve the estimates for intergenic parameters, we added a state corresponding to rRNA texture. Parameters of the composition associated to this state are not estimated but computed over the annotated structural RNAs.

Genes are present on both strands, and therefore are read on both the direct and complementary strands. With minor modifications, the complementary strand model can be derived from the direct strand one. A full description containing each of the hidden states of our model can be found in the software documentation.

Self-training : Parameter estimation Our software implements the EM algorithm, also known as the Baum-Welch algorithm (see [Rab89, DEKM98] for tutorials in the HMM framework). The EM algorithm has already been used to estimate HMMs for the segmentation of DNA sequences into homogeneous regions [Chu89, PG99, NBM+02].

The EM algorithm is an iterative procedure that alternates between two steps : the so-called E-step (for Expectation) and M-step (for Maximization). Given the current value of the parameter $\theta^{(m-1)}$, the E-step consists of computing, for each position in the sequence, the probability, conditional on the observed sequence X_1^n, of couples of hidden states $P_{\theta^{(m-1)}}(S_t =$

$u, S_{t+1} = v \mid X_1^n)$, from which follows $P_{\theta^{(m-1)}}(S_t = u \mid X_1^n)$. It is performed using a dynamic programming procedure often called the forward-backward algorithm. The M-step computes the new values of the parameter $\theta^{(m)}$ using Equation 4.1 for $a^{(m)}$ and Equation 4.2 for $b^{(m)}$:

$$a^{(m)}(u,v) = \frac{\sum_t P_{\theta^{(m-1)}}(S_t = u, S_{t+1} = v \mid X_1^n)}{\sum_t P_{\theta^{(m-1)}}(S_t = u \mid X_1^n)} \tag{4.1}$$

$$b_u^{(m)}(x;w) = \frac{\sum_t 1\{X_{t-r_u}^{t-1} = w, X_t = x\} P_{\theta^{(m-1)}}(S_t = u \mid X_1^n)}{\sum_t 1\{X_{t-r_u}^{t-1} = w\} P_{\theta^{(m-1)}}(S_t = u \mid X_1^n)} \tag{4.2}$$

where $1\{X_{t-r_u}^{t-1} = w\} = 1$ if $X_{t-r_u}^{t-1} = w$, and 0 elsewhere. This procedure ensures the increase of the likelihood at each iteration m : $P_{\theta^{(m)}}(X_1^n) \geq P_{\theta^{(m-1)}}(X_1^n)$.

The EM algorithm has been shown to converge toward the maximum likelihood provided the starting point is near enough from the true value. The major drawback of this algorithm is that it can be trapped in local maxima of the likelihood function. In order to circumvent this problem, we used multiple random parameter initial values, and after convergence, we selected the best parameter set with regard to likelihood. Moreover, we performed a two-stage estimation of the model. In the first stage, a relatively simple model is estimated which does not contain RBS and CDS overlaps states. This first estimation is followed by a second stage, where the estimated parameters are used as initial values for the estimation of the complete model.

In order to favour symmetry of parameters for both coding directions, and to make the estimation of poorly represented states more accurate, some parameters have been tied. In particular, those concerning frequencies of the start codons and overlap frequencies in the distinct coding classes. A more complete description of the estimation procedure can be found in our software documentation.

Sequence parsing to predict CDSs and RBSs We use the probabilities $P_\theta(S_t = u \mid X_1^n)$ computed by the forward-backward algorithm to predict the modelled biological features along the sequence. It allows us to determine for each ORF the probability of being a CDS (equal to the probability of being a stop codon at the end of the ORF) and the probability of each start site for multiple start positions (equal to the probability of being a start codon at each possible position). Then, we choose a probability threshold beyond which we predict CDSs, which we denote the "CDS prediction probability threshold". For each predicted CDS, the best start (with the highest probability) is predicted. Other possible start sites are also reported if the ratio

between their probability and the probability of the best start is higher than a chosen "multiple start prediction threshold" set to 0.1 here. In a similar manner, RBS predictions are provided with probabilities of being in the RBS hidden state.

4.2.3 Searching clues to confirm short gene predictions

We used SSEARCH [Pea91] with default parameters, with an E-value threshold of 0.01, to compare each of the predicted short genes ($\leq$ 50 aa) to four different sets of putative genes. Choice of the SSEARCH program is based on its good performances [Pea98, BCH98] compared to BLAST [AMS+97] or FASTA [PL88]. It uses the exact Smith-Waterman alignment algorithm, thus it does not miss any matches due to the use of a heuristic. Moreover, its computation of the E-value has been shown to be accurate, even concerning relatively high E-values of the order of 0.01.

Set 1 : the GenBank annotated genes. This set consists of the GenBank annotated genes for the 15 genomes. This enables us to find similarities with genes of every length, whereas the following comparison sets concern only short genes. Thus, it allows us to identify probable pseudogenes as predicted genes presenting similarities with parts of longer genes.

Set 2 : predicted short genes of similar length to the query length. Here we use the set of translations of all predicted genes found by our program with length between the length of the query plus/minus 10 aa. This set may allow us to discover similarities between unannotated short genes. Moreover, its relative small size could allow similarities to be found which may not be significant within a larger set consisting of all predicted genes.

Set 3 : all predicted genes shorter than 100 aa. This comparison set includes all predicted genes of length less than 100 aa. Motivation for the use of this set is to find similarities between unannotated short genes, even if they are of different sizes.

Set 4 : short genes belonging to a conserved gene organization. Our aim is to find poorly conserved short genes by their synteny with a conserved gene. Given a putative predicted short gene (the query), and its two annotated adjacent genes, the first step consists of looking in the 15 studied genomes for annotated genes similar to the annotated ones of the query. We used BLAST with default parameters and an E-value threshold of 0.0001. The second step consists of selecting, within the 15 genomes, all predicted short genes which are adjacent to one of the conserved genes found during the first step, with the same gene organisation (position, strand) as

the query and also its corresponding adjacent annotated gene. Moreover, we keep only predicted short genes of similar length to the query (± 10 aa). The PRSS program provides a shuffle based p-value associated with the alignment of a single pair of amino-acid sequences [Pea96]. In order to obtain an E-value for the query alignment containing each putative short gene with the same genic organization, the p-value is multiplied by the number of single pair comparisons done with the query.

Assessment of the coding nature of a similarity found at the amino-acid level In order to confirm the coding nature of a couple of putative short genes found by similarity, we set-up a statistical test allowing us to reject the hypothesis of a homogeneous mutation rate between the three codon positions along the alignment of the putative CDSs. Within a coding region, the number of mutations at the third codon position is typically the greatest, while the number of mutations at the second position is the smallest. Therefore, we restricted our region of rejection to the case where the observed number of mutations at the third position is greater than the number of mutations at the second position.

Let $(m1, m2, m3)$ be the number of mismatches in first, second, and third codon positions respectively, calculated along the alignment of N amino-acids between the two putative CDS protein sequences. If $m3 \geq m2$, we compute $p' = P(m1, m2, m3 \mid N, m)$, the probability of $(m1, m2, m3)$ conditional on the total number of mismatches $m = m1 + m2 + m3$, under the hypothesis of a homogeneous mutation rate between the three codon positions, and independence between mutation sites. In particular, we assume that a mutation at a given position does not modify the probability of a mutation at an adjacent position. The ratio between the combinations possible with and without the constraints on codon position mismatches gives us the required probability :

$$p' = \frac{{}^{N}\mathrm{C}_{m1}\,{}^{N}\mathrm{C}_{m2}\,{}^{N}\mathrm{C}_{m3}}{{}^{3N}\mathrm{C}_{m}}.$$

We use this quantity p' as the test statistic. The p-value of our test p is equal to the probability, under the hypothesis of homogeneous repartition of mismatches, that a repartition of the mismatches among the three codon positions $(x1, x2, x3)$ belongs to the set $S(N, m, p')$, defined as follows. $S(N, m, p')$ corresponds to the set of repartitions with probabilities less than p' :

$$S(N, m, p') = \Big\{ (x1, x2, x3) \in [0 \ldots N]^3 \quad \text{with} \quad x1 + x2 + x3 = m, \quad x3 \geq x2 \quad \text{and} \quad P(x1, x2, x3 \mid N, m) \leq p' \Big\}.$$

We have

$$p = \sum_{(x1,x2,x3) \in S(N,m,p')} P(x1, x2, x3 \mid N, m).$$

Bacteria	Annot. total nb	Pred. total nb	Agreements 3' nb	3' Sn	3' Sp	5' and 3' nb	5' and 3' Sn	5' and 3' mult.	3' discrepancies total +	total −	≤ 100 aa +	≤ 100 aa −	l ≤ 50 aa +	l ≤ 50 aa −
E. coli K12	4279	4213	4095	95.7%	97.2%	3142	73.4%	3435	120	184	87	121	43	36
EcoGene dataset	839		833	99.3%		780	93.0%	820		6		2		1
B. subtilis	4112	4200	4009	97.5%	95.5%	3400	82.7%	3650	191	103	169	81	81	28
B. halodurans	4066	4045	3958	97.3%	97.8%	3574	87.9%	3769	87	108	62	99	25	60
L. innocua	2981	3006	2947	98.9%	98.0%	2700	90.6%	2833	59	32	50	28	19	3
L. monocytogenes	2855	2850	2819	98.7%	98.9%	2599	91.0%	2726	31	35	29	24	21	5
S. pneumoniae TIGR4	2094	2098	1892	90.4%	90.2%	1636	78.1%	1724	206	202	98	186	30	122
S. pneumoniae R6	2043	1980	1920	94.0%	97.0%	1451	71.0%	1555	60	123	53	109	38	19
S. pyogenes	1697	1747	1654	97.5%	94.7%	1455	85.7%	1532	93	43	55	39	25	10
L. lactis	2267	2367	2214	97.7%	93.5%	1937	85.4%	2046	153	53	106	42	40	4
C. perfringens	2660	2693	2638	99.2%	98.0%	2410	90.6%	2465	55	22	55	19	51	3
C. acetobutylicum	3672	3726	3587	97.7%	96.7%	3279	89.3%	3427	139	85	107	77	71	16
S. aureus Mu50	2714	2656	2608	96.1%	98.2%	2304	84.9%	2410	48	106	47	98	32	38
S. aureus N135	2594	2548	2503	96.5%	98.2%	2250	86.7%	2343	45	91	43	81	31	26
M. tuberculosis H37Rv	3927	3994	3767	95.9%	94.3%	2804	71.4%	3240	227	160	173	54	79	7
M. tuberculosis CDC1551	4187	4125	3725	89.0%	90.3%	2446	58.4%	2856	400	462	174	303	96	130

TAB. 4.2 – Comparison between CDS predictions and CDS GenBank annotations. Probability threshold used for prediction was equal to 0.99. The column "Sn" refers to the sensitivity of the method, and "Sp" the specificity of the method. The "mult." column contains the number of genes having their annotated start belonging to the multiple predicted starts. In the right part of the table, "+" refers to the predicted but not annotated CDSs, and "−" to the annotated but not predicted CDSs.

4.3 Results

4.3.1 Program behaviour

To obtain the results presented in this study, we used 10 random starting points for both stages of the parameter estimation. The time required to obtain gene prediction for each of the genome sequences grows approximately linearly with the sequence length. Using a Linux PC with a 1 GHz processor, *B. subtilis* gene prediction needs around 2 days.

Figure 4.2 gives a graphical representation of the gene detection along the 100 Kbp segment of the *B. subtilis* genome between positions $3,400,001$ bp and $3,500,000$ bp. This representation is superimposed on GenBank annotations. For most positions along the sequence, probabilities of being in a coding region is close to either 0 or 1. Hence the prediction of coding regions are generally well defined. Moreover, a unique start codon position is proposed for most of the predicted genes, and this start codon is often preceded by a predicted RBS when intergenic length is sufficient.

Within coding regions, the type of composition associated with each position can be clearly identified. In the case of the *B. subtilis* genome, the four estimated coding composition classes correspond to : (i) the majority of the genes, (ii) part of the genes coding for hydrophobic amino-acids sequence (e.g. ABC transporter genes *fhuG, fhuC*), (iii) atypical a+t-rich genes (e.g. gene *yvaZ* belonging to a putative horizontally transferred region [NBM+02]), (iv) highly expressed genes (e.g. from *gap* to *eno* there are 5 genes encoding some of the glycolysis enzymes). This kind of clustering of coding regions when estimating 4 types of coding sequence seems relatively common but not universal. In particular, in the *M. tuberculosis* genome, one of the coding types is associated with repeated regions shared by proteins of the PE_PGRS family. Some genes are segmented in different coding composition types. Within gene segmentation is generally related to hydrophobic regions within the proteins (e.g. *yvrG* and *yvgP*), but more complex profiles may also appear (e.g. *yvgX* and *yvaJ*).

Finally, we notice the prediction of short unannotated genes on this segment which are subject to further investigation in the last part of this study.

4.3.2 Comparison of predictions with GenBank annotation

The global agreement between predictions and annotation shown in Figure 4.2 is accurately quantified for all genomes in Table 4.2. We use a CDS prediction probability threshold of 0.99 since we observed that this results in a similar number of predicted genes to the number of GenBank annotated genes. In all cases, the difference is less than 100 genes.

The 3' agreement gives the correspondence between the 3'-end of predicted and annotated genes. Considering annotations as the truth, we computed the sensitivity (True Positives / True Positives + False Negatives) and the

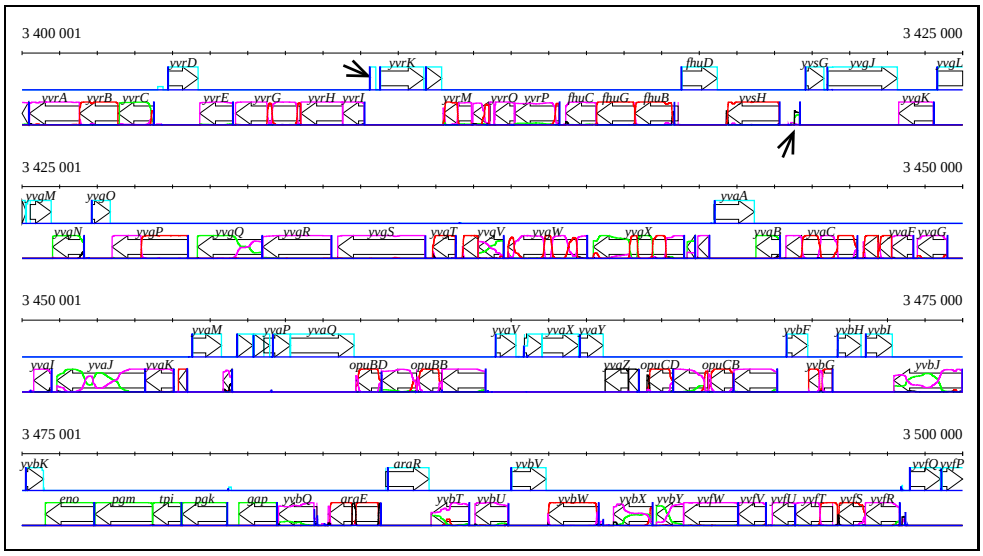

FIG. 4.2 – Coding sequence detection along the 100 Kbp of the *B. subtilis* chromosome. The probabilities of being in a coding region, as computed using the forward-backward algorithm, for each position of the sequence are superimposed on the GenBank file annotation (wide open arrows). For clarity the four coding types are displayed only for genes predicted on the reverse strand. Colors rules for the probability curves are : cyan curve for the coding regions on the direct strand ; magenta ("majority"), red ("hydrophobic protein"), green ("highly expressed") and black ("a+t-rich horizontal gene transfer") for the coding composition types of the reverse strand ; blue for the RBS on the two strands. Thin black arrows point out two short putative genes predicted by the program but not annotated.

specificity (True Positives / True Positives + False Positives). Sensitivity and specificity values are both higher than 95.0% for 9 of the 15 genomes. Sensitivity ranges from 89.0% for *M. tuberculosis* CDC1551, associated with a specificity of 90.3%, to 99.2% for *C. perfringens*, associated with a specificity of 98.0%. It must be noticed that *M. tuberculosis* has the greatest g+c-content (65.6%) and that *C. perfringens* has the lowest g+c-content (28.6%). Thus these differences probably reflect the increasing difficulty of gene detection with g+c-content. However, annotation policy is also involved in the observed differences. For instance, sensitivity of the 3' detection is 95.9% and specificity 94.4% in *M. tuberculosis* H37Rv, which is much higher than in *M. tuberculosis* CDC1551. Another illustrative example is given by the differences between strains of *S. pneumoniae* : sensitivity and specificity values are 90.4% and 90.2% for the TIGR4 strain, but become 94.0% and 97.0% for strain R6.

As shown in Table 4.2, discrepancies between predictions and annotation are largely due to differences between the prediction and annotation of short genes. Considering the *B. subtilis* genome annotation, the total number of predicted but not annotated genes is 191, and the total number of annotated but not predicted genes is 103. Now genes shorter than 50 aa represent only 1.41% of total annotated genes, however among the 191 predicted but unannotated genes, 169 are shorter than 100 aa, and 81 are shorter than 50 aa. Reciprocally, among the 103 annotated but unpredicted genes, 81 are shorter than 100 aa, and 28 are shorter than 50 aa. Similar observations can be made concerning other genome annotations. *M. tuberculosis* CDC1551 is an exception where discrepancies also affect genes longer than 100 aa, but with its genome being g+c-rich, non-coding ORFs longer than 100 aa are much more frequent than in a+t-rich genomes.

5' and 3' agreement means exact correspondence between predicted and annotated genes. Taking annotations as the reference, we compute sensitivity of the prediction. This sensitivity ranges from 58.4% for *M. tuberculosis* CDC1551 to 91.0% for *L. monocytogenes* (being 90.6% for *C. perfringens*). Sensitivity is higher than 80.0% for 10 of the 15 studied genomes. This sensitivity seems partly related to the proportion of genes not having an in frame ATG upstream of the annotated start. The two genome annotations presenting the lowest proportion, *L. lactis* (2.35%) and *E. coli* (6.75%), have respectively the second and fourth lowest sensitivity, 71.0% and 73.4%. However, this proportion is relatively high for *M. tuberculosis* CDC1551 (17.8%), whereas the sensitivity is the lowest. Prediction of multiple start positions shows that in some of the predicted genes the annotated start is not in the first, but in the second, or even in a lower rank. It must be noticed that multiple start prediction is relatively unusual, only 753 of the 4200 predicted genes of the *B. subtilis* genome are concerned, with the mean number of predicted starts over these genes being 2.28. Taking into account all predicted starts, the number of annotated genes in *B. subtilis* which are in complete

agreement grows from 3400 (82.7%) to 3650 (88.8%).

4.3.3 Comparison with experimentally determined *E. coli* start sites

Comparison between predictions and coordinates of 839 proteins with verified N-terminal sequences (EcoGene dataset) are shown in Table 4.2. The 3'-ends of 833 of the 839 CDSs are predicted using the 0.99 probability threshold, sensitivity reaching 99.3% for the EcoGene dataset. The 6 missing genes are : *glpR* (EcoGene id. : EG10400), *priB* (EG10764), *vsr* (EG11068), *rpmJ* (EG11232), *torR* (EG12615) and *rmf* (EG50004). The *glpR*, *torR* and *vsr* genes 3'-ends overlap with 3'-end regions of other genes ("stop near stop" overlaps). The three other missed genes, *priB* (105 aa), *rpmJ* (38 aa), and *rmf* (55 aa) are relatively short ones. Whereas *priB* is missed even with a 0.5 threshold, *rpmJ* and *rmf* are predicted, but with a probability lower than 0.99, respectively 0.82 and 0.83. Predictions agree in 3' and 5' for 780 of the 839 genes. The corresponding sensitivity is 93.0%. Taking into account multiple start predictions with threshold 0.1, 820 of the 839 genes have their verified starts predicted.

It is remarkable that the proportions of genes in the EcoGene dataset having their 3'-ends predicted (99.3%) and exactly predicted (93.0%) is much higher than in the *E. coli* GenBank annotation (95.7% and 73.5%). Moreover, these proportions are the highest of all considered annotations.

4.3.4 Short gene assessment

Uncertainty concerning short gene detection

For *B. subtilis*, 113 genes among the 4200 predicted genes using a 0.99 probability threshold have a length less than or equal to 50 aa. If the probability threshold is brought down to 0.50, the total number of predicted genes increases to 4503, and the number of predicted short genes reaches 335. Thus, the number of predicted short genes increases about threefold, with 73% of the predicted genes in the probability range $[0.5; 0.99]$ being shorter than 50 aa. Together with this threefold increase, the proportion of predicted annotated genes which are shorter than 50 aa increases from 30/58 to 45/58. Figure 4.3 displays the gene prediction probability as a function of the length of the predicted gene. Clearly, the variation in prediction probabilities increases dramatically as the length of the predicted gene decreases. This expresses the uncertainty concerning the detection of short genes.

For each studied genome, the total number of predicted short genes using 0.99, 0.90 and 0.50 thresholds are displayed in the first three columns of Table 4.3, together with the number of these putative short genes which are annotated. With a 0.50 threshold, the number of these putative genes per

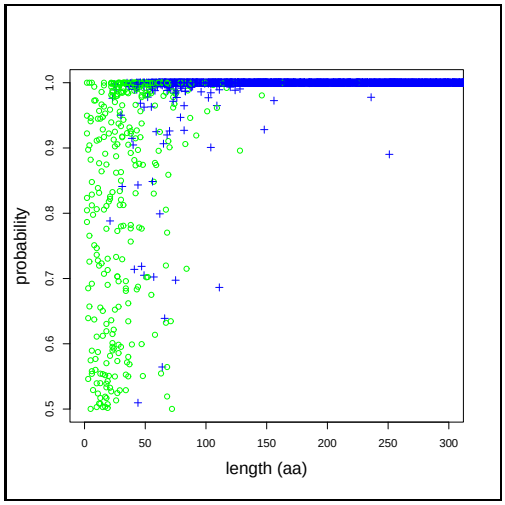

FIG. 4.3 – Plot of the CDS prediction probability against CDS length for the *B. subtilis* genome. Blue crosses indicate the gene is annotated in GenBank, otherwise green circles are used.

Bacteria	HMM predicted short genes 0.99	0.90	0.50	Probable short genes 0.99	0.90	0.50	Probable pseudogenes 0.99	0.90	0.50
E. coli K12	52(9)	97(15)	203(19)	8(2)	11(3)	12(4)	5(1)	7(1)	9(1)
B. subtilis	113(32)	187(41)	335(48)	28(9)	40(12)	42(14)	18(3)	28(4)	35(4)
B. halodurans	88(63)	158(88)	309(101)	27(26)	37(32)	43(34)	4(4)	10(6)	21(7)
L. innocua	30(11)	56(12)	108(13)	13(6)	21(7)	24(7)	2(1)	3(1)	4(2)
L. monocytogenes	25(4)	48(5)	88(6)	9(4)	11(4)	12(4)	1(0)	2(0)	2(0)
S. pne. TIGR4	64(34)	116(45)	199(55)	8(5)	11(7)	15(7)	18(6)	28(8)	43(8)
S. pne. R6	58(20)	111(30)	186(34)	7(3)	10(4)	12(4)	23(13)	36(18)	47(21)
S. pyogenes	31(6)	60(7)	115(11)	7(3)	9(3)	10(3)	7(1)	11(1)	17(1)
L. lactis	50(10)	77(11)	146(13)	8(4)	9(5)	9(5)	5(0)	5(0)	5(0)
C. perfringens	63(12)	90(12)	138(13)	9(5)	10(5)	10(5)	11(1)	11(1)	11(1)
C. acetobutylicum	83(12)	151(16)	266(18)	8(3)	13(4)	14(4)	8(3)	9(3)	13(4)
S. aureus Mu50	49(17)	86(23)	149(26)	16(5)	19(6)	21(7)	2(1)	5(2)	10(2)
S. aureus N135	46(15)	78(17)	144(18)	16(5)	16(5)	21(5)	3(1)	5(1)	8(1)
M. tub. H37Rv	84(5)	151(6)	328(8)	2(0)	2(0)	7(0)	13(3)	19(3)	30(5)
M. tub. CDC1551	129(33)	199(44)	378(71)	4(0)	4(0)	8(1)	30(15)	35(15)	50(19)

TAB. 4.3 – Summary of the results concerning evidence of short genes (≤ 50 aa). The "Probable short genes" and "Probable pseudogenes" are disjoint subsets of the total predicted short genes defined according to our criteria (see text). For each of these three sets, number of predicted short genes having probability greater than 0.99, 0.90 and 0.50 are given. Numbers between parentheses refers to the number of considered genes which are annotated in GenBank.

chromosome ranges from 88 for *L. monocytogenes* to 378 for *M. tuberculosis* CDC1551, being 335 for *B. subtilis*.

Experimentally known *B. subtilis* short genes

Experimental or other strong evidence is given in the literature to support the coding nature of 20 of the 58 annotated short genes.

Three genes *rpmG* (49 aa), *rpmJ* (37 aa), and *rpmH* (44 aa), respectively encode ribosomal proteins L33, L36 and L34. A fourth probable gene *yqgNa* (49 aa) encodes a homolog to the ribosomal protein L33. These four genes are detected by our program with probabilities : 0.71 for *rpmG*, 0.99 for *rpmJ*, 1.00 for *rpmH* and 0.96 for *yqgNa*.

Two genes *cotK* and *cotL* are annotated as encoding spore coat proteins in the GenBank file, but these two products have been described as more probably minor small acid-soluble proteins (SASP) after the complete genome publication [BSS98, CHS00], and thus have also been designated as *sspO* and *sspP*. Whereas the product of *cotK* is experimentally found in acid extracts from spore [BSS98], the *cotL* expression has only been revealed by translational *LacZ* fusion [CHS00]. Five other short unannotated genes have also been found with *cotK* in acid extracts of spore [BSS98] : *sspG* (48 aa), *sspJ* (46 aa), *sspK* (50 aa), *sspL* (42 aa) and *sspM* (34 aa). These six proteins which are experimentally found in acid extracts of spore are detected with probability 1.00, whereas the *cotL* gene is not detected.

Translation of three short annotated genes is known to regulate expression of downstream genes (leader peptides) : *ask* (24 aa), *usd* (36 aa) and *tetL* (20 aa). The *ask* gene is an attenuator of the aspartokinase II gene [CHP87], translation of the *usd* gene is required for translation of *spoIIID* [DMKL97], and *tetL* is the tetracycline leader peptide [SFB98]. None of these three genes are detected using a 0.50 probability threshold.

Products of the three short genes *comS* (46 aa), *spoVM* (26 aa) and *degQ* (46 aa) are identified as regulators. The *comS* gene encodes a protein which regulates genetic competence [DNZ94] through overcoming the *MecA*/*ClpC* inhibition of *comK* [THVD97], the expression of the *comS* gene being controlled by cell density. The expression of the *spoVM* gene, whose product interacts with the ATP-dependent protease FtsH, is required for normal spore cortex and coat synthesis during sporulation stage V [CAL+97]. *DegQ* regulates, by an unknown mechanism, the hyperproduction of extracellular degradative enzymes [YFCH86, MKKR91]. The *comS* gene is not detected, certainly due to its localisation within the long *srfAB* gene. The two other genes, *spoVM* and *degQ*, are detected with probability 0.99 and 1.00.

The product of the *sbo* (43 aa) gene is bacteriocin subtilosin [BTSK85], *sbo* is detected with probability 1.00.

Two short annotated genes *phrA* (44 aa) and *phrC* (40 aa), respectively lying just downstream of the *rapA* and *rapC* genes encoding phosphatases,

have been shown to inhibit their cognate phosphatase (reviewed in [PB01]). Nine other phosphatase gene homologs to *rapA* and *rapC* are found in the *B. subtilis* chromosome, five of them, *rapE*, *rapF*, *rapG*, *rapI*, and *rapK* having potential short annotated *phr* genes downstream from them [KOM$^+$97] : *phrE* (44 aa), *phrF* (39 aa), *phrG* (38 aa), *phrI* (39 aa), and *phrK* (40 aa). The *phrA* gene encodes a precursor of a 5 amino-acid peptide that inhibits *rapA*. The *phrC* gene also encodes the precursor of a 5 amino-acid peptide known as CSF (Competence and Sporulation Factor), which inhibits *rapC*, and probably also *rapB* encoded phosphatases. The *phrA*, *phrC*, *phrE*, *phrF*, *phrG* and *phrK* genes are detected with probability 1.00, 1.00, 0.51, 0.79, 1.00 and 0.90 respectively. The *phrI* gene is not detected using the 0.50 threshold, but this gene is only putative, and its 5'-part overlaps about 66% of its length (77 bp) with the 3'-part of *rapI*.

Detection of new probable short genes

We searched for sequence similarities in order to confirm the existence of new predicted genes with length $\leq$ 50 aa and probability higher than 0.5. Comparison sets 2, 3 and 4, described in the Material and Method section, are constructed using all predictions with probability higher than 0.5.

We considered that a predicted short gene is probably real if a significant similarity match within comparison sets 1, 2 or 4 could be found. A significant match was defined as having an E-value less than 0.01, the similar sequence having the same length as the query $\pm$10 aa, and homogeneous mutation repartition is rejected at a significance level of 5%. Results of all comparisons which produced similarity matches with E-value less than 0.01 are browsable (http ://www-mig.jouy.inra.fr/ssb/SHOW/shortgenes/).

We report in Table 4.3 the number of probable short genes identified in each of the genomes using these criteria. The number of probable short genes ranges from 7 in *M. tuberculosis* H37Rv to 43 in *B. halodurans*, equalling 42 in *B. subtilis*. The two *Bacillus* genomes are clearly distinct compared to other genomes concerning the number of probable short genes found by our similarity criterion. The third ranked genome according to number of probable short genes is *L. innocua* with 24. The main reason for this difference may be technical, the evolutionary distance between both *Bacillus* species is within a range where similarity search is informative concerning our problem. The evolutionary distance is not too long, and thus numerous homologs can be found between both species. Also, the evolutionary distance is not too short, and thus the number of mutations between homologs is sufficient to reject homogeneous mutation rate. As an illustration, for 21 putative *B. subtilis* short genes, the closest similar sequences in set 2 belong to *B. halodurans* and homogeneous mutation rate is rejected for 18 of these genes (data not shown, see web page results). On the contrary, the two *Listeria* genomes are evolutionary too close, for 24 putative short genes of *L.*

Id.	Coordinates			Length	Proba.	Evidence	Comments
830	−	429765	429854	29 aa	1.000	1, 2, 3	*yosA* family
832	−	429936	430049	37 aa	1.000	1, 2, 3	*yosA* family
1008	−	505879	506010	43 aa	1.000	2, 3	*ydbN*, sim. to id. 4100 (41 aa), *BH2296* (48 aa) and *senS* (65 aa)
1934	−	927619	927771	50 aa	1.000	1, 2, 4	sim. to *BH0927* (47 aa) ‖ (*yfhJ*)
2038	−	972365	972454	29 aa	0.997	1, 2	sim. to *BH1024* (29 aa)
2259	+	1070878	1070964	28 aa	1.000	1, 2, 3, 4	*yosA* family ; sim. to *BH1178* (28 aa) ‖ (*prsA*)
2261	+	1071089	1071175	28 aa	1.000	1, 2, 3, 4	*yosA* family ; sim. to *BH1178* (28 aa) ‖ (*prsA*)
2659	+	1252574	1252723	49 aa	1.000	1, 2, 3	*yosA* family
2997	+	1394836	1394973	45 aa	1.000	1, 3	*ykzD* sim. to part of *splA* (79 aa)
3295	−	1524524	1524595	23 aa	0.979	2, 4	sim. to Bhal (25 aa) ‖ (*ykrA*)
3318	+	1533525	1533644	39 aa	0.994	2	sim. to Bhal (41 aa)
4100	+	1929329	1929454	41 aa	1.000	1, 2	sim. to *BH2296* (45 aa) and *BH0880* (48 aa)
4104	+	1929519	1929665	48 aa	1.000	1, 2	sim. to *BH2295* (46 aa)
4363	+	2069138	2069230	30 aa	0.933	2	sim. to a id. 5935 (27 aa) and *BH0344* (27 aa)
4559	−	2145063	2145176	37 aa	1.000	1, 2, 3	*yosA* family
4659	+	2169050	2169169	39 aa	1.000	1, 2, 3	*yosA*
5101	+	2329112	2329264	50 aa	0.980	2	sim. to *BH0926* (51 aa)
5110	−	2332337	2332432	31 aa	0.916	4	sim. to *BH1766* (29 aa) ‖ (*cotD*, *yprB*)
5135	+	2338912	2339016	34 aa	1.000	2, 4	sim. to *BH1705* (38 aa) ‖ (*yppC*)
5482	+	2488070	2488162	30 aa	0.986	2, 4	sim. to Linn (31 aa), Lmon (31 aa) and *BH1467* (28 aa) ‖ (*yqjC*)
5763	−	2609026	2609169	47 aa	0.996	1, 2, 4	sim. to *BH1368* (49 aa) ‖ (*bex*, *yqxN*)
5828	+	2636635	2636769	44 aa	1.000	1, 2, 4	sim. to *BH1336* (45 aa) ‖ (*yqeN*, *comEC*)
5851	−	2647019	2647111	30 aa	1.000	2	sim. to id. 2259 (28 aa)
5935	+	2678075	2678158	27 aa	0.980	2	sim. to id. 4363 (30 aa), *BH0344* (27 aa) and Saur (35 aa)
5973	−	2692726	2692869	47 aa	0.978	2	sim. to Bsub (55 aa)
6126	+	2740038	2740115	25 aa	0.925	1, 2, 3	*yosA* family
7323	−	3298775	3298918	47 aa	0.719	1, 2, 4	*yuiA* ; sim. to *BH3406* ‖ (*yuiB*, *yumB*)
7410	+	3334472	3334603	43 aa	1.000	1, 2, 3	*yosA* family
7466	−	3360033	3360170	45 aa	0.999	1, 3	sim. to *BH3472* (62 aa)
7498	−	3371628	3371774	48 aa	1.000	2	sim. to *Bhal* cotK gene (53 aa)
8681	+	3917668	3917817	49 aa	1.000	1, 2, 3	*ywzA* sim. to SpneTIGR4 (54 aa) and *ydaS* (85 aa)
9052	+	4091316	4091459	47 aa	1.000	1, 2	sim. to Spyo (48 aa), and Saur (272 aa) (probable pseudogene)
9154	−	4134247	4134375	42 aa	1.000	2	sim. to *BH4008* (52 aa)

TAB. 4.4 – New probable short genes found on the *B. subtilis* chromosome. The column 'Evidence' indicates in which data sets similar sequences were found. The column 'Comments' reports : gene names if annotated in GenBank file (underlined) ; names of similar annotated genes in *B. subtilis* or *B. halodurans* ; organisms in which similar sequences were found (four characters abbreviation, omitted for *B. subtilis*, or when gene names are *BH*xxxx indicating *B. halodurans* genes) ; within table cross references use the 'Id.' column ; conserved genes adjacent to the query (Evidence set 4) are indicated after parallel symbol.

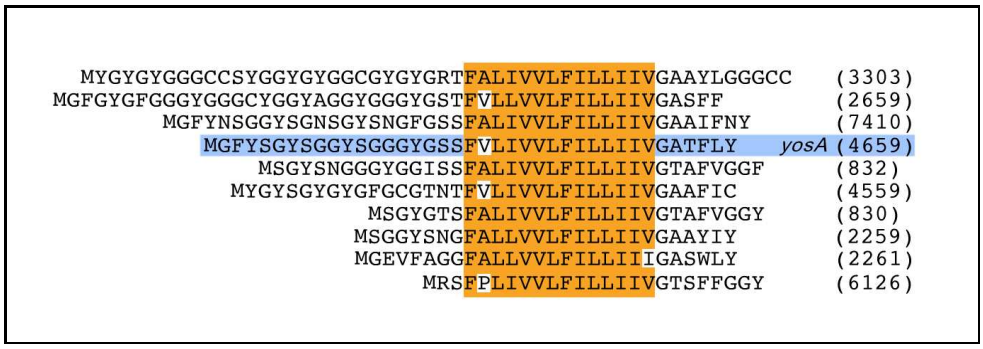

FIG. 4.4 – The *yosA* family of short genes found in the *B. subtilis* chromosome. The first ten protein sequences are arranged by size, they share a well conserved hydrophobic region (orange rectangle, white cell if the amino-acid is distinct from the consensus). Alignment has been done manually by aligning the well conserved hydrophobic regions.

innocua the most significant similar sequences in comparison set 2 belongs to *L. monocytogenes*, but homogeneous mutation rate is rejected only in 5 cases.

Among these sets of 7, 43 and 42 probable short genes, the number of annotated genes is respectively 0, 34 and 14. Hence, the proportion of probable short genes annotated in the GenBank file is highly variable, ranging from 0% for *M. tuberculosis* H37Rv to 79% for *B. halodurans*, and equalling 33% for *B. subtilis*.

Most of the probable short genes have a probability greater than 0.90. For instance in *B.subtilis* 95% of probable short genes have a probability greater than 0.90, whereas only 56% of all the putative short genes predicted by the HMM.

Concerning *B. subtilis*, 28 unannotated probable short genes were identified on the chromosome. Moreover, 34 of the 42 probable genes are not biologically characterized. Table 4.4 reports for each of these genes their location, length, probability, comparison sets where similar sequences were found, and a short comment.

One surprising feature is the existence of a family of short genes containing 10 members having lengths between 52 and 26 aa. Figure 4.4 displays a crude alignment between the proteins in this family. Among this family 9 have a length less than 50 aa and the only one annotated is *yosA*, thus we refer to this family as the *yosA* family. All these genes share a similar organization consisting of three parts : the N-term part, of variable length, is glycine+tyrosine-rich, the central part is highly conserved and clearly hydrophobic, and the C-term part has a variable length. We do not have any idea of the biological function of the genes belonging to *yosA* family. This

family is also found in *B. halodurans*.

We observed that comparison set 4, that is using conserved gene organization, rarely detects significant similarities which are not detected using comparison set 2 (matches of a similar length to the query). Set 4 reveals similarities between *phrA* and *phrC*, and between *phrG* and *phrK*, but as Table 4.4 reveals, there is only a single gene for which significant similarity is only found in set 4 (id. 5110).

Probable pseudogenes

Any predicted gene which did not fully satisfy the criteria to be a probable short gene was considered as a possible pseudogene. A number of additional criteria were also used for pseudogenes including the existence of a similarity in comparison sets 1 or 3 with an E-value less than 0.01 and a length greater than the query +10 aa. Also, homogeneous mutation repartition had to be significant at 5% (set 1 or set 3), or the similar sequence had to have a length greater than 150 aa (set 1). The comparison results enabling the identification of pseudogenes are included on the Web, together with all other statistically significant matches.

The last three columns of Table 4.3 gives the number of probable pseudogenes found in each genome. This number appears to be highly variable. The number of probable pseudogenes is lowest in *L. monocytogenes* (2), and highest in *M. tuberculosis* CDC1551 (50). The large number of putative short genes classified as probable pseudogenes in *M. tuberculosis* CDC1551 is partly due to the existence of numerous unresolved base pairs (175). Thus a number of predicted genes in *M. tuberculosis* CDC1551 have imprecise locations and their amino-acid sequences are found similar to parts of other proteins (often *M. tuberculosis* H37Rv). However, the number of true probable pseudogenes must be close to 30, which is the number found in *M. tuberculosis* H37Rv. The second largest number of probable pseudogenes is found in *S. pneumoniae* R6 (47). Surprisingly, despite the proximity between both *Bacillus* species, we found 35 probable pseudogenes in *B. subtilis*, but only 21 in *B. halodurans*.

4.4 Discussion

Novel features of our gene detection program

The main HMM based prokaryotic gene detection programs are GeneMark.hmm, and its self-training version GeneMarkS [BLB01]. Here we present methodological and model differences between our program and GeneMark.hmm. We also compare our program to other programs when this is relevant.

The first model difference is that GeneMark.hmm uses a generalized hidden Markov model (GHMM, also known as a hidden semi-Markov model). GHMMs are also used for eukaryotic gene detection by some programs such as Genie [KHRE96, RKTH00] and GenScan [BK97]. GHMMs are an interesting extension of HMMs in which hidden state duration distributions can be modelled, whereas in a HMM, they follow geometric distribution. Indeed, very short prokaryotic CDSs and eukaryotic exons seem to be less frequent than longer ones, and their lengths have been modelled by gamma distributions [BK97, LB98]. The main drawback of a GHMM, in comparison with a HMM, is the increase in time-complexity of classical algorithms (Viterbi, forward-backward). Fortunately, ingenious optimization can be performed, by taking into account peculiarities of the model used (see for instance [Bur97]). Thanks to these optimizations, iterative GHMM self-training was implemented in GenMarkS, but the number of iterations is not so important than in our case. Moreover, ingenious model specific optimization makes implementation of effective algorithms in the framework of flexible modelling harder. With regard to our results concerning short gene detection, the choice of a prokaryotic CDS length distribution might also be a difficult task.

The second major difference is the method we employ for gene prediction, which is based on the forward-backward algorithm instead of the Viterbi method which seems to be used almost everywhere else (see for instance [KMH94, BK97, LB98, RKTH00, Kro00]). The Viterbi algorithm finds the segmentation with the highest probability. The forward-backward algorithm computes, at each position, the probability of each hidden state. It has already been used to provide a confidence measure for Viterbi based predictions [BK97]. One of the interests of Viterbi based segmentations is that they conform to the structure described in the model. On the contrary, probabilities computed by the forward-backward algorithm do not immediately provide a segmentation consistent with the model. For instance, if we choose to predict a CDS start position if the probability of being in a start codon at a position is higher than a threshold we could potentially have multiple start codons predicted for the same CDS. Prediction of multiple start codons for the same gene does not conform to the model CDS structure. It is however useful information, not only because it reports the statistical incertitude about the prediction, but also because it proposes potential alternative predictions.

We observed a surprising behaviour of the Viterbi algorithm which encourages us to do both parsing and confidence computations using the forward-backward algorithm. Some of the probabilities computed using forward-backward for CDSs predicted by Viterbi are very low (< 0.1). Such predicted CDSs belong in fact to RNA genes. The probability of the RNA state is much higher (> 0.9) than the probability of the CDS state. However, the "best" segmentation, in the Viterbi sense, predicts a CDS because the RNA state probability was summed over numerous distinct paths, each ha-

ving a low probability ; on the contrary, since a CDS must contain an in frame start-stop couple, the number of such paths is greatly reduced, and individual probabilities increased. This is a relatively harmless example, but it highlights problems which could occur when using Viterbi, in particular when making the model more complex and thus multiplying the number of possible paths (for instance, when allowing changes of texture within genes, and thus multiplying the number of paths when predicting a CDS).

Other distinctive features are allowing texture changes within CDSs, in order to model heterogeneous proteins, for example those displaying parts with "hydrophobic texture". RBSs are also included in the HMM, which permits better start prediction (this is also done in GeneMark.hmm v2.0 [BLB01]). Model flexibility allied with rigorous self-training of parameters, although processor-intensive, is also an interesting distinctive feature. It allows the model designer to carefully estimate model details such as the frequencies of different start codons, textures, RBS consensus and overlap frequencies.

Quality of predictions

Overall, we find good global agreement between predictions and GenBank annotations. Using a 0.99 CDS prediction threshold, our gene prediction method reach a sensitivity of 97.5% and a specificity of 95.5% when compared to *B. subtilis* annotations. These results are slightly better than those of GeneMarkS runned through its Web server : sensitivity 96.6% and specificity 94.3%.

Disagreements between predictions and annotations are restricted to some specific points where annotation and prediction problems still exist. Saying this, it is questionable to benchmark gene detection programs using GenBank annotations.

The use of the experimental data set collected for *E. coli* seems to be presently the only way to precisely evaluate the start site predictions. The best published results for these experimental data were obtained by GeneMarkS with 94.4% correct prediction on a 195 gene sub-set of our 839 gene set. Over the entire 839 genes, GeneMarkS correctly predict 93.1% (781 genes) of the translation start sites. Here, we obtain a very close result : 93.0% (780 genes). Moreover, taking into account multiple start site prediction, 97.7% of these genes have their starts in the predicted set.

Short gene prediction

A point we have especially addressed in this study is the prediction of short genes. We have shown that short gene annotation remains largely a matter of annotation policy. Moreover, discrepancies between predictions and GenBank annotations consist mostly of putative short genes which have

been predicted, but have are not annotated, or which are annotated but have not been predicted. For instance, using a 0.99 probability threshold for *B. subtilis*, 88.5% of the predicted genes which are not annotated are shorter than 100 aa, and reciprocally 78.6% of the annotated genes which are not predicted are less than 100 aa long. This could be explained by the uncertainty of short gene detection, compared to the detection of longer genes. This uncertainty could be due to a number of factors. First, as gene length diminishes, compositional information used for prediction also diminishes, and thus short genes are *a priori* more difficult to predict than longer ones. Secondly, short ORFs looking like genes are expected to occur with a probability greater than longer ones. A third explanation to the number of short genes predicted with low probabilities could be that some fossil genes (pseudogenes) perturb gene detection ; these pseudogenes being not genes but having potentially some characteristics of them. Finally, it must be noticed that intergenic regions are certainly less homogeneous than supposed by the model, even in the absence of pseudogenes, and thus the probabilities assigned to some gene predictions are probably over-estimated by the model.

Some biologically known short genes are predicted with probabilities lower than 0.99, establishing that it is worthwhile searching for other clues to confirm the existence of short genes predicted, even when they have relatively low probabilities. The approach we used provides strong evidence to suppose that there are more than 30 short genes which are biologically unknown. The approach may fail to point out some of the short genes for two reasons. The first reason is that some short genes are not detected by the gene detection program. This concerns typically leader peptides which are not only very short, but also could present highly atypical composition features due to their special function, their translation is itself a regulatory mechanism. The second reason is that for some detected and true short genes, no homologs could be found, or their number of mutations was too low to be able to conclude about their translated nature. On that subject, Lawrence [Law03] has also pointed out that leader peptides have atypical mutation profiles.

In order to understand evolutionary processes, the presence of pseudogenes in bacteria has recently became a subject of interest (see for instance [AA01, HFK$^+$02, Bab03]). We hypothesize that the lower number of probable pseudogenes detected in *B. halodurans* than in *B. subtilis* might be related to the lack of competence of *B. halodurans*.

In *B. subtilis*, a low estimation of the total number of short CDSs must take into account the 30 probable short genes pointed out here, the approximately twenty annotated and biologically known short genes, and the observation that some biologically known short genes are missed by our approach. Thus, in *B. subtilis*, the actual number of short genes is probably close to or higher than sixty, which is roughly the number of annotated ones. This estimation leads, at least concerning genes shorter than 50 aa in the *B. subtilis* genome, to temperate conclusions about the over-annotation of

short genes reported by Skovgaard *et al.* [SJB+01]. However, it must be noticed that the number of these "very" short genes annotated is small, and hence they are unlikely to be the candidates held responsible for global over-annotation. The genes being over-annotated on a global scale are certainly short, but probably longer than the ones we consider here. We expect that experimental studies on the predicted short genes, in particular functional studies of those pointed out as probable, will have an important impact on the understanding of the strange world of the very short genes. Moreover, as for most comparative genomics approaches, the power of short gene detection will probably increases rapidly with the number of sequenced genomes.

Results and tools availability

Complete predictions are available on our Web site :

http ://www-mig.jouy.inra.fr/ssb/SHOW.

The program is distributed totally free of charge under the GPL license. The full availability of our software allows three levels of use : as a fully automatic system, modifications to model settings and estimation strategy, and source code modification. The program can be used for gene detection with incomplete genomes (typically in the course of a sequencing project), by estimating the parameters from the currently available sequences, and allowing it to predict parts of genes.

Chapitre 5

A Reversible Jump Markov Chain Monte Carlo Algorithm for Bacterial Promoter Motifs Discovery

Pierre Nicolas, Anne-Sophie Tocquet, Vincent Miele and Florence Muri

Abstract : Effective probabilistic modeling approaches have been developed to find motifs of biological function in DNA sequences. However, the problem of automated model choice remains largely open and becomes more essential as the number of sequences to be analyzed is constantly increasing. Here we propose a reversible jump Markov chain Monte Carlo algorithm for estimating both parameters and model dimension of a Bayesian hidden semi-Markov model dedicated to bacterial promoter motif discovery. Bacterial promoters are complex motifs composed of two boxes separated by a spacer of variable but constrained length and occurring closed to the protein translation start site. The algorithm allows simultaneous estimations of the width of the boxes, of the support size of the spacer length distribution and of the order of the Markovian model used for the "background" nucleotide composition. The application of this method on three sequence sets points out the good behavior of the algorithm and the biological relevance of the estimated promoter motifs.

5.1 Introduction

Finding motifs involved in biological functions is one of the most important challenge of the nucleic acid and protein sequences analysis. Among the numerous statistical methods developed to carry out this task, two main approaches can be distinguished. The first one is based on searching words or sets of words having a number of occurrences significantly higher than expected, given a "null" model - usually an homogeneous Markov model [RSW00]. Whereas the second one attempts to model motifs in sequences. These two approaches fundamentally differ : while the first approach addresses a test problem, the second one sets a modeling and estimation problem.

Methods based on motif modeling describe individual simple motifs as sets of positions each having an associated probability of occurrence for symbols of the sequence. Probabilistic modeling requires more than motif models : composition of the "background" sequence and motif occurrences along the sequence are also to be modeled. Usual models for the motif occurrence positions include mixture models with identical independent probability of occurrence at each position [LAB$^+$93, LNL95, HETC00] and hidden Markov models (HMMs). HMMs enable both flexible modeling of non-uniform distributions of the occurrences along the sequence and dependence between occurrences of simple motifs [GBEB97, LNL99, JLK$^+$01]. Depending on the level of correlation between simple motif occurrences, it is sometimes preferable to refer to complex or structured motifs. In this work, a complex motif and its surrounding sequence is described using a hidden semi-Markov model (HSMM), which is a generalization of HMM allowing the state duration to be explicitly modeled whereas in HMM, state duration implicitly follows a geometric distribution [BK97, Gué03].

Estimation of motif models have been performed in maximum likelihood and Bayesian frameworks, respectively through Expectation-Maximization (EM) algorithms [LR90, BE95] and Markov chain Monte Carlo (MCMC) algorithms [LWK94, LNL99]. However, the problem of motif model choice remains widely open. The use of the parameters maximum *a posteriori* (MAP) estimate has already been proposed [LNL99]. Recently, Qin *et al.* [QMT$^+$03] have performed inference on posterior model distribution using a MCMC (Gibbs) sampler. This approach enables to determine informative motif positions within a Bayesian mixture framework. We extend their work to more sophisticated models and to the choice of other model components. The alternative approach we use is based on a recent MCMC methodology called reversible jump MCMC in which model dimension and other parameters are simultaneously estimated through a single MCMC sampling. Reversible jump MCMC were introduced by Green [Gre95]; Richardson and Green [RG97] applied this methodology to determine the number of components in a normal mixture model. In the HMM framework, reversible jump MCMC were first used to estimate the number of hidden states associated to centered nor-

mal distributions [RRT00]. Further applications to DNA sequence modeling were developed in order to provide an estimation of the number of hidden states corresponding to distinct DNA composition types [BH02a] and to estimate the order of the Markov chain modeling DNA composition associated to the hidden states [BH02b].

In this work we focus on identifying the central motif of the bacterial promoters : the RNA polymerase binding motifs. RNA polymerase binds to DNA through one of its subunits called the Sigma factor. Different kinds of Sigma factors can be expressed in a same bacteria leading to the transcription of different set of genes. Therefore, identifying motifs responsible for these transcriptional regulations would be an important step toward the interpretation of genomic sequences. Sigma factor binding sites are typical complex motifs : they could generally be described as two boxes separated by a spacer of variable but constrained length [PG01]. For most Sigma factors these boxes are called "-35" and "-10" boxes, where the number is referring to the distance between the box and the DNA transcription start site. To identify these motifs, approaches based on algorithmic extraction of over-represented structured motifs and evaluation of their statistical significance given a "null" Markov model have been developed [MS00, RDR$^+$02]. Nevertheless, most methods are based on motif modeling such as HMM models estimated through EM algorithm [JLK$^+$01, PLU$^+$03], or mixture related model estimated through MCMC [XLBL01, EJC$^+$03]. Latter models also enable dimer motifs separated by a specified gap range to be identified.

The use of the HSMM framework allows us to consider straight-forward modeling to describe the distance between RNA polymerase binding site and translation start site, the width of the two boxes and the distribution of the spacer length between both boxes. Here the dimensions of the HSMM we estimate are : (i) the width of the boxes, (ii) the size of the support of the spacer length distribution and (iii) the Markovian order of the background composition model. Behavior of the reversible jump MCMC algorithm is described using three sets of *Bacillus subtilis* sequences extending upstream from protein coding regions.

5.2 Bayesian model

5.2.1 Hidden Semi-Markov model

HSMMs are composed of an hidden state process (corresponding here to the structure of the promoter region) and an observed one (corresponding here to the observed DNA sequence). The hidden state process is a semi-Markov chain composed of both an embedded first-order Markov chain, describing transitions between states, and of state sojourn time distributions d_u attached to any non absorbing state u. The transition matrix of the states is denoted by a. As in HMM, the distribution b_u for the observed sequence

depends on the hidden state u. Each observed nucleotide takes its value in an alphabet denoted by $\mathcal{X} = \{a, g, c, t\}$ of size $|\mathcal{X}| = 4$.

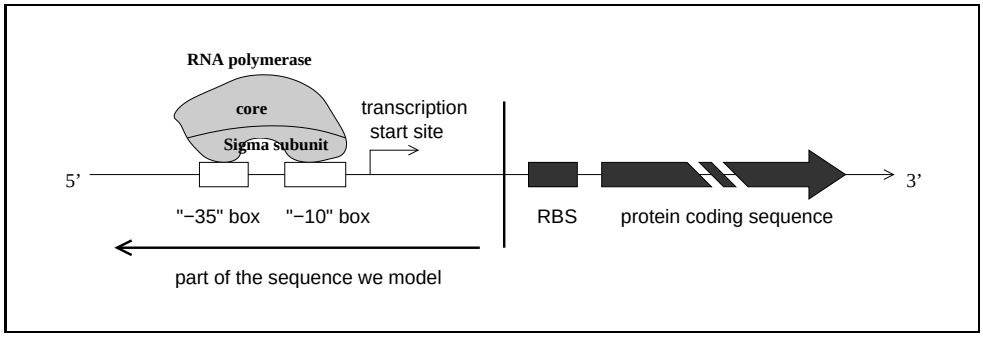

FIG. 5.1 – Schematic representation of the modeled biological sequences.

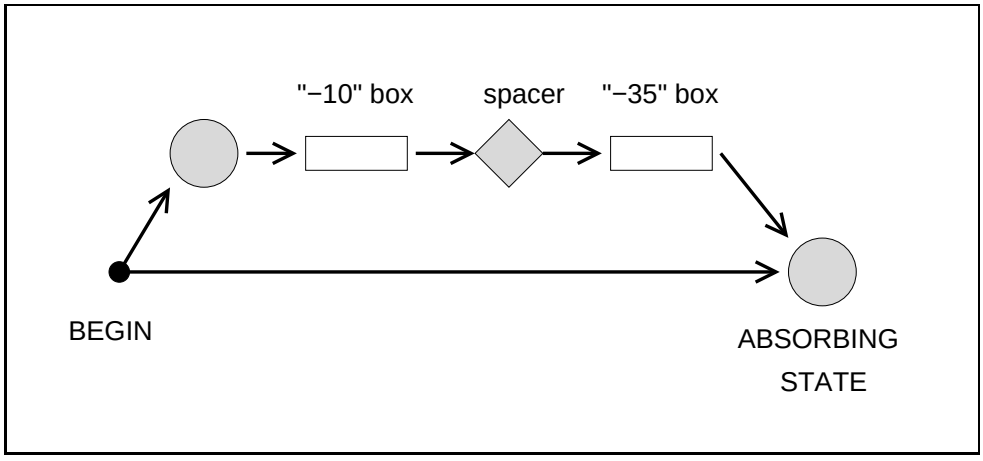

FIG. 5.2 – Structure of the HSMM. Gray filled objects correspond to hidden states associated to the "background" Markovian model. Model with optional "-35" box would allow to skip the "-35" box, adding a direct transition from the "-10" box to the absorbing state.

The promoter region sequence we model is illustrated in Figure 5.1. These sequences are assumed to be independent and to be generated by the same HSMM. Figure 5.2 describes the structure of the models we use in this study. Sequences are modeled in the opposite direction than the one of the transcription process : the "-10" box is the first one to be reached. Each box is modeled by a hidden state with fixed width (assumed not to be known). In each box state u, the probability $b_u(x; k)$ of emitting a nucleotide x depends on the position k inside that box. In every other states, nucleotides are emitted from the same background homogeneous Markovian model of

order $r_u = r$ (assumed not to be known). Nucleotide transitions are denoted by $b_u(x; w)$ for $x \in \mathcal{X}$ and a nucleotide context w of length r_u. A spacer state models the varying distance between both boxes through a multinomial sojourn time distribution whose support is unknown. The two boxes and the spacer describe the complex motif we are interested in. The distance between the beginning of the sequence and the motif is modeled by a shifted negative binomial distribution associated to a pre-motif hidden state. After the motif, the sequence is modeled by an absorbing state. The hidden path starts in a silent "begin" state and go through either the pre-motif state or directly reaches the absorbing state. According to this model, the complex motif occurs no more than once in each sequence. This HSMM can be extended to model additional transcription factor binding sites upstream from the RNA polymerase binding sites.

The shifted negative binomial sojourn time distribution for the pre-motif state is defined as $d_u(k; n_u, p_u, h_u) \propto p_u^{n_u}(1-p_u)^{k-h_u}\mathbb{I}\{k \geq h_u\}$, where $\mathbb{I}\{k \geq h_u\} = 1$ if $k \geq h_u$ and 0 elsewhere. Distributions for the length of the spacer state and the width of the box states are respectively defined as $d_u(k; p_u, h_u, l_u) = p_u(k)\mathbb{I}\{h_u \leq k \leq h_u + l_u\}$ and $d_u(k; h_u) = \mathbb{I}\{k = h_u\}$. As d denotes the sojourn time distributions, it partially includes model dimensions : support size $l_u + 1$ of the multinomial distribution and width h_u of box states.

We denote by $\theta = (a, b, d, r)$ the parameters of our HSMM, X_1^n the set of observed sequences and S_1^n the associated hidden state paths. The complete likelihood could be expressed with intuitive occurrence numbers as

$$P_\theta(X_1^n, S_1^n) = \prod_u a(u)^{N_u} \prod_{u,v} a(u,v)^{N_{uv}}$$
$$\times \prod_u \prod_k d_u(k)^{N_{u,k}} \prod_u \prod_{k_c} D_u(k_c)^{N_{u,k_c}} \prod_u \prod_{x,w} \prod_k b_u(x; w, k)^{N_{wx,k;u}}$$

where : N_u is the number of sequences beginning in state u, N_{uv} is the number of jumps from state u to state v in all sequences, $N_{u,k}$ the number of visits to state u during exactly k, N_{u,k_c} is the number of sequences where the last state u was visited during a censored time k_c, and $N_{wx,k;u}$ is the number of words wx in state u which has been reached a time k ago, $D_u(k) = \sum_{k' \geq k} d_u(k')$ is the survivor function of d_u. For sequence positions t smaller than $r_u + 1$, nucleotides are assumed to be drawn from uniform distributions. Notice that all nucleotide emission probabilities are denoted by $b_u(x; w, k)$ with the convention that $b_u(x; w, k)$ is set to $b_u(x; k)$ for heterogeneous model and to $b_u(x; w)$ for background model.

Given a set of sequences, we aim to estimate θ and simultaneously to predict motif position in each sequence from the joint posterior distribution $\pi(\theta, S_1^n \mid X_1^n)$.

5.2.2 Prior distributions

For Bayesian inference, we need to specify the prior distribution for the model parameters

$$\pi(\theta) = \pi(a)\pi(d)\pi(r)\pi(b \mid r, d).$$

We use independent conjugate Dirichlet priors $D_{|V_u|}(\alpha_u, \ldots, \alpha_u)$ for each row $a(u, \bullet)$ of the matrix a [RCD93], where V_u denotes the set of states reachable from state u. Parameters p_u and n_u of the shifted negative binomial distribution are respectively drawn from conjugate Beta prior $Be(\gamma_{u,1}, \gamma_{u_2})$ and continuous uniform on $(1, n_{u,max})$ ($n_{u,max}$ is set to 5). The shift parameter h_u is fixed to 1. Parameters h_u and l_u defining the support of the multinomial distribution are uniform on $\{h_{u,min}, \ldots, h_{u,max}\}$ and on $\{l_{u,min}, \ldots, l_{u,max}\}$. We use conjugate Dirichlet prior for $p_u(\bullet) : (p_u(h_u), \cdots, p_u(h_u + l_u)) \sim D_{l_u+1}(\lambda_u, \ldots, \lambda_u)$. Width of box states h_u are *a priori* uniform on $\{h_{u,min}, \ldots, h_{u,max}\}$. Order r of the background Markovian model is uniform on $\{0 \ldots r_{max}\}$. Markovian emission probabilities b_u are the same for all states associated to background model (pre-motif, spacer and absorbing states) and are drawn for all word w from independent conjugate Dirichlet priors : $b_u(\bullet; w) \sim D_{|\mathcal{X}|}(\beta_u, \ldots, \beta_u)$ [Mur98, BHW00]. In box states, emission probabilities are also independent Dirichlet : $b_u(\bullet; k) \sim D_{|\mathcal{X}|}(\beta_u, \ldots, \beta_u)$, for $k \in \{1 \ldots h_u\}$.

We use a fully non informative prior by setting to 1 the parameters of all Dirichlet and Beta priors (α_u, β_u, λ_u and γ_u).

5.3 MCMC algorithm

5.3.1 Reversible Jump MCMC algorithm

We use MCMC methodology to sample in the joint posterior distribution $\pi(\theta, S_1^n \mid X_1^n)$ deduced from the full distribution

$$\pi(\theta, S_1^n \mid X_1^n) \propto \pi(\theta, S_1^n, X_1^n) = \pi(\theta) P_\theta(X_1^n, S_1^n).$$

Using MCMC algorithms [Tie94] a sample from a target distribution is generated as realizations of the stationary distribution of an ergodic Markov chain. One sweep of our MCMC algorithm consists in the 8 following moves that preserve the target distribution $\pi(\theta, S_1^n \mid X_1^n)$:

- **(a)** updating the state transition matrix a,
- **(b)** updating the emission probabilities b,
- **(c)** updating the parameters n_u, p_u of the negative binomial sojourn time distribution,
- **(d)** updating the parameters $p_u(\bullet)$ of the multinomial sojourn time distribution,
- **(e)** updating the hidden state paths S_1^n,

(f) updating the width h_u of the box states,

(g) updating the support $\{h_u, \ldots, h_u + l_u\}$ of the multinomial sojourn time distribution,

(h) updating the order r of the Markovian model of the background.

Moves **(a-e)** are Gibbs or Metropolis-Hastings (MH) moves leaving the model dimension unchanged while moves **(f-h)** are reversible jump MH moves as they do change the model dimension.

Reversible jump MH move is a generalization of the MH move for variable dimension sampling [Gre95, RG97] which allows jumps between sub-spaces of different dimensions through pairs of reversible moves type (m^+, m^-). Considering the current state of the sampler being (θ, S_1^n), the move type m^+ proposes a new candidate $(\tilde{\theta}, \tilde{S}_1^n)$ in a higher-dimensional sub-space. First, a random vector ω of dimension $dim(\omega) \geq dim(\tilde{\theta}) - dim(\theta)$ is drawn, then $\tilde{\theta}$ is deterministically proposed from θ and ω : $\tilde{\theta} = h_+(\theta, \omega)$. In the reverse of the move m^-, θ is proposed as $\theta = h_-(\tilde{\theta}, \tilde{\omega})$ where $\tilde{\omega}$ is a random vector of which dimension matches $dim(\tilde{\omega}) + dim(\tilde{\theta}) = dim(\omega) + dim(\theta)$. To ensure detailed balance equilibrium, an invertible deterministic mapping f is needed : $(\tilde{\theta}, \tilde{\omega}) = f(\theta, \omega)$. The target distribution is preserved if move type m^+ and candidate $(\tilde{\theta}, \tilde{S}_1^n)$ are accepted with probability $\alpha_{m^+}(\theta, S_1^n; \tilde{\theta}, \tilde{S}_1^n) = \min(1, A_{m^+})$ where

$$A_{m^+} = \frac{\pi(\tilde{\theta}, \tilde{S}_1^n, X)}{\pi(\theta, S_1^n, X)} \times \frac{g_{\tilde{\theta}, \tilde{S}_1^n}(m^-, \tilde{\omega}, S_1^n)}{g_{\theta, S_1^n}(m^+, \omega, \tilde{S}_1^n)} \times J$$

with $g_{\theta, S_1^n}(m^+, \omega, \tilde{S}_1^n)$ denoting the proposal density for choosing move type m^+, drawing ω and $\tilde{S}_1^n$ given the current state of the sampler (θ, S_1^n) ; $J = \left| \frac{\partial f(\theta, \omega)}{\partial \theta \partial \omega} \right|$ is the Jacobian determinant arising from the change of variable from (θ, ω) to $\tilde{\theta}$. The reverse move (from $(\tilde{\theta}, \tilde{S}_1^n)$ to (θ, S_1^n)) is accepted with probability $\alpha_{m^-}(\tilde{\theta}, \tilde{S}_1^n; \theta, S_1^n) = \min(1, 1/A_{m^+})$.

5.3.2 Moves without changing model dimension

Moves **(a)** and **(b)** are Gibbs moves similar to the HMM framework [Mur98, BHW00]. They consist in simulating each row $a(u, \bullet)$ of the state transition matrix and each row $b_u(\bullet; w, k)$ of the nucleotides emission distributions from their full conditional Dirichlet distribution :

$$\begin{aligned} a(u, \bullet) \mid S_1^n &\sim D_{|V_u|}(\alpha_u + N_{u1}, \ldots \alpha_u + N_{u|V_u|}) \\ b_u(\bullet; w) \mid r, S_1^n, X_1^n &\sim D_{|\mathcal{X}|}(\beta_u + N_{w1;u}, \ldots, \beta_u + N_{w|\mathcal{X}|;u}) \\ b_u(\bullet; k) \mid S_1^n, X_1^n &\sim D_{|\mathcal{X}|}(\beta_u + N_{1,k;u}, \ldots, \beta_u + N_{|\mathcal{X}|,k}). \end{aligned}$$

$b_u(x; w)$'s are common to all background composition model associated states. Then, $N_{wx;u}$'s are computed over all the positions allocated to these states.

During moves **(c)** and **(d)**, parameters of the sojourn time distributions are updated. To achieve a Gibbs move (see Chib [Chi92]), the right-censored times t' spent in the last visited state u is first to be sampled from its full conditional distribution

$$\pi(k' \mid k' \geq k_c, \theta, S_1^n, X_1^n) = \frac{d_u(k')}{D_u(k')} \mathbb{I}\{k' \geq k_c\} \tag{5.1}$$

where k_c is the observed time spent in the last visited state. A second step consists in updating parameters knowing the censored times. In move **(c)**, p_u is drawn from its Beta full conditional distribution $p_u \mid k_u, n_u \sim Be(\gamma_{u,1} + n_u \sum_{k_u} N_{u,k_u}, \gamma_{u,2} + \sum_{k_u}(k_u - h_u)N_{u,k_u})$ where k_u contains all the (observed and simulated) durations in sate u. Similarly in move **(d)**, $p_u(\bullet)$ is updated after simulating the censored times (5.1) from its Dirichlet full conditional distribution $D_{l_u+1}(\lambda_u + N_{u,h_u}, \ldots, \lambda_u + N_{u,h_u+l_u})$.

To update the parameter n_u in move **(c)**, we first used MH move with proposal $q_{n_u}(\tilde{n}_u) = \mathcal{N}(n_u; 1)$. In order to increase the acceptance rate, we finally chose a proposal that changes n_u without changing the mean $h_u + n_u(1-p_u)/p_u^2$ of the distribution. The candidate couple $(\tilde{n}_u, \tilde{p}_u)$ is proposed in the following way : draw $\tilde{n}_u$ from the proposal $q_{n_u}(\tilde{n}_u)$ and set $\tilde{p}_u$ as

$$\tilde{p}_u = \frac{p_u \tilde{n}_u}{\tilde{n}_u p_u + n_u(1-p_u)}. \tag{5.2}$$

The proposed value is accepted with probability

$$min\Big(1; \frac{\pi(\tilde{\theta}, S_1^n, X_1^n) q_{\tilde{n}_u}(n_u)}{\pi(\theta, S_1^n, X_1^n) q_{n_u}(\tilde{n}_u)} \times \frac{n_u \tilde{n}_u}{\tilde{n}_u p_u + n_u(1-p_u)^2}\Big)$$

where the last term in the ratio is the Jacobian determinant induced by the deterministic transformation (5.2).

Move **(e)** is a Gibbs move which consists in drawing the hidden state paths from their full conditional distribution $\pi(S_1^n \mid X_1^n, \theta)$. This is performed through a "forward-backward" algorithm similar (but more time-consuming) to the HMM framework (see Appendix : section 5.6). As in HMM, this algorithm allows simultaneous computation of the incomplete likelihood $P_\theta(X_1^n)$ used in the following sections. The forward recurrence is realized as in Guédon [Gué03] and the simulation algorithm is slightly different from the one proposed by Plagnol [Pla01]. In the particular case of the two boxes model with a highly constrained spacer length, a more efficient algorithm can be used for hidden state allocation : direct evaluation of each path is permitted by the restricted number of authorized hidden paths. The forward-backward algorithm would be required in a more complex model setting.

5.3.3 Updating the width of box states

Moves **(f)** allow the width h_u of a box state u to increase $(+1)$ or to decrease (-1) by one on the left side (l) or on the right side (r). Several steps are required to achieve these moves. First, we randomly choose a box u and a move type with probability $g_\theta(u, \pm 1, l \text{ or } r) = 1/8$ as there are two box states and 4 move types. Remaining steps consist in updating first parameters $(b,\ d)$ involved in the move, and second the hidden state paths S_1^n. We only describe the "increase-decrease" pair of reversible moves on the left side which is easy to transpose for the right side.

In the "increase" move, the new width is given by $\tilde{h}_u = h_u + 1$. We draw a random vector $\omega_{1;u}$ of size $|\mathcal{X}|$, from the Dirichlet

$$\mathcal{D}_{|\mathcal{X}|}\left(\beta_u + N_{1,-1;u}, \cdots, \beta_u + N_{q,-1;u}\right)$$

to get parameters of the nucleotides distribution associated to the first position of the box u

$$\forall x \in \mathcal{X}, \quad \begin{cases} \tilde{b}_u(x,1) = \omega_{1;u}(x) \\ \tilde{b}_u(x;k) = b_u(x;k-1) \quad \forall k \in \{2,\ldots,\tilde{h}_u\} \end{cases}$$

where $N_{x,-1;u}$ is the number of nucleotide x adjacent to the left side of the box u. Parameters and support of the multinomial sojourn time distribution associated to every state v preceding state u $(u \in V_v)$ are shifted as

$$\forall v \quad \text{s.t.} \quad u \in V_v, \quad \begin{cases} \tilde{h}_v = h_v - 1 \\ \tilde{p}_v(k) = p_v(k+1) \quad \forall k \in \{\tilde{h}_v,\ldots,\tilde{h}_v + l_v\}. \end{cases}$$

Finally, new $\tilde{S}_1^n$ are drawn from their full conditional distribution $\pi(S_1^n \mid \tilde{\theta}, X_1^n)$.

In the reverse "decrease" move, the new width is given by $h_u = \tilde{h}_u - 1$. Nucleotide emission probabilities belonging to the box u are shifted as

$$\forall k \in \{1,\ldots,h_u\}, \forall x \in \mathcal{X} \quad b_u(x;k) = \tilde{b}_u(x;k+1).$$

Parameters of the multinomial sojourn time distributions are shifted as

$$\forall v \quad \text{s.t.} \quad u \in V_v, \quad \begin{cases} h_v = \tilde{h}_v + 1 \\ \forall \quad h_v \le k \le h_v + \tilde{l}_v \quad p_v(k) = \tilde{p}_v(k-1). \end{cases}$$

Finally, states are reallocated from $\pi(S_1^n \mid \theta, X_1^n)$.

The acceptance probabilities for the "increase" and "decrease" moves on the left side are $\alpha\big(\theta, S_1^n; \tilde{\theta}, \tilde{S}_1^n\big) = \min(1, A_{+1}^l)$ and $\alpha\big(\tilde{\theta}, \tilde{S}_1^n; \theta, S_1^n\big) = \min(1, 1/A_{+1}^l)$ where

$$A_{+1}^l = \frac{\pi(\tilde{\theta}, \tilde{S}_1^n, X_1^n)}{\pi(\theta, S_1^n, X_1^n)} \times \frac{g_{\tilde{\theta}}(u,-1,l)}{g_\theta(u,+1,l)} \frac{1}{g(\omega_{1;u})} \frac{\pi(S_1^n \mid \theta, X_1^n)}{\pi(\tilde{S}_1^n \mid \tilde{\theta}, X_1^n)} \times J$$

with g denoting the density of the Dirichlet proposal of $\omega_{1;u}$ and with the Jacobian J equals 1. The move is obviously rejected if the new parameters do not fall into the support of the model prior distribution. The ratio A^l_{+1} simplifies to

$$A^l_{+1} = \frac{P_{\tilde{\theta}}(X_1^n)}{P_\theta(X_1^n)} \times \frac{\pi(\tilde{\theta})}{\pi(\theta) g(\omega_{1;u})}$$

where incomplete likelihoods $P_\theta(X_1^n)$ and $P_{\tilde{\theta}}(X_1^n)$ are computed during state paths reallocations. The right part of the ratio simplifies to $\pi(\tilde{b}_{u,1})/g(\omega_{1;u})$.

5.3.4 Updating the support of the spacer

Moves **(g)** consist in increasing or decreasing by one the support of the multinomial sojourn time distribution of state u (chosen with probability 1 as there is only one spacer state u). Then the support size is either increased or decreased according to the probabilities p_l and $1 - p_l$ with $p_{l_{u,max}} = 1 - p_{l_{u,min}} = 1$ and $p_l = 1/2$ elsewhere. The side the support size is modified is defined using probability p_h^+ for the left side and $1 - p_h^+$ for the right side with $p^+_{h_{u,min}} = p^-_{h_{u,max}} = 0$ and $p_h^+ = p_h^- = 1/2$ elsewhere (similar probabilities p_h^- are needed for the "decrease" move). For instance support is increased on the left side with probability $g_\theta(u, +1, l) = p_l p_h^+$. Here, we only describe the moves on the left side.

In the "increase" move, the support of the spacer state u is set to $\tilde{h}_u = h_u - 1$ and $\tilde{l}_u = l_u + 1$. We draw ω_u from $Be\big(\lambda_u, \tilde{l}_u \times \lambda_u\big)$, the marginal density of the prior Dirichlet distribution of $p_u(\bullet)$, and we set :

$$\begin{cases} \tilde{p}_u(\tilde{h}_u) = \omega_u \\ \tilde{p}_u(k) = (1 - \omega_u) p_u(k) \quad \forall k \in \{\tilde{h}_u + 1, \ldots, \tilde{h}_u + \tilde{l}_u\}. \end{cases} \tag{5.3}$$

Finally, states are reallocated from the full conditional posterior $\pi(S_1^n \mid \tilde{\theta}, X_1^n)$.

The "decrease" move is realized according to the the following steps. First, $h_u = \tilde{h}_u + 1$ and $l_u = \tilde{l}_u - 1$ are set. Second, probabilities $p_u(k)$ are rescaled :

$$p_u(k) = \frac{\tilde{p}_u(k)}{1 - \tilde{p}_u(\tilde{h}_u)} \quad \forall k \in \{h_u, \ldots, h_u + l_u\}.$$

Finally, states are reallocated from $\pi(S_1^n \mid \theta, X_1^n)$.

Acceptance probabilities for "increase" and "decrease" moves on the left side are $\min(1, A^l_{+1})$ and $\min(1, 1/A^l_{+1})$ where

$$A^l_{+1} = \frac{\pi(\tilde{\theta}, \tilde{S}_1^n, X_1^n)}{\pi(\theta, S_1^n, X_1^n)} \times \frac{g_{\tilde{\theta}}(u, -1, l)}{g_\theta(u, +1, l)} \frac{1}{g(\omega_u)} \frac{\pi(S_1^n \mid \theta, X_1^n)}{\pi(\tilde{S}_1^n \mid \tilde{\theta}, X_1^n)} \times J$$

with g being the Beta proposal density of ω_u and J being the Jacobian of the transformation (5.3) equal to $(1-\omega_u)^{\tilde{l}_u-1}$. The ratio A^l_{+1} simplifies to

$$A^l_{+1} = \frac{P_{\tilde{\theta}}(X_1^n)}{P_\theta(X_1^n)} \frac{g_{\tilde{\theta}}(u,-1,l)}{g_\theta(u,+1,l)} \frac{\pi(\tilde{d}_u)}{\pi(d_u)} \frac{1}{g(\omega_u)} (1-\omega_u)^{\tilde{l}_u-1}.$$

5.3.5 Updating the Markovian order of the background

Move **(h)** attempts to increase or decrease by one the order o of the background Markovian model. An alternative way of changing the Markov order consists in drawing it from its conditional density leading the acceptance ratio to equal 1 [BH02b]. Here, "increase" and "decrease" moves are chosen with probabilities $p_r = g_\theta(+1)$ and $1-p_r = g_\theta(-1)$, where $p_r = 1 - p_{r_{max}} = 1$ and $p_r = 1/2$ elsewhere. For the "increase" move, we set $\tilde{r} = r+1$ and we draw new $\tilde{b}_u(\bullet; w)$ from their full conditional Dirichlet distributions $\pi(\tilde{b}_u \mid \tilde{r}, S_1^n, X_1^n) = D_{|\mathcal{X}|}(\beta_u + N_{w1;u}, \ldots, \beta_u + N_{w|\mathcal{X}|;u})$. The hidden states allocation S_1^n remains unchanged and thus this reversible jump move is the less time-consuming one. Similarly, in the "decrease" move, we set $r = \tilde{r} - 1$ and we simulate new $b_u(\bullet, w)$ from their full conditional Dirichlet distribution.

The acceptance probabilities of the "increase" and "decrease" moves of the Markovian order are $\min(1, A)$ and $\min(1, 1/A)$ where

$$A = \frac{\pi(\tilde{\theta}, S_1^n, X_1^n)}{\pi(\theta, S_1^n, X_1^n)} \times \frac{g_{\tilde{\theta}}(-1)}{g_\theta(+1)} \frac{\pi(b_u \mid r, S_1^n, X_1^n)}{\pi(\tilde{b}_u \mid \tilde{r}, S_1^n, X_1^n)} \times 1.$$

A can be rewritten as

$$A = \frac{\pi(\tilde{r} \mid S_1^n, X_1^n)}{\pi(r \mid S_1^n, X_1^n)} \times \frac{g_{\tilde{\theta}}(-1)}{g_\theta(+1)}$$

and then does not depend on parameters $\tilde{b}_u$ and b_u, probabilities $\pi(\tilde{r} \mid S_1^n, X_1^n)$ and $\pi(r \mid S_1^n, X_1^n)$ can be obtained in closed form as in [BH02b]. Thus, the proposed move is accepted or rejected before drawing new parameters $\tilde{b}_u$.

5.4 Applications

5.4.1 Data sets

We implemented the MCMC algorithm in a C++ program using the GNU Scientific Library for both random number generation and probabilistic density functions computation.

To investigate the behavior of the MCMC algorithm, three sets of *B. subtilis* DNA sequences have been extracted from the complete chromosome sequence. Because protein translation start sites are known to be not reliably annotated [BLB01], we base extraction on our own coding sequences prediction program (http ://www-mig.jouy.inra.fr/ssb/SHOW/). Only non coding sequences extending upstream from a translation start site which are longer than 80 nucleotides are selected. As most of the promoter motifs are closed to the translation start site, these sequences have been truncated 150 nucleotides away from the translation start site. Moreover, as nucleotides standing just before the translation start sites contain the ribosome binding site (RBS), they are likely to perturb the estimation of the promoter motif model. Therefore the first 15 nucleotides extending just upstream from the translation start site have been removed. A set of 2811 sequences was obtained, further called the "full data set". Two other sets, much smaller, have been extracted in the same way by selecting only upstream sequence of genes whose transcription is expected to be dependent of a particular Sigma factor according to published transcriptional analysis results [AYK+03, PFC+01]. These two sets are further called the "Sigma M data set" and the "Sigma B data set" and contain respectively 24 and 87 sequences.

Software and data sets are available upon request to the authors.

5.4.2 Results

Results presented in this section are obtained by running 5000 sweeps of the MCMC algorithm in the space delimited by $h_{\text{box},min} = 1$, $h_{\text{box},max} = 25$, $h_{\text{spacer},min} = 1$, $h_{\text{spacer},max} = 25$, $l_{\text{spacer},min} = 0$, $l_{\text{spacer},max} = 5$, $r_{\text{bg},min} = 0$ and $r_{\text{bg},max} = 5$. Width of boxes are initially set to 7 ($h_{\text{box}} = 7$), the range of spacer length is initialized to $\{15, 16, 17\}$ ($h_{\text{spacer}} = 15$ and $l_{\text{spacer}} = 2$), parameters of the shifted negative binomial distribution modeling the distance between the beginning of the sequence and the motif are initially set to ($h = 1$, $n = 3$, $p = 0.15$). Other parameters are initially drawn from their respective prior distribution. The computation time grows linearly with the number of sequences and is about 70 min for the "Sigma B data set" using a 1 GHz Pentium 3 processor. As the hidden state allocation is performed three times a sweep, the computation is approximately three times longer than it would be in a fixed dimension setting. For reasons discussed below, the model used for the "full data set" is a model with optional "-35" box (see Figure 5.2).

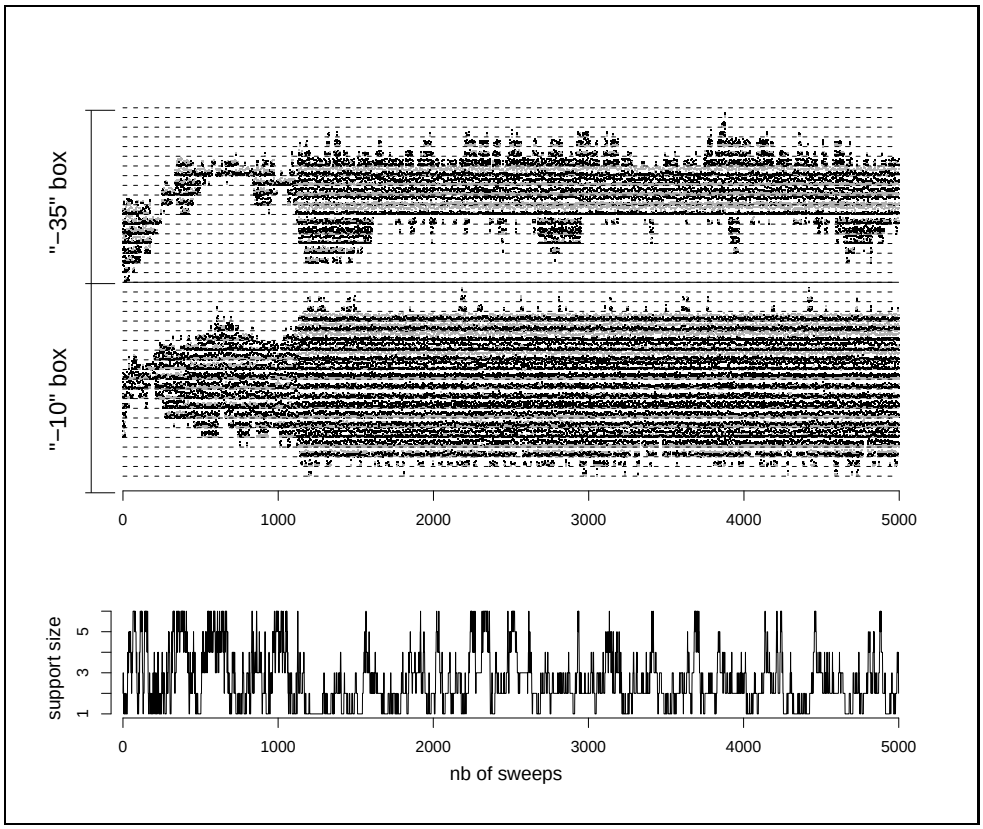

FIG. 5.3 – 5000 sweeps of the MCMC algorithm run on the "Sigma M data set". Upper panel : Width and nucleotide probabilities associated to each positions of the "-10" and "-35" boxes plotted every 5 sweep. Dashed lines separate positions in the box. Between dashed lines, black points correspond to the probabilities of a and t, gray points to the probabilities of g and c. If a box is enlarged, a line is added to represent the new position, if it is shorten a line is removed. Finally, if move takes place respectively to the left side or to the right side of the box, change in the graph occurs below or above. Lower panel : support size of the spacer length distribution plotted every sweep.

Figure 5.3 displays values of some parameters sampled during the first 5000 sweeps of the MCMC algorithm run on the "Sigma M data set". According to Figure 5.3, the burn-in time appears to be shorter than 2000 sweeps. Even shorter burn-in are observed on the two other data sets. Acceptance rates for moves that change the width of the boxes, computed over the last 3000 sweeps are respectively 27.8%, 18.7% and 0.5% for the "Sigma M data set", the "Sigma B data set" and the "full data set". Acceptance rates for moves that change the spacer length support size are respectively 21.0%, 16.7% and 6.4% for the same data sets. The acceptance rates decrease when the amount of data increases, probably due to the sharpness of the posterior density function which tends to increase. However, the proportion of accepted reversible jump MH moves is fully satisfying except for the acceptance rate of the move that change box width on the "full data set". On the "full data set" we observe (results not shown) that the MCMC is sometimes trapped after a short burn-in in a local mode of the posterior distribution, which corresponds to a "-35" box model fitted on the "-10" motif of the Sigma A binding site. Unlike other model dimensions, the Markovian order of the background model is determined without any uncertainty in a few sweeps. Then, out of the burn-in time, moves that change Markovian order are never accepted. Estimated order increases with the size of the data set and is respectively 1, 2 and 3.

Figures 5.4 and 5.5 display posterior distribution of the parameters estimated over the last 3000 sweeps. Figure 5.4 shows posterior distribution of box widths and nucleotide compositions estimated on each data set. On Figure 5.5 estimated spacer length distribution are presented. Models estimated using the "Sigma B data set" and the "full data set" are compatible respectively to previous descriptions of the Sigma B and Sigma A binding site motifs. Finding Sigma A binding site motif when analyzing the "full data set" was expected as Sigma A is the main Sigma factor, thus its binding sites are the most frequent in that data set. Consensus sequences for the Sigma B binding sites have been previously identified as *gtttaa* and $ggg(a/t)a(a/t)$ separated by a spacer 13 :15 nucleotides long [PBG+99]. Consensus sequences of the "-35" and "-10" boxes of the Sigma A binding site have been previously identified as *ttgaca* and *tataat* separated by a spacer 16 :18 nucleotides long [Hel95]. Moreover, the few positions included in the estimated "-10" box upstream from *tataat* are known to be the extended "-10" box. It is also to be noticed that the estimated "-10" box extends 12 positions downstream from *tataat* due to the a+t enrichment of the sequence around the transcription start site. As far as we know, the Sigma M binding site motif has never been described before. However, the motif estimated using the "Sigma M data set" looks realistic and consistent with the mapping of the 5'-end of the *sigM* mRNA by primer extension analysis [HM99]. In Figure 5.5, posterior inference of the spacer length distribution is reported for two different HSMM structures : with optional or obligatory "-35" box. Actually, it has been descri-

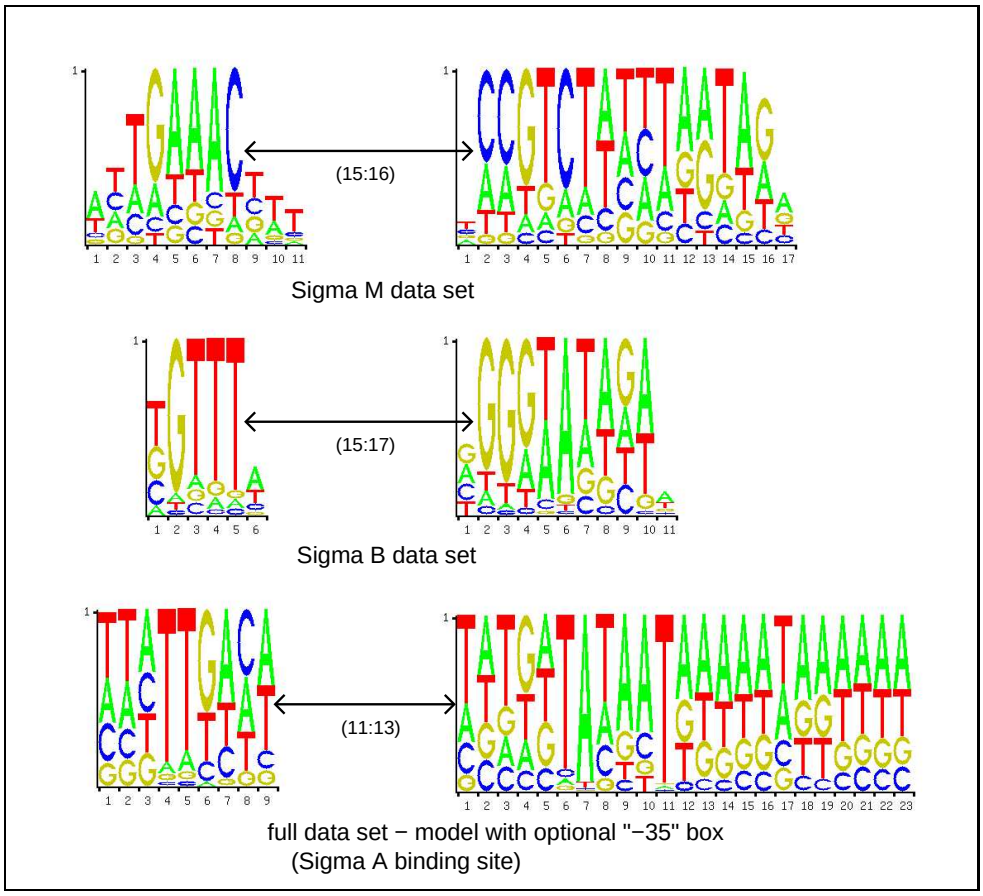

FIG. 5.4 – Promoter motifs estimated on the three datasets. "-35" and "-10" boxes are displayed respectively on left and right sides. Motifs are displayed as "sequence logos" where height associated to each motif position corresponds to the posterior probability that the position is included in the box. At each position, relative height of the letters are proportional to the estimated nucleotide posterior probabilities. Double sided black arrows point the motif positions taken as reference when reporting the spacer length posterior distribution. Couples of numbers between parentheses report the posterior mode of the support range of the spacer length distribution. Logos have been drawn using the TFBS : :Matrix : :ICM Perl module.

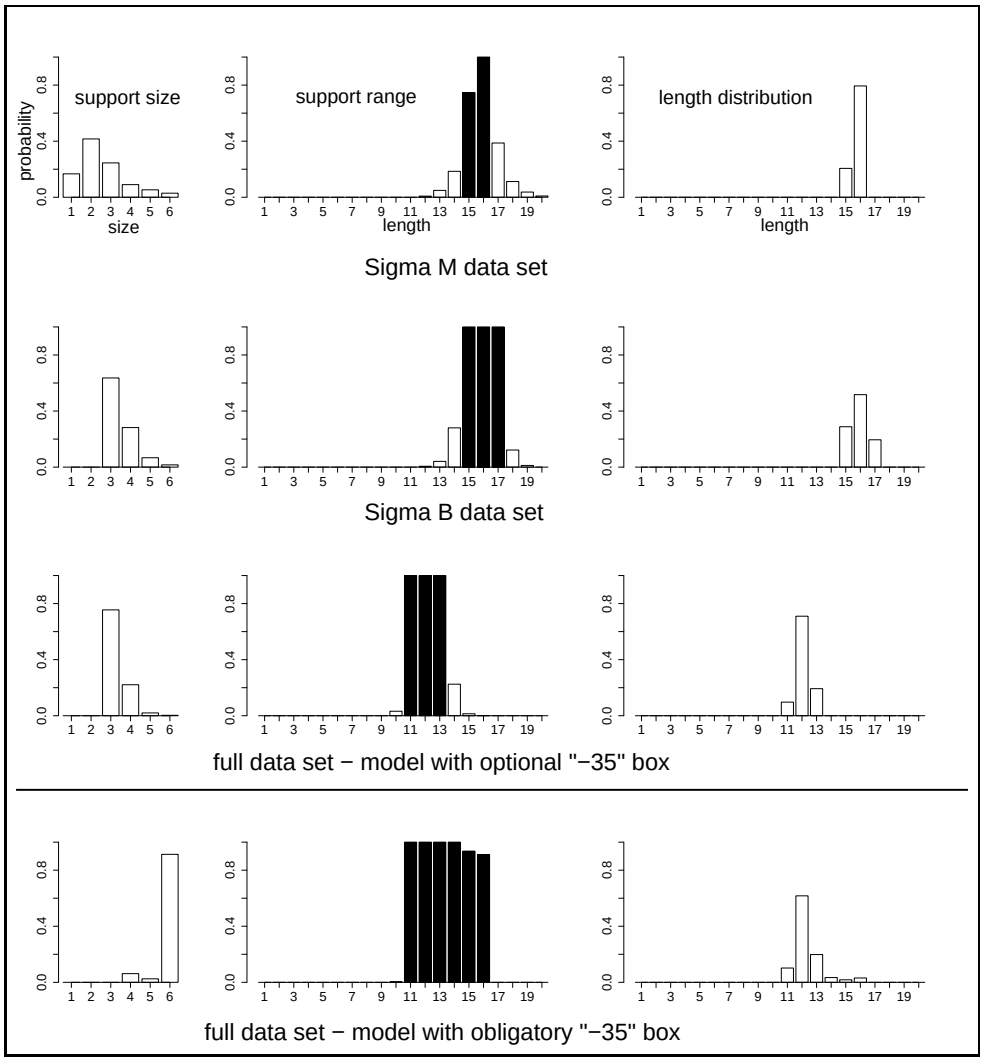

FIG. 5.5 – Spacer length distributions estimated on the three datasets. First column corresponds to the posterior distribution of the support size $l_u + 1$. Second column displays, for each length, the posterior probability to belong to the spacer length range. Black bars indicate lengths belonging to the posterior mode of the spacer length range. Third column displays the posterior probabilities of the lengths conditionally on the equality between the spacer length range and its posterior mode. The motif positions pointed by double sided black arrows in Figure 5.4 are used to report spacer lengths in second and third columns.

bed that the "-35" box is missing within some Sigma A binding sites [Hel95]. Then imposing a "-35" box to be present probably do not conform to the data. Indeed, whereas estimated probability of the spacer lengths $\{11, 12, 13\}$ are closed for these two models (Figure 5.5, third column), the estimated support size of the spacer length distributions completely differ (Figure 5.5, first and second columns). This behavior points out the fact the model selection is highly sensitive to the set of explored model.

5.5 Conclusion

The reversible jump MCMC algorithm proposed in this study gives biologically reliable results and is a straight-forward approach for HSMM dimension choice in the MCMC framework. The acceptance probabilities achieved by the algorithm are rather satisfying. This good behavior of our sampler is a consequence of the reversible jump moves we propose that either change by one the model dimension (update of the support size of the spacer) either allow higher dimension jumps, using informative proposal distributions (update of both Markovian order and width of the boxes). Then the methodology used appears to be particularly suitable to our set-up. Moreover, computational complexity does not raise that much while considering variable rather than fixed dimension sampling. The use of reversible jump MCMC would probably be easy to extend to other models dedicated to motif discovery such as HMMs and mixture models. It is worth to mention that in an earlier version of our algorithm, we attempted to change the width of the boxes without reallocating the hidden state paths from their full conditional distribution (in a similar way as [QMT$^+$03]). These moves consists in a deterministic proposal for the hidden state paths by modifying the current allocations according to the proposed width. This approach allows to get computationally cheap reversible jump moves with an acceptance rate comparable to the one shown in this study. However, numerous moves which should end up in an increase on the left side of the "-10" box width are rejected because the proposed allocations do not conform to the HSMM. Indeed, these allocations contain a few "-10" boxes that begin at the first position of a sequence. The use of this kind of moves is efficient in other model frameworks, in which they enable the estimation of models describing simultaneously a large number of distinct motifs.

5.6 Appendix

Computing the full conditional posterior $\pi(S_1^n \mid X_1^n, \theta)$ is done with a forward-backward recurrence involving the probability of leaving state v at time t given observed nucleotides X_1^t, $F_v(t) = P_\theta(S_t = v, S_{t+1} \neq v \mid X_1^t)$ and the probability of nucleotide X_t given the previous ones $N_t = P_\theta(X_t \mid X_1^{t-1})$.
Forward recurrence : compute as in Guédon [Gué03]

$$
\begin{aligned}
F_v(t) &= \frac{b_v(X_t; X_{t-r_v}^{t-1}, t)}{N_t} \\
&\quad \times \left\{ \sum_{k=1}^{t} \left(\prod_{t'=1}^{k-1} \frac{b_v(X_{t-t'}; X_{t-t'-r_v}^{t-t'-1}, t-t')}{N_{t-t'}} \right) d_v(k) \sum_{u \neq v} a(u,v) F_u(t-k) \right. \\
&\quad \left. + \left(\prod_{t'=1}^{t} \frac{b_v(X_{t-t'}; X_{t-t'-r_v}^{t-t'-1}, t-t')}{N_{t-t'}} \right) d_v(t+1) a(v) \right\} \text{ for } t = 1, \ldots, n-1 \\
F_v(n) &= \frac{b_v(X_n; X_{n-r_v}^{n-1}, n)}{N_n} \\
&\quad \times \left\{ \sum_{k=1}^{n} \left(\prod_{t=1}^{k-1} \frac{b_v(X_{n-t}; X_{n-r_v}^{n-t-1}, n-t)}{N_{n-t}} \right) D_v(k) \sum_{u \neq v} a(u,v) F_u(n-k) \right. \\
&\quad \left. + \left(\prod_{t=1}^{n} \frac{b_v(X_{n-t}; X_{n-t-r_v}^{n-t-1}, n-t)}{N_{n-t}} \right) D_v(n) a(v) \right\} \\
N_t &= \sum_v b_v(X_t; X_{t-r_v}^{t-1}, t) \\
&\quad \times \left\{ \sum_{k=1}^{t} \left(\prod_{t'=1}^{k-1} \frac{b_v(X_{t-t'}; X_{t-t'-r_v}^{t-t'-1}, t-t')}{N_{t-t'}} \right) D_v(k) \sum_{u \neq v} a(u,v) F_u(t-k) \right. \\
&\quad \left. + \left(\prod_{t'=1}^{t} \frac{b_v(X_{t-t'}; X_{t-t'-r_v}^{t-t'-1}, t-t')}{N_{t-t'}} \right) D_v(t+1) a(v) \right\} \text{ for } t = 1, \ldots, n
\end{aligned}
$$

Note that the normalizing factors N_i allow to compute the incomplete likelihood as $P_\theta(X_1^n) = \prod_{t=1}^n N_t$.
Backward recurrence :

1. Draw S_n from $P_\theta(S_n = u \mid X_1^n) = F_u(n)$
2. Draw the last jumping time t' (before visiting state v) and $S_{t'} \neq v$ from

$$
\begin{aligned}
&P_\theta(S_{t'} = u, S_{t'+1}^{n-1} = v \mid S_n = v, X_1^n) \\
&\quad = \left(\prod_{t=t'+1}^{n} \frac{b_v(X_t; X_{i-r_v}^{t-1}, t-t')}{N_t} \right) \frac{D_v(n-t') F_u(t') a(u,v)}{F_n(v)}
\end{aligned}
$$

and

$$P_\theta(S_1^{n-1} = v \mid S_n = v, X_1^n) = \left(\prod_{t=1}^{n} \frac{b_v(X_t; X_{t-r_v}^{t-1}, t-t')}{N_t}\right) \frac{D_v(n)a(v)}{F_n(v)}$$

3. While $t' \geq 2$, $\tilde{s}_{t'}^{t-1} := (u, v^{t-t'-1})$, $t := t'$, $v := u$, draw t' and $S_{t'} \neq v$ from

$$\begin{aligned} & P_\theta(S_{t'} = u, S_{t'+1}^{t-1} = v \mid S_t^n = \tilde{s}_t^n, X_1^n) \\ &= P_\theta(S_{t'} = u, S_{t'+1}^{t-1} = v \mid S_t = v, S_{t+1} \neq v, X_1^n) \\ &= \left(\prod_{k=t'+1}^{t} \frac{b_v(X_k; X_{k-r_v}^{k-1}, k-t')}{N_k}\right) \frac{d_u(t-t')F_u(t')a(u,v)}{F_v(t)} \end{aligned}$$

and

$$\begin{aligned} P_\theta(S_1^{t-1} = v \mid S_t^n = \tilde{s}_t^n, X_1^n) &= P_\theta(S_1^{t-1} = v \mid S_t = v, S_{t+1} \neq v, X_1^n) \\ &= \left(\prod_{k=1}^{t} \frac{b_v(X_k; X_{k-r_v}^{k-1}, k-t')}{N_k}\right) \frac{d_v(t)a(v)}{F_v(t)} \end{aligned}$$

All these equations are valid for absorbing state u by choosing $d_u(t) = 0$ and $D_u(t) = 1$ for all t.

Conclusions et perspectives

Trois domaines d'applications des HMM pour l'analyse des séquences d'ADN ont été abordés au cours de cette thèse. Le fil conducteur de cette conclusion est de positionner ces travaux dans la perspective d'une description globale des chromosomes bactériens, depuis la description des hétérogénéités de séquence, jusqu'à l'identification des motifs régulateurs, en passant par la détection des gènes.

L'hétérogénéité des séquences

Dans le chapitre 3, un modèle HMM est utilisé pour l'analyse de l'hétérogénéité du génome de *B. subtilis*. Initialement développée pour la recherche des transferts horizontaux, cette approche a effectivement permis l'identification de régions de composition atypique. Des indices supplémentaires de l'origine exogène de ces régions ont également été trouvés (fonctions d'adaptation, répétitions). Cette étude a contribué à raviver les interrogations sur la source de la majorité des transferts horizontaux. En effet, les transferts horizontaux, ici identifiés grâce au HMM, comme ceux identifiés par d'autres méthodes d'étude de la composition des séquences, présentent la particularité d'être plus riches en $a+t$ que le reste du génome. Ce caractère commun à la plupart des transferts est intriguant si l'on considère à la fois qu'ils proviennent d'organismes divers et que leur composition reflète celle de l'organisme source. La relative homogénéité des transferts pourrait s'expliquer par leur appartenance à un ensemble de gènes mobiles « voyageant » entre des espèces relativement proches (voir des souches d'une même espèce), et donc subissant un même mécanisme évolutif. Cette hypothèse n'explique cependant pas le caractère $a+t$ riche. D'autres études se sont penchées sur cette caractéristique. Ainsi, Rocha et Danchin [RD02] ont invoqué le moindre coût énergétique de la synthèse des nucléotides a et t par rapport à celle de g et c. D'autre part, Daubin *et al.* [DLP03] ont confirmé la concordance entre le groupe de gènes distingués par leur composition atypique et la vaste majorité des gènes récemment transférés, identifiés par comparaison de génomes.

De manière remarquable, le modèle HMM, très simple, a aussi permis de rendre compte d'autres niveaux d'hétérogénéités du chromosome liés aux propriétés des gènes (gènes d'ARN de structure, niveau d'expression, hy-

drophobicité des protéines codées). Deux directions ont été choisies pour poursuivre la modélisation des chromosomes bactériens en tenant compte des spécificités de composition des régions codantes. Ainsi, l'approche mise au point pour la prédiction de gènes dans le chapitre 4 peut être considérée comme le prolongement naturel de l'estimation d'un HMM pour analyser l'hétérogénéité des chromosomes. Quant à la méthode de sélection de modèle développée pour la recherche des motifs promoteurs, elle s'inscrit dans une démarche de modélisation séparée des régions non codantes.

D'autres prolongements plus directs de l'utilisation d'un HMM pour l'étude de l'hétérogénéité des séquences sont possibles. Tout d'abord, il est possible de modéliser, grâce à un HMM, la séquence d'ADN conditionnellement à la séquence protéique. Alors que les gènes sont considérés comme l'unité de composition homogène élémentaire dans les approches classiques d'étude de l'usage des codons, cette modélisation présente l'originalité de permettre la mise en évidence de variations d'usage des codons au sein des gènes. Elle constitue de plus une alternative aux méthodes d'analyse factorielle qui ne tiennent pas compte de l'incertitude sur l'estimation des fréquences relatives d'usage des codons associées à chaque gène [PT02]. Par ailleurs, un HMM dédié à la modélisation de l'hétérogénéité de composition des séquences pourrait constituer une alternative à l'utilisation des modèles markoviens homogènes pris comme modèle nul dans de nombreux tests statistiques (par exemple pour identifier les mots apparaissant à une fréquence exceptionnelle). Enfin, chez les vertébrés homéothermes, un HMM pourrait peut-être permettre de rendre compte d'hétérogénéités de composition au sein des isochores, dont l'homogénéité a été récemment remise en cause [NL00, LBGCO03].

La prédiction des gènes

L'estimation d'un HMM modélisant de façon beaucoup plus réaliste les régions codantes a permis la mise au point d'un logiciel de prédiction de gènes. Ses performances le situent au niveau des meilleurs. De plus, ce logiciel ne requiert ni la construction d'un ensemble d'apprentissage, ni aucune autre information extérieure à la séquence étudiée. Pour un logiciel de prédiction de gènes bactériens fondé sur un HMM, il a aussi la particularité de quantifier l'incertitude concernant les prédictions réalisées (Larsen et Krogh [LK03] ont récemment proposé une approche alternative pour ce problème). Des prédictions de multiples codons *start* sont notamment réalisées lorsque la prédiction d'un unique codon *start* est trop risquée. La quantification de l'incertitude liée aux prédictions a révélé l'existence de nombreux petits ORF ressemblant, par leur composition, à des séquences codantes.

Actuellement, peu de petits gènes sont connus, et leur « annotation » relève essentiellement du choix d'une politique d'annotation. Leur prédiction est probablement l'un des derniers grands problèmes de la prédiction des

séquences codantes chez les bactéries. Ainsi, chez *B. subtilis*, seuls 20 gènes de taille inférieure à 50 aa ont été identifiés biologiquement. *B. subtilis* est pourtant sans doute la bactérie dont on connaît le plus grand nombre de très petits gènes. Nous avons donc commencé une recherche *in silico* systématique des petits gènes chez les bactéries. À notre connaissance, cette étude est la première à se fixer cet objectif. Un test statistique, fondé sur la répartition des mutations aux trois positions des codons, a été construit pour confirmer la nature codante des ORF grâce aux homologies de séquences. Chez *B. subtilis*, on a ainsi pu proposer plus de 30 ORF de moins de 50 aa dont la nature codante semble presque certaine. Parmi eux, seuls 5 sont annotés dans le fichier GenBank. Ce résultat révèle clairement notre méconnaissance actuelle des petits gènes. Certains sont pourtant connus pour jouer des rôles très importants, notamment dans les interactions entre bactéries. Actuellement, nous identifions des candidats chez *Streptococcus thermophilus* et *Bacillus cereus* pour des études expérimentales de ces fonctions. Dans un futur proche, cette approche pourra être affinée. Il semble notamment intéressant de comparer le test construit pour confirmer par homologie la nature codante avec un test fondé sur le nombre de mutations synonymes et non synonymes [NML02, NCL03]. Toujours par comparaison de génomes, il serait aussi pertinent d'envisager la recherche d'indices négatifs de la nature codante : il s'agirait par exemple d'identifier les ORF dont la séquence n'est pas conservée alors que les gènes voisins sont conservés.

Par ailleurs, la mise au point du détecteur de gènes a aussi conduit au développement du logiciel SHOW (annexe B). Ce logiciel permet la conception et l'utilisation de HMM pour l'analyse des séquences. Probablement moins ambitieux que le projet de librairie GHMM [KSS+02], SHOW s'est néanmoins déjà montré stable et performant. La conception d'un détecteur de gènes pour les levures constitue une perspective intéressante. En effet, l'estimation non supervisée pourrait s'avérer particulièrement appropriée à la modélisation de la diversité des caractéristiques des génomes de levures. Au contraire, chez les eucaryotes supérieurs, la faible proportion des séquences codantes et l'hétérogénéité des régions non codantes rend peu probable le succès d'une prédiction de gènes fondée sur ce principe.

La recherche des motifs régulateurs

Dans le chapitre 5, la recherche de motifs promoteurs par HMM a été abordée. Ce travail prolonge une utilisation récente de l'approche bayésienne de sélection de modèles par MCMC, employée dans le but de classifier des séquences de sites de fixation d'autres facteurs de transcription [QMT+03]. Lors de cette précédente étude, les séquences ont été pré-sélectionnées par identification des régions conservées entre génomes proches. La sélection de modèle est ici réalisée sans traitement préalable des séquences. Le modèle, adapté à la recherche des sites de fixation des facteurs sigma de l'ARN po-

lymérase, est aussi plus riche : les motifs recherchés sont constitués de deux boîtes séparées par un « *spacer* » de longueur variable, et la position des motifs par rapport au site d'initiation de la traduction est prise en compte. Dans ce cadre, la sélection de modèle est étendue à la détermination de l'ordre markovien du modèle de « *background* » et au choix du support des longueurs autorisées pour le « *spacer* ».

Les résultats présentés montrent que l'utilisation d'algorithmes MCMC pour la sélection bayésienne de modèles est particulièrement bien adaptée à la recherche de motifs. D'une part, l'algorithme permet d'étudier une loi *a posteriori* définie sur un immense ensemble de modèles ($25 \times 25 \times 25 \times 6 \times 6 = 562\,500$ modèles différents : 25 longueurs pour chacune des deux boîtes, 25×6 supports de longueur pour le *spacer* et 6 ordres markoviens différents pour le modèle de *background*). D'autre part, la pertinence des modèles sélectionnés a été montrée sur des motifs connus, et le motif du site de fixation du facteur Sigma M de *B. subtilis* a pu être identifié. Étant donné l'importance de l'identification des motifs régulateurs pour l'interprétation des génomes, les perspectives offertes par l'implémentation d'algorithmes *Reversible Jump* MCMC sont prometteuses.

La majorité des motifs régulateurs n'ont que peu d'occurrences sur le chromosome. L'utilisation de données expérimentales peut permettre l'identification de ces motifs. Par exemple, l'utilisation de données d'expression a permis ici l'identification des motifs des sites de fixation des facteurs Sigma B et Sigma M. La mise au point d'approches purement *in silico* pour la recherche de ces motifs relativement peu fréquents reste un défi. La conservation des séquences régulatrices au cours de l'évolution peut faciliter considérablement l'identification des motifs peu représentés sur la séquence [MTC+01, RSZS02, MTCL02]. Pour l'instant, cette information a été prise en compte dans le cadre de modèles de mélanges [vNZRS02, QMT+03] mais il est envisageable de le faire dans un HMM. Pour ce qui est de la recherche des sites de fixation des facteurs sigma secondaires, un travail préliminaire d'évaluation semble nécessaire. En effet, les facteurs sigma secondaires interviennent dans l'expression de gènes impliqués dans des fonctions physiologiques (comme la réponse aux stress pour Sigma B) qui peuvent ne pas être conservés, même dans des bactéries relativement proches. On pourrait prendre en compte, dans une approche complémentaire, les fonctions probables des gènes, prédites à partir des homologies de séquences protéiques. La conception d'un modèle probabiliste pour aborder ce problème apparaît comme une perspective intéressante. Enfin, la prédiction des sites de fixation des facteurs sigma de l'ARN polymérase devrait faciliter la recherche des sites de fixation d'autres facteurs de régulation transcriptionnelle. En effet, ces derniers interagissant avec l'ARN polymérase, les positions de leurs sites de fixation peuvent être plus ou moins contraintes par celles des sites de fixation des facteurs sigma.

Annexe A

Prédiction de gènes

Pierre Nicolas et Hélène Chiapello

A.1 Prédire les gènes

À partir du début de la décennie 1990, le séquençage à grande échelle de l'ADN génomique a changé le problème de la détection de gènes. Il est devenu important de pouvoir prédire où sont les gènes sur une séquence d'ADN. Ces prédictions, réalisées à l'aide d'outils informatique, sont dites *in silico*. Actuellement, le terme « prédiction de gènes » est souvent utilisé dans le sens restreint de prédiction des séquences codant pour des protéines, les CDS (*CoDing Sequences*). C'est dans ce sens qu'il est employé dans ce texte. À cause de l'importance fonctionnelle des protéines, la prédiction des CDS est l'un des principaux objectifs du séquençage de l'ADN. Il s'agit souvent de l'étape préalable à l'interprétation des données génomiques et aux expériences d'analyse fonctionnelle. D'un point de vue fondamental, prédire les gènes à partir de la séquence d'ADN est aussi la seule façon de connaitre toutes les protéines, même celles qui ne sont exprimées que dans des conditions particulières. Compte tenu de l'importance des ces enjeux, il est utile d'avoir une idée des principes et des limites des différentes méthodes de prédiction.

On peut distinguer des approches extrinsèques et intrinsèques. Les approches extrinsèques utilisent essentiellement les comparaisons avec d'autres séquences. Ces séquences peuvent être déjà dans des banques de données ou provenir d'expériences dont un des buts est de détecter les gènes. Les approches intrinsèques, au contraire, se fondent uniquement sur les propriétés locales de la séquence étudiée : composition et présence de signaux. La prédiction est alors dite *ab initio*, c'est l'aspect de la prédiction de gènes le plus développé dans ce chapitre.

A.2 Modélisation de la séquence d'ADN et prédiction *ab initio*

Chez les bactéries, une prédiction approximative des CDS peut être réalisée en cherchant les longs cadres ouverts de lecture (*Open Reading Frames*, ORF) : régions sans codon *stop* dans une des six phases de lecture (figure A.1a). Cependant, cette démarche ne permet ni de localiser de façon précise les débuts de traduction, ni de détecter les petits gènes. D'autre part, plus le génome est pauvre en A et T, plus les longs ORF qui ne correspondent pas à des CDS sont fréquents (figure A.1b). En effet, les codons *stop* (TGA, TAA, TAG) sont riches en nucléotides A et T. Chez les organismes dont les CDS sont morcelés en exons cette méthode n'est pratiquement pas applicable, de nombreux exons étant trop courts pour être détectés (figure A.1c).

Une prédiction plus fine des CDS peut être réalisée en prenant en compte la composition en nucléotides, ou texture, de la séquence. À cause des contraintes exercées par la séquence protéique sur la séquence nucléique, les nucléotides

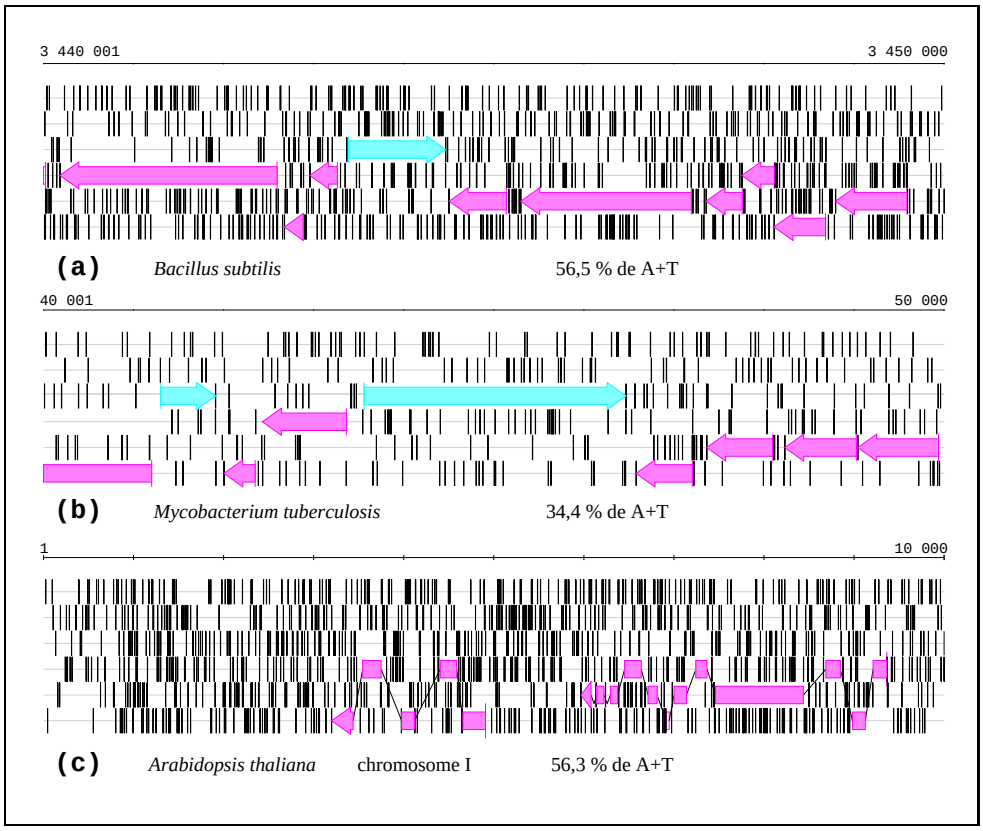

FIG. A.1 – Positions des CDS annotés (flèches) et des codons *stop* (barres verticales noires) dans les six phases de lecture de trois séquences de 10000 nucléotides. Les fragments codants ne contiennent pas de *stop* dans leur phase de lecture. Les séquences (a) et (b) proviennent de bactéries qui diffèrent notamment par leur composition en A+T. La fréquence des codons *stop* (TAG, TGA et TAA) est nettement plus élevée dans la séquence (a), plus riche en A+T. La séquence (c) est une séquence eucaryote dont les CDS sont morcelés en exons.

apparaissent avec des fréquences différentes selon la nature localement codante ou non codante de la séquence. De plus, en raison des propriétés du code génétique, la fréquence des nucléotides varie aussi dans un CDS selon la position dans le codon.

Les différences de composition en nucléotides entre régions codantes et non codantes ont encouragé l'emergence de méthodes de prédiction fondées sur une modélisation probabiliste des séquences d'ADN. Un modèle probabiliste est un moyen d'affecter une probabilité à chaque séquence. Les modèles doivent rendre compte des propriétés de la séquence sur lesquelles on veut appuyer les prédictions, ils n'ont pas vocation à rendre compte de la façon dont la séquence est réellement apparue.

A.2.1 Modélisation de la texture

Le modèle le plus simple de la composition en mono-nucléotides consiste à imaginer que les nucléotides apparaissent indépendamment les uns des autres le long de la séquence et avec des fréquences différentes pour A, C, G et T. Pour rendre compte de la composition en di-nucléotides on peut imaginer qu'à chaque position les nucléotides apparaissent avec une probabilité qui dépend uniquement du nucléotide apparu à la position précédente. Ce modèle est une chaîne de Markov, il se généralise aisément en faisant dépendre la probabilité d'apparition d'un nucléotide des n-nucléotides qui ont précédé. Il s'agit alors d'un modèle de Markov d'ordre n. Par extension, le modèle d'indépendance des nucléotides est parfois désigné comme le modèle de Markov d'ordre 0. Un modèle de Markov d'ordre n modélise uniquement la composition en oligo-nucléotides de longueur $n+1$. Ainsi, les modèles markoviens n'ont qu'une mémoire très courte. Pour modéliser la périodicité des régions codantes il suffit d'utiliser une loi de probabilité différente pour l'apparitions des nucléotides dans chacune des trois positions dans le codon. On parle alors de modèle de Markov tri-périodique. Le logiciel GenMark [BM93] utilise un modèle markovien tri-périodique et généralement d'ordre 5.

L'estimation des paramètres d'un modèle markovien d'ordre n à partir d'un ensemble de séquences d'apprentissage est très simple. Il suffit de calculer les fréquences observées d'apparition des nucléotides dans chacun des contextes (après chacun des oligo-nucléotides de longueur n). D'une manière générale, si les paramètres sont bien estimés la séquence est d'autant mieux modélisée que l'ordre du modèle est élevé. Cependant le nombre de paramètres augmente très rapidement avec l'ordre du modèle. Les paramètres tendent donc à être de moins en moins bien estimés. Pour résoudre ce problème, GLIMMER1.0 [SDKW98] propose l'utilisation d'un modèle de Markov dit à ordre variable dans lequel la longueur du contexte pris en compte dépend de la séquence du contexte. Pour qu'un contexte long soit pris en compte, il faut tout d'abord que suffisamment de données soit disponibles pour l'estimation des fréquences, mais aussi que ce contexte apporte effec-

tivement plus d'information sur le nucléotide qui le suivra qu'un contexte plus court. Cette généralisation des modèles markoviens a été étendue aux contextes « à trous » qui tiennent compte du fait que ce ne sont pas forcément les nucléotides juste adjacents qui apportent le plus d'information. Une procédure automatique permet à GLIMMER2.0 [DHK+99] de choisir, dans un contexte de 12 nucléotides, quelles sont les positions à prendre en compte. On appelle ces modèles des arbres de contexte.

A.2.2 Prédiction des gènes à partir de la modélisation de la texture

Plusieurs approches sont possibles pour prédire les gènes à l'aide de modèles de composition de l'ADN pour les régions codantes et non codantes. Le programme GenMark utilise une fenêtre glissante le long de la séquence. La probabilité de la séquence contenue dans la fenêtre est calculée sous chacun des 7 modèles : non codant, codant dans chacune des trois phases sur chacun des deux brins. L'introduction d'une probabilité *a priori* de chaque modèle permet alors d'obtenir grâce à la formule de Bayes la probabilité de chaque modèle connaissant la séquence. La probabilité de chaque modèle est proportionnelle au produit de (i) la probabilité d'apparition de la séquence sous le modèle et de (ii) la probabilité *a priori* du modèle. L'utilisation d'une fenêtre courte permet d'avoir une information plus locale sur la séquence, mais l'information diminue rapidement lorsque la fenêtre est trop petite. En faisant glisser la fenêtre, on obtient une probabilité de la nature localement codante de la séquence dans chaque phase le long de la séquence. Il faut ensuite réaliser un post-traitement des résultats pour choisir les ORF à prédire comme codants et essayer de localiser leur codon *start*. Une alternative simple aux fenêtres glissantes est mise en oeuvre par GLIMMER, elle consiste à classer directement les ORF selon leur nature codante ou non codante.

L'utilisation de fenêtres glissantes ou la classification des ORF ne permet pas de localiser simplement les frontières entre régions codantes et non codantes. D'un point de vue probabiliste, la façon naturelle de résoudre ce problème consiste à modéliser, non seulement la composition des régions codantes et non codantes, mais aussi l'alternance de ces régions. Cette modélisation permet alors de répondre à la question « où sont les régions des différents types ? » et non plus « étant donnée une région, de quel type est-elle ? ».

A.2.3 Modéliser de l'hétérogénéité des séquences pour prédire les frontières des gènes

Les modèles de chaîne de Markov cachées permettent de modéliser la séquence d'ADN comme une mosaïque de régions, d'un nombre de types relativement restreint et avec chacun leurs propriétés de composition. Tous

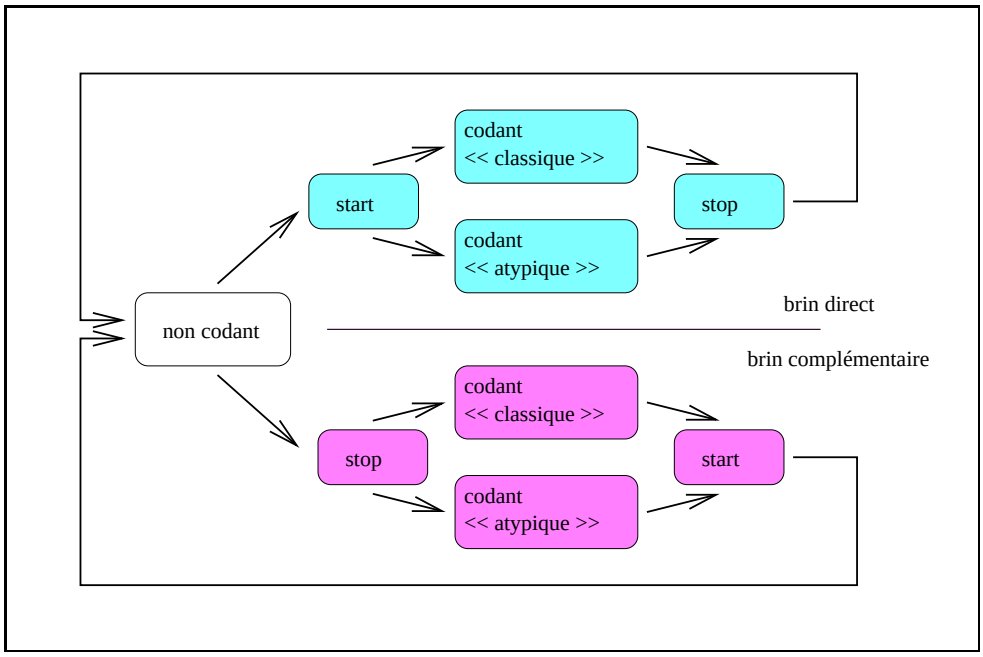

FIG. A.2 – Graphe représentant le modèle caché du HMM de GeneMark.hmm [LB98]. Ce graphe décrit l'ensemble des transitions possibles (flèches) entre états cachés (rectangles arrondis). Le HMM de GeneMark.hmm a été ultérieurement enrichi de la modélisation des RBS.

les modèles de texture présentés précédemment peuvent être utilisés pour rendre compte de l'émission des nucléotides dans les différents types de régions. L'alternance des types de régions le long de la séquence est modélisée par une chaîne de Markov. Cette chaîne est cachée car elle n'est observable qu'à travers les hétérogénéités de la composition de la séquence d'ADN. Les types de régions, sont appelés états cachés du modèle.

Les modèles de chaînes de Markov cachées, ou HMM (*Hidden Markov Model*), constituent un cadre simple et souple de modélisation de l'alternance de régions au sein de la séquence d'ADN. Les paramètres de ces modèles sont les paramètres des modèles d'émission des nucléotides au sein des états cachés et les probabilités de transition entre les états cachés. De nombreuses probabilités de transition entre états cachés peuvent être fixées nulles, permettant ainsi de structurer la suite des états cachés. Le graphe des transitions autorisées entre états cachés apporte souvent beaucoup d'information sur le modèle, la figure A.2 présente le graphe du modèle caché de GeneMark.hmm [LB98].

De nombreux algorithmes ont été développés pour l'utilisation des HMM. En prédiction de gènes, le plus utilisé est l'algorithme de Viterbi. Il permet, à partir d'un modèle dont les paramètres ont été estimés, de trouver la suite

des états cachés la plus probable pour une séquence d'ADN donnée. Cette suite des états cachés est alors, en un certain sens, la meilleure prédiction que l'on peut faire avec le modèle utilisé pour la séquence observée. À notre connaissance, cet algorithme est actuellement utilisé par tous les programmes qui se fondent sur les chaînes de Markov cachés pour prédire les gènes. Après avoir prédit les gènes grâce à l'algorithme de Viterbi, GENSCAN [BK97] attribue grâce à un autre algorithme (*Forward-Backward*) une probabilité pour chacun des exons prédits.

Un reproche souvent fait aux HMM est d'introduire implicitement une mauvaise modélisation des longueurs des régions codantes. C'est pourquoi, les modèles utilisés pour la prédiction de gènes sont en fait souvent une extension des HMM qui permet la modélisation explicite de la durée des séjours dans les états cachés. On trouve ces modèles désignés par de nombreux noms : *Hidden Semi-Markov Models*, *Explicit Duration State HMM*, *Generalized HMM* ou tout simplement *HMM*. Ils sont utilisés par exemple par GeneMark.hmm, GENSCAN et Genie [KHRE96]. La principale contrepartie de l'utilisation de ces modèles est d'augmenter beaucoup le coût des calculs.

A.2.4 Modélisation des signaux et détection des gènes à introns

Les CDS des eucaryotes supérieurs sont souvent morcelés sur le chromosome en de nombreux exons relativement courts séparés par des introns. Par exemple, chez *Arabidopsis thaliana* il y a en moyenne environ 5 exons par CDS d'une longueur moyenne d'environ 250 nucléotides, la longueur moyenne des introns étant d'environ 170 nucléotides [The00]. Les introns sont excisés lors de l'épissage de l'ARN pré-messager qui donne ainsi naissance à l'ARN messager dans lequel les exons sont bout à bout et dans la même phase de lecture. L'excision des introns est un phénomène complexe dans lequel interviennent plus de 50 molécules (protéines et de petits ARN nucléaires) qui constituent le *spliceosome*. Presque tous les introns commencent par GT et finissent par AG, mais cette information ne suffit pas au *spliceosome* pour choisir les sites d'épissage. La reconnaissance d'autres signaux autour de ces di-nucléotides joue un rôle important. La prédiction des sites d'épissage est un aspect essentiel de la prédiction *ab initio* des gènes à introns. Cette prédiction est non seulement nécessaire pour localiser précisément les frontières des exons lorsque ceux-ci sont révélés par leur composition mais peut aussi aider à détecter les exons courts.

La majorité des méthodes de reconnaissance des sites d'épissage repose essentiellement sur l'évaluation de l'adéquation des séquences des sites potentiels à un modèle probabiliste qui décrit position par position la composition en nucléotides des sites d'épissage réels. Lorsque l'apparition des nucléotides est modélisée comme indépendante entre positions adjacentes, il s'agit d'un modèle parfois appelé WAM (*Weighted Array Model*). Un exemple de

nucléotide	position								
	exon			intron					
	-3	-2	-1	+1	+2	+3	+4	+5	+6
A	$\underline{0,33}$	$\underline{0,60}$	0,08	0	0	$\underline{0,49}$	$\underline{0,71}$	0,06	0,15
C	$\underline{0,37}$	0,13	0,04	0	0	0,03	0,07	0,05	0,19
G	0,18	0,14	$\underline{0,81}$	$\underline{1}$	0	$\underline{0,45}$	0,12	$\underline{0,84}$	0,20
T	0,12	0,13	0,07	0	$\underline{1}$	0,03	0,09	0,05	$\underline{0,46}$

TAB. A.1 – Fréquences position par position des nucléotides autour du site donneur d'epissage chez l'Homme [BK97]. Ces fréquences correspondent aux paramètres d'un WAM pour la reconnaissance du site. Les fréquences des nucléotides majoritaires à chaque position sont soulignées.

composition en mono-nucléotides autour d'un site d'épissage est donné Tableau A.1. Lorsque, non seulement la position dans le site mais aussi les dépendances entre nucléotides adjacents sont prises en compte, ils sont parfois désignés WMM (*Weighted Matrix Model*). Une limite de ces modèles est de ne pas modéliser de dépendances entre positions non adjacentes du site. Une solution consiste à utiliser une collection de modèles, plutôt qu'un unique modèle. Dans ce cas, chaque modèle décrit un « type » de site qui correspond en fait à une corrélation des fréquences d'apparitions des nucléotides entre positions non adjacentes. Une procédure automatique, la MDD (*Maximal Dependance Decomposition*), a été proposée [BK97] pour construire la collection de modèles correspondant à un site d'épissage. Cette procédure est utilisée par GENSCAN et GeneSplicer [PLS01].

Ces modèles de la composition des sites d'épissages peuvent être introduits sous la forme de nouveaux états cachés dans un HMM. L'utilisation de ces HMM permet de réaliser une prédiction qui tient compte, non seulement des différences de composition entre régions codantes et non codantes, mais aussi des propriétés de séquence des sites d'épissage. Selon la structure du HMM les prédictions peuvent être de nature différentes. Le plus souvent, comme le font par exemple GENSCAN et HMMgene [Kro97], la structure complète des gènes est prédite. Cependant certains programmes ne garantissent pas que les exons qui composent le CDS prédit sont bien en phase les uns avec les autres, c'est le cas de la version de GeneMark.hmm pour les gènes à introns (pas de publication).

A.2.5 Prédire sans modéliser la séquence

D'une façon très générale toute méthode de prédiction nécessite la mise au point d'un critère pour distinguer les bonnes prédictions de l'ensemble des prédictions possibles. La modélisation probabiliste est un cadre qui permet de construire de façon assez directe ces critères tout en leur donnant

une signification. Cependant, il est parfois difficile de modéliser ce dont on souhaite tenir compte dans une prédiction. De nombreuses méthodes qui ne se fondent pas sur une modélisation probabiliste complète de la séquence d'ADN ont été développées pour la prédiction de gènes. En s'affranchissant de la modélisation complète de la séquence, ces méthodes sont confrontées au problème de la recherche du bon critère. Il s'agit souvent de combiner des informations provenant de différentes mesures ou « senseurs ».

Pour « sentir » la texture ou la présence d'un signal d'épissage, les mesures font souvent intervenir les probabilités d'apparition de la séquence sous des modèles probabilistes. Par exemple, le programme de prédiction des sites d'épissage GeneSplicer, combine des mesures de textures codante et non codante, de part et d'autre du site, avec une mesure du signal d'épissage. Dans le cas de GeneSplicer, il est possible d'avoir une idée assez claire de la bonne façon de construire le critère à partir des différentes mesures. Cependant, la construction du critère se fait bien souvent en se plaçant dans un cadre méthodologique particulier qui définit le type de critère et propose des algorithmes pour optimiser la qualité des prédictions sur un ensemble d'apprentissage.

Quelques exemples vont permettre d'illustrer la diversité des méthodes employées. Pour prédire des sites d'épissage, NetPlantGene [HKT+96] utilise des réseaux de neurones artificiels et fait une prédiction directement à partir de la séquence d'ADN. SplicePredictor [KHV+96], quant à lui, combine la mesure du signal d'épissage par des modèles probabilistes à deux autres mesures dont le but est de percevoir l'environnement du signal : variation de la composition en G+C et en pyrimidines (C+T) de part et d'autre du site potentiel. La combinaison de ces mesures est réalisée et optimisée dans le cadre d'un modèle logistique. Le programme MZEF [Zha97] utilise pour la détection des exons humains 9 mesures différentes. L'optimisation de l'utilisation de ces 9 mesures est faite par analyse discrimante quadratique alors que pour une tâche proche GRAIL [UM91] utilise un réseau de neurones artificiels.

L'élaboration d'un critère ne suffit pas pour permettre la prédiction de la structure d'un gène composé de multiples exons. Il faut ensuite rechercher parmi toutes les combinaisons d'exons possibles celles qui satisfont au mieux le critère choisi. Le nombre immense de combinaisons interdit de les évaluer une à une. Sous réserve de certaines conditions sur le critère choisi, les algorithmes dits de programmation dynamique permettent d'explorer rapidement l'espace de toutes les combinaisons pour en extraire les meilleures. L'algorithme de Viterbi pour les HMM en est un exemple. C'est en utilisant un algorithme de programmation dynamique que la version de GLIMMER adaptée à la détection des gènes à introns [SPD+99], recherche les structures de gènes qui maximisent le critère correspondant à la somme des mesures de la texture codante des exons et de la qualité des sites d'épissages.

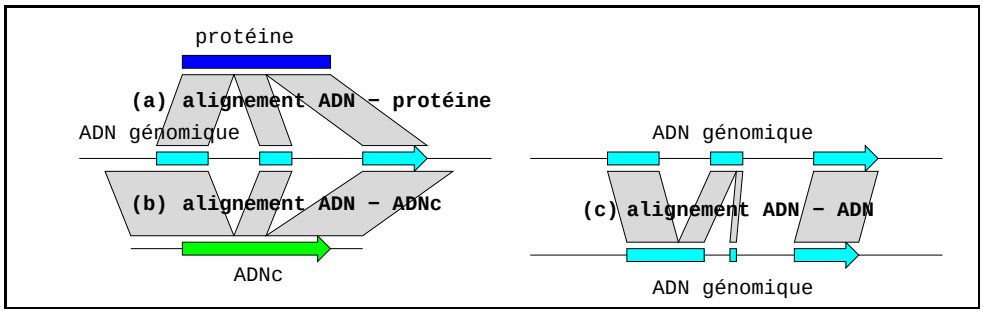

FIG. A.3 – Les trois grandes approches extrinsèques pour la prédiction de gènes : **(a)** alignement ADN génomique - protéine, **(b)** alignement ADN génomique - ADNc, **(c)** alignement ADN génomique - ADN génomique. Les parallélogrammes réprésentent les alignements sur lesquels reposent les différentes approches.

A.3 Méthodes extrinsèques

Contrairement aux méthodes intrinsèques, les méthodes extrinsèques utilisent des informations extérieures à la séquence elle-même pour prédire les gènes. La plupart des algorithmes développés prennent en compte un des trois types d'informations extérieures représentés Figure A.3 : séquences protéiques connues, séquences d'ARN messagers ou séquences d'ADN. Dans ces trois cas, il s'agit fondamentalement d'un problème d'alignement entre la séquence d'ADN où l'on cherche les gènes et une séquence qui lui ressemble plus ou moins. L'alignement est souvent amélioré par une sélection de sites d'épissage potentiels.

A.3.1 Comparaison ADN-protéine

Dans le cas de la similitude ADN-protéine l'idée est de prédire les CDS qui codent pour des protéines proches de protéines déjà contenues dans les banques de données. BLASTX [GS93] et FASTX [PWZM97] permettent de mettre en évidence des similitudes par alignement local entre une séquence d'ADN et une banque de séquences protéiques. Cependant, pour prédire la structure d'un CDS composé de multiples exons, il faut aligner la séquence d'ADN et la séquence protéique en tenant compte du fait que les introns constituent de longs fragments de la séquence d'ADN qui ne s'alignent pas avec la séquence protéique. PROCRUSTES [GMP96] a été le premier programme développé pour résoudre ce problème dit d'alignement épissé.

A.3.2 Comparaison ADN-ARNm

Le séquençage de l'ARNm par l'intermédiaire de l'ADNc (ADN copie de l'ARNm) est une technique expérimentale de détermination de la séquence des gènes transcrits. Le séquençage ayant lieu après l'épissage, ces séquences permettent de déterminer très précisément la structure introns-exons des gènes. Le séquençage à grande echelle des ARNm produit des séquences dites EST (*Expressed Sequence Tags*). On peut voir ce séquençage comme une technique de détection expérimentale des gènes. Cependant, déterminer la structure des gènes sur l'ADN à partir des EST peut s'avérer délicat car ces séquences ont pour caractéristiques d'être généralement incomplètes (souvent séquençage en 3'), de mauvaise qualité (séquençées une seule fois) et redondantes (fonction de l'abondance de l'ARNm). L'alignement ADN-ARNm est aussi parfois utilisé pour prédire des gènes alors que la séquence de l'ARNm a été obtenue dans un organisme différent. Des algorithmes d'alignements ADN-ADNc et ADN-EST tels que SIM4 [FHZ+98] et GeneSequer [UZB00] ont été développés pour automatiser ces taches.

A.3.3 Comparaison ADN-ADN

Une approche que l'on peut qualifier d'extrinsèque et pourtant *ab initio* émerge. Elle consiste à prédire un couple de génes qui se ressemblent simultanément sur deux séquences d'ADN. Actuellement, les programmes qui prédisent ainsi la structure complète des gènes se fondent essentiellement sur la conservation souvent bien meilleure des séquences codantes par rapport à celle des non codantes. Un exemple est le programme ROSETTA [BPM+00]. D'autres critères tels que la comparaison des taux de mutations synonymes et non synonymes devrait permettre de distinguer, même au sein des séquences conservées, les parties codantes des autres [NML01], [RDM99].

A.4 Succès et limites actuelles de la prédiction de gènes

A.4.1 Illustration de la complexité des facteurs du succès d'un détecteur de gènes

On peut tenter d'expliquer certaines des raisons pour lesquelles les HMM ont mis du temps à révéler leur supériorité pour la prédiction *ab initio* des gènes bactériens. Chronologiquement, le programme EcoParse [KMH94] proposa un HMM pour la détection des gènes d'*Escherichia coli* peu après les premières versions du programme GenMark [BM93] qui lui utilise une fenêtre glissante. Cependant, le HMM d'EcoParse ne prenait pas en compte certaines des caractéristiques participant au succès de GenMark. Il s'agit tout d'abord de la présence des gènes sur les deux brins qui évite que l'empreinte d'un

gène sur le brin complémentaire soit parfois prédit comme un gène sur le brin étudié. Ensuite, GenMark a proposé rapidement de prendre en compte l'existence d'une proportion relativement importante de gènes de composition atypique (riches en A et T), non prédits autrement [BMK+95]. Il fallut attendre GeneMark.hmm [LB98], pour qu'un HMM prenne en compte ces deux caractéristiques et se révele meilleur que GenMark.

A.4.2 Diversité et intégration des approches

Les approches évoquées dans ce chapitre abordent différents aspects de la détection de gènes à travers la prédiction des exons codants, des sites d'épissages ou de la structure complète des gènes. Pour ces prédictions, différentes informations sont prises en compte par les différentes méthodes : composition, similitudes de séquences, données expérimentales. Les types d'algorithmes et de modèles employés sont eux aussi variés et n'ont pas toujours un rapport direct avec la qualité des prédictions. La comparaison des performances des différentes approches est un travail fastidieux et difficile. La difficulté vient en partie de la diversité des informations prises en compte comme de la diversité de la nature des prédictions réalisées (structure complète des CDS, exons, sites d'epissage). Cette difficulté est encore accrue par la diversité des espèces pour lesquelles elles ont été conçues. En effet, tout comme la qualité des méthodes intrinsèques dépend clairement de leur ajustement à une espèce donnée, celles des méthodes extrinsèques dépend du degré de similitude entre les séquences comparées. Dans tous les cas, comme en témoigne le nombre d'approches développées, la difficulté du problème fait que jusqu'à présent aucune méthode n'est pleinement satisfaisante.

Il semble que les méthodes intrinsèques *ab initio* prédisent environ 80% des exons codants correctement chez *A. thaliana* [PRD+99] et moins chez les mammifères [GAA+00] [RMO01]. Le chiffre de 80% peut paraitre assez élevé mais on en déduit que la probabilité de prédire parfaitement un CDS composé de 5 exons est faible $0.8^5 = 33\%$ alors celle de ne prédire aucun des exons correctement est très faible $(1 - 0.8)^5 = 0.03\%$. On s'attend donc à une proportion faible de gènes complétement oubliés mais à une proportion importante de gènes annotés de façon incorrecte. Bien sûr, ce calcul est trop simple car les exons d'un gène sont rarement prédits et annotés indépendamment les uns des autres. De plus, il ne prend pas en compte l'un des problèmes important de ces prédictions : la difficulté de prédire les frontières entre gènes. Il est en effet difficile de distinguer les régions intergéniques des introns, en l'absence de méthode efficace de détection d'éléments tels que les promoteurs et les signaux de poly-adénylation.

Lors de l'annotation d'un génome complet les CDS sont mieux prédits que ne le laisse supposer ces petits calculs. En effet, l'annotation ne repose qu'en partie sur des méthodes *ab initio* intrinsèques et les informations provenant de similitudes et de données expérimentales peuvent beaucoup

améliorer ces chiffres. Il y a deux grandes voies d'intégration des différentes approches. La première consiste utiliser séparément les programmes puis à réaliser un post-traitement des résultats. La seconde consiste à développer des programmes qui fondent leurs prédictions simultanément sur des critères intrinsèques et extrinsèques. Aujourd'hui, la prise en compte d'informations extrinsèques par les méthodes intrinsèques est une tendance générale qui donne lieu à des développements méthodologiques et à des résultats intéressants [KFDB01] [YLB01]. Une voie intermédiaire est empruntée par le programme EuGène [SMR01], pour *Arabidopsis*, qui met en oeuvre une stratégie originale d'intégration de prédictions de natures différentes, réalisées des programmes indépendants.

A.4.3 Vers une bonne annotation des séquences génomiques ?

Malgré l'utilisation de toutes ces sources d'informations pour annoter les CDS, l'incertitude de ces annotations est souvent insuffisamment soulignée lors de leur publication. Ces annotations sont ensuite progressivement améliorées, en grande partie grâce à l'augmentation des performances des méthodes extrinsèques, résultant de l'augmentation du nombre de séquences connues dont celles provenant des projets de séquençage d'ARNm.

Parmi ces projets de séquençage d'ARNm, le séquençage de bonne qualité et à grande échelle d'ADNc complet (*full length cDNA*) semble prometteur. Pour donner un exemple de l'ampleur des modifications d'annotations, chez *Arabidopsis* parmi plus de 5000 séquences d'ADNc complets, 62% correspondaient parfaitement à des CDS annotés, 33% à des CDS qui devaient être corrigés et 5% à de probables nouveaux gènes [HVT+02]. Chez *Drosophila melanogaster* une révision de l'annotation a conduit à la modification de 45% des CDS [MCM+02]. En dehors de la correction de l'annotation des CDS, un autre intérêt des séquences d'ADNc est de permettre des annotations qui ne peuvent être aujourd'hui réalisées de façon fiable par des approches purement *in silico* : annotation des exons non codants, localisation du début de la transcription, détection de phénomènes d'épissage alternatif [ML02].

Grâce à sa sensibilité, la prédiction de gènes *in silico* à partir de la séquence génomique devrait cependant rester essentielle, notamment pour l'étude des CDS peu ou rarement exprimés. De plus, elle est rapide et peu couteuse dès lors que la séquence génomique est disponible. Enfin, les performances des approches *in silico* augmenteront probablement avec la quantité de données d'apprentissage, le nombre de séquences disponibles et les connaissances biologiques. On peut aussi espérer des progrès méthodologiques. Actuellement, le succès relatif de la prédiction de gènes *ab initio* repose en grande partie la texture tri-périodique des CDS. Or, cette texture n'est qu'une conséquence de la traduction en protéines. C'est une des raisons pour lesquelles la prédiction des gènes qui ne codent pas pour des protéines est pour l'instant très difficile.

Pour plus d'informations sur la prédiction de gènes, le lecteur intéressé est invité à se reporter aux synthèses récentes de Zhang [Zha02] et de Mathé *et al.* [MSSR02]. Il existe aussi un site web qui recense de nombreuses publications sur le sujet (http ://linkage.rockefeller.edu/wli/gene/).

Annexe B

SHOW User Manual

Pierre Nicolas, Anne-Sophie Tocquet et Florence Muri

B.1 Introduction

SHOW stands for Structured HOmogeneities Watcher. It is a set of programs implementing different uses of Hidden Markov Models (HMMs) for DNA sequences. SHOW enables self-learning of HMM on a set of sequences, sequence segmentation based on the Baum-Welch or the Viterbi algorithms, and sequence simulation under a given HMM. We have designed these programs to allow the user to specify any highly structured model and also to process large sets of sequences. To date it has been successfully used in diverse tasks such as DNA segmentation in homogeneous segments, bacterial gene prediction and human splice sites detection.

The three following programs are available :

- *show_emfit* enables to fit an HMM on sequences using EM algorithm (learning) and to reconstruct the hidden state path using forward-backward algorithm (segmentation). When used with fixed parameters, *show_emfit* only produces the sequence segmentation with the forward-backward algorithm.
- *show_viterbi* implements the viterbi algorithm to find the most probable hidden path given the observed sequence (segmentation). The HMM parameters can first be learned with *show_emfit*.
- *show_simul* enables to simulate a hidden state sequence and a DNA sequence under a specified HMM.

All these programs share a same format for the HMM specification file which is presented in the first section. The three following sections present detailed explanations of the different executables and how to use them. Section B.6 intends to deal with the source code design in order to facilitate further developments of the software. Finally, section B.7 presents a Perl script making easy the use of SHOW for bacterial gene finding.

The source code of SHOW is freely available, this software is protected by the GNU Public Licence. Installation instruction can be found in the INSTALL file of the distribution.

Keywords : DNA segmentation, Hidden Markov Models, maximum likelihood estimation, EM algorithm, Viterbi algorithm, Baum-Welch algorithm, forward-backward algorithm, HMM simulation, gene detection, DNA sequence heterogeneity

B.2 Hidden Markov Models, HMMs

B.2.1 SHOW's HMMs for DNA sequences

HMMs are basically implemented in the same way in SHOW and RHOM [NBM+02].

We note $X_1^n = X_1, X_2, \ldots, X_n$ the observed DNA sequence with $X_t \in \mathcal{X} = (a, g, c, t)$ and $S_1^n = S_1, S_2, \ldots, S_n$ the corresponding hidden state path, each S_t taken from a finite set $\mathcal{S} = (1, \cdots, q)$ defined by the user.

The S_t are generated according to a first order Markov chain of transitions

$$a(u, v) = P(S_{t+1} = v \mid S_t = u) \ , \quad u, v \in \mathcal{S}$$

The initial distribution of the chain is

$$a(v) = P(S_1 = v) \ , \quad v \in \mathcal{S}$$

Unlike the RHOM software, designs of the algorithms implemented in SHOW are optimized for large sparse HMMs where most of the transitions between hidden states are null.

The X_t are generated according to a markov model of order r_u which depends on the actual hidden state $S_t = u$. Transitions from the letters $x_{-r_u}, \ldots, x_{-1}$ to the letter x in state u are

$$b_u(x; w) = P(X_t = x \mid S_t = u, X_{t-r_u}^{t-1} = w) \ , \quad w \in \mathcal{X}^{r_u}, \ x \in \mathcal{X}, \ u \in \mathcal{S}$$

For the first $r_u - 1$ positions of the sequence, we will use for $0 \leq t < r_u$

$$b_u(x; w) = P(X_{t+1} = x \mid S_{t+1} = u, X_1^t = w) \ , \quad w \in \mathcal{X}^t, \ x \in \mathcal{X}, \ u \in \mathcal{S}$$

In the source code, the state transitions a and the emission observation transitions b will be denoted respectively by *ptrans* and *pobs*.

B.2.2 Example of a simple model for gene detection

Figure B.1 displays the structure of a simple HMM (represented by the hidden state transitions) for which SHOW has been designed. Circles represent the considered states of the HMM and arrows the allowed transitions between states. This graph modelises the alternation of intergenic regions with CoDing Sequences (CDS) on the both strands of a DNA molecule. The model contains 23 hidden states grouped in 7 groups (in dotted line). Here follows a short description of the biological meaning of this graph (we only give details of the model structure).

- When the actual hidden state corresponds to an intergenic region at position t, the arrows indicate that we can stay in this region at position $t + 1$ or leave it towards the first position of a start codon on the $(+)$ strand (triplets : atg, gtg, ttg) or towards a last position of a stop codon on the $(-)$ strand (inverse complementary of tga, tag, taa). Intergenic state can only be reached from the last position of a stop codon on the $(+)$ strand or the first position of a start codon on the $(-)$ strand.

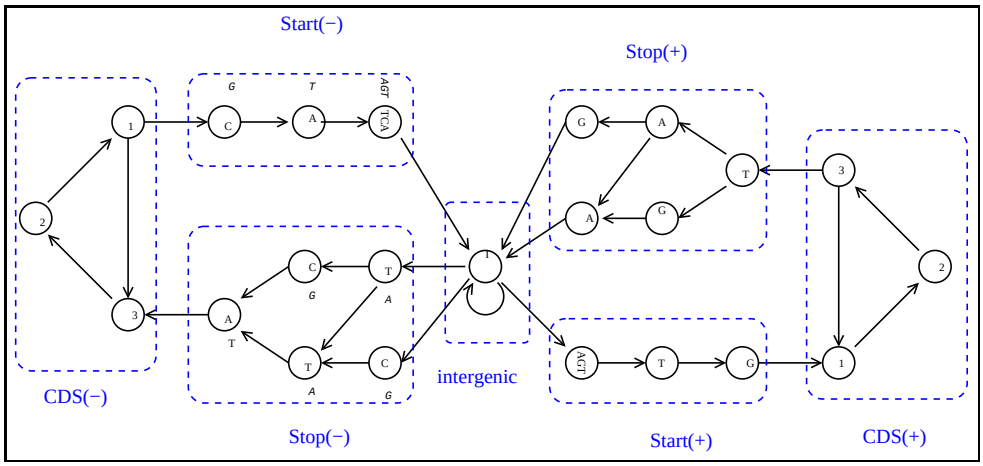

FIG. B.1 – Example of a simple HMM dedicated to bacterial coding sequences detection

- Leaving the third position of a start codon on the (+) strand, the hidden path goes through a CDS which is a succession of codons. Codons are modelised by using a cycle of three hidden states, one for each position inside the codon. This modelisation ensures that the length of the CDS will be a multiple of three and enables to take into account distinct compositions of the DNA according to the codon position (defined by the emission transitions b which are not described here).
- At the third codon position on the (+) strand, it is necessary to forbid the appearance of an in frame stop codon. Thus, the emission model associated with the third position needs to verify : $b_{\text{CDS+\_3}}(a; tg) = P(X_t = a \mid S_t = \text{CDS+\_3}, X_{t-2}^{t-1} = tg) = 0$, $b_{\text{CDS+\_3}}(g; ta) = P(X_t = g \mid S_t = \text{CDS+\_3}, X_{t-2}^{t-1} = ta) = 0$ and $b_{\text{CDS+\_3}}(a; ta) = P(X_t = a \mid S_t = \text{CDS+\_3}, X_{t-2}^{t-1} = ta) = 0$.
- The first position of the stop codon on the (+) strand can only be reached from the third position of a translated codon and corresponds to the first nucleotide of the triplets : tga, tag, taa.
- From the third stop codon position on the (+) strand, the path goes through intergenic.
- From the intergenic state, the third position of a stop codon on the (−) strand could be reached. This transition enable the beginning of a CDS on the (−) strand.

In the next section, we will present the syntax allowing the definition of such HMM.

B.2.3 HMM specification file : -model <file>

Model specification consists in the specification of each hidden state (state and emission observation transitions). It corresponds to the description of each of the nodes of the graph. This file does not only contain the description of the model structure, but also indicates which parameters of the model (b and a) are fixed or must be estimated when using the model as input of the show_emfit executable. When running show_emfit, values found in this file for the parameters described as "to be estimated" are used as the starting point of the iterative EM algorithm.

Hidden state definition

The HMM definition file is organized as a succession of hidden state definitions that makes it highly modular, easy to edit by 'copy/paste' operations and makes an existing model easy to extend. The following shows an example of a hidden state which could be the intergenic hidden state of the figure B.1 definition :

```
BEGIN_STATE
state_id: intergenic                        # identifier of the state
    BEGIN_TRANSITIONS
       type: 1                    # allows estimation of the transition
       state: start+_1            # transition towards start+_1 state
       ptrans: 0.00432589         # probability of this transition
       type: 1
       state: stop-_3*1
       ptrans: 0.00426073
       type: 1
       state: stop-_3*2
       ptrans: 0.000892905
       type: 1
       state: intergenic
       ptrans: 0.99052
    END_TRANSITIONS
    BEGIN_OBSERVATIONS
       seq: genomic_dna   # a sequence id which must be the same as in the -seq <file>
       type: 1            # allows estimation of the observation distribution
       order: 2           # markov order of the observation distribution
       pobs:
              0.314514 0.183587 0.185804 0.316094    # a  g  c  t

              0.385992 0.177465 0.146873 0.28967     # aa ag ac at
              0.327029 0.217127 0.207901 0.247942    # ga gg gc gt
              0.310991 0.158126 0.221846 0.309037
              0.23778 0.185229 0.190936 0.386055

              0.44321 0.181248 0.140779 0.234764    # aaa aag aac aat
              0.33124 0.25435 0.217354 0.197056     # gaa gag gac gat
              0.382986 0.161845 0.2005 0.25467
              0.303587 0.204104 0.178921 0.313388
              0.38761 0.202831 0.138278 0.27128
              0.342915 0.224382 0.201373 0.231331
              0.290107 0.172784 0.214286 0.322822
              0.255216 0.193496 0.177495 0.373793
              0.326314 0.198495 0.157725 0.317467
```

```
            0.260414 0.258744 0.223338 0.257503
            0.22166 0.186155 0.233092 0.359093
            0.184989 0.211447 0.218147 0.385417
            0.334371 0.134578 0.155746 0.375305
            0.345444 0.155348 0.195448 0.30376
            0.328272 0.126919 0.235736 0.309073
            0.204668 0.155128 0.192557 0.447647  # tta ttg ttc ttt
   END_OBSERVATIONS
END_STATE
```

Hidden state description begins with the BEGIN_STATE keyword and ends with the END_STATE keyword. It contains the identifier of the state that is used in the description of the outgoing transitions and that must be unique. The state transition description is separated from the description of the emission observation transitions.

Outgoing transition description from the hidden state begins with the BEGIN_TRANSITIONS keyword and ends with the END_TRANSITIONS keyword. It contains the description of each allowed transition from the hidden state.

The type : must be specified for each outgoing transition; 0 means no estimation of the parameter and 1 means estimation (by the EM algorithm). The state : refers to the identifier of the state to which the transition is allowed. The ptrans : keyword preceeds a numeric value of the state outgoing transition probability $a(u, v)$. This numerical value is fixed when the type : is equal to 0, and corresponds to the initial value required to run EM when the type : is set to 1 (in this last case, the value of ptrans will evolve during iterations of EM).

The keyword label : can be set on the first line of the transition description. It enables to choose an identifier which can be used with the keyword tied_to : when setting up models with tied parameters. The example below shows how to use this feature.

```
BEGIN_STATE
state_id: cds1+_3
   BEGIN_TRANSITIONS
      label: trans_cds+_3  # identifier is trans_cds+_3
      type: 1
      state: cds1+_1
      ptrans: 0.99
      type: 1
      state: stop+_1
      ptrans: 0.01
   END_TRANSITIONS
   BEGIN_OBSERVATIONS
      seq: genomic_dna
      type: 1
      order: 2
      pobs:    random
      excepted: TGA TAG TAA
   END_OBSERVATIONS
END_STATE
```

```
BEGIN_STATE
state_id: cds2+_3
    BEGIN_TRANSITIONS
       tied_to: trans_cds+_3
       state: cds2+_1     # P(cds2+_3 -> cds2+_1) =  P(cds1+_3 -> cds1+_1)
       state: stop+_1     # P(cds2+_3 -> stop+_1) =  P(cds1+_3 -> stop+_1)
    END_TRANSITIONS
    BEGIN_OBSERVATIONS
       seq: genomic_dna
       type: 1
       order: 2
       pobs:    random
       excepted: TGA TAG TAA
    END_OBSERVATIONS
END_STATE
```

In this example the states cds1+_3 and cds2+_3 correspond to the third codon position of a model allowing two types of compositions for coding sequences. The outgoing transitions of this two states are tied : they are identical and simultaneously estimated when running show_emfit. This feature can be used to ensure that the state transition probabilities will be the same for two or more hidden states and can also be useful to decrease the number of parameters and then give an easier and better estimation.

Observation emission transition probabilities description conditionally on the hidden state begins with the keyword BEGIN_OBSERVATIONS and ends with the keyword END_OBSERVATIONS.

The first keyword found in the observation description must be seq : It gives an identifier that must be the same as the identifier given in the file referenced by the -seq argument of the command line (see the section 6 for more explanations).

The keyword type : indicates how the observation emission transition probabilities should be processed during estimation. Type 0 stands for no estimation (constant value) and type 1 means estimation. Types 2 and 3 must be used only for tied observations : 2 means identical to the referenced observation emission distribution, while 3 means complementary to the referenced one. Type 3 can only be used when the order r_u of the referenced observation emission distribution is 0. The observation emission transition probabilities will be estimated when type is set to 2 or 3 only if the referenced one are estimated.

The keyword order : is used to indicate the order r_u of the markov chain of the observation emission distribution. The keyword pobs : preceedes numerical values given by the user for the observation transition probabilities $b_u(x; x_{-r_u}^{-1})$. After pobs :, $\sum_{t=1}^{r_u+1} 4^t$ numerical values must be set to the values of $b_u(x; x_{-t}^{-1})$, for t increasing from 0 to r_u. The parameters $\left(b_u(x; x_{-t}^{-1})\right)_{0 \leq t \leq r_u}$ are used only at the beginning of the sequence, i.e. when no sufficient context is known to use the parameters $b_u(x; x_{-r_u}^{-1})$. These values correspond to the

starting point required to process EM when type is set to 1, and are fixed otherwise.

A random choice of the pobs values can be done by setting pobs : to the keyword random. In this case, the model cannot be used directly for viterbi reconstruction of the hidden path or for simulation, but must first be estimated using show_emfit. Note that when using directly show_viterbi or show_simul, the pobs : (and ptrans :) values will be considered as fixed for these executables, even if the type : of these parameters is set to 1. Thus, show_emfit must first be used to allow (some) parameter estimation.

Finally, the keyword excepted : can be used to forbid the emission of some words ; this keyword was used in the previous example to forbid an in frame stop codon tag, tga and taa. The length of the forbidden word must be $\geq r_u + 1$. If the length of a word l_w forbidden by the excepted : keyword is greater than $r_u + 1$, then SHOW uses a special kind of Markov model for the observation emission process. This Markov model corresponds to a model of order r_u conditionally on that some words of length l_w do not appear, the resulting model is in fact a Markov model of order $l_w + 1$ but with the same number of parameters than a Markov model of order r_u. Then we refer to this kind of Markov model as a Markov model of pseudo-order $l_w + 1$. The following matrix gives an example of the parameters of a Markov model of pseudo-order 1 corresponding to a Markov model of order 0 conditionally on that ag dinucleotide does not appear.

$$b'_u(\bullet;\bullet) = \begin{pmatrix} \frac{b_u(a)}{b_u(a)+b_u(c)+b_u(t)} & 0 & \frac{b_u(c)}{b_u(a)+b_u(c)+b_u(t)} & \frac{b_u(t)}{b_u(a)+b_u(c)+b_u(t)} \\ b_u(a) & b_u(g) & b_u(c) & b_u(t) \\ b_u(a) & b_u(g) & b_u(c) & b_u(t) \\ b_u(a) & b_u(g) & b_u(c) & b_u(t) \end{pmatrix}$$

The example below shows how to use the keywords label : and tied_to : in the observation description.

```
BEGIN_STATE
state_id: start+_1
    BEGIN_TRANSITIONS
       type: 0
       state: start+_2
       ptrans: 1
    END_TRANSITIONS
    BEGIN_OBSERVATIONS
       seq: genomic_dna
       label: start+_1_obs
       type: 1
       order: 0
       pobs:
             0.3 0.3 0 0.4 # a, g or t.
    END_OBSERVATIONS
END_STATE

BEGIN_STATE
```

```
state_id: start-_1
    BEGIN_TRANSITIONS
       type: 0
       state: intergenic
       ptrans: 1
    END_TRANSITIONS
    BEGIN_OBSERVATIONS
       seq: genomic_dna
       tied_to: start+_1_obs
       type: 3
    END_OBSERVATIONS
END_STATE
```

Two distinct modelisations of the boundaries of the sequence

SHOW allows two distinct modelisations of the sequence's boundaries.

The first one enables to work conditionally on the length of the sequence. In this case the sequence length is not modelised and the sequence begins and ends in any of the hidden states. This is the default modelisation when no 'bound' state is specified.

The alternative is to modelise the length of the sequence. This is done by imposing a state of which the identifier is set to bound. The state named bound corresponds to the beginning and the end of the sequence : the sequence begins when going out from the bound state and ends when reaching back the bound state. No observations are described in the bound state description. The description of the outgoing transitions from the bound state follows the same rules as any other state.

An application example of such modelisation is how to distinguish false and true sites corresponding to some signals. When presenting a potential site, the likelihood of a true and false model is computed and the decision of predicting a true or a false site is taken according to the likelihood ratio. Figure B.2 displays the graph corresponding to a HMM which can be used to predict a ten nucleotide length signal. This model enables to take into account some kinds of correlations along the motif corresponding to the signal. It could be learned using show_emfit on a learning set. The prediction will be done by computing the likelihood of the models corresponding to the true and false sites given the sequence when running show_emfit with the two models.

The description of the beginning and the end of the model corresponding to the figure B.2 is given below.

```
BEGIN_STATE
state_id: bound  # the 'bound' state contains no observation description
    BEGIN_TRANSITIONS
       type: 1
       state: motif_1*1
       ptrans: 0.5
       type: 1
       state: motif_1*2
       ptrans: 0.5
    END_TRANSITIONS
```

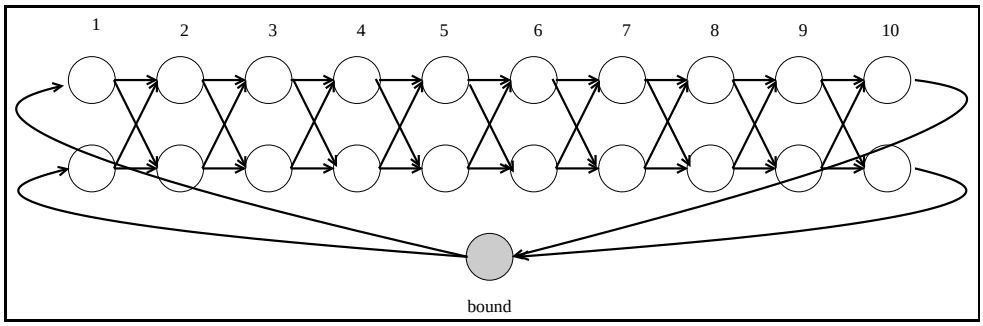

FIG. B.2 – Example of a HMM dedicated to a ten nucleotides length motif detection

```
END_STATE

BEGIN_STATE
state_id: motif_1*1
    BEGIN_TRANSITIONS
       type: 1
       state: motif_2*1
       ptrans: 0.5
       type: 1
       state: motif_2*2
       ptrans: 0.5
END_TRANSITIONS
BEGIN_OBSERVATIONS
       seq: genomic_dna
       type: 1
       order: 0
       pobs: random
END_OBSERVATIONS
END_STATE

BEGIN_STATE
state_id: motif_1*1
    BEGIN_TRANSITIONS
       type: 1
       state: motif_2*1
       ptrans: 0.5
       type: 1
       state: motif_2*2
       ptrans: 0.5
    END_TRANSITIONS
    BEGIN_OBSERVATIONS
       seq: genomic_dna
       type: 1
       order: 0
       pobs: random
END_OBSERVATIONS
END_STATE

           .
           .
           .
           .
```

```
BEGIN_STATE
state_id: motif_10*1
    BEGIN_TRANSITIONS
       type: 0
       state: bound
       ptrans: 1
    END_TRANSITIONS
    BEGIN_OBSERVATIONS
       seq: genomic_dna
       type: 1
       order: 0
       pobs: random
    END_OBSERVATIONS
END_STATE

BEGIN_STATE
state_id: motif_10*2
    BEGIN_TRANSITIONS
       type: 0
       state: bound
       ptrans: 1
    END_TRANSITIONS
    BEGIN_OBSERVATIONS
       seq: genomic_dna
       type: 1
       order: 0
       pobs: random
    END_OBSERVATIONS
END_STATE
```

B.2.4 Observed sequences file list : -seq <file>

The SHOW executables can work on a single sequence or a set of sequences. The sequences to process are referenced in the -seq <file>. An example of such a file is given below. It must contain the three keywords seq_identifier, seq_type and seq_files.

```
seq_identifier: genomic_dna
seq_type: dna
seq_files:
           ContigI.dna
           ContigII.dna
```

Keyword seq_identifier refers to any chosen identifier of the sequence, that must be the same as in the -model <file> (keyword seq : in observation emission description). The seq_type corresponds to the nature of the observed sequence, currently only dna is properly supported. The seq_files are the name of the files containing sequences to be analyzed. Sequences in GenBank and Fasta format are accepted. In order to process a sequence set, it is possible to store all the sequences in the same Fasta file or to use multiple files, each of them referenced in the -seq <file>.

When processing a set of sequences, the sequences are considered as independent realisations of a same HMM.

B.3 The show_emfit executable

The show_emfit executable simultaneously enables to estimate the parameters of the model by likelihood maximization and to segment the sequence using the EM algorithm. The show_emfit executable can also be used with fixed parameters to segment the sequences according to the Baum-Welch algorithm or to compute the likelihood of a model for a given sequence.

B.3.1 EM algorithm / Baum-Welch algorithm

This section describes the EM algorithm implemented in SHOW. We denote by θ the whole parameters of the HMM (state and observation transitions a and b) that we want to estimate (some parameters can of course be fixed). Given a starting value $\theta^{(0)}$ of the parameters, the EM algorithm is an iterative procedure that produces an approximation of the maximum likelihood estimation (MLE) $\theta^{(m)}$ that is updated at each iteration m. This procedure ensures the increase of the likelihood at each iteration $m : P_{\theta^{(m+1)}}(X) \geq P_{\theta^{(m)}}(X)$.
EM consists in alternating two steps : the so-called E-step (for Expectation) and M-step (for Maximization).

E-step / forward-backward algorithm

The E-step on a HMM is also named Baum-Welch or forward-backward algorithm. It computes the probability values $P_{\theta^{(m-1)}}(S_t = u, S_{t+1} = v \mid X_1^n = x_1^n)$ for each position of the sequence $t \in [1, \ldots, n-1]$ and each couple of hidden states $(u, v) \in [1, \ldots, q]^2$. Values of $P_{\theta^{(m-1)}}(S_t = u \mid X_1^n = x_1^n)$ are further deduced from $P_{\theta^{(m-1)}}(S_t = u, S_{t+1} = v \mid X_1^n = x_1^n)$.

The forward processing of the sequence starts with $t = 1$ and use recursively $(t = 2, \cdots, n)$ both following equations until the calculation of the value $P_{\theta^{(m-1)}}(S_n = v | x_1^n)$:

Predictive equation (1) :

$$\text{if} \quad t > 1 \quad P_{\theta^{(m-1)}}(S_t = v | x_1^{t-1}) = \sum_{u=1}^{q} a^{(m-1)}(u, v) P_{\theta^{(m-1)}}(S_{t-1} = u \mid x_1^{t-1})$$

$$\text{else} \quad P_{\theta^{(m-1)}}(S_1 = v) = a^{(m-1)}(v)$$

Filtring equation (2) :

$$P_{\theta^{(m-1)}}(S_t = v | x_1^t) = \frac{b_v^{(m-1)}(x_t; x_{t-r_v}^{t-1}) P_{\theta^{(m-1)}}(S_t = v, \mid x_1^{t-1})}{\sum_{u=1}^{q} b_u^{(m-1)}(x_t; x_{t-r_v}^{t-1}) P_{\theta^{(m-1)}}(S_t = u, \mid x_1^{t-1})}$$

For the first $r_u - 1$ positions of the sequence, the probabilities $b_v^{(m-1)}(x_t; x_1^{t-1})$ are used instead of the transitions $b_v^{(m-1)}(x_t; x_{t-r_u}^{t-1})$.

The backward processing of the sequence starts from $t = n$ and use recursively $(t = n-1, \ldots, 1)$ the equations :

Smoothing equation (3) :

$$\begin{aligned} & P_{\theta^{(m-1)}}(S_{t-1} = u, S_t = v \mid x_1^n) \\ = \; & \frac{a^{(m-1)}(u,v) P_{\theta^{(m-1)}}(S_{t-1} = u \mid x_1^{t-1}) P_{\theta^{(m-1)}}(S_t = v \mid x_1^n)}{P_{\theta^{(m-1)}}(S_t = v \mid x_1^{t-1})} \end{aligned}$$

Equation (4) :

$$P_{\theta^{(m-1)}}(S_{t-1} = u \mid x_1^n) = \sum_{v=1}^{q} P_{\theta^{(m-1)}}(S_{t-1} = u, S_t = v \mid x_1^n)$$

M-step

M-step consists in updating the parameter θ by choosing

$$\theta^{(m)} = \arg\max_{\theta} E_{\theta^{(m-1)}}\big(\log P_\theta(X_1^n, S_1^n) \mid X_1^n\big)$$

i.e. the conditional expectation (to the current candidate $\theta^{(m-1)}$ and the sequence X_1^n) of the complete loglikelihood. This maximization step uses the segmentation obtained during the E-step and leads to the intuitive estimators.
Equation (5)

$$a^{(m)}(u,v) = \frac{\sum_{t=2}^{n} P_{\theta^{(m-1)}}(S_{t-1} = u, S_t = v \mid x_1^n)}{\sum_{t=2}^{n} P_{\theta^{(m-1)}}(S_{t-1} = u \mid x_1^n)}$$

and

$$b_v^{(m)}(x; w) \quad = \quad \frac{\sum_{t=r_v+1}^{n} P_{\theta^{(m-1)}}(S_t = v \mid x_1^n) 1_{\{X_t = x, X_{t-r_v}^{t-1} = w\}}}{\sum_{t=r_v+1}^{n} P_{\theta^{(m-1)}}(S_t = v \mid x_1^n) 1_{\{X_{t-r_v}^{t-1} = w\}}}$$

In the special case of a state u with an observation emission Markov process with pseudo-order (See Section) the maximization of $b_u(\ldots)$ could not be performed analytcally. Thus, this maximization is performed using a multidimensional maximization routine of the GSL (GNU Scientific Library).

Computation of the loglikelihood

The computation of the incomplete loglikelihood is performed recursively (for $t = 1, \ldots, n$) during the E-step using equation :

$$\begin{aligned}\log\Big(P_{\theta^{(m-1)}}(X_1^t = x_1^t)\Big) &= \log\Big(\sum_{v=1}^{q} b_v^{(m-1)}(x_t; x_{t-r_v}^{t-1}) P_{\theta^{(m-1)}}(S_t = v | x_1^{t-1})\Big) \\ &\quad + \log\Big(P_{\theta^{(m-1)}}(X_1^{t-1} = x_1^{t-1})\Big)\end{aligned}$$

Precautions must be taken for the $r_v - 1$ first positions.

Stopping criteria for EM

It can be shown that every limit point of the sequence $(\theta^{(m)})_{m\geq 0}$ generated by EM (alternating the E and M steps previously described) satisfies the incomplete log-likelihood equations and that $(\theta^{(m)})_{m\geq 0}$ converges towards the maximum likelihood estimate (MLE) if the starting point $\theta^{(0)}$ is not too far from the true value. From a practical point of view, EM can thus converge to a local maximum.

The E and M steps are alternated until an iteration M for which convergence can be stated. The stopping rule is $|\log P_{\theta^{(M+1)}}(X) - \log P_{\theta^{(M)}}(X)| \leq \epsilon$, for a given ϵ (the value of ϵ is fixed by the user). The parameters θ are then estimated by $\theta^{(M)}$ ($a^{(M)}$ and $b^{(M)}$), and for all positions t in the sequence, the probability of the state S_t to be $v = 1, \cdots, q$ is estimated by $P_{\theta^{(M)}}(S_t = v \mid X)$.

Memory saving approximation

As the forward-backward algorithm requires the storage of the probabilities $P_{\theta^{(m-1)}}(S_t = u \mid X_1^{t-1})$ and $P_{\theta^{(m-1)}}(S_t = u \mid X_1^t)$ during the forward step, this procedure can consume a large amount of active memory when processing large sequences with a big model. A solution could be to use hard memory (files) to store this probabilities.

Another solution that we think more effective in terms of execution speed have been implemented in SHOW. It consists in approximating $P_{\theta^{(m-1)}}(S_t = u \mid X_1^n)$ by $P_{\theta^{(m-1)}}(S_t = u \mid X_s^{s'})$ where s and s' are boundaries of overlapping segments of the sequence. This approximation is mathematically justified because the dependence between two positions in a Markov chain decreases geometrically with the distance separating the two positions.

Bypassing local maxima of the likelihood function

One of the major drawbacks of the EM algorithm is that it can be stuck in local maxima of the likelihood function.

The implemented solution consists in starting EM with distinct initial value $\theta^{(0)}$ and to choose the best path which leads to the highest likelihood value. As likelihood rapidly increases during the first iterations of the algorithm, it is interesting to stop the maximization after a few iterations and then to choose the model for which the likelihood maximization procedure is performed further. The figure B.3 shows the likelihood increase during such a maximization performed from ten random starting points.

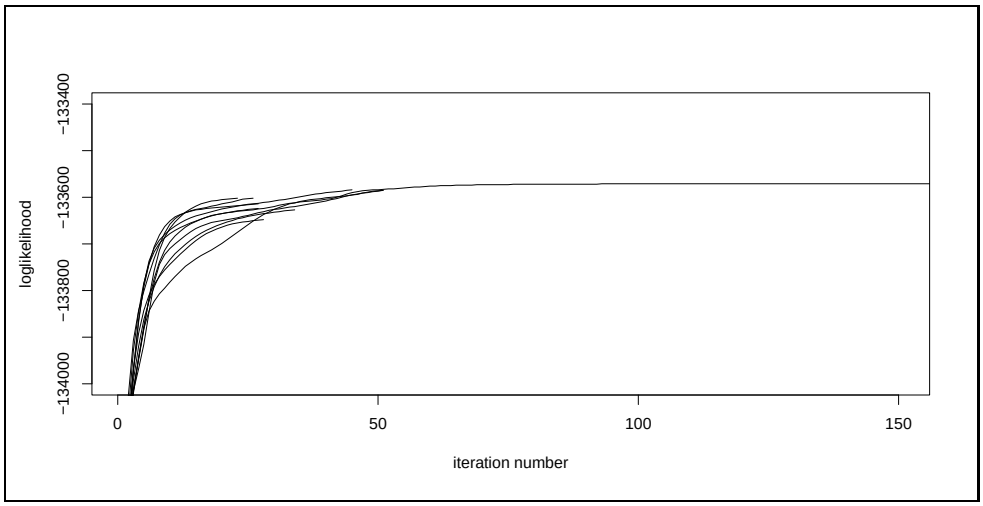

FIG. B.3 – Example of likelihood increase during EM iterations. The loglikelihood maximization begins with 10 starting points (during 50 iterations), only the likelihood of the best model is performed further than 50 iterations

B.3.2 Input files

-model <file>

The syntax of this file is described in the section B.2.3.

-em <file>

This file contains the information required to initialize and run the EM algorithm. An example of such a file is given below.

```
estep_segment: 20000
estep_overlap: 1000
niter: 1000
epsi: 0.00001
```

Keywords estep_segment and estep_overlap refer to the length of the segment used for the memory saving approximation during the forward-backward algorithm (E-step). For a given model and set of sequences, the memory needed by the program grows linearly with estep_segment. niter and epsi define the stopping criteria of the EM-algorithm : EM iterations stop when the loglikelihood increase between two consecutive iterations is lower than epsi or the maximal number of iterations niter has been reached. If parameters of the emission observation process associated to some hidden states are randomly initialized (i.e. model definition file contains pobs : random), supplementary keywords nb_sel, niter_sel and eps_sel are expected in the -em <file>. These supplementary keywords are used in the following example.

```
estep_segment: 20000
estep_overlap: 1000
nb_sel: 3
niter_sel: 100
eps_sel: 0.01
niter: 1000
epsi: 0.0001
```

nb_sel corresponds to the number of random starting points for the EM algorithm. niter_sel refers to the maximal number of iterations performed from each starting point. eps_sel corresponds to the stopping criteria of the EM algorithm running from each starting point.

-seq <file>

The syntax of this file is described in the section B.2.4.

Optional -output <file>

This file is optional and is required for the output .e file generation. The -output <file> contains the output of the last iteration I of the forward-backward algorithm, i.e. the probabilities $P_{\theta^{(M)}}(S_t = u \mid X_1^n)$ and $P_{\theta^{(M)}}(S_t = u, S_{t+1} = v \mid X_1^n)$ for each state u, v and each position t of the sequences. An output file containing these probabilities for all considered states can be very large and difficult to use and analyze. Then, a summary of selected states can be done.

The following example gives the syntax of such an output .e file :

```
(rhom1) (rhom2 ; rhom3) (rhom2 -> rhom3)
```

If rhom1, rhom2, rhom3 are three (hidden) states, the output file will contain three columns (of length n) corresponding to $P_{\theta^{(M)}}(S_t = \text{rhom1} \mid X_1^n)$,

$P_{\theta(M)}(S_t = \text{rhom2} \mid X_1^n) + P_{\theta(M)}(S_t = \text{rhom3} \mid X_1^n)$ and $P_{\theta(M)}(S_t = \text{rhom2}, S_{t+1} = \text{rhom3} \mid X_1^n)$. Note that a same probability cannot be summed in two columns.

B.3.3 Output files

Some files are created in the current directory when running show_emfit. The names of these files are generated by adding suffixes to the name of -seq <file>, excepted output files with the suffix .e which are created using the names of the individual sequence files referenced in -seq <file>. Files with suffixes containing .select are generated only if random initializations of the EM-algorithm are used. Files with the suffix .e are generated only if an -output <file> is specified.

.select.traces file

This file contains likelihoods of the models at each iteration of the EM-algorithm running from each starting point.

```
**************************************
model 0
iter 0 logl -71378.6
iter 1 logl -68449.2 diff 2929.44
iter 2 logl -68067.4 diff 381.803
iter 3 logl -67971.1 diff 96.317
iter 4 logl -67928.4 diff 42.6526
iter 5 logl -67903.5 diff 24.9115
iter 6 logl -67886.6 diff 16.8449
iter 7 logl -67874.3 diff 12.3817
iter 8 logl -67864.8 diff 9.50504
iter 9 logl -67857.3 diff 7.44326
iter 10 logl -67851.4
**************************************
model 1
iter 0 logl -71490.3
iter 1 logl -68347.4 diff 3142.88
       .
       .
       .
```

.select.likelihoods file

This file summarizes the final likelihoods obtained from each starting point and reports the model which has been further fitted during the final stage of the estimation.

```
model 0 loglikelihood -67851.4
model 1 loglikelihood -67883.3
model 2 loglikelihood -67901
```

```
best model found 0 loglikelihood -67851.4
```

.select.models file

This file contains the definition of the models obtained from each starting point ; Models are described using the same format as in the -model <file>.

.trace file

This file contains likelihoods of the models at each iteration of the EM-algorithm running from the model of the -model <file> if random initializations are not used or from the best model found after random initializations.

.model file

This file contains the final model obtained after the estimation of the parameters ; model is described using the same format as in the -model <file>.

.e file

These files are only generated if an -output <file> is specified. One file with the .e suffix is created for each of the sequence files referenced in the -seq <file>. For each sequence, the .e file contains output of the forward-backward algorithm as defined in the -output <file>. Each line corresponds to a position along the sequence.

```
# ( hello ) ( hello2 )
#
0.209391        0.790609
0.174707        0.825293
0.0927552       0.907245
0.081693        0.918307
0.0806847       0.919315
0.125776        0.874224
0.161663        0.838337
0.235672        0.764328
0.221932        0.778068
0.178636        0.821364
0.114087        0.885913
0.0881827       0.911817
0.0877433       0.912257
0.0741231       0.925877
       .
       .
       .
```

B.4 The show_viterbi executable

The show_viterbi executable performs the Viterbi algorithm on a sequence or a set of sequences.

B.4.1 Viterbi algorithm

The Viterbi algorithm enables the computation of the most likely hidden state path $s^\star$ given the observed sequence x_1^n and the model parameters $\theta = (a, b)$:

$$s^\star = \arg\max_{s_1^n} P_\theta(S_1^n = s_1^n \mid X_1^n = x_1^n) = \arg\max_{s_1^n} P_\theta(S_1^n = s_1^n, X_1^n = x_1^n)$$

To overcome numerical problems, we present a version of Viterbi by taking the logarithm of all the probabilities involved in the algorithm.

Starts with $t = 1$ with :

$$\log P_\theta(S_1 = v, x_1) = \log(b_v(x_1)a(v))$$

For t increasing from 2 to $n-1$, compute by recurrence the logarithm of the maximal probability of the most probable path (enabling to generate the observations $x_1, \ldots, x_{t+1}$) that ends in position $t+1$ in state $S_{t+1} = v$:

$$\begin{aligned}
&\max_{s_1^t} \log P_\theta(S_1^t = s_1^t, S_{t+1} = v, x_1^{t+1}) \\
&\quad = \max_u \Big(\log(b_v(x_{t+1}; x_{t-r_v+1}^t)a(u, v)) + \\
&\qquad \max_{s_1^{t-1}} \log P_\theta(S_1^{t-1} = s_1^{t-1}, S_t = u, x_1^t) \Big)
\end{aligned}$$

Find $s^\star = (s_1^\star, s_2^\star, \ldots, s_n^\star)$ by backtracing :

$$s_n^\star = \arg\max_v \Big(\max_{s_1^{n-1}} \log P_\theta(S_1^{n-1} = s_1^{n-1}, S_n = v, x_1^n) \Big)$$

$$\begin{aligned}
s_t^\star \quad = \quad & \arg\max_u \Big(\log(b_v(x_{t+1}; x_{t-r_v+1}^t)a(u, s_{t+1}^\star)) \\
& + \max_{s_1^{t-1}} \log P_\theta(S_1^{t-1} = s_1^{t-1}, S_t = u, x_1^t) \Big)
\end{aligned}$$

The same memory saving approximation has been implemented as for forward-backward algorithm (See section B.3.1).

B.4.2 Input files

-model <file>

The syntax of this file is described in the section B.2.3.Keep in mind that all the parameters of the model must be fully specified, *i.e.* the file must not contain any pobs : random.

-vit <file>

This file contains the information concerning the parameters of the memory saving approximation. An example of such a file is given below.

```
vit_segment: 20000
vit_overlap: 4000
```

-seq <file>

The syntax of this file is described in the section B.2.4.

B.4.3 Output files

Only one kind of output file is generated by the show_viterbi executable. One output file with the .vit suffix is created for each sequence file referenced after the seq_files keyword in the -seq <file>. For each sequence file, the .vit output file contains the most probable hidden path. An example of such a file is given below : the mapping between numbers and hidden state identifiers is given in the header of the file, each line of the file corresponds to a position along the sequence (excepted in the header).

```
# viterbi reconstruction
# 0 : (hello) 1 : (hello2)
0
0
0
0
0
0
0
0
0
0
0
0
0
0
      .
      .
      .

0
0
0
1
1
1
1
      .
      .
      .
```

B.5 The show_simul executable

The show_simul executable enables to simulate an HMM given a fully specified HMM.

B.5.1 Simulating an HMM

The simulation of an HMM is easy because it only consists in simulating the hidden path given its markov model and simulating the observed sequence conditionally on the hidden path :

$$S_1 \sim \mathcal{M}_1(a(1), a(2), \ldots, a(q))$$

$$S_t \mid S_{t-1} = u \sim \mathcal{M}_1(a(u,1), a(u,2), \ldots, a(u,q))$$

$$X_t \mid S_t = v, X_{t-r_v}^{t-1} = x_{t-r_v}^{t-1} \sim \mathcal{M}_1(b_v(1; x_{t-r_v}^{t-1}), b_v(2; x_{t-r_v}^{t-1}), \ldots, b_v(4; x_{t-r_v}^{t-1}))$$

B.5.2 Input files

-model <file>

The syntax of this file is described in the section B.2.3. Keep in mind that all the parameters of the model must be fully specified, *i.e.* the file must not contain any pobs : random.

-simul <file>

This file contains only the length of the sequence to be simulated.

```
lg: 10000
```

-seq <file>

The syntax of this file is described in the section B.2.4. It may surprise that such a file should be used by a sequence simulation program. However, the sequence referenced in this file is absolutely not used for the simulation of the new sequence, but the seq_type information of the -seq <file> is necessary.

B.5.3 Output files

simulated.hidden_states file

This file conforms to the same format as output files of the show_viterbi executable. An example of such a file is given below.

```
# hidden states simulation
# 0 : (state1)  1 : (state2)    2 : (state3)    3 : (state4)
1
1
1
1
1
1
1
1

     .
     .
     .

1
1
1
3
3
3

     .
     .
     .
```

simulated_0.dna file

A DNA sequence in Fasta format.

```
> dna observations simulation
TTAGATCAAAAACAGCAGGTAAATAAAAGTTCTATTACTGAAGCGAAGGTGTTTCAACAC
CAAAAATGTTCTAAGGATATTCCTGACCTTAGTGGACAGCATCAAATGATATCTTTTCAT
GCAAATATACAAGTCTAGAAAAATTATTCATTTAATTGCAAAACTAGAAGGCTTTTTCTG
CCAGCGCCTACTAAGAATGGGCAAGTTCGTTAAACTTAAGATGGTACAAAGGTTTCTTTG
GCACAAGTGTGCTTATTTTATGCTAGATGAATTCCTTACATACTATGATAATCTTTGTTT
ATCAGATGACGGTAATAGTATTTGTCAAGTTTTCTCTTTAGGAAGAACTATCACTTGATT

      .

      .

      .
```

If the model includes a "bound" state, the output sequence is a concatenation of sequences of total length corresponding to the length specified in -simul <file>. Appearances of "bound" states delimit individual sequences, "bound" state positions along the concatenated sequence is given in the simulated.hidden_states file in the same way than other hidden states, and in the simulated_0.dna file by "X" in the simulated sequence.

B.6 Some precisions concerning the design of the source code

These considerations are dedicated to users interested in extending the source code.

The C++ programming language has been chosen to implement this program because it is widely used, it allows object-oriented programming and it is efficient for applications requiring intense numeric calculus. The object-oriented design is well adapted for HMM based programs in particular because of the modular nature of the HMM. However, an object oriented design implies numerous calls to function (methods) and then could slow down the execution. Thus, a balanced design must be chosen in order to keep modularity and efficiency properties. The three programs show_emfit, show_viterbi and show_simul share widely the same object components. These objects are well designed for some further extensions but not for all. Here we present what we think possible or not.

Concerning observation modelisation, extends of the program enabling to deal with other kinds of sequences such as amino-acids sequences, codon sequences (conditioning the model with the translation) are possible. The use of multiple sources of observation that could be taken into account for hidden path reconstruction has been considered and could be easily implemented. This is the reason why the seq : identifier is used in the observation distribution description. Concerning the estimation procedure we think that HMM estimation algorithm based on bayesian methodology could be done. In contrast, we do not think that the design of the source code could allow extension to hidden semi-Markovian models (generalized hidden Markov model).

B.7 bactgeneSHOW a Perl script invoking SHOW for bacterial gene detection

B.7.1 Motivation

SHOW is a generic program that uses various HMMs to analyze biological sequences (especially DNA). The design of the HMM and the processing of the output files is a sizeable task. Thus we propose a Perl script that enables to use SHOW for bacterial gene detection using a single command line.

B.7.2 The bactgeneSHOW command line

```
Usage: bactgeneSHOW -i <dnafile> -o <outputfile> [options]
  <dnafile>     Fasta file containing the DNA sequences to be analyze
  <outputfile>  Annotation file containing the results of the gene detection
OPTIONS:
1- CHOICE OF THE MODEL AND PARAMETERS ESTIMATION
  -m      <modelid>   1c | 2c | 3c | 4c | 1c_si | 2c_si | 3c_si | 4c_si
                      Number of coding types to use; "si" stands for short
                      intergenic (default is 4c_si).
  -rna    <rnafile>   Optional fasta file containing DNA sequences
                      of structural RNA genes (used only in order
                      to compute nucleotides frequencies).
  -rbs    <modelid>   m0 | m1 | double_m0 (default is m0).
  -em     <niter epsi niter_sel nb_sel eps_sel>
```

```
                    Parameters for the EM algorithm
                    (default is 20 0.01 50 10 10).
2- USE OF AN ALREADY ESTIMATED MODEL
  -fm     <showmodel> A bacterial gene detection model already fitted by SHOW.
                    If used, any option of the SECTION 1- is ignored.
3- TEMP FILES AND PARAMATERS OF THE GENE PREDICTION OUTPUT
  -d      <tmpdir>    Location of the "temporary" directory where SHOW I/O
                    used by the Perl script will be located
                    (default is /tmp/).
  -cdst   <float>     Probability threshold for CDS prediction (default is 0.5).
  -startt <float>     Probability threshold for multiple starts prediction
                    (default is 0.1).
  -rbst   <float>     Probability threshold for RBS prediction (default is 0.1).
```

B.7.3 HMM for bacterial gene detection

Figure B.4 displays the structure of the models used by bactgeneSHOW.

Intergenic sequences

Intergenic sequences are modelled using a single back looped hidden state, emitting the observed DNA sequence according to a second order Markov chain. The structure of the model displayed in Figure B.4 contains a single intergenic state (at the center of the graph). However, if -m <modelid> refers to a model identifier containing "si" (short intergenic), two supplementary states are added for short intergenic regions modelling. The purpose of the use of these states is explained in the paragraph concerning RBS modelling.

Coding sequences

Coding sequences have a three-periodic composition "core", represented by a cycle of three hidden states, corresponding to the three positions within a codon. Each of these hidden states emits nucleotides according to a second order Markov chain. In frame stop codons are prevented by a zero probability of emitting a stop codon in the third state of the cycle. All states concerned with this constraint are filled in Figure B.4. In order to ensure that start and stop codons delimit CDSs, appropriate sub-models are added upstream and downstream from the core of the CDS.

Heterogeneities of coding sequences composition have been shown to be important features of the bacterial chromosome composition. In particular, taking into account the atypical a+t-rich composition of some horizontally transferred genes has been shown to greatly improve gene detection sensitivity. In order to take these features into account, different sub-models corresponding to distinct coding types are reachable downstream from a start codon. Moreover, composition type can change within genes. This is more general than using disjoint gene types, and also more realistic, in particular with regard to the existence of hydrophobic regions in proteins. The model displayed in Figure B.4 contains two types of coding regions. The user can

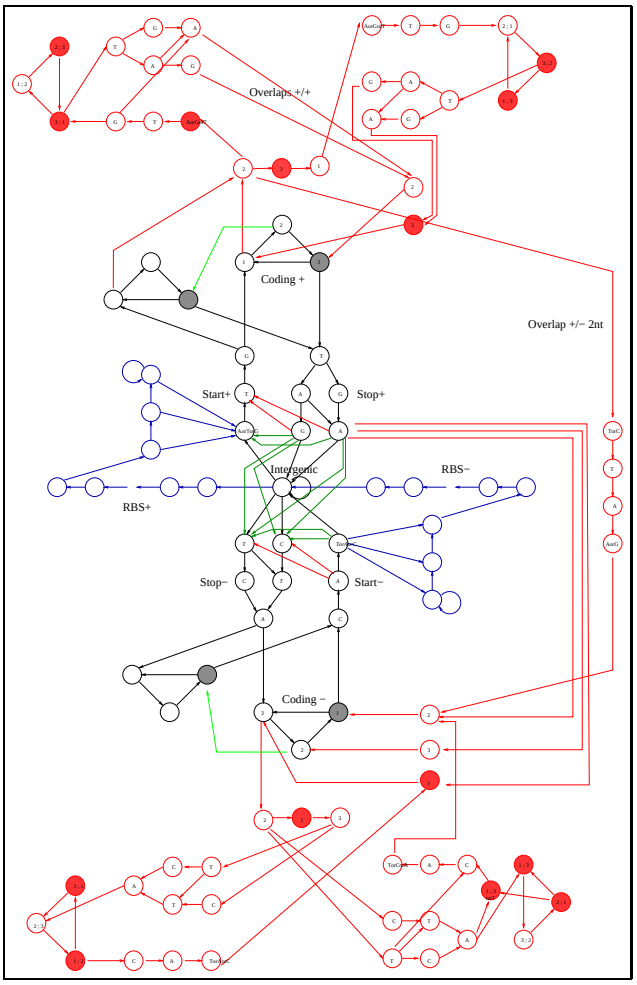

FIG. B.4 – Structure of a HMM dedicated to bacterial coding sequences detection used by bactgeneSHOW. Colors are used in text in order to identify parts of the model.

choose the number of coding sequence compositions of the HMM according to the -m <modelid> option. Model identifiers containing "1c", "2c", "3c" and "4c" refer respectively to HMM with a single, two, three and four types of coding compositions.

Overlap between coding sequences

Overlaps between CDSs are an important feature of the bacterial genome organization. Our model distinguishes the very frequent short overlaps of 1, 2 or 4 nucleotides from the rare longer overlaps. Composition of long overlaps is modelled in the same way as non overlapping CDSs. As not so much data

is available in a single genome for their composition estimation, we use for these overlaps an emission process of order 1. In order to prevent from the appearence of in frame stop codons we used a pseudo-second order model, corresponding to a Markov model of order 1 conditionally on the absence of stop codons. Hidden states and transitions used to model the overlaps are colored in red in Figure B.4.

RBS modelling

Taking into account that the ribosome binding site (RBS) position has been shown to improve precise start site prediction, this can also help to predict short genes where CDS composition does not provide sufficient information. The script enables the user to choose between distinct RBS models according to the -rbs <modelid> options : -rbs m0 RBS sequences are modelled by a positional compositional matrix of Markov order 0 and length 14 ; -rbs m1 same as -rbs m0 but with Markov order 1 ; -rbs double_m0 the positional compositional matrix is duplicated enabling to model two distinct consensus for RBSs (proposed in [BLB01]). The RBS is followed by a "spacer" sequence of minimal length 1. Hidden states corresponding to RBS and spacer are colored in blue in the Figure B.4. The HMM does not enforce each CDS to be preceeded by RBS : the probability to find a RBS before a start codon is estimated. The addition of two states corresponding to "short intergenic" regions could improve this estimation. These two states are the same as the central intergenic state in terms of composition (tied observations) but their transitions differ. They are dedicated to model short intergenic regions, separating CDSs on the same strand : one for CDSs on the forward strand and the other for CDSs on the backward strand. RBS model is not reachable from the state which corresponds to short intergenic regions on the forward strand (reciprocally, short intergenic regions state is not reachable from RBS model on the backward strand). One of the main interest of the use of these states is to distinguish intergenic regions too short to contain RBS from the others, and thus to enable a better estimation of the probability of finding a RBS before a CDS when intergenic is not so short.

Structural RNA modelling

Structural RNA composition differ from intergenic composition. In particular, it has been shown to be related to environmental conditions for the organism, such as temperature. In order to prevent prediction of CDSs within rRNA genes, and to improve the estimates for intergenic parameters, the user can choose to add two states corresponding to rRNA texture (one on the forward, the other on the complementary strand) by the use of the -rna <rnafile> option. These states are not displayed in Figure B.4. Parameters of the composition associated to this state are computed on the sequence in

Fasta format contained in the <rnafile>.

CDSs on the complementary strand

Genes exist on both strands, and therefore are read on both the direct and complementary strands. With minor modifications, the complementary strand model can be derived from the direct strand one, hidden states modelling the complementary strand are displayed in the lower half part of Figure B.4.

B.7.4 What does bactgeneSHOW ?

The bactgeneSHOW script creates a new "temporary" directory in the location specified by the -d <tmpdir> option. This directory contains all the intervening files, in particular input and output files of the show_emfit executable.

We will now describe the different steps performed by bactgeneSHOW.

- The DNA sequence is copied in the temporary directory.
- An initial model corresponding to the model specified by the user is copied in the directory (for instance gene_4c_si.model). This file is a -model <file> for show_emfit (see section B.2.3) containing flags between { and } or between two !. These flags are used by the script to determin respectively where to add some hidden states and which states must be used for gene prediction (states modelling coding regions).
- If -rna <rnafile> option is used, Fasta file containing structural RNA gene sequences is copied in the directory and a file having the suffix .invcomp containing the reverse complementary sequences is created. Composition of structural RNA is computed over these two files, and the two corresponding states are added in the HMM ; the new model file have rna_ prefix (for instance rna_gene_4c_si.model).
- Input files for show_emfit are created : em.desc (see section B.3.2) according to the -em <niter epsi niter_sel nb_sel eps_sel> option, start.set (see section B.2.4).
- show_emfit estimates the parameters of the initial model, the following files are created : start.select.traces, start.select.likelihoods, start.select.-model, start.trace, start.model (see section B.3.3).
- The initial model is enriched after estimation (start.model), hidden states modelling overlaps between CDSs and RBS are added, the resulting model file is named startbis.model.
- show_emfit estimates the parameters of the enriched model (startbis.-model), it produces the output files : final.select.traces, final.select.likelihoods, final.select.model, final.trace, final.model (see section B.3.3). The file final.model contains the final fitted model which will be used for gene prediction.

- An output description file named outputfromparse.desc (see section B.3.2) is created by parsing final.model according to the flags between two !.
- show_emfit performs forward-backward algorithm (using the files final.model, outputfromparse.desc and em_final.desc) and produces an output file with suffix .e (see section B.3.3).
- The .e output file is parsed to produce the annotation file, either in GFF format or in GenBank feature format.

B.7.5 How to retrieve a fitted model for use with the -fm <showmodel> option

Copy the file named final.model from the temporary directory created during a preceding run of bactgeneSHOW.

B.8 Acknowledgments

We thank Antoine MARIN, Gregory NUEL and Vincent MIELE for their help in accessing powerful computers which enables us to try numerous HMM for varied biological problems. We are grateful to Mark HOEBEKE for advices in C++ object-oriented development, which unfortunatly have not always been followed !

Annexe C

Publications

Articles publiés :

K. Marrocco, A. Lecureuil, P. Nicolas et P. Guerche
The Arabidopsis SKP1-like genes present a spectrum of expression profiles.
Plant Mol. Biol. 2003 Jul ;52(4) :715-27.

M. Hoebeke, P. Nicolas et P. Bessières
MuGeN : simultaneous exploration of multiple genomes and computer analysis results.
Bioinformatics. 2003 May 1 ;19(7) :859-64.

P. Nicolas, L. Bize, F. Muri, M. Hoebeke, F. Rodolphe, S.D. Ehrlich, B. Prum et P. Bessières
Mining Bacillus subtilis chromosome heterogeneities using hidden Markov models.
Nucleic Acids Res. 2002 Mar 15 ;30(6) :1418-26.

À paraître :

P. Nicolas, A.-S. Tocquet, V. Miele et F. Muri
A reversible jump Monte-Carlo Markov chain algorithm for bacterial promoter motifs discovery.
Accepté dans *J. Comput. Biol.*.

Chapitre de livre :

P. Nicolas et H. Chiapello. *Prédiction de gènes*
in Dynamique du Génome Végétal, à paraitre
(Ed. J.-F. Maurot-Gaudry and J.-F. Briat) Éditions-INRA.

Bibliographie

[AA01] J.O. Andersson and S.G. Andersson. Pseudogenes, junk DNA, and the dynamics of *Rickettsia* genomes. *Mol. Biol. Evol.*, 18 :829–839, 2001.

[Aka74] H. Akaike. A new look at the statistical model identification. *IEEE T. Automat. Contr.*, AC-19 :716–723, 1974.

[AMS+97] S. F. Altschul, T. L. Madden, A. A. Schäffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman. Gapped BLAST and PSI-BLAST : a new generation of protein database search programs. *Nucleic Acids Res.*, 25 :3389–3402, 1997.

[AYK+03] K. Asai, H. Yamaguchi, C.M. Kang, K. Yoshida, Y. Fujita, and Y. Sadaie. DNA microarray analysis of Bacillus subtilis sigma factors of extracytoplasmic function family. *FEMS Microbiol. Lett.*, 220 :155–160, 2003.

[BA98] K. P. Burnham and D. R. Anderson. *Model Selection and Inference : A Practical Information-Theoretic Approach.* Springer-Verlag, 1998.

[Bab03] M. Madan Babu. Did the loss of sigma factors initiate pseudogene accumulation in *M. leprae*? *Trends Microbiol.*, 11 :59–61, 2003.

[BB98] P. Baldi and S. Brunak. *Bioinformatics. The machine learning approach.* MIT Press, 1998.

[BB99] J. Besemer and M. Borodovsky. Heuristic approach to deriving models for gene finding. *Nucleic Acids Res.*, 27 :3911–3920, 1999.

[BCH98] S.E. Brenner, C. Chothia, and T.J. Hubbard. Assessing sequence comparison methods with reliable structurally identified distant evolutionary relationships. *P. Natl. Acad. Sci. USA*, 95 :6073–6078, 1998.

[BDDL+98] P. Bork, T. Dandekar, Y. Diaz-Lazcoz, F. Eisenhaber, M. Huynen, and Y. Yuan. Predicting function : from genes to genomes and back. *J. Mol. Biol.*, 283 :707–725, 1998.

[BE95] T.L. Bailey and C. Elkan. Unsupervised learning of multiple motifs in biopolymers using Expectation Maximization. *Mach. Learn.*, 21 :51–83, 1995.

[Ber00] G. Bernardi. Isochores and the evolutionary genomics of vertebrates. *Gene*, 241 :3–17, 2000.

[BH02a] R. J. Boys and D. A. Henderson. A comparison of reversible jump MCMC algorithms for DNA sequence segmentation using hidden Markov models. *Computing Science and Statistics*, 33 :35–49, 2002.

[BH02b] R. J. Boys and D. A. Henderson. On determining the order of markov dependence of an observed process governed by a hidden Markov model. *Special Issue of Scientific Programming*, 10 :241–251, 2002.

[BHB97] M.A. Basrai, P. Hieter, and J.D. Boeke. Small open reading frames : beautiful needles in the haystack. *Genome Res.*, 7 :768–771, 1997.

[BHW00] R. J. Boys, D. A. Henderson, and D. J. Wilkinson. Detecting homogenous segments in DNA sequences by using hidden Markov models. *Appl. Stat-J. Roy. St. C*, 49 :269–285, 2000.

[BK97] C. Burge and S. Karlin. Prediction of complete gene structures in human genomic DNA. *J. Mol. Biol.*, 268 :801–807, 1997.

[BK01] N. Balaban and N. Koyfman. Peptides : bacteria's point of view. *Peptides*, 22 :1517–1518, 2001.

[BLB01] J. Besemer, A. Lomsadze, and M. Borodovsky. GeneMarkS : a self-training method for prediction of gene starts in microbial genomes. Implications for finding sequence motifs in regulatory regions. *Nucleic Acid Res.*, 29 :2607–2618, 2001.

[BM93] M. Borodovsky and J. McIninch. GENMARK : parallel gene recognition for both DNA strands. *Comput. Chem.*, 17 :123–133, 1993.

[BM98] J. V. Braun and H. G. Muller. Statistical methods for DNA sequence segmentation. *Stat. Sci.*, 13 :142–162, 1998.

[BMK+95] M. Borodovsky, J. D. McIninch, E. V. Koonin, K. E. Rudd, C. Médigue, and A. Danchin. Detection of new genes in a bacterial genome using Markov models for three gene classes. *Nucleic Acids Res.*, 23 :3554–3562, 1995.

[BMS+99] L. Bize, F. Muri, F. Samson, F. Rodolphe, S.D. Ehrlich, B. Prum, and P. Bessières. Searching gene transfers on *Bacillus subtilis* using hidden Markov models. In *Recomb'99 Proc. of 3rd Ann. Int. Conf. on Comput. Mol. Biol.*, pages 43–49, 1999.

[BO99] J.H. Badger and G.J. Olsen. CRITICA : coding region identification tool invoking comparative analysis. *Mol Biol Evol.*, 16 :512–524, 1999.

[BP66] L. E. Baum and T. Petrie. Statistical inference for probabilistic functions of finite state Markov chains. *Ann. Math. Stat.*, 37 :1554–1563, 1966.

[BPB+97] F.R. Blattner, G. III Plunkett, C.A. Bloch, N.T. Perna, V. Burland, M. Riley, J. Collado-Vides, J.D. Glasner, C.K. Rode, and G.F. Mayhew. The complete genome sequence of *Escherichia coli* K-12. *Science*, 277 :1453–1474, 1997.

[BPM+00] S. Batzoglou, L. Pachter, J. P. Mesirov, B. Berger, and E. S. Lander. Human and mouse gene structure : comparative analysis and application to exon prediction. *Genome Res.*, 10 :950–958, 2000.

[BPSW70] L.E. Baum, T. Petrie, G. Soules, and N. Weiss. A maximization technique occuring in statistical analysis of probabilistic functions of Markov chains. *Ann. Math. Stat.*, 41 :164–171, 1970.

[Bro98] S.P. Brooks. Markov chain Monte Carlo method and its application. *Statistician*, 47 :69–100, 1998.

[BSB97] V. Biaudet, F. Samson, and P. Béssières. Micado — a network-oriented database for microbial genomes. *Comput. Appl. Biosci.*, 13 :431–438, 1997.

[BSS98] I. Bagyan, B. Setlow, and P. Setlow. New small, acid-soluble proteins unique to spores of *Bacillus subtilis* : identification of the coding genes and regulation and function of two of these genes. *J. Bacteriol.*, 180 :6704–6712, 1998.

[BTSK85] K. Babasaki, T. Takao, Y. Shimonishi, and K. Kurahashi. Subtilosin A, a new antibiotic peptide produced by *Bacillus subtilis* 168 : isolation, structural analysis, and biogenesis. *J. Biochem.*, 98 :585–603, 1985.

[Bur97] C. Burge. *Identification of genes in human genomic DNA.* PhD thesis, Stanford University, 1997.

[BWM+01] A. Bolotin, P. Wincker, S. Mauger, O. Jaillon, K. Malarme, J. Weissenbach, S.D. Ehrlich, and A. Sorokin. The complete genome sequence of the lactic acid bacterium *Lactococcus lactis* ssp. *lactis* IL1403. *Genome Res.*, 11 :731–753, 2001.

[CAL+97] S. Cutting, M. Anderson, E. Lysenko, A. Page, T. Tomoyasu, K. Tatematsu, T. Tatsuta, L. Kroos, and T. Ogura. SpoVM, a small protein essential to development in *Bacillus subtilis*, interacts with the ATP-dependent protease FtsH. *J. Bacteriol.*, 179 :5534–5542, 1997.

[CBP+98] S.T. Cole, R. Brosch, J. Parkhill, T. Garnier, C. Churcher, D. Harris, S.V. Gordon, K. Eiglmeier, S. Gas, and C.E. Barry III. Deciphering the biology of *Mycobacterium tuberculosis* from the complete genome sequence. *Nature*, 393 :537–544, 1998.

[CC95] B. P. Carlin and S. Chib. Bayesian model choice via Markov chain Monte Carlo. *J. Roy. Stat. Soc. B*, 57 :473–484, 1995.

[Chi92] S. Chib. Bayes inference in the tobit censored regression model. *J. Econometrics*, 51 :79–99, 1992.

[Chi95] S. Chib. Marginal lokelihood from the Gibbs output. *J. Am. Stat. Assoc.*, 90 :1313–1321, 1995.

[CHP87] N.Y. Chen, F.M. Hu, and H. Paulus. Nucleotide sequence of the overlapping genes for the subunits of *Bacillus subtilis* aspartokinase II and their control regions. *J. Biol. Chem.*, 262 :8787–8798, 1987.

[CHS00] A. Cabrera-Hernandez and P. Setlow. Analysis of the regulation and function of five genes encoding small, acid-soluble spore proteins of *Bacillus subtilis*. *Gene*, 248 :169–181, 2000.

[Chu89] G. A. Churchill. Stochastic models for heterogeneous DNA sequences. *B. Math. Biol.*, 51 :79–94, 1989.

[Cin69] E. Cinlar. Markov renewal theory. *Adv. Appl. Probab.*, 1 :123–187, 1969.

[CJRS94] C. Cervantes, G. Ji, J. L. Ramirez, and S. Silver. Resistance to arsenic compounds in microorganisms. *FEMS Microbiol. Rev.*, 15 :355–367, 1994.

[CRR03] O. Cappé, C. P. Robert, and T. Rydén. Reversible jump, birth-and-death Markov chain Monte Carlo and more general continuous time samplers. *J. Roy. Stat. Soc. B*, pages 679–700, 2003.

[CS00] I. Csiszar and P. Shields. The consistency of the BIC Markov order estimator. *Ann. Stat.*, 28 :1601–1619, 2000.

[Dav94] J. Davies. Inactivation of antibiotics and the dissemination of resistance genes. *Science*, 264 :375–382, 1994.

[Dav99] J. Davison. Genetic exchange between bacteria in the environment. *Plasmid*, 42 :73–91, 1999.

[DEKM98] R. Durbin, S. R. Eddy, A. Krogh, and G. Mitchison. *Biological Sequence Analysis : Probabilistic models of proteins and nucleic acids*. Cambridge University Press, Cambridge, 1998.

[DHK+99] A. L. Delcher, D. Harmon, S. Kasif, O. White, and S. L. Salzberg. Improved microbial gene identification with GLIMMER. *Nucleic Acids Res.*, 27 :4636–4641, 1999.

[dlCD00] I. de la Cruz and I. Davies. Horizontal gene transfer and the origin of species : lessons from bacteria. *Trends Microbiol.*, 8 :128–133, 2000.

[DLP03] V. Daubin, E. Lerat, and G. Perriere. The source of laterally transferred genes in bacterial genomes. *Genome Biol.*, 4, 2003.

[DLR77] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *J. Roy. Stat. Soc. B.*, 39 :1–38, 1977.

[DMKL97] A. Decatur, M.T. McMurry, B.N. Kunkel, and R. Losick. Translation of the mRNA for the sporulation gene *spoIIID* of *Bacillus subtilis* is dependent upon translation of a small upstream open reading frame. *J. Bacteriol.*, 179 :1324–1328, 1997.

[DNZ94] C. D'Souza, M.M. Nakano, and P. Zuber. Identification of *comS*, a gene of the *srfA* operon that regulates the establishment of genetic competence in *Bacillus subtilis*. *P. Natl. Acad. Sci U S A*, 91 :9397–9401, 1994.

[DSO78] M. Dayhoff, R.M. Schwartz, and B.C Orcutt. *Atlas of Protein Sequence and Structure*, chapter A model of evolutionary change in proteins. Silver Spring, MD : National Biomedical Research Foundation, 1978.

[ED94] S. R. Eddy and R. Durbin. RNA sequence analysis using covariance models. *Nucleic Acids Res.*, 22 :2019–2088, 1994.

[EJC+03] P. Eichenberger, S.T. Jensen, E.M. Conlon, C. van Ooij, J. Silvaggi, J.E. Gonzalez-Pastor, M. Fujita, S. Ben-Yehuda, P. Stragier, J.S. Liu, and R. Losick. The sigmaE regulon and the identification of additional sporulation genes in *Bacillus subtilis*. *J. Mol. Biol.*, 327 :945–972, 2003.

[EM02] Y. Ephraim and N. Merhav. Hidden Markov processes. *IEEE T. Inform. Theory*, 48 :1518–1569, 2002.

[FAE+02] R.D. Fleischmann, D. Alland, J.A. Eisen, L. Carpenter, O. White, J. Peterson, R. DeBoy, R. Dodson, M. Gwinn, and D. Haft. Whole-genome comparison of *Mycobacterium tuberculosis* clinical and laboratory strains. *J. Bacteriol.*, 184 :5479–5490, 2002.

[FAW+95] R. D. Fleischmann, M. D. Adams, O. White, R. A. Clayton, E. F. Kirkness, A. R. Kerlavage, C. J. Bult, J. F. Tomb, B. A. Dougherty, J. M. Merrick, et al. Whole-genome random sequencing and assembly of *Haemophilus influenzae Rd*. *Science*, 269 :496–512, 1995.

[FEN+02] C. Fraser, J.A. Eisen, K.E. Nelson, I.T. Paulsen, and S.L. Salzberg. The value of complete microbial genome sequencing (you get what you pay for). *J. Bacteriol.*, 184 :6403–6405, 2002.

[FGW+95] C.M. Fraser, J.D. Gocayne, O. White, M.D. Adams, R.A. Clayton, R.D. Fleischmann, C.J. Bult, A.R. Kerlavage, G. Sutton, J.M. Kelley, et al. The minimal gene complement of *Mycoplasma genitalium*. *Science*, 270 :397–403, 1995.

[FHZ+98] L. Florea, G. Hartzell, Z. Zhang, G. GM. Rubin, and W. Miller. A computer program for aligning a cDNA sequence with a genomic DNA sequence. *Genome Res.*, 8 :967–974, 1998.

[Fin90] L. Finesso. *Estimation of the order of a finite Markov chain.* PhD thesis, University of Maryland, 1990.

[FL99] A. C. Frank and J. R. Lobry. Asymmetric substitution patterns : a review of possible underlying mutational or selective mechanisms. *Gene*, 238 :65–77, 1999.

[FM00] D. Freitag and A. McCallum. Information extraction with HMM structures learned by stochastic optimization. In *AAAI/IAAI*, pages 584–589, 2000.

[FMA+01] J.J. Ferretti, W.M. McShan, D. Adjic, D. Savic, G. Savic, K. Lyon, C. Primeaux, S.S. Sezate, A.N. Surorov, and S. Kenton. Complete genome sequence of an M1 strain of *Streptococcus pyogenes. P. Natl. Acad. Sci. USA*, 98 :4658–4663, 2001.

[FMMG98] D. Frishman, A. Mironov, H.W. Mewes, and M. Gelfand. Combining diverse evidence for gene recognition in completely sequenced bacterial genomes. *Nucleic Acids Res.*, 26 :2941–2947, 1998.

[FT99] T.H. Fan and C.A. Tsai. A Bayesian method in determining the order of a finite state Markov chain. *Comm. Stat. A-Theory*, 28 :1711–1730, 1999.

[GAA+00] R. Guigo, P. Agarwal, J. F. Abril, M. Burset, and J. W. Fickett. An assessment of gene prediction accuracy in large DNA sequences. *Genome Res.*, 10 :1631–1642, 2000.

[GB03] E. Gassiat and S. Boucheron. Optimal error exponents in hidden Markov models order estimation. *IEEE T. Inform. Theory*, 48 :964–980, 2003.

[GBEB97] W.N. Grundy, T.L. Bailey, C.P. Elkan, and M.E. Baker. Meta-MEME : motif-based hidden Markov models of protein families. *Comput. Appl. Biosci.*, 13 :397–406, 1997.

[Gel96] A. E. Gelfand. *Markov Chain Monte Carlo in Practice*, chapter Model determination using sampling-based methods. Chapman and Hall, London, 1996.

[GFB+01] P. Glaser, L. Frangeul, C. Buchrieser, A. Amend, F. Baquero, P. Berche, H. Bloecker, P. Brandt, T. Chakraborty, and A. Charbit. Comparative genomics of *Listeria* species. *Science*, 294 :849–852, 2001.

[GG82] M. Gouy and C. Gautier. Codon usage in bacteria : correlation with gene expressivity. *Nucleic Acids Res.*, 10 :7055–7074, 1982.

[GG84] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE T. Pattern Anal.*, 6 :721–741, 1984.

[GK00] E. Gassiat and C. Keribin. The likelihood ratio test for the number of components in a mixture with Markov regime. *ESAIM Prob. and Stat.*, 4 :25–52, 2000.

[GL94] J. L. Gauvain and C. H. Lee. Maximum a posteriori estimation for multivariate Gaussian mixture observations of Markov chains. *IEEE T. Speech Audio P.*, 2 :291–298, 1994.

[GL97] N. Galtier and J. R. Lobry. Relationships between genomic G+C content, RNA secondary structures, and optimal growth temperature in prokaryotes. *J. Mol. Evol.*, 44 :632–636, 1997.

[GMP96] M. S. Gelfand, A. A. Mironov, and P. A. Pevzner. Gene recognition via spliced sequence alignment. *P. Natl. Acad. Sci. USA*, 93 :9061–9066, 1996.

[GO97] E. A. Groisman and H. Ochman. How *Salmonella* became a pathogen. *Trends Microbiol.*, 5 :343–349, 1997.

[Gre95] P. J. Green. Reversible jump Markov Chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82 :711–732, 1995.

[GS90] A.E. Gelfand and A.F.M. Smith. Sampling based approaches to calculating marginal densities. *J. Am. Stat. Assoc.*, 85 :398–409, 1990.

[GS93] W. Gish and D. J. States. Identification of protein coding regions by database similarity search. *Nat. Genet.*, 3 :266–272, 1993.

[Gué03] Y. Guédon. Estimating semi-Markov chains from discrete sequences, 2003. to appear in *J. Comput. Graph. Stat.*

[GVPR99] S. Garcia-Vallvé, J. Palau, and A. Romeu. Horizontal gene transfer in glycosyl hydrolases inferred from codon usage in *Escherichia coli* and *Bacillus subtilis*. *Mol. Biol. Evol.*, 16 :1125–1134, 1999.

[HAA+01] J.A. Hoskins, W. J Alborn, J. Arnold, L. Blaszczak, S. Burgett, B.S. DeHoff, S. Estrem, L. Fritz, D.-J Fu, W. Fuller, et al.

Genome of the bacterium *Streptococcus pneumoniae* strain R6. *J. Bacteriol.*, 183 :5709–5717, 2001.

[Has70] W.K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57 :97–109, 1970.

[HB98] W.S. Hayes and M. Borodovsky. How to interpret an anonymous bacterial genome : machine learning approach to gene identification. *Genome Res.*, 8 :1154–1171, 1998.

[HBOMT97] J. Hacker, G. Blum-Oehler, I. Muhldorfer, and H. Tschape. Pathogenicity islands of virulent bacteria : structure, function and impact on microbial evolution. *Mol. Microbiol.*, 23 :1089–1097, 1997.

[Hel95] J.D. Helmann. Compilation and analysis of Bacillus subtilis sigma A-dependent promoter sequences : evidence for extended contact between RNA polymerase and upstream promoter DNA. *Nucleic Acids Res.*, 23 :2351–2360, 1995.

[HETC00] J.D. Hughes, P.W. Estep, S. Tavazoie, and G.M. Church. Computational identification of cis-regulatory elements associated with groups of functionally related genes in saccharomyces cerevisiae. *J. Mol. Biol.*, 296 :1205–1214, 2000.

[HFK+02] K. Homma, S. Fukuchi, T. Kawabata, M. Ota, and K. Nishikawa. A systematic investigation identifies a significant number of probable pseudogenes in the *Escherichia coli* genome. *Gene*, 294 :25–33, 2002.

[HH92] S. Henikoff and J.G. Henikoff. Amino acid substitution matrices from proteins blocks. *P. Natl. Acad. Sci. USA*, 89 :10915–10919, 1992.

[HIHB97] M. Hirosawa, K. Isono, W. Hayes, and M. Borodovsky. Gene identification and classification in the *synechocystis* genomic sequence by recursive GeneMark analysis. *DNA Sequence*, 8 :17–29, 1997.

[HK00] J. Hacker and J. B. Kaper. Pathogenicity islands and the evolution of microbes. *Annu. Rev. Microbiol.*, 54 :641–679, 2000.

[HKT+96] S. M. Hebsgaard, P. G. Korning, N. Tolstrup, J. Engelbrecht, P. Rouze, and S. Brunak. Splice site prediction in Arabidopsis thaliana pre-mRNA by combining local and global sequence information. *Nucleic Acids Res.*, 24 :3439–3452, 1996.

[HM99] M.J. Horsburgh and A. Moir. Sigma M, an ECF RNA polymerase sigma factor of Bacillus subtilis 168, is essential for growth and survival in high concentrations of salt. *Mol. Microbiol.*, 32 :41–50, 1999.

[HSF97] J. Henderson, S. Salzberg, and K. H. Fasman. Finding genes in DNA with a hidden Markov model. *J. Comput. Biol.*, 4 :127–141, 1997.

[HVT+02] B. J. Haas, N. Volfovsky, C. D. Town, M. Troukhan, N. Alexandrov, K. A. Feldmann, R. B. Flavell, O. White, and S. L. Salzberg. Full-length messenger rna sequences greatly improve genome annotation. *Genome Biol.*, 3, 2002.

[JLK+01] H. Jarmer, T.S. Larsen, A. Krogh, H.H. Saxild, S. Brunak, and S. Knudsen. Sigma A recognition sites in the *Bacillus subtilis* genome. *Microbiology*, 147 :2417–2424, 2001.

[Kar01] S. Karlin. Detecting anomalous gene clusters and pathogenicity islands in diverse bacterial genomes. *Trends Microbiol.*, 9 :335–343, 2001.

[KBM+94] A. Krogh, M. Brown, I.S. Mian, K. Sjolander, and D. Haussler. Hidden Markov models in computational biology : Applications to protein modeling. *J. Mol. Biol.*, 235 :1501–1531, 1994.

[KEA+03] K. Kobayashi, S.D. Ehrlich, A. Albertini, G. Amati, K.K. Andersen, M. Arnaud, K. Asai, S. Ashikaga, S. Aymerich, P. Bessières, et al. Essential *Bacillus subtilis* genes. *P. Natl. Acad. Sci. USA*, 100 :4678–4683, 2003.

[KFDB01] I. Korf, P. Flicek, D. Duan, and M. R. Brent. Integrating genomic homology into gene structure prediction. *Bioinformatics*, 17 :S140–148, 2001.

[KHRE96] D. Kulp, D. Haussler, M. G. Reese, and F. H. Eeckman. A generalized hidden Markov model for the recognition of human genes in DNA. *Proc. Int. Conf. Intell. Syst. Mol. Biol.*, 4 :134–142, 1996.

[KHV+96] J. Kleffe, K. Hermann, W. Vahrson, B. Wittig, and V. Brendel. Logitlinear models for the prediction of splice sites in plant premRNA sequences. *Nucleic Acids Res.*, 24 :4709–4718, 1996.

[KMH94] A. Krogh, I. S. Mian, and D. Haussler. A hidden Markov model that finds genes in *Escherichia coli* DNA. *Nucleic Acids Res.*, 22 :4768–4778, 1994.

[KNKTU99] I. Kobayashi, A. Nobusato, N. Kobayashi-Takahashi, and I. Uchiyama. Shaping the genome–restriction-modification systems as mobile genetic elements. *Curr. Opin. Genet. Dev.*, 9 :645–656, 1999.

[KOM+97] F. Kunst, N. Ogasawara, I. Moszer, A. M. Albertini, G. Alloni, V. Azevedo, M. G. Bertero, P. Béssières, A. Bolotin, S. Borchert, et al. The complete genome of the gram-positive bacterium *Bacillus subtilis*. *Nature*, 390 :249–256, 1997.

[KOU+01] M. Kuroda, T. Ohta, I. Uchiyama, T. Baba, H. Yuzawa, I. Kobayashi, L. Cui, A. Oguchi, K. Aoki, and Y. Nagai. Whole genome sequencing of meticillin-resistant *Staphylococcus aureus*. *Lancet*, 357 :1225–1240, 2001.

[KR95] R. E. Kass and A. E. Raftery. Bayes Factors. *J. Am. Stat. Assoc.*, 90 :773–795, 1995.

[Kro94] A. Krogh. Hidden Markov Models for labeled sequences. In *12th IAPR Int. Conf. on Pattern Recognition*, pages 140–144. IEEE Computer Society Press, 1994.

[Kro97] A. Krogh. Two methods for improving performance of an HMM and their application for gene finding. *Proc. Int. Conf. Intell. Syst. Mol. Biol.*, 5 :179–186, 1997.

[Kro00] A. Krogh. Using database matches with for HMMGene for automated gene detection in Drosophila. *Genome Res.*, 10 :523–528, 2000.

[KSS+01] J. Kawai, A. Shinagawa, K. Shibata, M. Yoshino, M. Itoh, Y. Ishii, T. Arakawa, A. Hara, Y. Fukunishi, H. Konno, et al. Functional annotation of a full-length mouse cdna collection. *Nature*, 409 :685–690, 2001.

[KSS+02] B. Knab, A. Schliep, B. Steckemetz, B. Wichern, A. Gädke, P. Pipenbacher, and D. Thorarinsdottir. Ghmm and hmmed. In *BOSC2002*, 2002.

[Kul97] V. G. Kulkarni. *Modeling and analysis of stochastic systems.* Chapman and Hall, London, 1997.

[KW95] R. E. Kass and L. Wasserman. A reference bayesian test for nested hypothesis and its relation to the Schwartz criterion. *J. Am. Stat. Assoc.*, 90 :928–934, 1995.

[LAB+93] C.E. Lawrence, S.F. Altschul, M.S. Boguski, J.S. Liu, A.F. Neuwald, and J.C. Wootton. Detecting subtle sequence signals : a Gibbs sampling strategy for multiple alignment. *Science*, 262 :208–214, 1993.

[Lau96] S. L. Lauritzen. *Graphical models.* Oxford University Press, 1996.

[Law99] J. G. Lawrence. Selfish operons : the evolutionary impact of gene clustering in prokaryotes and eukaryotes. *Curr. Opin. Genet. Dev.*, 9 :642–648, 1999.

[Law03] J. Lawrence. When ELFs are ORFs, but don't act like them. *Trends Genet.*, 19 :131–132, 2003.

[LB98] A. V. Lukashin and M. Borodovsky. GeneMark.hmm : new solutions for gene finding. *Nucleic Acids Res.*, 26 :1107–1115, 1998.

[LBGCO03] W. Li, P. Bernaola-Galvan, P. Carpena, and J.L. Oliver. Isochores merit the prefix 'iso'. *Comput. Biol. Chem.*, 27 :5–10, 2003.

[LG94] J. R. Lobry and C. Gautier. Hydrophobicity, expressivity and aromaticity are the major trends of amino-acid usage in 999 *Escherichia coli* chromosome-encoded genes. *Nucleic Acids Res.*, 15 :3174–3180, 1994.

[lGM00] F. le Gland and L. Mevel. Exponential forgetting and geometric ergodicity in hidden Markov models. *Math. Control Signal*, 13 :63–93, 2000.

[LHS99] C. A. Liebert, R. M. Hall, and A. O. Summers. Transposon Tn21, flagship of the floating genome. *Microbiol. Mol. Biol. R.*, 63 :2925–2929, 1999.

[LK03] T.S. Larsen and A. Krogh. EasyGene - a prokaryotic gene finder that ranks ORFs by statistical significance. *BMC Bioinformatics*, 4, 2003.

[LL99] J. S. Liu and C. E. Lawrence. Bayesian inference on biopolymer models. *Bioinformatics*, 15 :38–52, 1999.

[LLB+01] E.S. Lander, L.M. Linton, B. Birren, C. Nusbaum, M.C. Zody, J. Baldwin, K. Devon, K. Dewar, M. Doyle, W. FitzHugh, et al. Initial sequencing and analysis of the human genome. *Nature*, 409 :860–921, 2001.

[LMS+95] V. Lazarevic, C. Mauel, P. Soldo, P. P. Freymond, P. Margot, and D. Karamata. Sequence analysis of the 308 to 311 segment of the *Bacillus subtilis 168* chromosome, a region devoted to cell wall metabolism, containing non-coding grey holes which reveal chromosomal rearrangements. *Microbiology*, 141 :329–335, 1995.

[LNL95] J.S. Liu, A.F. Neuwald, and C.E. Lawrence. Bayesian models for multiple local alignment and Gibbs sampling strategies. *J. Am. Stat. Assoc.*, 90 :1156–1170, 1995.

[LNL99] J.S. Liu, A.F. Neuwald, and C.E. Lawrence. Markovian structures in biological sequence alignments. *J. Am. Stat. Assoc.*, 94 :1–15, 1999.

[LO98] J. G. Lawrence and H. Ochman. Molecular archeology of the *Escherichia coli* genome. *P. Natl. Acad. Sci. USA*, 95 :9413–9417, 1998.

[Lob96] J. R. Lobry. Asymmetric substitution patterns in the two DNA strands of bacteria. *Mol. Biol. Evol.*, 13 :660–665, 1996.

[LP92] B.G. Leroux and M.L. Puterman. Maximum-penalized-likelihhood estimation for independent and Markov-dependent mixture models. *Biometrics*, 48 :545–558, 1992.

[LR90] C.E. Lawrence and A.A. Reilly. An expectation maximization (EM) algorithm for the identification and characterization of common sites in unaligned biopolymer sequences. *Proteins*, 7 :41–51, 1990.

[LWK94] J. S. Liu, W. H. Wong, and A. Kong. Covariance structure of the Gibbs sampler with applications to the comparisons of estimators and sampling schemes. *Biometrika*, 81 :27–40, 1994.

[Mak96] S. C. Makrides. Strategies for achieving high-level expression of genes in *Escherichia coli*. *Microbiol. Rev.*, 60 :512–538, 1996.

[MCC98] F. Muri, D. Chauveau, and D. Cellier. *Lecture Notes in Statistics : Discretization and MCMC convergence assessment*, chapter Convergence assessment in latent variable models : DNA applications, pages 127–146. Springer-Verlag, 1998.

[MCM+02] S. Misra, M. A. Crosby, C. J. Mungall, B. B. Matthews, K. S. Campbell, P. Hradecky, Y. Huang, J. S. Kaminker, G. H. Millburn, S. E. Prochnik, et al. Annotation of the *Drosophila melanogaster* euchromatic genome : a systematic review. *Genome Biol.*, 3, 2002.

[Mev97] L. Mevel. *Statistique asymptotique pour les modèles de Markov cachés*. PhD thesis, Université Rennes 1, 1997.

[MKKR91] T. Msadek, F. Kunst, A. Klier, and G. Rapoport. DegS-DegU and ComP-ComA modulator-effector pairs control expression of the Bacillus subtilis pleiotropic regulatory gene *degQ*. *J. Bacteriol.*, 173 :2366–2377, 1991.

[ML02] B. Modrek and C. Lee. A genomic view of alternative splicing. *Nat. Genet.*, 30 :13–19, 2002.

[MRD99] I. Moszer, E. Rocha, and A. Danchin. Codon usage and lateral gene transfer in *Bacillus subtilis*. *Curr. Opin. Microbiol.*, 2 :524–528, 1999.

[MRV+91] C. Médigue, T. Rouxel, P. Vigier, A. Hénaut, and A. Danchin. Evidence for horizontal gene transfer in *Escherichia coli* speciation. *J. Mol. Biol.*, 222 :851–856, 1991.

[MS00] L. Marsan and M.F. Sagot. Algorithms for extracting structured motifs using a suffix tree with an application to promoter and regulatory site consensus identification. *J. Comput. Biol.*, 7 :345–362, 2000.

[MSSR02] C. Mathé, M. F. Sagot, T. Schiex, and P. Rouze. Current methods of gene prediction, their strengths and weaknesses. *Nucleic Acids Res.*, 30 :4103–4117, 2002.

[MTC+01] L.A. McCue, W. Thompson, C. Carmack, M.P. Ryan, J.S. Liu, V. Derbyshire, and C.E. Lawrence. Phylogenetic footprinting

of transcription factor binding sites in proteobacterial genomes. *Nucleic Acids Res.*, 29 :774–782, 2001.

[MTCL02] L.A. McCue, W. Thompson, C.S. Carmack, and C.E. Lawrence. Factors influencing the identification of transcription factor binding sites by cross-species comparison. *Genome Res.*, 12 :1523–1532, 2002.

[Mur97] F. Muri. *Comparaison d'algorithmes d'identification de chaînes de Markov cachées et application à la détection de régions homogènes dans les séquences d'ADN.* PhD thesis, Université René Descartes, Paris V, 1997.

[Mur98] F. Muri. Modelling bacterial genomes using hidden Markov models. In Physica-Verlag, editor, *Compstat'98 Proc. in Comput. Stat.*, pages 98–100, 1998.

[MWD98] M. J. McLean, K. H. Wolfe, and K. M. Devine. Base composition skews, replication orientation, and gene orientation in 12 prokaryote genomes. *J. Mol. Evol.*, 47 :691–696, 1998.

[MZ97] I. L. MacDonald and W. Zucchini. *Hidden Markov and other models for discrete-valued time series.* Chapman and Hall, London, 1997.

[NBM+02] P. Nicolas, L. Bize, F. Muri, M. Hoebeke, F. Rodolphe, S.D. Ehrlich, B. Prum, and P. Bessières. Mining *Bacillus subtilis* chromosome heterogeneities using hidden Markov models. *Nucleic Acids Res.*, 30 :1418–1426, 2002.

[NBO+01] J. Nolling, G. Breton, M.V. Omelchenko, K.S. Markarova, Q. Zeng, R. Gibson, H.M. Lee, J. Dubois, D. Qiu, and J. Hitti. Genome sequence and comparative analysis of the solvent-producing bacterium *Clostridium acetobutylicum*. *J. Bacteriol.*, 183 :4823–4838, 2001.

[NCL03] A. Nekrutenko, W.Y. Chung, and W.H. Li. ETOPE : Evolutionary test of predicted exons. *Nucleic Acids Res.*, 31 :3564–3567, 2003.

[NL00] A. Nekrutenko and W.H. Li. Assessment of compositional heterogeneity within and between eukaryotic genomes. *Genome Res.*, 10 :1986–1995, 2000.

[NM91] Y. Normandin and S. D. Mogera. An improved MMIE training algorithm for speaker-independent, small vocabulary, continuous speech recognition. *Proc. ICASSP*, pages 537–540, 1991.

[NML01] A. Nekrutenko, K. D. Makova, and W. H. Li. The Ka/Ks ratio test for assessing the protein-coding potential of genomic regions : an empirical and simulation study. *Genome Res.*, 12 :198–202, 2001.

[NML02] A. Nekrutenko, K.D. Makova, and W.H. Li. The K(A)/K(S) ratio test for assessing the protein-coding potential of genomic regions : an empirical and simulation study. *Genome Res.*, 12 :198–202, 2002.

[NW70] S.B. Needleman and C.D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J. Mol. Biol.*, 48 :443–453, 1970.

[OBSR97] A. M. Osborn, K. D. Bruce, P. Strike, and D. A. Ritchie. Distribution, diversity and evolution of the bacterial mercury resistance (*mer*) operon. *FEMS Microbiol. Rev.*, 19 :239–262, 1997.

[Och02] H. Ochman. Distinguishing the ORFs from the ELFs : short bacterial genes and the annotation of genomes. *Trends Genet.*, 18 :335–337, 2002.

[OLG00] H. Ochman, J. G. Lawrence, and E. A. Groisman. Lateral gene transfer and the nature of bacterial innovation. *Nature*, 405 :299–304, 2000.

[Ols93] M.V. Olson. The human genome project. *P. Natl. Acad. Sci. USA*, pages 4338–4344, 1993.

[ORR+99] J. L. Oliver, R. Roman-Roldan, , J. Perez, and P. Bernaola-Galvan. Segment : identifying compositional domains in DNA sequences. *Bioinformatics*, 15 :974–979, 1999.

[PB01] M. Perego and J.A. Brannigan. Pentapeptide regulation of aspartyl-phosphate phosphatases. *Peptides*, 22 :1541–1547, 2001.

[PBG+99] A. Petersohn, J. Bernhardt, U. Gerth, D. Hoper, T. Koburger, U. Volker, and M. Hecker. Identification of sigma(B)-dependent genes in Bacillus subtilis using a promoter consensus-directed search and oligonucleotide hybridization. *J. Bacteriol.*, 181 :5719–5724, 1999.

[Pea91] W.R. Pearson. Searching protein sequence libraries : comparison of the sensitivity and selectivity of the Smith-Waterman and FASTA algorithms. *Genomics*, 11 :635–650, 1991.

[Pea96] W.R. Pearson. Effective protein sequence comparison. *Method Enzymol.*, 266 :227–258, 1996.

[Pea98] W.R. Pearson. Empirical statistical estimates for sequence similarity searches. *J. Mol. Biol.*, 276 :71–84, 1998.

[PFC+01] C.W. Price, P. Fawcett, H. Ceremonie, N. Su, C.K. Murphy, and P. Youngman. Genome-wide analysis of the general stress response in Bacillus subtilis. *Mol. Microbiol.*, 41 :757–774, 2001.

[PG99] L. Peshkin and M. S. Gelfand. Segmentation of yeast DNA using hidden Markov models. *Bioinformatics*, 15 :980–986, 1999.

[PG01] M. Ptashne and A. Gann. *Genes and Signals*. Cold Spring Harbor Laboratory press, New York, 2001.

[PL88] W.R. Pearson and D.J. Lipman. Improved tools for biological sequence comparison. *P. Natl. Acad. Sci. USA*, 85 :2444–2448, 1988.

[Pla01] V. Plagnol. Detection de gènes par modèles de chaînes de markov cachées, 2001. Master's thesis.

[PLS01] M. Pertea, X. Lin, and S. L. Salzberg. GeneSplicer : a new computational method for splice site prediction. *Nucleic Acids Res.*, 29 :1185–1190, 2001.

[PLU+03] L. Petersen, T.S. Larsen, D.W. Ussery, S.L.W. On, and A. Krogh. Rpo D promoters in *Campylobacter jejuni* exhibit a strong periodic signal instead of a -35 box. *J. Mol. Biol.*, 326 :1361–1372, 2003.

[PRD+99] N. Pavy, S. Rombauts, P. Déhais, C. Mathé, D. V. V. Ramana, P. Leroy, and P. Rouzé. Evaluation of gene prediction software using a genomic data set : application to *Arabidopsis thaliana* sequences. *Bioinformatics*, 15 :887–899, 1999.

[PS96] D. B. Phillips and A. F. M. Smith. *Markov Chain Monte Carlo in practice*, chapter Bayesian model comparison via jump diffusions, pages 215–240. Chapman and Hall, London, 1996.

[PT02] G. Perriere and J. Thioulouse. Use and misuse of correspondence analysis in codon usage studies. *Nucleic Acids Res.*, 2002.

[PWZM97] W. R. Pearson, T. Wood, Z. Zhang, and W. Miller. Comparison of DNA sequences with protein sequences. *Genomics*, 15 :24–36, 1997.

[QMT+03] Z.S. Qin, L.A. McCue, W. Thompson, L. Mayerhofer, C.E. Lawrence, and J.S Liu. Identification of co-regulated genes through Bayesian clustering of predicted regulatory binding sites. *Nat Biotechnol.*, 21 :435–439, 2003.

[QT90] W. Qian and D. M. Titterington. Parameter estimation for hidden Gibbs chains. *Stat. Probabil. Lett.*, 10 :49–58, 1990.

[QT91] W. Qian and D. M. Titterington. Estimation of parameters in hidden Markov models. *Philos. T. Roy. Soc. A*, 337 :407–428, 1991.

[Rab89] L. R. A. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *P. IEEE*, 77 :257–286, 1989.

[Raf96] A. E. Raftery. *Markov Chain Monte Carlo in Practice*, chapter Hypothesis testing and model selection. Chapman and Hall, London, 1996.

[RCD93] C. P. Robert, G. Celeux, and J. Diebolt. Bayesian estimation of hidden Markov chains : a stochastic implementation. *Stat. Probabil. Lett.*, 16 :77–83, 1993.

[RD02] E.P. Rocha and A. Danchin. Base composition bias might result from competition for metabolic resources. *Trends Genet.*, 18 :291–294, 2002.

[RDM99] I. B. Rogozin, D. D'Angelo, and L. Milanesi. Protein-coding regions prediction combining similarity searches and conservative evolutionary properties of protein-coding sequences. *Gene*, 226 :129–137, 1999.

[RDR+02] S. Robin, J.J. Daudin, H. Richard, M.F. Sagot, and S. Schbath. Occurrence probability of structured motifs in random sequences. *J. Comput. Biol.*, 9 :761–773, 2002.

[RDV99a] E. Rocha, A. Danchin, and A. Viari. Analysis of long repeats in bacterial genomes reveals alternative evolutionary mechanisms in *Bacillus subtilis* and other competent prokaryotes. *Mol. Biol. Evol.*, 16 :1219–1230, 1999.

[RDV99b] E. P. Rocha, A. Danchin, and A. Viari. Universal replication biases in bacteria. *Mol. Microbiol.*, 32 :11–16, 1999.

[RG97] S. Richardson and P.J. Green. On Bayesian analysis of mixtures with an unknown number of components (with discussion). *J. Roy. Stat. Soc. B.*, 59 :731–792, 1997.

[RG98] S. Richardson and P.J. Green. Corrigendum : On Bayesian analysis of mixtures with an unknown number of components. *J. Roy. Stat. Soc. B.*, 60 :661, 1998.

[RKTH00] M.G. Reese, D. Kulp, H. Tammana, and D. Haussler. Genie–gene finding in *Drosophila melanogaster*. *Genome Res.*, 10 :529–538, 2000.

[RMO01] S. Rogic, A. K. Mackworth, and F. B. F. Ouellette. Evaluation of gene-finding programs on mammalian sequences. *Genome Res.*, 11 :817–832, 2001.

[Rob92] C. P. Robert. *L'analyse statistique bayésienne*. Economica, 1992.

[Rob95] C.P. Robert. Convergence control methods for Markov Chains Monte Carlo algorithms. *Stat. Sci.*, 10(3) :231–253, 1995.

[Rob96] C. P. Robert. *Méthodes de Monte Carlo par Chaînes de Markov*. Economica, 1996.

[Rob98] C.P. Robert, editor. *Lecture Notes in Statistics : Discretization and MCMC convergence assessment*. Springer-Verlag, 1998.

[RRT00] C. P. Robert, T. Rydén, and D. M. Titterington. Bayesian inference in hidden Markov models through the reversible jump markov chain Monte Carlo Method. *J. Roy. Stat. Soc. B*, 62 :57–75, 2000.

[RSR+01] J.C. Rain, L. Selig, H. De Reuse, V. Battaglia, C. Reverdy, S. Simon, G. Lenzen, F. Petel, J. Wojcik, V. Schachter, Y. Chemama, A. Labigne, and P. Legrain. The protein-protein interaction map of *helicobacter pylori*. *Nature*, 409 :211–215, 2001.

[RSW00] G. Reinert, S. Schbath, and M.S. Waterman. Probabilistic and statistical properties of words : an overview. *J. Comput. Biol.*, 7 :1–46, 2000.

[RSZS02] N. Rajewsky, N.D. Socci, M. Zapotocky, and E.D. Siggia. The evolution of DNA regulatory regions for proteo-gamma bacteria by interspecies comparisons. *Genome Res.*, 12 :298–308, 2002.

[RTA98] T. Ryden, T. Teräsvirta, and S. Asbrink. Stylized facts of daily return series and the hidden Markov model. *J. Appl. Econometrics*, 13 :217–244, 1998.

[Rud00] K.E. Rudd. EcoGene : a genome sequence database for *Escherichia coli* K-12. *Nucleic Acids Res.*, 28 :60–64, 2000.

[SBH+94] Y. Sakakibara, M. Brown, R. Hughey, I. S. Mian, K. Sjolander, R. C. Underwood, and D. Haussler. Stochastic context-free grammars for tRNA modeling. *Nucleic Acids Res.*, 22 :5112–5120, 1994.

[Sch78] G. Schwarz. Estimating the dimension of a model. *Ann. Stat.*, 6 :461–464, 1978.

[SCH+82] F. Sanger, A.R. Coulson, G.F. Hong, D.F. Hill, and G.B. Petersen. Nucleotide sequence of bacteriophage lambda DNA. *J. Mol. Biol.*, 162 :729–773, 1982.

[SDKW98] S. L. Salzberg, A. L. Delcher, S. Kasif, and O. White. Microbial gene identification using interpolated Markov models. *Nucleic Acids Res.*, 26 :544–548, 1998.

[SESS01] B.E. Suzek, M.D. Ermolaeva, M. Schreiber, and S.L. Salzberg. A probabilistic method for identifying start codons in bacterial genomes. *Bioinformatics*, 17 :1123–1130, 2001.

[SFB98] S.J. Stasinopoulos, G.A. Farr, and D.H. Bechhofer. *Bacillus subtilis tetA*(L) gene expression : evidence for regulation by translational reinitiation. *Mol. Microbiol.*, 30 :923–932, 1998.

[SJB+01] M. Skovgaard, L.J. Jensen, S. Brunak, D. Ussery, and A. Krogh. On the total number of genes and their length

distribution in complete microbial genomes. *Trends Genet.*, 17 :425–428, 2001.

[SMR01] T. Schiex, A. Moisan, and P. Rouzé. *LNCS 2066*, chapter EuGène : An Eucaryotic Gene Finder that combines several sources of evidence, pages 111–125. 2001.

[SNC77] F. Sanger, S. Nicklen, and A.R Coulson. DNA sequencing with chain-terminating inhibitors. *P. Natl. Acad. Sci. USA*, 74 :5463–7, 1977.

[SOH+02] T. Shimizu, K. Ohtani, H. Hirakawa, K. Ohshima, A. Yamashita, T. Shiba, N. Ogasawara, M. Hattori, S. Kuhara, and H. Hayashi. Complete genome sequence of *Clostridium perfringens*, an anaerobic flesh-eater. *P. Natl. Acad. Sci. USA*, 99 :996–1001, 2002.

[SPD+99] S. L. Salzberg, M. Pertea, A. L. Delcher, M. J. Gardner, and H. Tettelin. Interpolated Markov models for eukaryotic gene finding. *Genomics*, 59 :24–31, 1999.

[Spi98] D. J. Spiegelhalter. Bayesian graphical modelling : a case-study in monitoring health outcomes. *Appl. Stat-J. Roy. St. C*, 47 :115–133, 1998.

[SSZ+98] P.T. Spellman, G. Sherlock, M.Q. Zhang, V.R Iyer, K. Anders, M.B. Eisen, P.O. Brown, D. Botstein, and B. Futcher. Comprehensive identification of cell cycle-regulated genes of the yeast *Saccharomyces cerevisiae* by microarray hybridization. *Mol. Biol. Cell.*, 9 :3273–3297, 1998.

[Sta94] R. Staden. Measurements of the effects that coding for a protein has on a DNA sequence and their use for finding genes. *Nucleic Acids Res.*, 12 :551–567, 1994.

[Ste00] M. Stephens. Bayesian analysis of mixtures with an unknown number of components - an alternative to reversible jump methods. *Ann. Stat.*, 28 :40–74, 2000.

[Sto77] M. Stone. An asymptotic equivalence of choice of model by cross-validation and Akaike's criterion. *J. Roy. Stat. Soc. B*, 39 :44–47, 1977.

[Sto90] G.D. Stormo. Consensus patterns in DNA. *Methods Enzymol.*, 183 :211–221, 1990.

[Sto94] A. Stolcke. *Bayesian learning of Probabilistic language models.* PhD thesis, University of California, Berkeley, 1994.

[Sue99] N. Sueoka. Two aspects of DNA base composition : G+C content and translation-coupled deviation from intra-strand rule A=T and G=C. *J. Mol. Evol.*, 49 :49–62, 1999.

[Syv94] M. Syvanen. Horizontal gene transfer : evidence and possible consequences. *Annu. Rev. Genet.*, 28 :237–261, 1994.

[TC00] E. R. Tillier and R. A. Collins. The contributions of replication orientation, gene direction, and signal sequences to base-composition asymmetries in bacterial genomes. *J. Mol. Evol.*, 50 :249–257, 2000.

[The00] The Arabidopsis Genome Initiative. Analysis of the genome sequence of the flowering plant Arabidopsis thaliana. *Nature*, 408 :796–815, 2000.

[THVD97] K. Turgay, L.W. Hamoen, G. Venema, and D. Dubnau. Biochemical characterization of a molecular switch involving the heat shock protein ClpC, which controls the activity of ComK, the competence transcription factor of *Bacillus subtilis*. *Gene Dev.*, 11 :119–128, 1997.

[Tie94] L. Tierney. Markov chains for exploring posterior distributions (with discussion). *Ann. Stat.*, 22 :1701–1762, 1994.

[Til03] C. Tilstone. Vital statistics. *Nature*, 424 :610–612, 2003.

[TMS+95] K. Takemaru, M. Mizuno, T. Sato, M. Takeuchi, and Y. Kobayashi. Complete nucleotide sequence of a *skin* element excised by DNA rearrangement during sporulation in *Bacillus subtilis*. *Microbiology*, 141 :323–327, 1995.

[TNP+01] H. Tettelin, K.E. Nelson, I.T. Paulsen, J.A. Eisen, T.D. Read, S. Peterson, J. Heidelberg, R.T. DeBoy, D.H. Haft, and R.J. Dodson. Complete genome sequence of a virulent isolate of *Streptococcus pneumoniae*. *Science*, 293 :498–506, 2001.

[TNT+00] H. Takami, K. Nakasone, Y. Takaki, G. Maeno, Y. Sasaki, N. Masui, F. Fuji, C. Hirama, Y. Nakamura, and N. Ogasawara. Complete genome sequence of the alkaliphilic bacterium *Bacillus halodurans* and genomic sequence comparison with *Bacillus subtilis*. *Nucleic Acids Res.*, 28 :4317–4331, 2000.

[TTU02] H. Takami, Y. Takaki, and I. Uchiyama. Genome sequence of *Oceanobacillus iheyensis* isolated from the Iheya Ridge and its unexpected adaptive capabilities to extreme environments. *Nucleic Acids Res.*, 30 :3927–3935, 2002.

[UM91] E. C. Uberbacher and R. J. Mural. Locating protein-coding regions in human DNA sequences by a multiple sensor-neural network approach. *P. Natl. Acad. Sci. USA*, 88 :1261–1265, 1991.

[UZB00] J. Usuka, W. Zhu, and V. Brendel. Optimal spliced alignment of homologous cDNA to a genomic DNA template. *Bioinformatics*, 16 :203–211, 2000.

[VAM+01] J.C. Venter, M.D. Adams, E.W. Myers, P.W. Li, R.J. Mural, G.G. Sutton, H.O. Smith, M. Yandell, C.A. Evans, R.A. Holt, et al. The sequence of the human genome. *Science*, 291 :1304–1351, 2001.

[vdLDK03] M. van der Laan, S. Dudoit, and S. Keles. Asymptotic optimality of likelihood-based cross-validation. Technical report, University of California, Berkeley, 2003.

[vNZRS02] E. van Nimwegen, M. Zavolan, N. Rajewsky, and E.D. Siggia. Probabilistic clustering of sequences : inferring new bacterial regulons by comparative genomics. *P. Natl. Acad. Sci. USA*, 99 :7323–7328, 2002.

[WDDM90] H. E. Wood, M. T. Dawson, K. M. Devine, and D. J. McConnell. Characterization of *PBSX*, a defective prophage of *Bacillus subtilis*. *J. Bacteriol.*, 172 :2667–2674, 1990.

[WRGS96] S. Richardson W. R. Gilks and D. J. Spiegelhalter. *Markov Chain Monte Carlo in practice*. Chapman and Hall, London, 1996.

[Wu83] C. F. Wu. On the convergence properties of the EM algorithm. *Ann. Stat.*, 11(1) :95–103, 1983.

[WWY01] M.P. Washburn, D. Wolters, and J.R. Yates. Large-scale analysis of the yeast proteome by multidimensional protein identification technology. *Nat. Biotechnol.*, pages 242–247, 2001.

[XLBL01] X. X. Liu, D.L. Brutlag, and J.S. Liu. BioProspector : discovering conserved DNA motifs in upstream regulatory regions of co-expressed genes. *Pac. Symp. Biocomput.*, pages 127–138, 2001.

[YFCH86] M. Yang, E. Ferrari, E. Chen, and D.J. Henner. Identification of the pleiotropic *sacQ* gene of *Bacillus subtilis*. *J. Bacteriol.*, 166 :113–119, 1986.

[YLB01] R. F. Yeh, L. P. Lim, and C. B. Burge. Computational inference of homologous gene structures in the human genome. *Genome Res.*, pages 803–816, 2001.

[Zha97] M. Q. Zhang. Identification of protein coding regions in the human genome by quadratic discriminant analysis. *P. Natl. Acad. Sci. USA*, 94 :565–568, 1997.

[Zha02] M. Q. Zhang. Computational prediction of eukaryotic protein-coding genes. *Nature*, 3 :698–709, 2002.

[ZKRH77] S. A. Zahler, R. Z. Korman, R. Rosenthal, and H. E. Hemphill. *Bacillus subtilis* bacteriophage $SP\beta$: localization of the prophage attachment site, and specialized transduction. *J. Bacteriol.*, 129 :556–558, 1977.

[Zuk91] M. Zuker. Suboptimal sequence alignment in molecular biology : alignment with error analysis. *J. Mol. Biol.*, 221 :403–420, 1991.