Université d'Evry Val d'Essonne

PMC pour l'étude des occurrences de motifs dans les séquences markoviennes

Mémoire présenté par

Grégory Nuel

en vue de l'obtention de l'

Habilitation à Diriger des Recherches

soutenu devant un jury composé de

Pr.	Maxime	Crochemore	
Pr.	James	Fu	Rapporteur
Pr.	Bernard	Prum	Directeur
Pr.	Stéphane	Robin	Rapporteur
Dr.	Eduardo	Rocha	
Pr.	Valeri	Stefanov	Rapporteur

le

vendredi 1er décembre 2006

Je tiens d'abord à remercier les membres du jury et tout particulièrement James Fu et Valeri Stéfanov qui, en plus de m'avoir fait l'honneur d'accepter d'être rapporteurs, ont eut la gentillesse de faire un long déplacement jusqu'à Paris spécialement pour l'occasion.

Je remercie également chaleureusement l'ensemble de mes collègues du laboratoire Statistique et Génome pour l'ambiance détendue et amicale qui accompagne mon quotidien professionnel grâce à eux.

Mes remerciements vont ensuite à ma famille et surtout à ma femme et à mes deux garçons qui ont si gentiment accepté de me partager avec cette amante jalouse et exigeante qu'on appelle "habilitation" et sans le soutien desquels je n'aurais jamais pu aller jusqu'au bout de ce travail.

Enfin, pour son soutien sans faille, pour sa disponibilité, pour son écoute, pour ses conseils et, plus généralement, pour ses immenses qualités humaines et scientifiques, je veux remercier Bernard Prum à qui je dédie cette HDR.

Ce document constitue la synthèse de l'ensemble de mes travaux de recherches depuis une dizaine d'années et se compose de trois parties principales. La première partie présente le contexte dans lequel c'est déroulé cette recherche et les thèmes scientifiques abordés. Elle comprend également une liste de mes communications (orales et écrites) ainsi que le détail de mes activités d'enseignement et d'encadrement de recherche.

La seconde partie du mémoire concerne l'étude des occurrences de motifs dans les séquences markoviennes et l'approche unifiée de ce problème à l'aide des PMC (*Pattern Markov Chain*) ; c'est le coeur de ce document.

Les parties suivantes contiennent une collection de mes articles scientifiques, classés par thèmes et par ordre chronologique. Une première partie (la plus importante en volume) concerne les motifs. La deuxième s'intéresse au études génétiques cas-témoins et la dernière liste les articles qui ne tombent dans aucune des deux catégories précédentes.

Table des matières

Première partie

Introduction

Chapitre 1

Présentation générale

1.1 Historique

A la suite d'une formation de mathématiques à l'université Paris XI, c'est en 1998 que commence ma première véritable expérience de la recherche scientifique avec un stage de DEA de quatre mois à l'INAP-G sous la direction de Stéphane Robin sur le thème de la distinction variétale. Bernard Prum me propose ensuite d'entreprendre avec lui une thèse ayant pour sujet l'utilisation des techniques de grandes déviations pour l'étude des statistiques de motifs. Je rejoins donc le laboratoire "Statistique et Génome" et commence dans le même temps à enseigner à l'université d'Evry Val d'Essonne (UEVE) en tant qu'allocataire-moniteur. Je soutiens mon doctorat en juillet 2001 et poursuit ma recherche au sein du même laboratoire en exerçant les fonctions d'ATER. J'obtiens un poste de Maître de Conférence à l'UEVE en 2002.

1.1.1 *Curriculum Vitae*

Formations et Diplômes

1990-1992 Classes préparatoires au lycée Jean-Baptiste Say, Paris 14$^{\text{ème}}$.

1992-1996 Premier et deuxième cycles en mathématiques à l'université Paris XI, Orsay.

1997 Agrégation externe de mathématiques (111$^{\text{ème}}$ sur 350).

1998 DEA "Modélisation stochastique et statistiques", université Paris XI, Orsay.

Juillet 1999 Participation à l'*International Summer School in Computational Biology* à Lipari, Italie (deux semaines).

Mars 1999 Participation à l'école thématique CNRS "Mathématiques, informatique et génomes" à Asnelles, Normandie (une semaine).

1999-2001 Formation personnelle à l'administration Linux/Unix et à la programmation (`C/C++`, `PERL`, `PHP`, bases de données, web).

Juillet 2001 Thèse de mathématiques, "Grandes déviations et chaînes de Markov pour l'étude des occurrences de mots dans les séquences biologiques", université d'Evry Val d'Essonne.
Jury :

Mme.	de Turckheim	Élisabeth	Rapporteur
M.	Ben Arous	Gérard	Rapporteur
	Prum	Bernard	Directeur de thèse
	Robin	Stéphane	Rapporteur
	Tavaré	Simon	Président

Mention : très honorable avec les félicitations du jury.

Stages et Emplois

1997-1998 Tutorat à l'université Paris XI, Orsay.

1998 Stage de recherche de quatre mois à l'INAPG, Paris.

1998-2001 Moniteur à l'université d'Evry Val d'Essonne.

2001-2002 ATER à l'université d'Evry Val d'Essonne.

depuis 2002 Maître de conférence à l'université d'Evry Val d'Essonne.

1.1.2 Communications orales

Conférences invitées

Novembre 1998 *Prédiction de distance phénotypiques à partir de données moléculaires pour la distinction variétale.* Journées méthodologiques du GEVES (Groupe d'Etude et de contrôle des Variétés et des Semences), Paris, France.

Septembre 2001 *Large deviations and Markov chains for the study or word counts in biological sequences.* European Young Statistician Meeting, Liptovski Mykulas, Slovaquie.

Mai 2004 *Déséquilibre de liaison et association à la maladie dans les études de SNPs cas-témoins à échelle des génomes entier.* Les industriels et les mathématiciens se parlent. 7 Mai 2004, IHP, Paris.

Août 2004 *Mathématiques et Génomes.* Université d'été : Sciences Mathématiques et Modélisation. Université Bordeaux 1.

mars 2006 *Exact Distribution of Local Score using Finite Markov Chain Imbedding : an effective approach.* International Congress on the Application of Mathematics (ICAM 2006), Universidad of Chile, Santiago de Chile, Chili.

Mai 2006 *Effective p-values computations using FMCI : application to local score and to pattern statistics.* International Workshop on Applied Probability (IWAP 2006), University of Connecticut, Etats-unis.

Conférences

Mars 2000 *Varietal distinctness assisted by molecular markers : a methodological approach.* Congrès MMH (Molecular Markers in Horticulturae), Montpellier, France.

Novembre 2000 *Grandes déviations et chaîne de Markov pour l'étude des occurrences de mots ou de familles de mots.* Colloque TAS (Traitement et Analyse des Séquences), Evry, France.

Mai 2001 *Grandes déviations et chaînes de Markov : occurrences de mots ou de motifs.* $33^{\text{ème}}$ journées de la Société Française de Statistiques, Nantes, France

Décembre 2001 *Unusual word frequencies in Markov chains : the large deviation approach.* SemStat congress about extremes, Goteborg, Suède.

1.1.3 Communications écrites

1998 Nuel. Prédiction de distances phénotypiques à partir de données moléculaires pour la distinction variétale. *Mémoire de DEA de l'université PARIS XI, Orsay.*

1999 Nuel, Baril, et Robin. Varietal distinctness assisted by molecular markers : a methodological approach. *Acta Horticulturae, Proceedings of the MMH congress (2001)*. 546 :65-71.

2001 Nuel, Robin, et Baril. Predicting distances using a linear model : the case of varietal distinctness. *Journal of Applied Statistics*. 28(5) :607-621.

2001 Nuel. Grandes déviations et chaînes de *Markov* pour l'étude des occurrences de mots dans les séquences biologiques. *Thèse de l'université d'Evry Val d'Essonne*.

2003 Richard et Nuel. SPA : Simple web tool to assess statistical significance of DNA patterns. *Nucleic Acids Research*. 31(13) :3679-3681.

2003 Goldstein, Fondrat, Muri, Nuel, Saragueta, Tocquet, et Prum. Short inverse complementary amino acid sequences generate protein complexity. *Comptes Rendus, Biologie*. 154(1-2) :170-80.

2003 Bocs, Cruveiller, Vellenet, Nuel, et Medigue. AMIGene : Annotation of Microbial Genes. *Nucleic Acids Research*. 31(13) :3723-3726.

2004 Bourguignon, Guedj, Képès, Matias, Nuel, Omont, et Prum. Recherche de gènes impliqués dans une maladie (Genset-Serono). *Matapli*. 74 :33-41.

2004 Nuel. LD-SPatt : Large Deviations Statistics for Patterns on Markov chains. *Journal of Computational Biology*. 11(6) :1023-1033.

2005 Nuel. S-SPatt : Simple Statistics for Patterns on Markov chains. *Bioinformatics*. 21(13) :3051-3052.

2005 Miele, Bourguignon, Robelin, Nuel, et Richard. seq++ : analyzing biological sequences with a range of Markov related models. *Bionformatics*. 21(11) :2783-2784.

2006 Nuel. Effective p-values computations using FMCI : application to local score and pattern statistics. *Algorithms for Molecular Biology*. 1(5).

2006 Nuel. Cumulative Distribution Function of a Geometric Poisson Distribution. Accepté par le *Journal of Statistical Computation and Simulation*.

2006 Nuel. Numerical solution for pattern statistics on Markov chains. Accepté par *Statistical Applications in Genetics and Molecular Biology*.

2006 Nuel. Pattern statistics on Markov chains and sensitivity to parameter estimation. Accepté par *Algorithms for Molecular Biology*.

2006 Guedj, Wojcik, Della-Chiesa, Nuel, et Forner. A fast, unbiased and exact allelic test for case-control association studies. *Human Heredity*. 61(4) :210-221.

2006 Guedj, Robelin, Hoebeke, Lamarine, Wojcik, et Nuel. Detecting local high-scoring segments : a first-stage approach for genome-wide association studies. Accepté par *Statistical Applications in Genetics and Molecular Biology*.

2006 Guedj, Della-Chiesa, Picard, et Nuel. Computing power in case-control association studies and application to meta-statistics. Accepté par *Annals of Human Genetics*.

2006 Nuel. Distribution of patterns on Markov chains : a unified approach using deterministic finite state automata. Soumis au *Journal of Applied Probability*.

1.2 Thèmes de recherche

Les thèmes présentés ici sont essentiellement abordés dans l'ordre chronologique. Pour chacun d'entre eux, les avancées scientifiques obtenues sont évoquées dans leur contexte et les perspectives éventuelles sont discutées.

1.2.1 Distinction Variétale

Ce thème a constitué le sujet du stage de DEA que j'ai effectué en 1998 au sein de l'INAP-G, sous la direction conjointe de Stéphane Robin (pour les statistiques) et Claire Baril (pour la génétique). Le but du stage était la prédiction de distances phénotypiques entre des variétés à partir de données de génotypage.

Pour qu'une nouvelle semence soit officiellement reconnue, il est nécessaire qu'elle soit certifiée par le GEVES (Groupe d'Etude des Variétés et des Semences) qui doit, entre autres critères, s'assurer que la variété candidate est bien distincte de l'ensemble de celles inscrites au catalogue ; c'est la distinction variétale.

Pour cela, on effectue au champ un grand nombre de mesures agro-morphologiques afin d'associer une distance phénotypique à chaque couple formé d'une variété de référence et d'une variété candidate. Les critères phénotypiques mesurés (taille, rendement, sensibilité à la verse, etc ...) étant fortement dépendants du climat et de l'environnement, il est donc nécessaire de cultiver chaque année, l'ensemble (toujours en expansion) des variétés du catalogue ainsi que toutes les variétés candidates. Le plan d'expérience résultant est aussi gigantesque que coûteux.

Dans le même temps, on dispose aisément pour ces mêmes variétés de données de génotypages qui permettent de caractériser leurs patrimoines génétiques de manière précise, rapide et peu coûteuse. Lorsque l'on compare les distances génétiques ainsi obtenues aux distances phénotypiques, on observe un graphe à la forme triangulaire bien marquée. Le but du stage était double : 1) comprendre cette corrélation particulière et 2) proposer une méthode permettant de tirer parti des données génétiques pour faciliter la distinction variétale.

Ces deux objectifs ont été remplis dans le cadre de mon travail de DEA qui a donné lieu à deux publications Nuel et al. (1999) et Nuel et al. (2001). La méthodologie proposée permet de réduire drastiquement le nombre de variétés de référence à expérimenter au champ en écartant par une phase prédictive celles qui seront nécessairement distinctes de l'ensemble des variétés candidates.

1.2.2 Occurrences de motifs

Il existe dans les séquences biologiques (ADN, Protéines, etc ...) un certain nombre de motifs fonctionnels (motifs de régulation, motifs CHI, sites de restriction, *binding sites* chez les protéines, etc ...). Du fait de leurs activités, le nombre d'occurrences de ces motifs ont très souvent un caractère *inhabituel* ; les motifs CHI (ayant un rôle protecteur pour le génome) sont par exemple très abondants tandis que les sites de restriction (dont la présence représente un danger pour l'organisme) sont rares.

La détection expérimentale de ces motifs étant longue et difficile, on se propose de les rechercher à partir des séquences en identifiant les motifs statistiquement exceptionnels, c'est à dire dont le nombre d'occurrences observées est étonnant par rapport à un modèle de référence. Le modèle markovien étant communément retenu pour jouer le rôle de

référant, on est donc confronté au problème de l'étude de la loi des occurrences d'un motif dans ce modèle.

Ce problème, s'il peut sembler simple de prime abord, est en fait d'une grande complexité. Les communautés des informaticiens, combinatoriciens, statisticiens et probabilistes s'y sont d'ailleurs largement intéressés (sous des formes plus ou moins générale : séquences indépendantes ou non, alphabets binaires ou plus complexes, simples répétitions ou motifs structurés, ...) au fil des ans.

Je présente ici, dans l'ordre chronologique, mes travaux de recherche sur ce thème.

Grandes déviations

Lorsque l'on s'intéresse au degré de significativité du nombre d'occurrences observées d'un motif dans une séquence réelle, il n'est pas rare de trouver des p-values de l'ordre de 10^{-50} à 10^{-300}. Ainsi, les événements auxquels on s'intéresse se trouvent souvent très loin dans les queues de distribution. Il semble donc particulièrement pertinent d'approcher ce problème avec une outil statistique adapté à de tels événements : la théorie des grandes déviations.

La loi des grand nombres et le théorème de la limite centrale nous informent que la moyenne d'une suite de n variables aléatoires tend vers son espérance et que les déviations autour de cette valeur sont de l'ordre de grandeur de $\sqrt{n}$. On parle de grandes déviations lorsque l'on observe des déviations de l'ordre de grandeur de n (par exemple, observer 75% de "pile" su 100 lancés de pile ou face équilibré).

Le théorème de Cramer-Chernov permet d'établir, dans le cas indépendant comme dans le cas markovien, que la probabilité de l'événement de grandes déviations décroît asymptotiquement à vitesse exponentielle avec n et donne le moyen de calculer cette vitesse en minimisant une certaine fonction de taux.

Ces résultats sur les déviations d'une moyenne empirique constituent le premier niveau des grandes déviations et permettent :

1) de calculer une approximation asymptotique pour la p-value associée au comptage d'un motif ;
2) de savoir que le logarithme de cette p-value est asymptotiquement proportionnel à la longueur de la séquence considérée.

Le second résultat amène de manière naturelle à considérer une représentation des p-values en échelle logarithmique (décimale) que nous adoptons dans toute la suite avec la convention d'utiliser une valeur positive pour les motifs sur-représentés (nombre d'occurrences observé supérieur au nombre d'occurrences attendu) et une valeur négative pour les motifs sous-représentés. (ex : on associe la valeur $+12.3$ à un motif sur-représenté dont la p-value associée est $10^{-12.3}$ et on associe la valeur -7.6 à un motif sous-représenté dont la p-value associée est $10^{-7.6}$).

En niveau 2, on s'intéresse non plus aux déviations de la moyenne mais à celles de la loi empirique ; on considère la loi jointe de l'ensemble des comptages des mots de longueur h. Là encore, on peut établir un principe de grandes déviations pour cette quantité.

Dans le niveau 1, on doit effectuer une minimisation unidimensionnelle d'une fonction de taux dont chaque évaluation est complexe (dans le cas des motifs, il faut calculer la plus grande valeur propre d'une matrice) alors que dans le niveau deux il faut minimiser une entropie relative (simple à évaluer) sous contraintes dans un espace de grandes dimensions.

Dans les deux cas, la mise en oeuvre pratique des calculs est une tâche difficile et constitue une part importante du travail accompli.

Mon travail de thèse (Nuel, 2001) a donné lieu à une publication dans le *Journal of Computational Biology* (Nuel, 2004). Dans le même temps, j'ai également mis à disposition un logiciel (LD-SPatt pour *Large Deviations Statistics for Patterns*) qui est désormais intégré au logiciel SPatt (*Statistics for Patterns*) dont nous allons maintenant parler de manière détaillée.

SPatt

Ayant, grâce aux grandes déviations, un outil pratique permettant de calculer (de manière approchée) des statistiques de motifs, je me suis naturellement posé la question de la pertinence de cet outil par rapport aux outils existants. J'ai alors découvert avec surprise que si la littérature se révélait extrêmement abondante en méthodes, assez peu d'entre elles faisait l'objet d'une implémentation pratique.

Hugues Richard ayant effectué ce travail pour les calculs exacts fondés sur les simples récurrences (Robin et al., 2002), nous avons entrepris de comparer les performances de cet outil aux grandes déviations. Cela a donné lieu à la publication de SPA (Richard et Nuel, 2003), une web-application permettant de choisir automatiquement la meilleure approche pour un problème donné.

En cherchant à comparer ce nouvel outil au logiciel RMES (Schbath, 1997) je me suis heurté à certaines limitations méthodologiques de ce dernier programme (ex : calcul d'espérance et variances de comptage dans le cas du modèle *shuffle*[1] uniquement) et une certaine instabilité numérique dans le calcul des fonctions de répartitions et de survie des lois de Poisson composées.

Ces constatations ont fini par m'amener à reprendre une implémentation de ces méthodologies en commençant par considérer les simples approximations binomiales[2] (asymptotiquement identiques aux approximations de Poisson). En utilisant des approches numériques efficaces (calcul de la loi stationnaire par algorithme d'Arnoldi et fonctions de répartition et de survie via la fonction beta incomplète), j'ai ainsi produit S-SPatt (*S-SPatt* Simple Statistics for Patterns) qui a constitué le premier composant du package. Cette approche élémentaire se révélant étonnamment fiable (Nuel, 2005), elle fût dès lors considérée comme une bonne heuristique.

Cherchant à comprendre l'origine des instabilités numériques évoquées plus haut dans le calcul des approximations de Poisson composées, je me suis intéressé de près au cas particulier des lois de Poisson géométriques pour lequel j'ai mis au point des algorithmes linéaires en temps et constants en espace avec le nombre d'occurrences observées[3] permettant le calcul des fonctions de répartition et de survie (Nuel, 2006a).

J'ai ensuite finalement repris intégralement le calcul asymptotique des moments d'ordre 1 et 2 (cas Markov et non *shuffle*) ainsi que les formules exactes fondées sur les simples récurrences en ayant à coeur de contrôler la qualité des approximations numériques utilisées. Toutes ces méthodes ont finalement été présentées et comparées dans Nuel (2006d)

[1] Il s'agit du modèle dans lequel on tire uniformément dans l'ensemble des séquences dont les comptages de tous les mots de longueur $m+1$ sont contraints ; ce modèle est asymptotiquement équivalent au modèle markovien d'ordre m dont les paramètres sont estimés sur la séquence initiale.

[2] Cette approximation consiste à supposer que les indicatrices de la présence d'un motif à une position donnée dans la séquences sont indépendantes et identiquement distribuées ; cette hypothèse est évidemment complètement fausse !

[3] Les algorithmes fondés sur les formules de Barbour et al. (1992) ont des complexités au moins cubique en temps et quadratique en espace.

qui constitue la comparaison la plus complète de méthodes permettant d'étudier les statistiques de motifs publiée à ce jour.

L'ensemble des réalisations pratiques associées à ce travail de recherche sont mises à disposition de la communauté sous la forme de la suite logicielle SPatt (*Statistics for Patterns*) dont le site web est consulté (hors robots) par une douzaine de visiteurs par semaine (provenant de France bien sûr, mais aussi d'Europe, de Chine, des Amériques, d'Inde, du Moyen-Orient, etc ...).

`http://stat.genopole.cnrs.fr/spatt/`

Sensibilité à l'estimation des paramètres

Parallèlement au travail précédent, je me suis aussi posé la question de la robustesse des statistiques de motifs. En effet, les paramètres du modèle markovien de référence devant être estimés, il est naturel de se poser la question de la sensibilité des statistiques de motifs à cette estimation. Les longueurs des séquences biologiques utilisées pour ces estimations étant en général très grandes (par exemple de l'ordre du million), cet aspect du problème est généralement écarté et il n'est pas rare de considérer des modèles markoviens d'ordre 5 ou 6 pour modéliser des séquences d'ADN.

Or dans un tel modèle, même de faibles variations des paramètres peuvent avoir des conséquences importantes sur des quantités faisant intervenir de nombreux produits des paramètres comme cela est par exemple le cas pour le calcul de l'espérance d'un comptage.

Afin d'y voir plus clair, je me suis proposé d'examiner la robustesse de l'approximation binomiale. Il s'agit en effet d'une approximation particulièrement simple puisque son seul paramètre (hormis la longueur de la séquence) est la probabilité pour le motif d'apparaître à une position donnée dans la séquence.

La technique proposée dans Nuel (2006e) consiste à utiliser une approximation gaussienne pour la loi de $(\mathbf{N}_m, \mathbf{N}_{m+1})$ le vecteur des fréquences de tous les mots de longueur m et $m+1$. Les estimateurs d'un modèle markovien d'ordre m s'écrivant ensuite comme des fonctions de ce vecteur de fréquences, on peut exprimer la statistique de motifs comme une fonctionnelle de ce vecteur (cette fonctionnelle fait intervenir la fonction beta incomplète). En utilisant un développement de Taylor d'ordre 1 autour de la moyenne (delta-méthode), on obtient ainsi une approximation gaussienne pour la statistique de motif.

Cette méthode a été validée par simulations et l'étude conclut que l'influence de l'estimation des paramètres peut être suffisante pour fausser de manière importante les résultats d'une étude sur les motifs si un ordre markovien trop important est considéré. Typiquement, sur un génome de 5 Mb, l'utilisation de modèles markoviens d'ordre supérieur à $m = 5$ semble être une pratique risquée. Pour un génome dix fois plus court, $m = 4$ semble être la limite supérieure. Bien évidemment, les chiffres donnés ici n'ont qu'une valeur indicative tant la sensibilité à l'estimation des paramètres varie en fonction de l'étude considérée (le classement des motifs a-t-il son importance ou seulement la valeur des statistiques ? S'agit-il de motifs court ou long ? etc ...).

L'approche par PMC

Je me suis ensuite penché sur les calculs exacts fondés sur la technique des FMCI. Cette approche propose de construire pour chaque motif et chaque modèle, une chaîne de Markov auxilaire d'ordre 1 qui contient à elle seule la complexité combinatoire du motif

et qui permet de calculer des probabilités exactes en considérant de simples puissances de la matrice de transition associée.

Cette méthode aussi puissante qu'élégante se heurte cependant à deux problèmes :

1) les calculs semblent délicats (voir impossibles) dans les cas concrets ;

2) comment construire cette FMCI de manière optimale.

Considérant la nature éminemment creuse de la matrice de transition de la FMCI, je suis arrivé à la conclusion qu'un algorithme efficace devait, par essence, s'appuyer sur cette propriété. Certains travaux avaient d'ailleurs déjà pris en compte se point en proposant des formules de récurrences dans certains cas particuliers. Dans Nuel (2006c), je propose de tirer parti de cette idée pour proposer un algorithme général dont la complexité est linéaire avec la longueur de la séquence et l'ordre de la matrice de transition. Deux applications de cet algorithme sont alors détaillées : la première concerne le calcul de la significativité des scores locaux et la seconde s'intéresse aux motifs. Dans les deux cas, cette nouvelle approche permet même d'obtenir d'intéressants développements asymptotiques.

En cherchant de quelle manière construire la meilleure FMCI associée à un motif donné, le lien de ce problème avec la théorie des langages et automates est soudain apparu comme évidente. Il est bien connu en *pattern matching* que les automates constituent un moyen efficace de localiser et compter les occurrences d'un motif dans une séquence. Cette idée a d'ailleurs été exploitée par plusieurs auteurs pour construire les fonctions génératrices du comptage d'un motif.

Cette approche nécessite néanmoins des calculs formels assez lourds (résolution formelle d'un système d'équations linéaires par exemple) et même une fois les fonctions génératrices calculées, le calcul de moments ou de probabilités associés à la distribution correspondante peuvent être numériquement délicats.

M'inspirant de ces méthodes, j'ai introduit dans Nuel (2006b) pour chaque motif et pour chaque modèle la notion de PMC (*Pattern Markov Chain*) qui est une chaîne de Markov d'ordre un dont la propriété est que les occurrences de motifs dans la séquence initiale correspondent aux occurrences d'un sous-ensemble de "lettres" dans la PMC. Grâce à cette PMC, on obtient une réécriture optimale du problème de motifs dont toute la complexité est prise en charge de manière optimale par la structure de la matrice de transition associée.

Grâce à cette réécriture, toutes les méthodes classiques (FMCI, approximations gaussiennes, binomiales, Poisson composées et grandes déviations) permettant le calcul de statistiques de motifs trouvent un cadre naturel dans lequel elles puisent de plus une efficacité accrue.

Ce travail constitue une avancée considérable dans le domaine des motifs. Cette approche permet en effet, pour la première fois de traiter de manière efficace et rigoureuse des motifs de forte cardinalité tels que les motifs à trous (ex : `atgc.(5-12)ccat`[4]), les motifs PROSITE et bien d'autres qui n'étaient jusqu'alors pris en charge que par un nombre très limité d'approches (fonction génératrice par exemple) ou pas du tout.

Je co-écris actuellement avec Bernard Prum un ouvrage intitulé "Analyse statistique des séquences biologiques : modélisation markovienne, alignements et motifs" (commandé par les éditions Hermes) dont plusieurs chapitres sont intégralement consacrés à cette approche du problème des motifs.

[4]Cela signifie que le motif correspond à une occurrence de `atgc` suivie, après 5 à 12 caractères quelconques, d'une occurrence de `ccat`

Des extraits de ce livre constituent la deuxième partie de ce mémoire qui présente de manière détaillée la notion de PMC et ses applications.

Perspectives

L'introduction des PMC ouvre la voie vers de nombreuses applications de SPatt dans des domaines aussi variés que la régulation des gènes (motifs à trous), la fiabilité (*scan statistics*), ...

Il existe également de nombreux problèmes liés aux motifs où l'on est confronté à une multitude de séquences assez courtes. Dans un tel cas, l'utilisation d'un modèle markovien stationnaire introduit un biais qui peut fausser les statistiques de motifs. Les calculs exacts (p-values et moments) présentés dans la seconde partie du mémoire permettent de prendre en compte ce phénomène et de le corriger. Une collaboration sur ce sujet est d'ailleurs en cours avec l'équipe EBGM (Paris 7) pour une application à la recherche de motifs structuraux dans les protéines.

Il reste également un travail important à accomplir sur le calcul de lois jointes et conditionnelles pour les comptages de motifs. On dispose déjà d'approches exactes, mais elles sont difficiles voire impossibles à mettre en oeuvre en pratique pour des questions de complexité. Les grandes déviations de niveau 2 constituent une piste intéressante, mais les problèmes numériques associés sont encore loin d'être résolus.

Citons enfin le thème du lien existant entre l'évolution et les motifs. En effet, l'exceptionalité d'un motifs fonctionnel étant le résultat de pressions de sélection s'exerçant au fil de l'évolution, il semble particulièrement pertinent d'étudier ce processus dans son ensemble et ses conséquences sur l'exceptionalité du motif associé (ex : `gctggtgg`), mais aussi, et c'est là le point clef, sur les motifs voisins (ex : `gatggtgg`, `gctgtgg`, ...). Plusieurs tentatives (stages) ont déjà été effectuées dans ce sens, mais le manque de statistiques conditionnelles fiables et efficace a toujours été un frein important et, de ce fait, il reste encore beaucoup de travail à accomplir.

1.2.3 Etudes cas-témoins

La plupart des maladies humaines dépendent à la fois de conditions environnementales (ex : consommation de tabac pour le cancer du poumon) et du patrimoine génétique de chaque individu. Certaines personnes peuvent avoir une prédisposition à la maladie et d'autre être naturellement protégées. Identifier cette composante génétique des maladies est un enjeu de taille pour le monde médical, et l'industrie pharmaceutique en particulier, dans la mesure où cela pourrait permettre de mieux comprendre le fonctionnement de ces maladies et de donner aux chercheurs de nouvelles pistes pour la mise au point de nouveaux traitements.

Avec les études cas-témoins, on recueille pour cela les échantillons sanguins d'une populations de malades (les cas), et d'une population de non malades (les témoins). On effectue ensuite un génotypage de ces individus afin de caractériser le patrimoine génétique de chacun par un ensemble de marqueurs. On procède ensuite à un traitement statistique des données pour mettre en évidence l'association de marqueurs à la maladie.

C'est précisément la démarche entreprise par *Serono Genetics Institute* (SGI) sur quatre maladies : la polyarthrite rhumatoide, la sclérose en plaques, le lupus et le psoriasis. Pour le génotypage, c'est l'approche *genome wide*[5] qui a été retenue en utilisant des puces

[5]Cela consiste à prendre un ensemble de marqueurs uniformément répartis sur l'ensemble du génome ;

de 100 000 SNPs[6] (puis de 500 000 SNPs par la suite).

En 2004, le laboratoire "Statistique et Génome" poursuit avec SGI une collaboration afin de fournir des approches méthodologiques adaptées à l'analyse statistique de leurs données. En ce qui me concerne, cette collaboration notamment la forme de l'encadrement du stage de DEA de Mickaël Guedj sur le thème de l'utilisation du score local pour l'analyse des statistiques d'associations. Ce travail fait presque immédiatement l'objet d'une communication (Bourguignon et al., 2004) et est très bien reçu par SGI qui propose à Mickaël Guedj le financement d'une thèse CIFRE dont j'assure encore aujourd'hui la direction scientifique.

Ce thème de recherche s'est révélé très productif puisque, à ce jour, il a déjà donné lieu à trois publications : Guedj et al. (2006c) discute de la validité d'un test d'association classique (le test allélique) et propose à son sujet plusieurs enrichissements aussi bien théoriques que numériques ; Guedj et al. (2006b) décrit en détails la technique du score local qui permet d'identifier des régions significativement associées à la maladie ; Guedj et al. (2006a) est une étude de puissance des différentes statistiques d'association disponibles afin d'aider à choisir celle qui sera le plus appropriée à un problème donné.

Ce réorientation (partielle) de mes recherches vers la génétique et le monde médical semble également très prometteuse pour l'avenir. Ce thème est en effet actuellement moteur de nombreux projets : intérêt des méta-statistiques d'association, étude détaillée de la validité et des propriétés du test de Hardy-Weinberg, prise en compte des hétérogénéités cachées (collaboration avec l'Institut de Recherche pour le Développement au Bénin, avec des données sur le paludisme et un post-doctorat — E. Della-Chiesa), tests multiples et FDR[7] (collaboration avec l'INAP-G), reconstruction haplotypiques (collaboration avec une équipe INSERM du CNAM, sur le SIDA).

1.2.4 Divers

J'ai également eu l'occasion de travailler de manière plus ponctuelle sur un certain nombre d'autres sujets : une étude sur la présence d'inverses- complémentaires dans les génomes et de leurs liens avec l'évolution du protéome (Goldstein et al., 2003) ; l'annotation de génome microbien par système expert (Bocs et al., 2003) ; une librairie de programmation en C++ pour la manipulation de séquences biologiques (Miele et al., 2005).

1.3 Encadrement de recherche

Depuis la fin de ma thèse je me suis efforcé d'encadrer régulièrement des stages de recherche d'étudiants. Cette activité, qui s'est toujours révélée pour moi (et je l'espère également pour mes étudiants) aussi enrichissante qu'utile, me semble aujourd'hui plus que jamais indissociable du métier de chercheur et d'enseignant-chercheur.

Je tiens d'ailleurs à remercier chaleureusement l'ensemble de ces étudiants pour leurs questions, leur enthousiasme et leur énergie.

---

cette approche est opposée à l'approche gène candidat où on se contente de tester un certain nombre de régions spécifiques.

[6] Il existe dans le génome des positions spécifiques possédant le polymorphisme d'un seul nucléotide isolé dans la population : une telle position s'appelle un SNP (*Single Nucleaotide Polymorphism*). On estime à trois millions le nombre de SNPs chez l'Homme.

[7] *False Discovery Rate*

1.3.1 Niveau L3

janvier 2002 Maxime Huvet, stage de deux mois, *Influence de l'évolution dans l'étude des motifs de fréquences exceptionnelles dans les séquences d'ADN.*

janvier 2004 David Gomes, stage de deux mois, *Etude du spectre des matrices markoviennes estimées sur des séquences d'ADN.*

janvier 2005 Alexandre Jacob, stage de deux mois, *Statistiques de Motifs dans des Séquences Biologiques Segmentées.*

1.3.2 Niveau M1

janvier 2002 Adrien Richard, stage de six mois, *WWbar : un outil pour étudier les courts inverse-complémentaires dans les séquences d'ADN.*

janvier 2003 Sabrina Serin, stage de six mois, *Etude des origines biologiques de la présence de courts inverses-complémentaires dans les séquences d'ADN.*

1.3.3 Niveau M2

janvier 2004 Maxime Huvet, stage de six mois, *Détection* ab initio *de motifs biologiques dans les génomes.*

mars 2004 Mickaël Guedj, stage de six mois, *Déséquilibre de liaison et association à la maladie dans les études de SNPs cas-témoins à grande échelle.*

septembre 2005 Adrien Gaillard, stage de quatorze semaines, *Minimisation multidimensionnelle sous contraintes et application aux grandes déviations de niveau 2 pour les occurrences de mots dans des chaînes de Markov.*

septembre 2005 Nathanaelle Brasseur, stage de quatorze semaines, *Calcul de la fonction de répartition d'une forme quadratique en variables normales par inversion numérique de la fonction caractéristique.*

février 2006 Allal Houssani, stage de cinq mois, *Chaînes de Markov cachées pour l'estimation de fonction de score.*

1.3.4 Niveau D

depuis octobre 2004 Mickaël Guedj, co-direction (B. Prum 10% et G. Nuel 90%) d'une thèse CIFRE en partenariat avec Serono Genetic Institute, *Déséquilibre de liaison et association à la maladie dans les études de SNPs cas-témoins à grande échelle.*

1.4 Enseignements et responsabilités collectives

1.4.1 Enseignements

DEUG Science de la vie

depuis 1998 TD d'analyse en première année.

depuis 2000 TD sur les variables aléatoires discrètes en première année.

depuis 2000 TD sur les variables aléatoires continues en deuxième année.

depuis 2000 TD sur les v.a. continues (avec le logiciel Scilab) en deuxième année.

depuis 2002 TP d'analyse (avec Scilab) en première année.

depuis 2002 TP sur les v.a. discrètes (avec Scilab) en première année.

depuis 2004 TP Analyse (avec le logiciel R) en première année.

depuis 2004 TP sur les v.a. discrètes (avec R) en première année.

IUP Génie Biologique et Informatique

depuis 2000 TP sur l'alignement en troisième année.

depuis 2002 Cours sur l'alignement en troisième année.

2005 TD sur les chaînes de Markov en deuxième année.

2005 TP sur les chaînes de Markov (Scilab) en deuxième année.

Licence Biologie

depuis 2000 TP sur l'alignement en troisième année.

depuis 2002 Cours sur l'alignement en troisième année.

Licence Maths

2001 TP sous Scilab.

1.4.2 Responsabilités collectives

2000 et 2003 Coordination de la mise en oeuvre de deux salles pédagogiques équipées de machines LINUX pour le département de biologie.

Depuis septembre 2002 Participation au renouveau de l'enseignement des mathématiques en filière sciences de la vie avec, notamment la création de travaux pratiques avec Scilab (et maintenant avec le logiciel R)

Depuis septembre 2003 Gestion de l'interface entre les utilisateurs de ces salles machines et le service informatique de l'université.

Depuis septembre 2004 Membre de la commission de spécialiste de Mathématiques.

Deuxième partie

PMC pour les statistiques de motifs

Chapitre 2

Synthèse (G. Nuel)

2.1 Introduction

Présentation du problème

On a vu en section 1.2.2 (page 6) que les motifs biologiques (ADN, protéines, ...) qui sont impliqués dans des mécanismes fonctionnels se caractérisent par des fréquences *exceptionnelles*. Il est donc naturel de rechercher de nouveaux motifs biologiques parmi ceux vérifiant cette propriété, mais la chose n'est pas facile. La section 5.1 (page 83) illustre en détails cette difficulté en considérant le texte complet de "Cyrano de Bergerac". On constate que les fréquences observées des mots de ce texte dépendent de la structure même des mots (possibilités d'auto-recouvrement) mais aussi (et surtout) de la composition du texte en lettres, en paires de lettres, en triplets, etc ...

Partant de cette simple constatation, on conclut donc à la nécessité de prendre en compte ces biais de compositions sans quoi on risque fort d'identifier à tort comme exceptionnel un motif simplement composé de lettres fréquentes (ou rares). Cette idée amène naturellement à travailler conditionnellement aux comptages des lettres, ou des mots de longueur 2, de longueur 3, etc, et ainsi à adopter la modélisation markovienne des séquences par un modèle d'ordre m (noté Mm). On se placera ainsi dans le modèle M0 pour prendre en compte la composition en lettres de la séquence, dans le modèle M1 pour considérer la composition en paires, et ainsi de suite ...

Ces modèles Markoviens et leurs liens avec les comptages de mots dans les séquences considérées sont présentés en détails dans le chapitre 3 (page 33). A partir de maintenant, on considère le cadre suivant : soit $X = X_1 \dots X_\ell$ une chaîne de Markov d'ordre m sur l'alphabet fini $\mathcal{A}$ (de cardinal k). On note μ_0 sa loi initiale, π ses transitions et on se fixe comme objectif l'étude de la distribution de $N(\mathcal{W})$ le nombre d'occurrences (recouvrantes ou renouvelantes) dans X d'un motif donné $\mathcal{W}$.

Notons que si l'on justifie ici l'intérêt porté à ce problème par la recherche de motifs biologiques, il existe également de nombreuses autres applications (linguistique, fiabilité, assurance, théorie de jeux, ...) qui donnent à ce thème de recherche une dimension plus grande et expliquent les efforts considérables qu'il suscite depuis un demi-siècle de la part de la communauté scientifique et des statisticiens et probabilistes en particulier.

Les approches existantes

Il existe un grand nombre d'approches concurrentes et complémentaires permettant d'étudier la distribution de $N(\mathcal{W})$ dans des cas particuliers (alphabet binaire, modèle M0 uniquement, $\mathcal{W}$ réduit à un simple mot, etc ...) ou dans le cas le plus général. Il serait vain de vouloir ici recenser l'ensemble de ces approches de manière exhaustive, aussi va-t-on se contenter de considérer les principales d'entre elles. On peut classer ces techniques selon deux grands domaines : les résultats exacts et les approximations asymptotiques.

Les approches exactes utilisent les séries génératrices (Pevzner et al., 1989; Régnier et Szpankowski, 1998; Régnier, 2000), les familles exponentielles (Stefanov et Pakes, 1997, 1999), les Automates Finis Déterministes – AFD – (Atteson, 1998; Nicodème et al., 2002; Crochemore et Stefanov, 2003), les *Finite Markov Chain Imbedding* – FMCI – (Fu et Koutras, 1994; Fu, 1996; Fu et Lou, 2003) ou encore de simples récurrences (Robin et Daudin, 1999, 2000; Robin et al., 2002). Notons que de par leur nature exacte, il est possible, pour l'essentiel de ces méthodes, de prendre en compte des modèles markoviens hétérogènes (voir section 3.8 page 52 pour un exemple de modèle de ce type).

Les approximations asymptotiques en revanche, nécessitent le plus souvent de considérer un modèle markovien homogène, ergodique et stationnaire. En effet, c'est précisément la convergence de la loi marginale de la chaîne de Markov vers sa loi stationnaire qui se trouve au coeur de la plupart de ces méthodes (voir section 3.4.2 page 43 pour plus de détails sur cette propriété). On pense d'abord naturellement aux approximations gaussiennes (Cowan, 1991; Kleffe et Borodovsky, 1992; Prum et al., 1995), mais on peut aussi effectuer de simples approximations binomiales (van Helden et al., 1998; Nuel, 2005) ou encore les approximations de Poisson et Poisson composées (Chrysaphinou et Papastavridis, 1988; Arratia et al., 1990; Geske et al., 1995; Schbath, 1995; Robin, 2002) ou bien enfin, approcher le problème par la théorie des grandes déviations (Nuel, 2001, 2004; Pudlo, 2004).

Motifs de forte cardinalité

Pour l'immense majorité de ces approches, la cardinalité r du motif $\mathcal{W}$ [1] intervient directement dans la complexité numérique des calculs (de manière linéaire ou quadratique). Lorsque l'on considère des motifs de forte cardinalité (*i. e.* des motifs fortement dégénérés) commes les motifs protéiques PROSITE ou des motifs nucléiques structurés (voir section 5.9.1 page 153 pour quelques exemples), on peut facilement rencontrer des valeurs de r de l'ordre de 10^{10} ou 10^{30} ce qui rend la plupart des calculs impraticables.

Même s'il est parfois possible de développer pour un type de ces motifs des méthodes spécifiques (c'est par exemple le cas pour les motifs structurés traités dans Robin et al., 2002) cela débouche en général sur des méthodes numériquement lourdes et aux capacités limitées. Une exception notable à cette règle est l'approche du problème des motifs par le biais des AFD (Atteson, 1998; Nicodème et al., 2002; Crochemore et Stefanov, 2003). Cet outil bien connu en informatique (*string* et *pattern matching* notamment) permet en effet de décomposer le mécanisme d'apparition d'une occurrence d'un motif en étapes élémentaires et ce, de manière optimale. Les auteurs de cette démarche s'appuient ensuite sur cette décomposition caractéristique du motif pour construire des séries génératrices du comptage qui permettent de nombreux calculs (moments, distribution). S'il faut indiscutablement saluer la pertinence et l'innovation de cette approche on peut cependant regretter que son application se limite au seul champ des séries génératrices. En effet, de nombreuses autres approches, parfois numériquement plus efficaces, peuvent elles aussi profiter de la caractérisation des motifs par les AFD et c'est précisément ce que nous nous proposons de faire ici (notons que l'article Nuel, 2006b, contient l'essentiel des résultats présentés dans ce chapitre, notamment ceux qui concernent la notion centrale de PMC).

[1] La cardinalité d'un motif est définie comme le nombre de mots qui le composent. Par exemple, la cardinalité du motif nucléique `atat` vaut 1, celle du motif `at.t` vaut 4 et cette de `at.(10)t` vaut 4^{10}

Plan

On va tout d'abord expliciter cette approche par AFD en introduisant la notion de *Pattern Markov Chain* – PMC – (section 2.2). Nous verrons ensuite comment l'utiliser dans le cadre des calculs exacts (section 2.3) puis dans celui des approximations asymptotiques (section 2.4). Nous effectuerons ensuite une comparaison de ces différentes approches (section 2.5) avant de conclure en examinant les perspectives de ce travail (section 2.6).

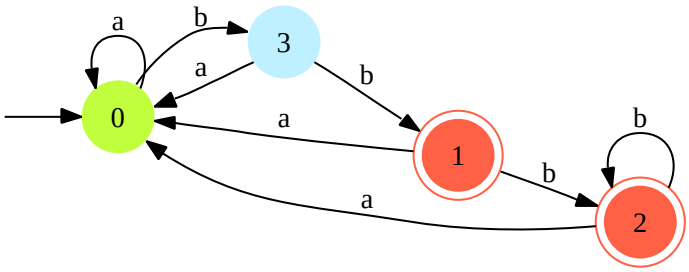
2.2 Notion de PMC

Langages et automates

On définit un langage $\mathcal{L}$ sur un alphabet fini $\mathcal{A}$ comme un ensemble quelconque de mots (voir section 4.2.1 page 60). On note $\mathcal{A}^*$ l'ensemble de tous les mots possibles. Un langage est régulier lorsqu'il peut être obtenu à partir de partie finie de $\mathcal{A}^*$ et d'opérations régulières. Ces opérations sont au nombre de trois : l'addition (*i. e.* réunion de langages), le produit (*i. e.* la concaténation de langage) et la clôture de Kleene noté * qui correspond à la réunion de toutes les puissances du langage.

Un AFD $(\mathcal{A}, \mathcal{Q}, s, \mathcal{F}, \delta)$ peut être représenté par un graphe dont les sommets sont les éléments de $\mathcal{Q}$, s désigne le sommet initial, $\mathcal{F}$ le sous ensemble des sommets terminaux et δ définit la structure des arêtes orientées de ce graphe de la manière suivante : $\delta(p, a) = q$ correspond à la présence d'une arête étiquetée par la lettre $a \in \mathcal{A}$ allant de p vers q. Le caractère déterministe de l'automate correspond au fait que $\delta : \mathcal{Q} \times \mathcal{A} \to \mathcal{Q}$ est une fonction. Une séquence composée des lettres rencontrées sur les arêtes en suivant dans l'automate un chemin allant de s à $\mathcal{F}$ est dite *reconnue* par l'automate. L'ensemble de toutes les séquences reconnues par l'automate est appelé son langage (voir section 4.2.2 page 61).

Prenons l'exemple de l'automate suivant :

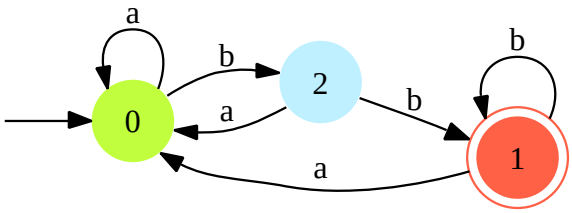


dont le language est simplement $\mathcal{L} = \{\mathtt{a}, \mathtt{b}\}^*\mathtt{bb}$. Il est en effet facile de vérifier que tout chemin reliant 0 à 1 ou 2 se termine par deux arêtes successives étiquetées par `b`.

Il existe un lien remarquable entre les langages réguliers et les AFD puisque tout langage reconnu par un AFD est régulier et, qu'à l'inverse, pour tout langage régulier il existe un unique (à isomorphisme près) plus petit AFD qui le reconnait.

La construction de l'AFD minimal associé à un langage régulier est en général non triviale et se fait en trois étapes : 1) construction d'un Automate Fini Non Déterministe – AFND – reconnaissant le langage (cette construction est élémentaire), 2) déterminisation de cet AFND pour produire un AFD (a priori non minimal), 3) minimisation de cet AFD. Ces trois étapes sont décrites en détails dans la section 4.3.1 page 63.

Voici par exemple l'automate minimal reconnaissant le langage $\mathcal{L} = \{\mathtt{a}, \mathtt{b}\}^*\mathtt{bb}$:



cet automate compte 3 états contre 4 pour la version non optimale précédemment présentée.

Le lien de cette théorie avec les motifs est mis en évidence par le résultat suivant :

Théorème 1. Pour tout motif $\mathcal{W}$ sur $\mathcal{A}$, et pour tout AFD $(\mathcal{A}, \mathcal{Q}, s, \mathcal{F}, \delta)$ reconnaissant le langage $\mathcal{L} = \mathcal{A}^*\mathcal{W}$ alors, si $x = x_1 \dots x_\ell$ est une séquence sur $\mathcal{A}$, la séquence sur $\mathcal{Q}$ définie par

$$y_0 = s \quad \text{et} \quad y_i = \delta(y_{i-1}, x_i) \quad \text{pour tout} \quad 1 \leqslant i \leqslant \ell$$

a la propriété suivante :

$$\mathcal{W} \text{ se termine en position } i \text{ dans } x \iff y_i \in \mathcal{F}.$$

Grâce à ce théorème il est très facile de compter les occurrences d'un motif $\mathcal{W}$ quelconque. Il suffit en effet de considérer un AFD qui lui est associé (minimal ou non) et de "passer" la séquence x à travers l'automate en notant les positions des éléments terminaux. Si on considère par exemple le motif $\mathcal{W} = \{\texttt{bb}\}$ sur l'alphabet binaire $\mathcal{A} = \{\texttt{a}, \texttt{b}\}$, on a vu ci-dessus deux exemples d'automates correspondants au langage $\mathcal{L} = \{\texttt{a}, \texttt{b}\}^*\texttt{bb}$. On considère pour simplifier l'AFD minimal et on cherche les occurrences de $\mathcal{W}$ dans $x = \texttt{abbababababbbaaabbab}$. Voici la séquence y que l'on produit alors grâce au théorème 1 :

x	–	a	b	b	a	b	a	b	a	b	b	b	a	a	a	b	b	a	b
y	0	0	2	**1**	0	2	0	2	0	2	**1**	**1**	0	0	0	2	**1**	0	2

où l'on voit bien se terminer une occurrence de $\mathcal{W}$ chaque fois que l'état 1 (seul état terminal) apparaît.

Notons qu'il est également aisé de modifier un AFD afin qu'il permette de compter les occurrences renouvelantes d'un motif (et non plus chevauchantes comme c'était le cas jusqu'à présent). On se reportera à la section 4.2.3 page 62 pour plus de détails sur ce point et plus généralement sur le comptage de motifs par le biais des automates.

Il est important de souligner que si les exemples considérés ici sont élémentaires (motif `bb` sur l'alphabet binaire) on pourra néanmoins appliquera rigoureusement la même technique aux motifs complexes (voir section 5.9.1 page 153 pour quelques exemples d'application).

PMC

La propriété de la séquence y définie dans le théorème 1 est tout à fait remarquable car elle permet de ramener un problème complexe – les occurrences d'un motif $\mathcal{W}$ dans la séquence x – à un problème plus simple – l'apparition du sous-ensemble $\mathcal{F}$ dans y –. Ainsi, toute la complexité du problème est intégrée, et ce de manière optimale dans le cas d'un AFD minimal, dans l'AFD associé au motif.

Le but de cette partie est d'étendre ce résultat au cas de $X = X_1 \dots X_\ell$ une chaîne de Markov d'ordre m. Pour cela, il est d'abord nécéssaire de modifier l'AFD correspondant au motif $\mathcal{W}$ afin que les transitions entre ses états soient définies de manière non ambiguë.

C'est précisément ce qui est fait dans la section 1 page 20 où l'on introduit la notion d'AFD non m-ambigu et où on donne un algorithme permettant d'obtenir des automates optimaux vérifiant cette propriété.

Il est alors possible d'introduire la notion de *Pattern Markov Chain* – PMC – définie par le résultat suivant (voir section 5.3 page 87 pour plus de détails)

Théorème 2. Soit $X = X_1 \dots X_\ell$ une séquence sur l'alphabet $\mathcal{A}$ sous le modèle Mm ($m \geqslant 0$) de loi initiale μ_0 et de paramètre π (π est réduit à μ dans le cas $m = 0$). Si $\mathcal{W}$ est un motif sur $\mathcal{A}$ et $(\mathcal{A}, \mathcal{Q}, s, \mathcal{F}, \delta)$ un AFD non m-ambigu (cette condition est vide lorsque $m = 0$) reconnaissant le langage $\mathcal{L} = \mathcal{A}^*\mathcal{W}$ alors la séquence $Y = Y_m \dots Y_\ell$ définie par

$$Y_0 = s \quad \text{et} \quad Y_i = \delta(Y_{i-1}, X_i) \quad \text{pour tout} \quad 1 \leqslant i \leqslant \ell$$

est une chaîne de Markov d'ordre 1 dont la matrice de transition Π vérifie pour tous $p, q \in \mathcal{Q}$

$$\Pi(p,q) = \begin{cases} \pi(a_1, \dots, a_m, b) & \text{si } \delta(p,b) = q \\ 0 & \text{si } q \notin \delta(p, \mathcal{A}) \end{cases}$$

où $a_1 \dots a_m$ désigne l'unique élément de $\delta^{-m}(p)$. Comme dans le cas M0 on a également

$$\mathcal{W} \text{ se termine en position } i \text{ dans } X \iff Y_i \in \mathcal{F}$$

$Y_m \dots Y_\ell$, la chaîne de Markov d'ordre 1 ainsi construite est une PMC associée au motif $\mathcal{W}$ et au modèle considéré. Notons que la loi initiale de la PMC (c'est à dire la loi de Y_m) correspond à la loi initiale de X (c'est à dire la loi de $X_1 \dots X_m$) mais que la PMC n'est *pas* stationnaire, même si X l'est.

Avant d'aller plus avant, faisons quelques remarques élémentaires concernant cette PMC. a) La séquence Y étant définie sur l'espace d'état $\mathcal{Q}$ de l'automate, il est clair que l'espace d'état de notre PMC sera optimal lorsque l'AFD utilisé pour sa construction le sera également ; on notera désormais L son cardinal. b) La matrice Π de transition de la PMC est creuse par nature puisqu'elle contient au plus $L \times k$ termes non nuls sur les L^2 possibles. c) On a la décomposition naturelle $\Pi = P + Q$ où P contient les transitions ne se terminant pas dans un état de comptage et Q uniquement de telles transitions.

Pour fixer les idées, reprenons l'exemple du motif `bb` sur l'alphabet binaire $\mathcal{A} = \{\mathtt{a}, \mathtt{b}\}$. L'AFD minimal qui lui correspond ayant la particularité d'être non 1-ambigu, on peut considérer le cas du modèle M1 de paramètres

$$\mu_0 = \begin{pmatrix} 0.5 & 0.5 \end{pmatrix} \quad \text{et} \quad \pi = \begin{pmatrix} 0.4 & 0.6 \\ 0.1 & 0.9 \end{pmatrix}$$

En utilisant le théorème 2 on construit alors la PMC $Y_1 \dots Y_m$ définie sur l'ensemble $\mathcal{Q} = \{0, 1, 2\}$ par

$$M_0 = \begin{pmatrix} 0.5 & 0 & 0.5 \end{pmatrix} \quad \text{et} \quad \Pi = P + Q = \begin{pmatrix} 0.4 & 0.0 & 0.6 \\ 0.1 & 0.0 & 0.0 \\ 0.1 & 0.0 & 0.0 \end{pmatrix} + \begin{pmatrix} 0.0 & 0.0 & 0.0 \\ 0.0 & 0.9 & 0.0 \\ 0.0 & 0.9 & 0.0 \end{pmatrix}$$

On est donc capable de ramener de manière canonique l'étude des occurrences d'un motif $\mathcal{W}$ dans une séquence $X_1 \dots X_\ell$ markovienne d'ordre m sur l'alphabet $\mathcal{A}$ (de cardinal k) à l'étude des occurrences du sous-ensemble d'états $\mathcal{F}$ (de cardinal F) dans la PMC $Y_m \dots Y_\ell$, chaîne de Markov d'ordre un sur un espace de cardinal L. C'est donc désormais ce problème plus simple que l'on va considérer dans la suite avec l'assurance que le caractère minimal de l'AFD utilisé nous garantit l'optimalité de la PMC associée (c'est à dire la valeur minimale pour le cardinal L).

2.3 Calculs exacts

On considère en premier lieu le cas des calculs exacts. Pour simplifier, on suppose ici la chaîne de Markov $X_1 \dots X_\ell$ (et donc également la PMC $Y_m \dots Y_\ell$) est homogène, tous les résultats proposés restent néanmoins valables dans le cas hétérogène.

Moments

Les résultats présentés ici sont détaillés dans la section 5.4.4 page 103 et s'inspirent largement de Nuel (2006d,b).

Pour tout $m \leqslant i \leqslant \ell$ la loi de Y_i est donnée par $\mu_0 \Pi^{i-m}$ où μ_0 est la loi initiale de la PMC et Π sa matrice de transition. Comme le comptage du motif $\mathcal{W}$ est simplement défini par

$$N(\mathcal{W}) = \sum_{i=m}^{\ell} \mathbb{I}\{Y_i \in \mathcal{F}\}$$

il est facile d'obtenir les moments de cette variable aléatoire grâce à la proposition suivante :

Théorème 3. Pour tout $a \in \mathcal{Q}$ on définit $N_\ell(a)$ le nombre d'occurrences de a (resp. b) dans la séquence $Y_m \dots Y_\ell$ on a alors

$$\mathbb{E}[N_\ell(a)] = \sum_{i=m}^{\ell} \mu_0 \Pi^{i-m} e'_a \quad \text{et} \quad \mathbb{E}[N_\ell(a) N_\ell(b)] = \mathbb{I}_{a=b} \times \mathbb{E}[N_\ell(a)] + C_\ell(a,b) + C_\ell(b,a)$$

avec

$$C_\ell(a,b) = \sum_{i=m}^{\ell-1} (\mu_0 \Pi^{i-m} e'_a) \sum_{j=i+1}^{\ell} (e_a \Pi^{j-i} e'_b).$$

où e_a (resp. e_b) désigne le vecteur (ligne) indicateur de la position a (resp. b).

Il ne reste alors plus qu'à considérer l'ensemble des $a, b \in \mathcal{F}$ pour obtenir l'espérance et la variance de $N(\mathcal{W})$ via l'algorithme 8 (page 106), ou bien, dans le cas homogène, grâce à la version plus rapide qui utilise la convergence de $\mu_0 \Pi^i$ vers la loi stationnaire pour accélerer sensiblement les calculs (voir algorithme 9 page 108).

On est donc capable de calculer de manière très rapide ($O(k \times L \times \ell)$ dans le cas hétérogène voire $O(k \times L \times \log \ell)$ dans le cas homogène et ergodique) les deux premiers moments de $N(\mathcal{W})$.

Temps d'attente

Il est également très facile d'obtenir des résultats concernant les temps d'attente d'une occurrence du motif comme cela est expliqué en détails dans la section 5.4.3 page 99 qui reprend en partie des résultats présentés dans Nuel (2006b).

Pour tout $i \geqslant m$ et $q \in \mathcal{Q}$ on commence par définir $\tau_i(q)$ le temps d'attente du premier $Y_j \in \mathcal{F}$ pour $j > i$ sachant que $Y_i = q$; il s'agit donc du temps d'attente de la première occurrence du motif partant de $Y_i = q$ et on établit de manière immédiate la proposition suivante :

Théorème 4. Pour tout $q \in \mathcal{Q}$, tout $f \in \mathcal{F}$ et tout $t \in \mathbb{N}$ on a :

$$\mathbb{P}(\tau_i(q) = t) = e_q P^{t-1} Q e'_{\mathcal{F}} \quad \text{et} \quad \mathbb{P}(\tau_i(q) = t, Y_{i+t} = f) = e_q P^{t-1} Q e'_f$$

et on a également la probabilité conditionnelle :

$$\mathbb{P}(Y_{i+t} = f | \tau_i(q) = t) = \frac{e_q P^{t-1} Q e'_f}{e_q P^{t-1} Q e'_{\mathcal{F}}}$$

où e_I désigne pour tout $I \subset \mathcal{Q}$ le vecteur (ligne) indicateur de l'ensemble I.

Comme dans le cas des calculs de moments, les calculs correspondants sont donc au plus d'une complexité en $O(k \times L \times \ell)$. Comme précédemment, on peut y remplacer ℓ par $\log \ell$ dans le cas homogène et ergodique et étudiant la convergence des puissances de P (attention cependant, cette matrice est sous-stochastique si bien que les résultats diffèrent légèrement du cas stochastique ; en particulier, la plus grande valeur propre $\lambda < 1$ de P joue ici un rôle clef).

On est donc très facilement capable de connaître la distribution des temps d'attente du motif ce qui peut servir à étudier la répartition des occurrences du motif le long de la séquence (section 5.4.3 page 100) ou encore à simuler une telle répartition sans avoir à générer la séquence sous-jacente (section 5.4.3 page 101).

FMCI

On s'intéresse maintenant directement à la distribution de $N(\mathcal{W})$ en introduisant la technique bien connue des *Finite Markov Chain Imbedding* – FMCI – tel que cela est fait en détails dans la section 5.4.1 page 94. Ces résultats proviennent de Nuel (2006c,b).

Pour tout $c \in \mathbb{N}$ on commence par définir la FMCI $Z = Z_m \dots Z_\ell$ de la manière suivante :

$$Z_j = \begin{cases} (Y_j, N_j) & \text{si } N_j < c \\ c_+ & \text{si } N_j \geqslant c \end{cases}$$

où N_j est le nombre d'occurrences du motif dans la séquence $X_1 \dots X_j$. Comme la PMC, Z est une chaîne de Markov d'ordre 1 et si on ordonne son espace d'états par N_j croissants, puis par Y_j et si l'on place c_+ en dernière position (ainsi, on a l'ordre suivant : $(1,0), \dots, (L,0), (1,1), \dots, (L,1), \dots$) sa matrice de transition T est alors définie par blocs par :

$$T = \left(\begin{array}{ccc|c} & R & & v \\ \hline 0 & \cdots & 0 & 1 \end{array} \right)$$

où $R = (R_{i,j})_{0 \leqslant i,j < c}$ de dimension $cL \times cL$ est formé de $c \times c$ blocs de taille $L \times L$ et $v = (v_i)_{0 \leqslant i < c}$ de dimension $cL \times 1$ de c blocs de taille $L \times 1$ vérifiant :

$$R_{i,j} = \begin{cases} P & \text{si i=j} \\ Q & \text{si i+1=j} \\ 0 & \text{sinon} \end{cases} \quad \text{et} \quad v_i = \begin{cases} 0 & \text{si } 0 \leqslant i < c-1 \\ \Sigma_Q = \sum_j Q(\cdot, j) & \text{si } i = c-1 \end{cases}$$

La distribution de $N_\ell = N(\mathcal{W})$ est alors simplement donnée par :

Théorème 5.

$$\mathbb{P}(N_\ell = i) = M_0 T^{\ell-m} E_i \text{ pour } 0 \leqslant i < c \quad \text{et} \quad \mathbb{P}(N_\ell \geqslant c) = u T^{\ell-m} E_{c_+}$$

avec

$$M_0 = (\mu_0\, 0 \cdots 0 \,|\, 0); \quad E_i = (0 \cdots 0\, 1\, 0 \cdots 0 \,|\, 0)'; \quad \text{et} \quad E_{c_+} = (0 \cdots 0 \,|\, 1)'$$

où le bloc de '1' dans E_i se trouve en $i^{\text{ème}}$ position.

En exploitant la structure creuse de T pour effectuer de manière itérative le calcul de $M_0 T^{\ell-m}$ on obtient l'algorithme de la proposition 97 (page 96) dont la complexité en temps est en $O(k \times c \times L \times \ell)$. Remarquons cependant qu'ici, contrairement aux calculs des moments ou des temps d'attentes, il n'est pas possible [2] d'utiliser la convergence des puissances de T vers sa loi stationnaire (qui est ici dégénérée puisque la FMCI possède un état absorbant).

Notons que ces résultats peuvent facilement s'étendre aux calculs de lois jointes au prix d'une augmentation (importante !) de l'espace d'état (voir section 5.4.5 page 110).

Plusieurs séquences

Jusqu'à présent on a considéré le problème des occurrences d'un motif dans une séquence donnée. Il est cependant possible de considérer le même problème avec un jeu de plusieurs séquences (éventuellement modélisée de manières différentes). Si la prise en compte de ce problème est extrêment délicate pour les méthodes asymptotiques (en particulier si l'on considère un grand nombre de courtes séquences), ce n'est pas le cas pour les méthodes exactes qui s'adaptent très facilement. La section 5.4.6 page 112 explique en détails comment il convient alors de procéder.

2.4 Approximations asymptotiques

A partir de maintenant on se place dans le cas d'un modèle homogène et ergodique que l'on supposera le plus souvent également stationnaire.

Approximations gaussiennes

Comme cela est expliqué dans la section 5.5 (page 114), le Théorème de la Limite Centrale – TLC – nous assure que la loi de $N(\mathcal{W})$ est asymptotiquement gaussienne. Comme on sait calculer (voir section 2.3) efficacement les deux premiers moments du comptage on peut asymptotiquement approcher sa loi par la gaussienne correspondante.

Ainsi, on associe à une observation de n occurrences du motifs le

$$\text{z-score} = \frac{n - \mathbb{E}\left[N(\mathcal{W})\right]}{\sqrt{\mathbb{V}\left[N(\mathcal{W})\right]}}$$

qui est asymptotiquement distribué selon une loi gaussienne centrée et réduite si on remplace n par $N(\mathcal{W})$.

[2] Il est cependant possible d'obtenir quelques développements asymptotiques intéressants en revenant aux matrices P et Q (voir section 5.4.2 page 98).

Lorsque l'on compare cette approximation à la loi exacte, on constate, comme on peut s'y attendre, une bonne adéquation dans la partie centrale de la distribution (voir figure 5.8 page 5.8). En revanche, comme c'est classiquement le cas avec le TLC, la qualité de l'approximation se dégrade fortement dans les queues de distribution.

On peut aussi obtenir les moments du comptage et une approximation similaire en se plaçant dans le modèle *shuffle* (voir section 5.5.5 page 121) plutôt que markovien. Dans ce modèle, on se donne une séquence observée, on fixe une longueur $h \geqslant 1$ et on tire notre séquence aléatoire dans l'ensemble des séquences ayant exactement les mêmes fréquences en mots de longueur h que la séquence initiale. Intuitivement (c'est même exactement le cas lorsque $h = 1$), on obtient notre séquence aléatoire en mélangeant (*shuffle* en anglais) les mots de longueur h. Il est facile de voir qu'un tel modèle est proche [3] d'un modèle markovien d'ordre $m = h-1$ estimé sur la séquence initiale, où les fréquences des mots de longueur h sont également respectées, mais seulement *en moyenne*. C'est historiquement cette approche qui était utilisée pour étudier les motifs dans les séquences. Comme il y a cependant comparativement beaucoup moins de résultats concernant les modèles shuffle que pour les modèles markoviens, cette modélisation est peu a peu tombée en désuétude et est désormais assez peu utilisée.

Il est très facile d'étendre ces résultats aux lois jointes de comptages, y compris à celles de $(\mathbf{N}_m, \mathbf{N}_{m+1})$ le vecteur aléatoire des comptages de tous les mots de longueurs m et $m+1$ (voir section 5.5.3 page 116). Cette dernière approximation est très intéressante car elle permet d'approcher la loi de l'estimateur d'une chaîne de Markov d'ordre m qui s'écrit comme une fonction de ce vecteur. Nous allons justement exploiter cette approximation dans la section suivante pour prendre en compte l'estimation des paramètres.

Approximations binomiales

On rappelle que

$$N(\mathcal{W}) = \sum_{i=m}^{\ell} \mathbb{I}\{Y_i \in \mathcal{F}\}$$

si on suppose que la PMC $Y_m \dots Y_\ell$ est stationnaire de loi μ [4] et que l'on suppose que ces indicatrices sont indépendantes les unes des autres [5] alors $N(\mathcal{W})$ suit une loi binomiale $\mathcal{B}(\ell - m + 1, \mu(\mathcal{F}))$ (voir section 5.6 page 125).

Cette approximation, a priori très grossière, se révèle cependant étonnament efficace, surtout lorsque l'on considère des motifs $\mathcal{W}$ peu ou pas recouvrants [6]. Cet observation a déjà été faite par van Helden et al. (1998) (par le biais de simulations) et plus récemment par Nuel (2005) (en comparant l'approximation binomiale à d'autres approximations).

Dans le cas particulier d'un motif non dégénéré réduit à un seul mot, on a une formule explicite de $\mu(\mathcal{F})$ à partir de la loi stationnaire du modèle Mm initial. On obtient alors une expression très simple de la p-value (en échelle logarithmique) associée à une observation donnée en utilisant les fonctions Beta complètes et incomplètes.

Jusqu'à présent, on a supposé les paramètres de notre modèle Mm connus alors qu'on sait bien qu'en réalité ils doivent être estimés. Ces estimateurs s'appuyant sur $(\mathbf{N}_m, \mathbf{N}_{m+1})$, le vecteur des fréquences des mots de longueurs m et $m+1$ dans les

[3] Ces deux modèles sont même asymptotiquement équivalents.
[4] On sait que cela n'est jamais le cas, même lorsque la séquence initiale est elle même stationnaire.
[5] Ce qui est évidemment complètement faux.
[6] On verra d'ailleurs une justification théorique à ce résultat dans la section suivante.

séquences servant à l'estimation, pour lequel on dispose d'une approximation gaussienne (voir section précédente). En utilisant un développement de Taylor d'ordre 1 autour de l'espérance [7], il est alors possible d'obtenir une approximation gaussienne pour la p-value (en échelle logarithmique) d'une observation. Cette technique initialement proposée dans Nuel (2006e) est détaillée dans la section 5.6.1 page 132.

Il est remarquable de noter que la variabilité ainsi mise en évidence est assez surprenante par son importance. A titre d'exemple, l'utilisation d'un modèle d'ordre $m \geqslant 5$ estimé sur un génome dont la taille se compte en Mb génère une variabilité d'estimation qui sera bien souvent inacceptable. Considérant le fait que le nombre de paramètres d'un modèle markovien croit géométriquement avec l'ordre m, il convient donc d'accorder une attention toute particulière à la variabilité due à l'estimation des paramètres dans les études de motifs. Dans ce but, l'approximation proposée ici peut être d'une grande aide même s'il est également possible de recourir aux simulations pour étudier ce phénomène qui est souvent négligé à tort.

Approximations de Poisson

La section 5.7 page 134 commence par rappeler un mécanisme important dans l'apparition des occurrences d'un mot ou d'un motif : l'auto-recouvrement. Il s'agit de la façon dont une première occurrence du motif peut recouvrir celle qui la suit et ainsi la faciliter. Prenons par exemple le cas du motif `aba`, il est clair que si une première occurrence de ce motif apparaît à une position donnée, il suffit de voir apparaître les lettres `ba` à sa suite pour obtenir une seconde occurrence recouvrant (ou chevauchant) la première. Ainsi les occurrences de `aba` arriveront par paquets distincts (*clump* en anglais) de une ou plusieurs occurrences

Ce phénomène central dans l'étude des occurrences de motifs avait jusqu'alors été occulté par l'AFD associé au motif et la PMC correspondante dont le rôle est précisément de prendre en charge ce type de comportement.

Dans les approches classiques, on définit une matrice d'auto-recouvrement du motif en faisant intervenir des notions comme la périodicité (principale ou non) des mots qui le composent et on montre que cette matrice détermine la loi du nombre d'occurrences de motif dans un paquet. Sous de bonnes conditions, la méthode de Chen-Stein permet alors d'approcher la loi du nombre de paquet par une loi de Poisson et ainsi d'obtenir une approximation de Poisson composée pour $N(\mathcal{W})$.

Dans la section 5.7.2 page 137 on généralise cette notion aux AFD afin de définir de manière optimale la matrice $\mathbf{A}$ (d'ordre F, le cardinal de $\mathcal{F}$) d'auto-recouvrement d'un motif donné et on obtient le résultat suivant [8] :

Théorème 6. $N(\mathcal{W})$ est approximativement distribué selon une loi de Poisson composée de paramètres

$$\lambda_\kappa = (\ell - m + 1)\mathbf{M}\mathbf{A}^{\kappa-1}(\mathbf{I} - \mathbf{A})^2\mathbf{1} \quad \kappa \geqslant 1$$

où $\mathbf{M} = \big(\mu(f)\big)_{f\in\mathcal{F}}$, $\mathbf{A} = (A_{i,j})_{i,j\in\mathcal{F}}$, $\mathbf{I}$ est la matrice identité et $\mathbf{1}$ est un vecteur colonne de 1.

[7] Cette technique est souvent désignée sous le nom de Delta-méthode.

[8] Ce résultat est dû à Roquain et Schbath (2006) dans le cas classique mais revient à Nuel (2006b) sous sa forme optimisée par l'approche par AFD.

ce qui veut dire que la loi du nombre de paquets suit une loi de Poisson de paramètre

$$\lambda = \sum_{\kappa \geqslant 1} \lambda_\kappa = (\ell - m + 1)\mathbf{M}(\mathbf{I} - \mathbf{A})\mathbf{1}$$

(la dernière égalité étant valable uniquement si $\mathbf{I} - \mathbf{A}$ est inversible) et la loi du nombre K d'occurrences par paquet est alors donnée par

$$\mathbb{P}(K = k) = \frac{\lambda_k}{\lambda}$$

Dans le cas où $\mathbf{A} \simeq 0$ (c'est à dire que l'on considère un motif non recouvrant [9]), $\lambda = \lambda_1 = (\ell - m + 1)\mu(\mathcal{F})$ et la distribution de $N(\mathcal{W})$ est donc approchée par une simple loi de Poisson. Notons au passage que cette approximation est évidemment très proche de l'approximation binomiale de la section précédente ce qui lui donne au passage à cette dernière une justification théorique de poids.

Lorsque $\mathbf{A} \simeq (1 - \theta)\mathbf{I}$, K se trouve distribué selon une loi géométrique et la loi de Poisson composée se ramène alors à une simple loi de Poisson géométrique. Si dans le cas d'une loi de Poisson, le calcul de la distribution ou des fonctions de répartition ou de survie est facile (par le biais de la fonction Gamma incomplète), il n'en est malheureusement rien dans le cas d'une loi de Poisson géométrique. Fort heureusement Nuel (2006a) propose une approche de ces calculs efficace grâce à une nouvelle relation de récurrence pour le calcul de la distribution et de la fonction de répartition qui est complétée en section 5.7.5 page 140 par une seconde récurrence pour le calcul de la fonction de survie. Les algorithmes proposés permettent d'effectuer l'ensemble de ces calculs avec une complexité linéaire en temps et constante en espace (à comparer aux complexités du cas général, voir ci-dessous)

Dans le cas le plus général (voir 5.7.6 page 145), le meilleur algorithme existant permet d'obtenir la distribution de la loi de Poisson composée avec une complexité cubique et temps et quadratique en espace. Cela constitue évidemment une sévère limitation à l'usage pratique de cette approximation.

Grandes déviations

On a vu avec les TLC que la qualité des approximations gaussiennes se dégrade lorsque l'on considère les queues de la distribution. Dans un tel cas, il est classique de recourir aux outils probabilistes dédiés à l'étude des extrêmes comme les grandes déviations, c'est précisément ce que l'on se propose de faire dans la section 5.8 page 147.

Grâce au TLC, on sait que les fluctuations de la somme d'un échantillon de taille n autour son espérance sont de l'ordre de grandeur de $\sqrt{n}$. Lorsque l'on considère des fluctuations linéaires avec n on rentre dans le cadre des grandes déviations (par opposition aux déviations normales du TLC) qui assure que la probabilité de l'événement considéré décroit à vitesse exponentielle.

C'est un résultat de ce type que l'on obtient avec le théorème suivant :

Théorème 7 (Cramer-Chernov)**.** Pour tout $a \geqslant 0$ on a

$$\lim_{\ell \to +\infty} \frac{1}{\ell - m + 1} \log \mathbb{P}\left(\frac{N(\mathcal{W})}{\ell - m + 1} \geqslant a \right) = -\Lambda^*(a) \quad \text{si } a \geqslant \frac{\mathbb{E}[N(\mathcal{W})]}{\ell - m + 1}$$

[9] C'est d'ailleurs systématiquement le cas lorsque l'on considère les comptages renouvelants d'un motif.

et

$$\lim_{\ell\to+\infty} \frac{1}{\ell-m+1} \log \mathbb{P}\left(\frac{N(\mathcal{W})}{\ell-m+1} \leqslant a\right) = -\Lambda^*(a) \quad \text{si } a \leqslant \frac{\mathbb{E}[N(\mathcal{W})]}{\ell-m+1}$$

avec

$$\Lambda^*(a) = \sup_{\theta\in\mathbb{R}}\{\theta.a - \Lambda(\theta)\}$$

où $\Lambda(\theta)$ est la logarithme de la plus grande valeur propre de $P + e^\theta Q$.

En utilisant la structure creuse des matrices P et Q, il est possible d'évaluer un $\Lambda(\theta)$ avec une complexité $O(k \times L)$. A l'aide d'une optimisation numérique de type Brent, il est donc possible d'évaluer $\Lambda^*(a) = \eta.a - \Lambda(\eta)$ (où η vérifie $\Lambda'(\eta) = a$) avec la même complexité. La mise en oeuvre pratique ce résultat et son application aux motifs est due à Nuel (2001, 2004).

Notons qu'il est également possible de raffiner l'approximation proposée par le théorème de Cramer-Chernov avec le résultat suivant :

Théorème 8 (Bahadur-Rao)**.** Pour tout $a \geqslant \mathbb{E}[N(\mathcal{W})]/(\ell-m+1)$ (resp. $\leqslant$) non nul, on a

$$\mathbb{P}\left(\frac{N(\mathcal{W})}{(\ell-m+1)} \geqslant a\right) \sim C(a) \times e^{-(\ell-m+1)\Lambda^*(a)} \quad \text{(resp. } \leqslant\text{)}$$

asymptotiquement lorsque ℓ tend vers l'infini avec

$$C(a) = \frac{1}{(1-e^{-|\eta|})\sqrt{2\pi(\ell-m+1)\Lambda''(\eta)}}$$

où η vérifie $\Lambda'(\eta) = a$.

Pour mettre en oeuvre ce théorème, il est cependant nécessaire de calculer numériquement $\Lambda''(\eta)$ (car on ne dispose malheureusement pas de forme analytique). Pour cela, on peut par exemple recourir à une approximation de la fonction Λ par des polynomes de Chebyshev au voisinage de η. Cette méthode donne d'assez bons résultats tout en conservant une complexité globale en $O(k \times L)$. L'application du théorème de Bahadur et Rao au motif apparaît dans Nuel (2006d). Le lien avec les PMC (et les matrices P et Q) est précisé dans Nuel (2006b).

Signalons enfin qu'il existe aussi des résultats de grandes déviations pour la distribution empirique des lettres, c'est à dire que l'on peut considérer la loi jointe de comptages (voir section 5.8.3 page 151). Il faut cependant noter que les calculs correspondants (minimisation sous contrainte d'une entropie relative dans un espace de dimension L^2) n'ont pas pour l'instant de solution numérique satisfaisante ce qui limite notablement l'application de ce résultat.

2.5 Comparaisons

Dans cette partie, on se propose de comparer les performances des différentes approches du problème à la fois en termes de vitesse de calcul et de qualité des résultats. Notons que toutes les méthodes que nous venons d'énumérer [10] sont implémentées dans le logiciel SPatt disponible à l'adresse suivante :

`http://stat.genopole.cnrs.fr/spatt/`

et dont il est ici fait un usage intensif.

[10] A l'exception des approximations gaussiennes dans le cas *shuffle* qui sont implémentées dans le logiciel RMES

Complexités

La table 5.1 (page 154) permet de comparer en détails les complexités des différentes méthodes. Ce qu'il faut d'abord retenir c'est que la complexité kL qui correspond au produit creux de la matrice de transition de la PMC par un vecteur apparaît dans toutes les méthodes. Pour les méthodes exactes, la longueur ℓ de la séquence intervient également avec, en prime, le nombre d'occurrence n du motif pour la distribution exacte. De ce fait, les méthodes exactes sont beaucoup plus lentes que les approximations asymptotiques. Ces dernières sont en général très rapides (à l'exception des approximations de Poisson composées dans le cas général qui sont en pratique inutilisables pour des valeurs importantes de n) même si les approximations issues des grandes déviations demandent plus de temps que les autres (on peut s'en rendre compte en consultant la table 5.3 page 156).

On constate donc que le cardinal L de l'espace d'état de la PMC joue un rôle prépondérant dans la complexité de ces méthodes. La table 5.2 page 155 donne pour quelques exemples de motifs et ordres m les valeurs de L rencontrées. Citons deux exemples permettant de mettre en avant toute la puissance de l'approche par PMC : 1) le motif nucléique structuré `atgca.(11)att.(12)gcatt` dont la cardinalité est $r = 4^{23} \simeq 7 \times 10^{13}$ pour lequel $L = 2220$ (cas $m = 0$) ; 2) le motif protéique

```
CG(2).(4-7)G.(3)C.(4-5)C.(3-5)[NHGS].[FYWMI].(2)QC
```

dont la cardinalité est $r \simeq 10^{30}$ et pour lequel $L = 16064$ ($m = 0$).

Pour le premier motif, la construction de l'automate correspondant requiert 0.27s, le comptage de ses occurrences sur le génome complet d'*Escherichia coli* ($\ell \simeq 4.5 \times 10^6$) et le calculs de l'approximation gaussienne des demande 0.16s de plus. Pour le second motif, il faut 56s pour contruire l'automate et le comptage de ses occurrences dans l'ensemble de la base de données Uniprot ($\ell \simeq 114 \times 10^6$) et le calcul de l'approximation gaussienne nécessite environ 4s de plus.

Qualité des approximations

Afin de juger de la qualité des approximations asymptotiques, on considère deux critères : l'erreur relative (sur la p-value en échelle logarithmique) et l'accord de rangs (tau de Kendall) en référence à la méthode exacte. Notons que les comparaisons effectuées dans toute la section 5.9 page 153 (et dans cette partie) proviennent pour l'essentiel de Nuel (2006d).

Si les approximations gaussiennes sont efficaces, comme on si attend, dans le centre de la distribution, elles sont très mauvaises dans les extrêmes que ce soit en terme d'erreur relative ou au niveau des rangs (mais dans une moindre mesure sur ce dernier point).

Comparativement, les approximations binomiales, bien que très élémentaires, sont bien meilleures au niveau de l'erreur relative et leurs performances en terme de rang sont du même ordre de grandeur que celles des approximations gaussiennes. Ceci permet donc de conclure comme dans Nuel (2005) que ces approximations très simples constituent une heuristique efficace. Comme la théorie le prédit, cette heuristique est cependant mauvaise lorsque l'on considère des motifs auto-recouvrants pour lesquels les approximations de Poisson composées sont bien plus adaptées.

La comparaison de ces dernières approximations (qui sont ramenées dans le cas de motif non dégénérés – simples mots – à des lois de Poisson géométriques) avec les valeurs exactes met en évidence une qualité que seules les grandes déviations précises parviennent à dépasser. Notons néanmoins que seules les grandes déviations conservent (et améliorent

même) leurs performances lorsque l'on considère spécifiquement des événements extrêmes (voir section 5.9.4 page 159).

Pour conclure on peut dire qu'en l'abscence de résultats exacts on devrait le plus possible éviter de recourir aux approximations gaussiennes pour les motifs les plus sur- ou sous-représentés (qui sont bien évidemment également les motifs les plus intéressants) pour lesquels les approximations binomiales peuvent donner une première idée du résultat qui devra néanmoins être confirmé par des approximations de Poisson (dont l'usage est cependant numériquement proscrit dans le cas de motifs fréquents pour lesquels on ne peut se ramener au cas Poisson ou géométrique Poisson, voir section précédente) ou par les grandes déviations précises.

2.6 Conclusion

Bilan

En introduisant la notion de PMC on dispose donc d'une méthode puissante pour considérer des motifs complexes dont le traitement était jusqu'ici difficile ou impossible. Cette nouvelle écriture du problème s'adapte de manière naturelle à de nombreuses méthodes tout en les optimisant. Dans le cas des FMCI par exemple, la construction optimale de la matrice de transition est évidemment un problème critique qui trouve avec les PMC sa solution. De la même façon, les grandes déviations bénéficient du caractère minimal de la PMC pour obtenir la meilleure performance possible. Dans le cas des approximations de Poisson composées enfin, la matrice d'auto-recouvrement obtenue à l'aide de l'AFD caractéristique du motif permet d'obtenir les mêmes résultats qu'avec les approches classiques mais bien plus efficacement. Dans tous les cas, c'est désormais le cardinal de l'espace d'état de la PMC et non plus la cardinalité des motifs qui se trouve finalement être la quantité d'intérêt.

Perspectives

Dans quelques cas particuliers (FMCI, approximation gaussiennes, grandes déviations) nous voyons qu'il est possible d'obtenir des lois jointes ou conditionnelles de comptages. Pour cela, il est cependant nécessaire que l'ensemble $\mathcal{F}$ des états terminaux de l'AFD puisse être partitionné en sous ensembles distincts correspondants chacun aux différents motifs intervenant dans le conditionnement. Or cette contrainte n'est en pas prise en compte dans la production de l'AFD minimal. Si on considère par exemple le motif nucléique `g.tggtgg`, l'AFD optimal qui lui est associé a 11 état dont un seul final. Il n'est donc matériellement pas possible, en considérant cet automate, de s'intéresser (par exemple) à la loi jointe de $(N(\texttt{g[at]tggtgg}), N(\texttt{g[cg]tggtgg}))$ car les comptages de ces deux motifs ne peuvent être distingués. Pour remédier à ce problème, il est donc nécessaire d'adapter les algorithmes de création des AFD optimaux pour qu'ils puissent prendre en compte de telles contraintes.

Même en ignorant ce premier verrou, l'étude de lois conditionnelles se heurte encore à de nombreuses difficultés selon l'approche envisagée. Avec les grandes déviations de niveau 2, c'est la minimisation sous contraintes d'une entropie relative qui pose problème. Avec les approximations binomiale, il est sans doute possible d'envisager un extension des résultats vers la loi multinomiale, mais pourrait-on obtenir une généralisation similaire pour les approximations de Poisson ?

La prise en compte de l'estimation des paramètres dans les statistiques de motifs est également en enjeu de taille. L'approche par Delta méthode utilisée dans le cadre des approximations binomiales pourrait certainement être relativement facilement généralisée aux approximations de Poisson (cas Poisson géométrique notamment) mais semble plus délicate à mettre en oeuvre avec les calculs exacts ou les grandes déviations. De plus, les résultats proposés ne sont actuellement valables que dans le cas de motifs non dégérés (de simples mots) car on dispose alors d'une expression simple pour $\mu(\mathcal{F})$. La généralisation de ce résultat à une PMC quelconque serait donc déjà une avancée considérable.

Nous avons également vu que les résultats exacts permettent de considérer des jeux de données composés de plusieurs séquences. Or il n'est pas rare dans le domaine des motif de se trouver confronté à un grand nombre de courtes séquences pour lesquelles l'utilisation d'un modèle stationnaire semble particulièrement dangeureuse. En effet dans une telle situation, les premières lettres de chacune des séquences ont une distribution qui est distincte de la loi stationnaire et, contrairement au cas d'une seule séquence, on ne peut se permettre de négliger cette différence car elle concerne ici une part importante des données. Il serait donc intéressant d'étudier de plus près ce phénomène et d'évaluer l'intérêt des méthodes exacte développées ici par rapport au méthodes faisant l'hypothèse de la stationnarité.

Avec les résultats sur les temps d'attente, nous avons vu un moyen simple et rapide pour s'intéresser à la répartition spatiale des motifs tout au long d'une séquence. En effet, il est possible grâce à ces résultats d'associer une p-value à chaque observation d'une occurrence du motif. La séquence de ces p-values peut donc certainement nous aider à localiser des régions riches ou pauvres en occurrences de motifs. En comparant ces régions avec des annotations connues (gène/non gène, sens de réplication, etc ...) on peut donc espérer faciliter la détection de motifs biologiques en tenant compte de leur contexte.

Citons enfin les modèles hétérogènes qui peuvent, comme on l'a vu, être considérés avec la même complexité numérique que les modèles homogènes par toutes les approches exactes (moments, temps d'attente, FMCI). Il serait sans doute particulièrement intéressant d'utiliser ces méthodes sur des modèles estimés par chaînes de Markov cachées (HMM) qui permettent de mieux modéliser la structure complexe des séquences biologiques.

Chapitre 3

Chaînes de Markov (B. Prum et G. Nuel)

3.1 Un exemple

La bactérie *Escherichia coli*, autrefois appelée le *Colibacille* est, avec la levure *Saccaromyces cerevisiae* le "microbe" le plus étudié depuis Pasteur – ne serait-ce que parce qu'il représente plus de 80 % de la flore intestinal de l'Homme. Donnons un exemple de phénomène biologique qui montrera que les gènes codant pour des protéines ne sont pas les seuls objets d'intérêt le long du génome, mais, surtout, qui introduira la démarche statistique de recherche de mots exceptionnels et justifiera l'introduction des modèles markoviens.

La figure 3.1 schématise le fait que, pour se protéger contre les intrusions virales, la cellule dispose d'un complexe protéique, Rec BCD. Celui-ci dégrade tout brin d'ADN en l'attaquant par son extrémité 5' et en "démontant" les uns après les autres les nucléotides qui le composent. Il détruit ainsi les virus (appelés "phages" en bactériologie) qui sont linéaires et présentent une extrémité 5' (phage simple brin), voire deux extrémités 5' (phages double brin). Normalement le génome de la cellule, qui est circulaire, ne présente pas d'extrémité que Rec BCD pourrait attaquer. Néanmoins, il est fréquent que ce génome s'ouvre momentanément ; si un complexe Rec BCD se trouve à proximité de la cassure, il peut entamer un processus de dégradation du génome cellulaire, qui risque d'être rapidement fatal pour la bactérie.

Pour la protéger contre ses propres défenses, existe un phénomène de type auto-immune : le long du chromosome de *E. coli* sont réparties des copies d'un mot, `gctggtgg`, appelé le CHI[1] de *coli*. Lorsque Rec BCD rencontre le CHI, il cesse la dégradation ; les mécanismes de réparation de l'ADN peuvent se mettre en fonction, copiant le texte détruit. Mais, comme celui-ci qui figure, inversé mais non détruit sur le brin opposé il y a possibilité de réparation par copie sur l'autre brin et la cellule survit "souvent" à cet épisode.

Pour rechercher s'il y a des CHI sur d'autres bactéries et mieux comprendre leur fonctionnement, on a pu tenir le raisonnement suivant : «Pour que le motif CHI soit efficace, il est clair qu'il est nécessaire qu'il présente un grand nombre d'occurrences. Recherchons donc le CHI d'une bactérie où il est inconnu parmi les mots – disons pour commencer de 8 lettres, comme chez *coli* – les plus fréquents».

De fait, cette démarche s'est soldée par un échec, dont la raison apparaît rapidement : les 4 lettres `t`, `c`, `a` et `g` ne sont pas présentes en la même proportion 1/4 dans la séquence. Prenons un exemple plus court que *E. coli* où les contrastes seront plus marqués : le génome du virus *HIV1* comporte 3410 fois le nucléotide `a`, pour seulement 1773 fois `c` (et 2370 et 2164 `a` et `g`). On comprend que dans ce virus, les mots les plus fréquents seront tendanciellement ceux composés de `a` et rares en `c`, de même qu'en langue française, on trouvera beaucoup de mots riches en `e` et peu de mots riches en `z` ou `w`!

[1] *Crossover Hotspot Instigator*

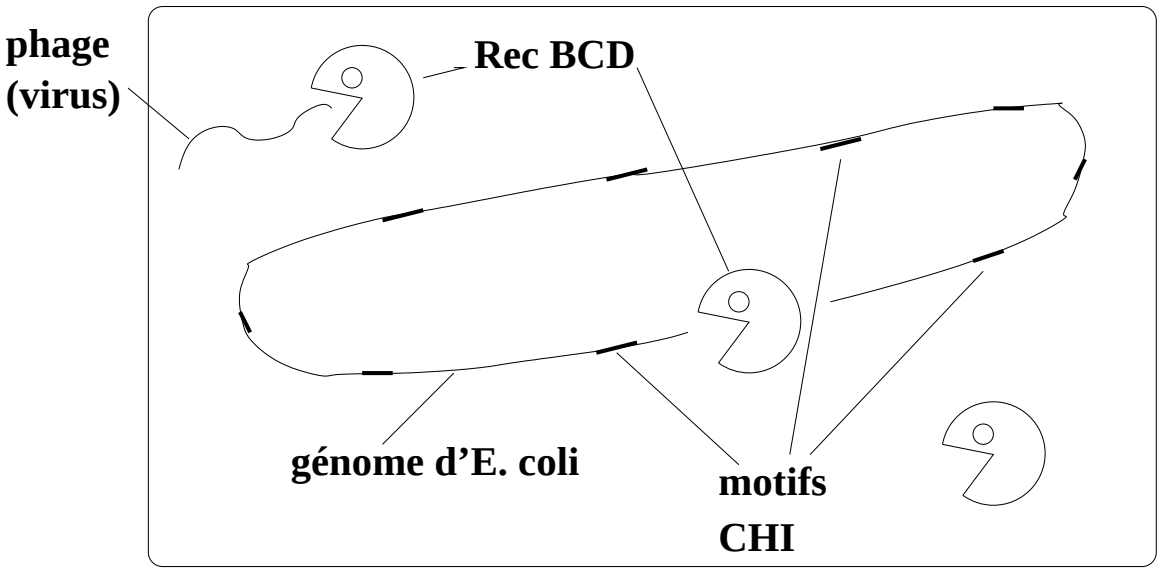


FIG. 3.1 – Entre autres fonctions, le CHI de coli protège le génome de la cellule contre sa dégradation par le complexe Rec BCD, chargé de la défense anti-virale.

Avant de déclarer un mot "exceptionnellement fréquent" (ou "exceptionnellement rare"), il convient de corriger (comment ?) de la richesse de la séquence en chacune des lettres.

Mais ceci ne suffit pas : deux lettres qui se suivent ne sont pas, en général, indépendantes au sens probabiliste. De même qu'en français la lettre `q` appelle le plus souvent la lettre `u` juste derrière, la probabilité d'observer dans un génome donné un `a`, par exemple, n'est pas la même après un `a` ou après un `g`, etc. Les biologistes savent par exemple que, chez tous les organismes, la succession `cg` est évitée (notons que `cg` est un palindrome [voir définition page 58] : s'il figure sur un brin, il figure en mêmes positions sur l'autre). De fait il favorise les mutations lors de la réplication et aura donc tendance à disparaître au cours de l'évolution. Par exemple, le génome de *HIV1* contient 470 occurrences du mot `ct`, 413 fois `cc`, 795 fois `ca` et seulement 95 fois `cg`. Autrement dit, sur 1773 `c`, 5.36% sont suivis par un `g`, contre près de 45 % par un `a`! Si le motif `cg` s'accumule dans une région du génome, on pourra penser que l'organisme en a *vraiment* besoin.

Pour évaluer le caractère exceptionnel d'un mot W, tel le CHI, il est donc indispensable de prendre en compte les fréquences des mots de deux lettres, donc des diverses probabilités $\pi(u, v)$ pour que la lettre v succède à la lettre u. C'est ce qui motive l'introduction des chaînes de Markov : l'exceptionalité dépend des décomptes des mots de deux lettres et ces décomptes spécifient le mode de succession des lettres (on formalisera ce lien dans la suite).

En tout état de cause, il apparaît que mesurer le caractère "normal" ou pas d'une statistique – ici le décompte du nombre d'occurrences d'un mot - ne peut se faire qu'en se plaçant dans un *modèle référant* dans lequel on pourra connaître, au moins partiellement, la loi de cette statistique, à commencer par son espérance, à laquelle on pourra comparer l'observation faite. Les modèles référant que nous utiliserons seront essentiellement des modèles de Markov.

3.2 Modèles d'indépendance

Le premier modèle référant qui vient à l'esprit consiste à supposer que toutes les positions dans les séquences sont indépendantes et qu'aucune lettre de l'alphabet n'est privilégiée par rapport à une autre ; nous nommerons ce modèle M00.

Définition 9 (modèle M00)**.** Le modèle M00 pour la séquence $X = X_1 \dots X_\ell$ sur l'alphabet $\mathcal{A} = \{a_1, \dots, a_k\}$ est défini par les conditions suivantes :

i) les X_i sont indépendants et identiquement distribués,

ii) $\mathbb{P}(X_i = a_j) = 1/k$ pour tout $1 \leqslant i \leqslant \ell$ et tout $1 \leqslant j \leqslant k$.

Par exemple, considérer le modèle M00 pour des séquences d'ADN ($k = 4$) consiste à supposer que les lettres de la séquence sont indépendantes et que chacune d'entre elle a la même probabilité 0.25 de prendre la valeur d'un acide nucléique donné.

Une extension naturelle du modèle M00 consiste à conserver l'hypothèse d'indépendance et d'identiques distributions mais d'autoriser les lettres à avoir des fréquences différentes.

Définition 10 (modèle M0 ou modèle de Bernoulli)**.** Le modèle M0 pour la séquence $X = X_1 \dots X_\ell$ sur l'alphabet $\mathcal{A} = \{a_1, \dots, a_k\}$ de paramètre μ (loi sur $\mathcal{A}$) est défini par les conditions suivantes :

i) les X_i sont indépendants et identiquement distribués,

ii) $\mathbb{P}(X_i = a_j) = \mu(a_j)$ pour tout $1 \leqslant i \leqslant \ell$ et tout $1 \leqslant j \leqslant k$

Sous le modèle M0 de paramètre μ, la probabilité d'observer la séquence $x = x_1 \dots x_\ell$ est donc

$$\mathbb{P}(X_1 = x_1, \dots, X_\ell = x_\ell) = \prod_{i=1}^{\ell} \mu(x_i) = \prod_{u \in \mathcal{A}} \mu(u)^{N_x(u)} \tag{3.1}$$

où $N_x(u)$ est le nombre de fois où la lettre u apparaît dans la séquence x.

Rappelons que si la loi d'une variable[2] X dépend d'un paramètre θ (ici à valeurs dans $\mathbb{R}^k$), on appelle *vraisemblance* de la valeur θ du paramètre au vu d'une observation $X = x$ la *probabilité* d'observer x quand le paramètre est égal à θ :

$$V_x(\theta) = \mathbb{P}_\theta(x) \tag{3.2}$$

L'estimateur du *maximum de vraisemblance* (EMV) est "la" valeur de θ qui maximise (3.2). Notons que, la fonction $y = \log(u)$ étant croissante, l'EMV maximise également la "log-vraisemblance"

$$L_x(\theta) = \log(V_x(\theta)) \tag{3.3}$$

Dans le cas d'une séquence X générée sous M0, la log-vraisemblance s'écrit à partir de (3.1)

$$L_x(\mu) = N_x(\mathtt{a}) \log \mu(\mathtt{a}) + N_x(\mathtt{c}) \log \mu(\mathtt{c}) + N_x(\mathtt{g}) \log \mu(\mathtt{g}) + N_x(\mathtt{t}) \log \mu(\mathtt{t}) \tag{3.4}$$

Chercher l'EMV, c'est maximiser cette quantité *conditionnellement à la contrainte* $\sum_u \mu(u) = 1$. Deux méthodes se présentent :

[2] on suppose ici, pour simplifier, que $X \in E$, où E est un ensemble fini.

1) On déduit de la contrainte $\mu(\mathtt{t}) = 1 - \mu(\mathtt{a}) - \mu(\mathtt{c}) - \mu(\mathtt{g})$, on reporte dans (3.4) avant de dériver par rapport aux trois paramètres encore présents. On obtient, par exemple $\dfrac{\partial L}{\partial \mu(\mathtt{a})} = \dfrac{N_x(\mathtt{a})}{\mu(\mathtt{a})} - \dfrac{N_x(\mathtt{t})}{\mu(\mathtt{t})}$. Annuler les trois dérivées partielle, c'est donc écrire $\dfrac{N_x(\mathtt{a})}{\mu(\mathtt{a})} = \dfrac{N_x(\mathtt{c})}{\mu(\mathtt{c})} = \dfrac{N_x(\mathtt{g})}{\mu(\mathtt{g})} = \dfrac{N_x(\mathtt{t})}{\mu(\mathtt{t})}$. Ces quatre fractions sont donc aussi égales au quotient de la somme de leurs numérateurs (qui vaut ℓ) à la somme de leurs dénominateurs (qui vaut 1). On en déduit l'EMV : $\forall u \in \mathcal{A} \quad \widehat{\mu}(u) = \dfrac{1}{\ell} N_x(u)$.
2) Multiplicateur de Lagrange : on maximise $L_x(\mu) + \lambda \times (\sum_u \mu(u) - 1)$, ce qui conduit, bien sûr, au même résultat.

Ce résultat, qui peut paraître évident (on estime les probabilités des différentes lettres par leurs fréquences), se généralisera au cas markovien. Il permet déjà de comparer deux séquences (test d'homogénéité des compositions en nucléotides), ou, comme nous allons le monter sur un exemple, de tester le caractère exceptionnel d'un mot. On notera, dans cet exemple, le fait que l'on ne tient pas compte du fait que μ a été estimé sur la même séquence sur laquelle on effectue le test, et que l'on utilise diverses approximations assez grossières. Nous présenterons plus loin d'autres méthodes plus exactes.

Exemple 11. On considère le génome complet de *Mycoplasma genitalium* ($\ell = 580\,074$) ; on observe les fréquences suivantes :

$$N(\mathtt{a}) = 200\,543 \quad N(\mathtt{c}) = 91\,524 \quad N(\mathtt{g}) = 92\,312 \quad N(\mathtt{t}) = 195\,695$$

On peut remarquer la grande disparité entre les fréquences des lettres `a-t` et `g-c` (les premières lettres apparaissant deux fois plus que les secondes). Il est bien connu que l'on peut classer les génomes en deux catégories selon qu'il sont riches ou pauvres en `g-c` ; le génome de *Mycoplasma genitalium* est à l'évidence un représentant de la seconde catégorie.

En utilisant ces comptages, on obtient donc l'estimateur suivant :

$$\widehat{\mu}(\mathtt{a}) = 0.3457 \quad \widehat{\mu}(\mathtt{c}) = 0.1578 \quad \widehat{\mu}(\mathtt{g}) = 0.1591 \quad \widehat{\mu}(\mathtt{t}) = 0.3374$$

Exemple 12. On peut par exemple comparer des fréquences en lettres de différentes séquences, voire de différents organismes. Le tableau suivant donne la composition en nucléotides de deux souches de virus HIV1 et HIV2 :

	t	c	a	g	total
HIV1	3411	1773	2370	2164	9 718
HIV2	2123	2132	3506	2598	10 359

La statistique T du χ^2 d'indépendance (qui a ici 3 ddl) prend la valeur 778.6, ce qui est bien sûr extrêmement significatif ; les deux distributions sont donc bien différentes.

3.2.1 Modèle *shuffle*

Notons qu'il existe une autre possibilité assez naturelle pour prendre en compte la fréquence des lettres dans une séquence : le modèle *shuffle*. Il consiste à mélanger (*to shuffle* en anglais) les lettres de la séquence initiale pour obtenir une séquence ayant exactement les mêmes fréquences de lettres. Autrement dit, on considère que la séquence

observée x a été tirée uniformément[3] dans l'ensemble $\mathcal{X}$ des $\ell\,!$ séquences ayant exactement la même composition en lettres que x.

Ce modèle peut être préféré au modèle M0 car il contraint les comptages de lettres à être exactement ceux observés dans la réalité, $N_x(u)$, alors que M0 impose seulement que *l'espérance* de ces comptages valent $N_x(u)$; il a connu un certain succès car il est très facile dans un démarche de simulation de tirer une séquence aléatoirement dans $\mathcal{X}$ – ce qui ne sera plus vrai dans un cadre markovien. Nous ne la privilégierons pas dans cet ouvrage (voir néanmoins 5.5.5), d'abord à cause de l'impossibilité d'estimer le modèle shuffle sur une séquence pour l'utiliser sur une séquence de longueur différente ; en outre, il existe beaucoup plus d'outils et résultats pour le modèle M0 que pour son pendant *shuffle*.

3.3 Rappels sur les chaînes de Markov

Avec les modèles M00 et M0, on suppose que les tirages des lettres sont indépendants d'une position à une autre. Une façon très naturelle de lever cette hypothèse consiste à introduire une dépendance markovienne d'ordre m ($m \geqslant 1$) : la probabilité d'observer une lettre à une position donnée dans la séquence est conditionnée par les lettres observées aux m positions précédentes.

Nous présenterons rapidement dans ce paragraphe le cadre mathématique des chaînes de Markov (CM), que nous utiliserons dans la suite pour modéliser des séquences d'ADN ou de protéines (voir 3.4).

3.3.1 Introduction

On appelle *processus* une famille, pourvue d'une loi conjointe, de variables aléatoires (X_t) indicée par une variable $t \in \mathcal{T}$. Le théorème de Kolmogorov dit que cette loi est déterminée dès que l'on en connaît toutes les marginales finies : si pour tout ensemble fini d'indices $T = \{t_1, t_2, \ldots, t_n\}$, on se donne la loi de $X_T = \{X_{t_j},\ j \in T\}$ [4], alors la loi de tout le processus, $X_{\mathcal{T}}$ est définie de manière unique.

Dans le cas où T est un ensemble d'entiers, on dit que le processus $\{X_k\}$ est une *chaîne* : ce sera par exemple le cas si k est la position sur une séquence biologique, ou le numéro de génération dans une évolution temporelle. Parfois l'indice sera un réel quelconque (par exemple lors d'une approximation continue d'une chaîne) ou un sommet dans un arbre, par exemple phylogénétique.

Dans le cas où *l'espace d'état*, c'est à dire l'ensemble dans lequel X_k prend ses valeurs est fini, on le qualifiera d'*alphabet*, on le notera généralement $\mathcal{A}$ et on notera k le nombre de ses éléments. On aura par exemple $\mathcal{A} = \{\mathtt{t}, \mathtt{c}, \mathtt{a}, \mathtt{g}\}$ et $k = 4$.

Définition 13. On dit que la chaîne $X_t,\ t \in \mathcal{T}$, est *stationnaire* si,,

$$\forall n, \forall \{t_1, \ldots, t_n\}, \forall s \quad \{X_{t_1+s}, \ldots, X_{t_n+s}\} \text{ a même loi que } \{X_{t_1}, \ldots, X_{t_n}\}$$

dès que tous les indices concernés appartiennent à $\mathcal{T}$.

[3] c'est à dire que chacune des séquences de $\mathcal{X}$ a la même probabilité, $1/\ell\,!$ d'être tirée. Notons que $\mathcal{X}$ comporte de fait 'seulement' $\ell\,!/(N_x(\mathtt{t})\,!\,N_x(\mathtt{c})\,!\,N_x(\mathtt{a})\,!\,N_x(\mathtt{g})\,!)$ éléments différents.

[4] ce qui implique une notion de cohérence : si T_1 et T_2 sont deux parties finies d'indices, les lois associées doivent avoir même marginale sur $T_1 \cap T_2$; on ne pourrait pas, sinon, définir la loi sur $T_1 \cup T2$.

Définition 14. On dit que la chaîne X_t, où t parcourt un intervalle $\mathcal{T} = (a, b)$ est markovienne, ou que c'est une *chaîne de Markov* si

$$\forall t \in \mathcal{T}^+, \forall x \in \mathcal{E}, \quad \mathbb{P}(X_t = x \mid \{X_s\,,\, s < t\}) = \mathbb{P}(X_t = x \mid X_{t-1})$$

où $\mathcal{T}^+$ désigne $\mathcal{T}$ privé de son premier élément a si celui-ci est fini et désigne $\mathcal{T}$ sinon.

On posera alors

$$\pi_t(x, y) = \mathbb{P}(X_t = y \mid X_{t-1} = x) \tag{3.5}$$

La matrice π_t de terme général $\pi_t(x, y)$ sera appelée la matrice de transition de la chaîne de Markov à la position t. C'est une *matrice stochastique* (i.e. à termes positifs tels que la somme de chaque ligne est égale à 1) .

On a alors

$$\mathbb{P}(X_{t+1} = z \mid X_{t-1} = x) = \sum_y \pi_t(x, y)\, \pi_{t+1}(y, z)$$

ce qui est le terme (x, z) du produit de matrices $\pi_t\, \pi_{t+1}$. Cette formule se généralise immédiatement pour deux indices t et $t + h$ quel que soit $h > 0$:

$$\mathbb{P}(X_{t+h} = v \,|\, X_t = u) = [\pi_{t+1} \times \pi_{t+2} \times \ldots \times \pi_{t+h}](u, v) \tag{3.6}$$

Cette formule montre *qu'il n'y a pas indépendance* entre X_t et X_{t+h}, même lorsque les positions t et $t+h$ sont éloignées (h grand) : X_t et X_{t+h} sont indépendants *conditionnellement* à la valeur de X en une position intermédiaire, disons X_{t+j}, avec $0 < j < h$.

La probabilité d'observer la séquence $\{X_t\}, t = 0, \ldots, \ell$ est égale à

$$\mu_0(X_0) \prod_{t=1}^{\ell} \pi_t(X_{t-1}, X_t) \tag{3.7}$$

où μ_0 est la loi de X_0. Ce pourra être la loi stationnaire définie ci dessous ; dans d'autres cas – quand on connaît la valeur, disons u_0 de X_0 – ce sera une "loi de Dirac" :

$$\mu_0(u_0) = 1 \qquad , \qquad \forall v \neq u_0\,,\, \mu_0(v) = 0. \tag{3.8}$$

Si $\mathcal{T}$ est un segment fini $[t_0, t_\ell]$ ou une demi droite discrète $[t_0, +\infty[$, la loi du processus sera définie si l'on se donne

– la loi initiale, c'est à dire la loi de X_{t_0} et
– la matrice de transition π_t pour tout $t > t_0$ de $\mathcal{T}$.

Notons[5] μ_t la loi de X_t

$$\forall x \in \mathcal{A}, \quad \mu_t(x) = \mathbb{P}(X_t = x)$$

La suite de ces lois vérifie la relation de récurrence $\mu_k = \mu_{t-1}\, \pi_t$.

[5]On sera amené à représenter un élément – ou un sous-ensemble – de $\mathcal{A}$ par un vecteur (= k lignes $\times$ 1 colonne), contenant des 0 et des 1 ; il est alors naturel de représenter une mesure (de probabilité) μ sur $\mathcal{A}$ comme un vecteur *ligne* de $\mathbb{R}^k$, assimilant $\mu(x)$ au produit de la ligne μ et du vecteur x. Ceci permet alors l'écrire simple $\mu_t = \mu_{t-1} \times \pi$.

3.3.2 Loi de X_t, loi stationnaire

Lorsque la matrice π_t ne dépend pas de t (on la note alors π), on dit que la chaîne de Markov est *homogène*.

La formule (3.6) s'écrit alors

$$\mathbb{P}(X_{t+h} = v \mid X_t = u) = \pi^h(u, v) \tag{3.9}$$

Définition 15. On dit que la loi de probabilité μ sur l'alphabet $\mathcal{A}$ est une *loi stationnaire* pour la chaîne de Markov homogène $\{X_t\}$ si $\mu_t = \mu \Rightarrow \mu_{t+1} = \mu$. On a alors

$$\mu = \mu\ \pi$$

Le caractère markovien d'une chaîne implique que si ses marginales d'ordre 1 (i.e. les lois de chaque X_t) sont égales à la loi stationnaire μ, alors ses marginales d'ordre quelconque h (lois de $X_t, X_{t+1}, \ldots, X_{t+h}$) sont aussi indépendantes de t. On parlera de *régime stationnaire* :

Proposition 16. Si $W = w_1 \ldots w_h$ est un mot de longueur h fixé, en régime stationnaire, quel que soit t, $\mathbb{P}(X_t = w_1, \ldots, X_{t+h-1} = w_h) = \mu(W)$ où

$$\mu(W) \stackrel{\Delta}{=} \mu(w_1) \times \pi(w_1, w_2) \times \pi(w_2, w_3) \ldots \times \pi(w_{h-1}, w_h)$$

Le nombre d'occurrences de W, $N(W) = \sum_{k=1}^{n-h+1} \mathbb{I}\{X_k = w_1, \ldots, X_{k+h-1} = w_h\}$, aura pour espérance $E(N(W) = (n - h + 1)\ \mu(W)$.

L'existence et de l'unicité de μ est assurée par le fait que π est stochastique (c'est le premier point du théorème de Perron-Frobénius, ci-dessous) et deux questions se posent : y a-t-il unicité et μ_k converge-t-elle vers μ ? De fait deux "pathologies" empêchent de répondre "oui" en général :

Définition 17. 1) On dit que π est réductible si l'espace d'état $\mathcal{A}$ peut être partitionné en deux sous ensembles non vides : $\mathcal{A} = \mathcal{B} \cup \mathcal{C}$ tels que, si $X_k \in \mathcal{C}$, quel que soit $d > 0$, on a encore $X_{k+d} \in \mathcal{C}$ (partant d'un point de $\mathcal{C}$, la chaîne n'atteint jamais $\mathcal{B}$).

2) On dit que π est périodique et de période $p > 1$ si $\forall x \in \mathcal{A}$, $\pi^d(x, x) \neq 0$ implique que d est divisible par p. (Partant d'un point x, il n'est possible d'y revenir qu'en p ou $2p$, ou $3p$, ... pas.)

Une chaîne non réductible est dite *irréductible*[6] ; une chaîne non périodique est dite *apériodique*. Les matrices de transition suivantes sur $\{\mathtt{t}, \mathtt{c}, \mathtt{a}, \mathtt{g}\}$ définissent des chaînes de Markov réductibles :

$$\pi_1 = \begin{pmatrix} 0.5 & 0.5 & 0 & 0 \\ 0.3 & 0.7 & 0 & 0 \\ 0 & 0 & 0.2 & 0.8 \\ 0 & 0 & 0.6 & 0.4 \end{pmatrix} \quad \text{et} \quad \pi_2 = \begin{pmatrix} 0.5 & 0.2 & 0.2 & 0.1 \\ 0.2 & 0.3 & 0.1 & 0.4 \\ 0 & 0 & 0.6 & 0.4 \\ 0 & 0 & 0 & 0.1 \end{pmatrix}$$

Pour π_1, la chaîne ne peut quitter ni $\mathcal{B} = \{\mathtt{t}, \mathtt{c}\}$, ni $\mathcal{C} = \{\mathtt{a}, \mathtt{g}\}$; pour π_2, la chaîne ne peut quitter ni $\mathcal{B} = \{\mathtt{a}, \mathtt{g}\}$, ni $\mathcal{B}' = \{\mathtt{g}\}$.

Les matrices de transition suivantes définissent des chaînes de Markov périodiques :

[6] $\forall u, v \in \mathcal{A}\ \exists d \geqslant 0$ tel que $\pi^d(u, v) > 0$

$$\pi_3 = \begin{pmatrix} 0 & 0 & 0.5 & 0.5 \\ 0 & 0 & 0.7 & 0.3 \\ 0.2 & 0.8 & 0 & 0 \\ 0.1 & 0.9 & 0 & 0 \end{pmatrix} \quad \text{et} \quad \pi_4 = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

La chaîne associée à π_3 est de période $p = 2$, X_k étant alternativement dans $\{\mathtt{t}, \mathtt{c}\}$ et dans $\{\mathtt{a}, \mathtt{g}\}$; la chaîne associée à π_4 est de période $p = 4$.

Une chaîne réductible peut avoir plusieurs lois stationnaires : pour π_1, on aura une loi stationnaire μ_y sur $\{\mathtt{t}, \mathtt{c}\}$, une autre, μ_r, sur $\{\mathtt{a}, \mathtt{g}\}$) (et donc toutes les lois $\alpha\,\mu_y + (1-\alpha)\,\mu_r$, pour $0 \leq \alpha \leq 1$ seront stationnaires) ; pour π_2, le singleton $\{\mathtt{g}\}$ est *absorbant*, la seule loi stationnaire est telle que $\mu(\mathtt{g}) = 1$.

Nous admettrons le théorème suivant :

Théorème 18. (Perron-Frobénius)

1) Une matrice stochastique π admet la valeur propre $\lambda_1 = 1$; toutes ses autres valeurs propres ont un module inférieur ou égal à 1.
2) Si la chaîne de Markov X_k est irréductible (resp. *primitive*[7]) et apériodique, alors
   - $\lambda_1 = 1$ est valeur propre simple ; toutes ses autres valeurs propres ont un module inférieur (resp. strictement) à 1,
   - la chaîne possède une unique loi stationnaire μ,
   - la loi μ_k converge vers μ à la vitesse ρ_2^k, où ρ_2 désigne le plus grand des modules des valeurs propres, une fois $\lambda_1 = 1$ enlevée.

Notons que les lois de probabilité μ, μ_k considérées ici sont des lois sur un espace de dimension finie, $\mathbb{R}^s$. Elles appartiennent à son dual qui est de même dimension. Sur $\mathbb{R}^s$, toutes les distances étant équivalentes, il n'est pas nécessaire de préciser pour quelle métrique $\|.\|$ on a

$$\|\mu_k - \mu\| \approx \rho_2^k$$

On pourra par exemple utiliser la distance en variation totale définie par

$$\|\mu - \nu\| = \frac{1}{2} \sum_{u \in \mathcal{A}} |\mu(u) - \nu(u)|$$

Définition 19. On dit que la chaîne de Markov homogène de transition π et de loi stationnaire μ est *réversible* si

$$\forall a, b \qquad \mu(a)\ \pi(a,b) = \mu(b)\ \pi(b,a)$$

Si X_t est une CM réversible en régime stationnaire, à l'observation $X = X_1 \dots X_\ell$, on peut associer $Y = Y_1 \dots Y_\ell$ défini par : $\forall t\ Y_t = X_{\ell - t}$. La réversibilité implique que X et Y ont même loi. Il suffit d'écrire

$$\begin{aligned} \mathbb{P}(X = x_1 \dots x_\ell) &= \mu(x_1) \times \pi(x_1, x_2) \times \pi(x_2, x_3) \times \cdots \times \pi(x_{\ell-1}, x_\ell) \\ &= \pi(x_2, x_1) \times \mu(x_2) \times \pi(x_2, x_3) \times \cdots \times \pi(x_{\ell-1}, x_\ell) \\ &= \pi(x_2, x_1) \times \pi(x_3, x_2) \times \mu(x_3) \times \cdots \times \pi(x_{\ell-1}, x_\ell) \\ &= \dots \\ &= \pi(x_2, x_1) \times \pi(x_3, x_2) \times \cdots \times \pi(x_{\ell-1}, x_\ell) \times \mu(x_\ell) \end{aligned}$$

[7] $\exists d \geqslant 0$ tel que $\forall u, v \in \mathcal{A}\ \pi^d(u, v) > 0$

qui est exactement $\mathbb{P}(Y = x_\ell \dots x_1)$ lorsque Y est une séquence stationnaire tirée selon la CM de même transition π.

Autrement dit, pour une CM réversible, il est impossible de distinguer "le sens du temps", c'est à dire des données ordonnées selon un temps croissant ou selon un temps décroissant.

3.3.3 Chaînes de Markov d'ordre supérieur à un

Définition 20. On dit que la chaîne (X_t) est une chaîne de Markov d'ordre m (ou de mémoire m) si

$$\mathbb{P}(X_t = b \mid \{X_s,\ s < t\}) = \mathbb{P}(X_t = b \mid X_{t-m}, X_{t-m+1}, \dots, X_{t-1})$$

On introduit donc la (matrice de) transition

$$\pi(a_1 \dots a_m, b) = \mathbb{P}(X_t = b \mid X_{t-m} = a_1, \dots, X_{t-1} = a_m)$$

où $a_1, \dots, a_m$ et b parcourent $\mathcal{A}$. Le mot $a_1 \dots a_m$ nécessaire pour savoir selon quelle loi conditionnelle est choisie la prochaine lettre de la séquence est appelé *prédicteur*.

Si k est le nombre de lettres de l'alphabet $\mathcal{A}$, ce modèle utilise k^m prédicteurs. π compte donc k^m lignes et k colonnes. La somme des termes de chaque ligne valant 1, ce modèle dépend donc de $(k-1) \times k^m$ paramètres indépendants.

La probabilité d'une séquence $X = X_1 \dots X_\ell$ sera obtenue comme le produit de la probabilité $\mu_0(X_1 \dots X_m)$ de ses m premiers termes par le produit de $\ell - m$ transitions $\pi(X_{k-m} \dots X_{k-1}, X_k)$.

On appellera Mm le modèle des chaînes de Markov d'ordre m, le modèle M1 étant donc celui des chaînes de Markov vues précédemment. Ceci est également cohérent avec la notation M0 pour le modèle "de Bernoulli" de la section 3.2

Remarque 21. Un modèle markovien d'ordre $m > 1$ peut être ramené à un modèle markovien d'ordre 1 de la façon suivante : on introduit l'alphabet $\mathcal{A}^m = \{a_1 \dots a_m, \text{avec } a_s \in \mathcal{A}\}$. Dans la séquence $\{X_t\}$ dont les éléments sont dans $\mathcal{A}$ pour $t = 1, \dots, \ell$, notons $\widetilde{x}_t$ le mot de longueur m commençant en position t ; t varie de 1 à $\ell - m + 1$; deux mots successifs vérifient la condition $R(\widetilde{x}_t, \widetilde{x}_{t-1})$, où, si w et w' sont des mots de m lettres,

$$\text{Condition } R(w, w') : \quad \begin{array}{l} \text{les } m-1 \text{ premières lettres de } w' \text{ sont égales} \\ \text{aux } m-1 \text{ dernières lettres de } w. \end{array}$$

Il y a bien sûr bijection entre les séquences de $X_t \in \mathcal{A}$ de longueur ℓ et les séquences de $\widetilde{x}_t \in \mathcal{A}^m$ de longueur $\ell - m + 1$ et vérifiant la condition $R(\widetilde{x}_{t-1}, \widetilde{x}_t)$ pour tout t. Si $\{X_t\}$ est dans $\mathcal{M}m$, la séquence $\widetilde{x}_t$ est une chaîne de Markov d'ordre 1, de transition

$$\widetilde{\pi}(x_1 x_2 \dots x_m, x_2 \dots x_m x_{m+1}) = \pi(x_1 x_2 \dots x_m, x_{m+1}) \tag{3.10}$$

et $\widetilde{\pi}(w, w') = 0$ si $R(w, w')$ n'est pas vérifiée.

Exemple 22. pour $m = 2$, on associe à la séquence $X =$ `tccgattgcat` la séquence

$$\widetilde{X} = \texttt{tc cc cg ga at tt tg gc ca at}.$$

La transition entre $\texttt{tc} \in \mathcal{A}^2$ et `cc` se fait avec la probabilité $\widetilde{\pi}(\texttt{tc}, \texttt{cc}) = \pi(\texttt{tc}, \texttt{c})$.

Il convient de vérifier que, si tous les termes de π sont non nuls, la chaîne de transition $\widetilde{\pi}$ est encore irréductible et apériodique, de sorte par exemple que $\widetilde{X}$ et donc X convergent vers une loi stationnaire (bien que seul un terme sur k de $\widetilde{\pi}$ soit non nul !).

aa	84538	aaa	34109	caa	16551	gaa	11320	gaa	11320
ac	30409	aac	16099	cac	5124	gac	2507	gac	2507
ag	33703	aag	14040	cag	6282	gag	3956	gag	3956
at	51893	aat	20290	cat	8563	gat	11024	gat	11024
ca	36520	aca	9987	cca	7427	gca	6876	gca	6876
cc	16593	acc	7006	ccc	3151	gcc	1718	gcc	1718
cg	5644	acg	1961	ccg	911	gcg	1448	gcg	1448
ct	32766	act	11454	cct	5104	gct	7334	gct	7334
ga	28807	aga	9385	cga	1447	gga	4826	gga	4826
gc	17376	agc	7415	cgc	1413	ggc	1858	ggc	1858
gg	16462	agg	5169	cgg	911	ggg	3312	ggg	3312
gt	29667	agt	11734	cgt	1873	ggt	6466	ggt	6466
ta	50678	ata	11814	cta	9556	gta	6410	gta	6410
tc	27146	atc	10695	ctc	3534	gtc	2514	gtc	2514
tg	36503	atg	9020	ctg	6114	gtg	5587	gtg	5587
tt	81368	att	20364	ctt	13562	gtt	15156	gtt	15156

FIG. 3.2 – Fréquences des mots de longueurs 2 et 3 dans le génome complet de *Mycoplasma genitalium* de longueur $\ell = 580\,074$.

3.4 Application aux séquences biologiques

3.4.1 Estimation du modèle, loi à portée d

Le modèle Mm pour la séquence $X = X_1 \dots X_\ell$ sur l'alphabet $\mathcal{A} = \{a_1, \dots, a_k\}$ de loi initial μ_0 (loi sur $\mathcal{A}^m$) et de paramètre $\pi \in \mathbb{R}^{m+1}$ est donc défini par [8] :

i) $\mathbb{P}(X_1 = x_1, \dots, X_m = x_m) = \mu_0(x_1 \dots x_m) = \mu_0(x_1^m)$

ii) $\mathbb{P}(X_i = x_i \mid X_1 = x_1, \dots, X_{i-1} = x_{i-1}) = \pi(x_{i-m} \dots x_{i-1}, x_i) = \pi(x_{i-m}^i)$ pour tout $i \geqslant m+1$.

Dans ce modèle, la probabilité d'observer la séquence $x = x_1 \dots x_\ell$ est donnée par :

$$\mathbb{P}(X_1 = x_1, \dots, X_\ell = x_\ell) = \mu_0(x_1 \dots x_m) \prod_{w \in \mathcal{A}^{m+1}} \pi(w)^{N_x(w)} \tag{3.11}$$

Considéré comme fonction des paramètres, ceci est la vraisemblance du modèle au vu des observations $x = x_1^\ell$. La maximisation de cette vraisemblance, analogue à ce qui a été fait sous M0 est laissée au lecteur. N'ayant pas supposé la stationnarité (i.e. que μ_0 est la loi stationnaire de π), l'EMV de μ_0 est la Dirac (voir 3.8) : $\mu_0(w) = \mathbb{I}\{w = x_1 \dots x_m\}$; pour les transitions, on trouve les estimateurs suivants :

$$\forall a = a_1 \dots a_m \in \mathcal{A}^m \quad \forall b \in \mathcal{A} \qquad \widehat{\pi}(a, b) = \frac{N_x(ab)}{N_x(a\,\cdot)}$$

où $N_x(a\,\cdot) = \sum_{c \in \mathcal{A}} N_x(ac)$.

Exemple 23. Tout comme dans l'exemple 11, on considère le génome complet de *Mycoplasma genitalium*. Dans le cas $m = 2$ on utilise les fréquences de la figure 3.2 pour

[8] si $i \leq j$, on notera toujours x_i^j la séquence $x_i x_{i+1} \dots x_j$.

obtenir :

$$\widehat{\pi}(\mathtt{ga},\mathtt{a}) = \frac{N(\mathtt{gaa})}{N(\mathtt{ga}\,\cdot)} = \frac{11320}{11320+2507+3956+11024} = 0.3929600$$

$$\text{ou encore} \quad \widehat{\pi}(\mathtt{g},\mathtt{ac}) = \frac{2507}{28807} = 0.0870275$$

Lorsqu'on se trouve dans le modèle M1 ($m = 1$), le paramètre π est la matrice de transition de la chaîne de Markov considérée et l'écriture matricielle permet d'obtenir $\mathbb{P}(X_{d+1} = x_{d+1}|X_1 = x_1)$ grâce à la formule 3.9.

Cette propriété se perd – au moins sous sa forme immédiate – dans les modèles Mm avec $m \geqslant 2$ puisque π n'a alors même pas la forme d'une matrice carrée. Fort heureusement, la remarque 21 permet de considérer la chaîne de Markov d'ordre m sur l'alphabet $\mathcal{A}$ comme une chaîne de Markov d'ordre 1 sur l'alphabet $\mathcal{A}^m$ dont la matrice de transition $\widetilde{\pi}$ est définie par (3.10). On peut alors généraliser la propriété (3.9) au le cas Mm : pour tout $d \geqslant 1$,

$$\mathbb{P}(X_{d+m} = x_{d+m}|X_1 = x_1, \ldots, X_m = x_m) = \widetilde{\pi}^d(x_1 \ldots x_m, x_{d+1} \ldots x_{d+m})$$

A noter que si la matrice $\widetilde{\pi}$ est creuse par construction, ce n'est pas le cas de ses puissances. Par conséquent, tout calcul utilisant les puissances de $\widetilde{\pi}$ devra, pour être efficace, éviter de faire intervenir explicitement $\widetilde{\pi}^d$. Par exemple, le calcul de $\widetilde{\pi}^d \times v$ (ou v est un vecteur colonne) peut se faire récursivement en remplaçant d fois v par $\widetilde{\pi} \times v$. Chacun de ces produits utilisant pleinement la structure creuse de la matrice de transition on obtient au finale une complexité mémoire en $O(k^{m+1})$ et une complexité temps en $O(d \times k^{m+1})$ (où k est la taille de l'alphabet).

Ces complexités sont à comparer avec celles de l'approches "naïve" consistant à calculer explicitement $\widetilde{\pi}^d$ puis à en faire le produit avec v : mémoire en $O(k^{2m})$ et temps en $O(d \times k^{3m})$. Une astuce permet de réduire cette complexité. Dans le cas où d est une puissance de 2, soit $d = 2^q$, le calcul de $\widetilde{\pi}^4 = \widetilde{\pi}^2 \times \widetilde{\pi}^2$, $\widetilde{\pi}^8 = \widetilde{\pi}^4 \times \widetilde{\pi}^4$, etc, permet le calcul de $\widetilde{\pi}^q$ par $\log_2 d$ produits de matrices. La généralisation au cas où d n'est plus une puissance de 2 s'obtient en écrivant d en base 2 : par exemple, de $45 = 32 + 8 + 4 + 1$, on déduit $\widetilde{\pi}^{45} = \widetilde{\pi}^{32} \times \widetilde{\pi}^8 \times \widetilde{\pi}^4 \times \widetilde{\pi}$, ce qui conduit à un nombre de produits de matrices à peine supérieur à $\log_2 d$.

En pratique, l'usage de cette astuce est restreint, car elle remplace un calcul de complexité globale en $O(d \times k^{m+1})$ par un calcul de complexité globale $O(\log_2 d \times k^{3m})$, en général plus élevée – par ailleurs, elle requiert une place mémoire en $O(k^{2m})$ contre $O(k^m)$.

En outre, nous allons voir au paragraphe suivant que les puissances d'une matrice de transition convergent "rapidement" vers une limite (liée à la loi stationnaire) ce qui conduira, comme nous le discuterons, à remplacer "assez vite" $\widetilde{\pi}^d$ par cette limite.

Des considérations de ce types seront au coeur de pratiquement tous les calculs que nous serons amenés à considérer et nous reviendrons donc largement sur ce point dans la suite.

3.4.2 Loi stationnaire

Supposons toujours que l'on se trouve dans le cas d'un modèle Mm, que l'on représentera comme chaîne d'ordre 1 sur $\mathcal{A}^m$ de loi initiale $\widetilde{\mu}_0$ et de matrice de transition $\widetilde{\pi}$; on la supposera irréductible et diagonalisable sur $\mathbb{C}$ (ces hypothèses ne sont pas restrictives en pratique car toutes les matrices de transitions estimées les vérifient).

Grâce au théorème de Perron-Frobénius, on a alors existence et unicité de la loi stationnaire $\widetilde{\mu}$ associée au modèle :

$$\widetilde{\mu} \times \widetilde{\pi} = \widetilde{\mu}$$

L'EMV $\widehat{\pi}$ des transitions sera encore donné par l'équation (3.11). La loi stationnaire μ n'est plus ici un paramètre libre, c'est une fonction de π ; son EMV sera donc la loi stationnaire de $\hat{\pi}$.

De plus, si on note $\nu < 1$ le module de la seconde plus grande valeur propre en module de $\widetilde{\pi}$ on a alors :

Proposition 24. Pour tout $a, b \in \mathcal{A}^m$ et pour tout $d \geqslant 0$ on a

$$\widetilde{\pi}^d(a, b) = \widetilde{\mu}(b) + O(\nu^d)$$

Démonstration. Comme $\widetilde{\pi}$ est diagonalisable, il existe une base de vecteurs propres. Grâce à l'irréductibilité de $\widetilde{\pi}$ et au théorème de Perron-Frobénius, le vecteur propre $\widetilde{\mu}$ est associé à la valeur propre 1, qui domine strictement en module les autres valeurs propres, dont la deuxième en module : ν. Le résultat est alors trivial. □

Cette propriété sera très souvent au coeur des calculs car il permet de remplacer les puissances de $\widetilde{\pi}^d$ pour d assez grand par la loi stationnaire tout en contrôlant l'erreur ainsi commise.

organisme	1	2	3	4	5	6
A. fulgidus	0.16 (21)	0.27 (29)	0.51 (55)	0.53 (59)	0.65 (86)	0.72 (113)
A. pernix	0.08 (15)	0.27 (29)	0.45 (47)	0.55 (62)	0.62 (78)	0.71 (108)
M. genitalium	0.16 (21)	0.28 (29)	0.40 (41)	0.47 (49)	0.54 (60)	0.65 (86)
B. subtilis	0.10 (16)	0.28 (29)	0.43 (44)	0.52 (57)	0.52 (57)	0.49 (52)
E. coli K12	0.09 (16)	0.25 (27)	0.48 (51)	0.50 (54)	0.55 (62)	0.64 (83)
L. lactis	0.12 (18)	0.16 (21)	0.39 (40)	0.42 (43)	0.56 (64)	0.67 (92)
C. elegans C1	0.15 (20)	0.30 (31)	0.38 (39)	0.42 (43)	0.49 (52)	0.57 (66)
S. cerevisiae C1	0.07 (14)	0.23 (26)	0.44 (45)	0.46 (48)	0.51 (55)	0.64 (83)

TAB. 3.1 – Valeurs du module ν de la seconde plus grande valeur propre en module pour différent modèles Mm (m variant de 1 à 6) et pour différents organismes. On donne également entre parenthèses le rang d à partir duquel l'erreur faite en remplaçant $\widetilde{\pi}^d$ par μ est de l'ordre de grandeur de 10^{-16}.

On peut voir sur la table (3.1) la valeur de ν pour quelques organismes et quelques choix d'ordre m. De manière empirique on peut déterminer un encadrement grossier pour ν lorsque l'on est confronté à des modèles Mm estimés sur des séquences d'ADN :

$$\frac{m}{10} \leqslant \nu \leqslant \frac{m+1}{10}$$

La même table donne également les rangs à partir desquels l'erreur faite en remplaçant les puissances de la matrice de transition par la loi stationnaire est négligeable. C'est aussi le nombre de pas nécessaires pour converger vers la loi stationnaire quelle que soit la distribution initiale considérée. Même dans les cas les moins favorables, ce rang reste de l'ordre de 100. Ainsi, lorsque l'on considère des séquences d'ADN dont la longueur ℓ est

grande devant 100 (ce qui est le plus souvent largement le cas), considérer que la chaîne de Markov est en régime stationnaire (c'est à dire que $\mu_k = \mu$), c'est faire une erreur négligeable. Ainsi, même dans le cas où l'on opte pour une initialisation déterministe de la chaîne de Markov en prenant pour μ_0 une loi concentrant toute la masse en un état particulier (souvent les m première lettres d'une séquence observée), on pourra supposer la stationnarité de la chaînes de Markov.

3.5 Choix d'un modèle

Avec les modèles M00 et Mm pour $m \geqslant 0$, on dispose donc d'un choix varié de modèles référents. Comment choisir un modèle plutôt qu'un autre ? Une façon de répondre à cette question est de l'approcher sous l'angle classique de sélection de modèle.

Les modèles sont emboîtés les uns dans les autres : M00 est un modèle M0 dont la loi μ est uniforme ; M0 peut également s'écrire comme un modèle M1 (de transition π) : si la loi de X_t est indépendante des X_s pour $s < t$, on a bien sûr indépendance de X_t et de X_s pour $s < t-1$, conditionnellement à X_{t-1} : la loi de X_t conditionnellement à $X_{t-1} = u$ sera la même quel que soit u, la matrice de transition π aura toutes ses lignes identiques : $\pi(u, v) = \mu(v)$.

Le même raisonnement peut être tenu aux ordres supérieurs. On a donc

$$\text{M00} \subset \text{M0} \subset \text{M1} \subset \text{M2} \subset \dots$$

Il est alors naturel de penser au test de rapport de vraisemblances pour départager deux de ces modèles[9].

Exemple 25. On considère le génome complet $x = x_1 \dots x_\ell$ du virus HIV1 (avec $\ell = 9\,718$ et $x_1 = \texttt{t}$). Notons L_{00} la log-vraisemblance du modèle M00 (aucun paramètre), L_m celle du modèle stationnaire Mm estimé sur la séquence (3×4^m paramètres libres). En arrondissant à l'entier le plus proche, on trouve $L_{00} = -13473$, $L_0 = -13182$, $L_1 = -12889$, $L_2 = -12796$, $L_3 = -12657$, $L_4 = -12287$ et $L_5 = -10945$. On peut donc construire les 6 tests de rapport de vraisemblances suivants :

test	L_{00} *vs* L_0	L_0 *vs* L_1	L_1 *vs* L_2	L_2 *vs* L_3	L_3 *vs* L_4	L_4 *vs* L_5
$2 \times D$	582	586	186	278	560	2684
ddl	3	9	36	144	576	2304
dds	10^{-125}	10^{-120}	10^{-22}	10^{-10}	0.68	10^{-7}

où $2 \times D$ est deux fois la différence des log-vraisemblances, "ddl" le nombre de degrés de liberté du chi-2 (la différence des nombres de paramètres) et "dds" est le degré de significativité. Par exemple le test L_2 *vs* L_1 conduit à écrire $2D = 2(L_2 - L_1) = 186$, ddl $= 48 - 12 = 36$ et dds $= \mathbb{P}(\chi^2(36) \geqslant 186) = 10^{-22}$.

On peut également penser à se tourner vers des critères de vraisemblance pénalisée comme les critères AIC (Akaïke Information Criterion) ou BIC (Bayesian Information Criterion) qui sont définis par

$$AIC = -2\,\text{log-vraisemblance} + 2K \quad \text{et} \quad BIC = -\text{log-vraisemblance} + K\log(\ell)$$

[9] Si $\mathcal{K}$ est modèle dépendant de k paramètres réels et si $\mathcal{H}$ est un sous-modèle de $\mathcal{K}$ dépendant de h paramètres, sous des hypothèses de régularité [p.ex. $\mathcal{H}$ est une variété deux fois dérivable dans $\mathcal{K}$, la vraisemblance est deux fois dérivable], la différence D du maximum $L(K)$ de la log-vraisemblance sur K et du maximum $L(H)$ de la vraisemblance sur H vérifie asymptotiquement $2 \times D \sim \chi^2(k-h)$.

où K est le nombre de paramètres libres du modèle. Le modèle minimisant ces critères sera le modèle choisi.

Exemple 26. Prenons les génomes complets du virus HIV1 ($\ell = 9\,718$) et de *Escherichia coli* K12 ($\ell = 4\,639\,221$) et calculons les valeurs de l'AIC ($\times 10^3$) pour différents modèles :

modèle	M00	M0	M1	M2	M3	M4	M5	M6	M7
HIV	26.95	26.37	25.80	**25.68**	25.70	26.10	28.03	40.00	106.00
E. coli	12863	12861	12743	12626	12546	12497	12456	**12435**	12443

et on peut faire la même chose pour le BIC ($\times 10^3$) :

modèle	M00	M0	M1	M2	M3	M4	M5	M6	M7
HIV	26.95	26.39	**25.89**	26.03	27.08	31.62	50.10	128.26	459.00
E. coli	12863	12862	12743	12627	12548	12508	**12497**	12599	13099

Dans chaque cas, on a indiqué en lettre grasses quel était le modèle retenu.

Notons que ces méthodes sont conçues pour trouver le modèle qui s'ajuste le mieux à la séquence : si il existe un "vrai" modèle M alors, quand la taille ℓ de la séquence tend vers l'infini, le critère BIC estime M de façon consistante (le modèle estimé "tend vers" le vrai modèle). S'il est concevable d'appliquer un tel résultat non pas quand $\ell \to \infty$, mais pour ℓ grand, encore faut-il une hypothèse d'homogénéité (c'est le *même* modèle qui est valide tout au long de la séquence), qui sera difficile à admettre face à des données biologiques.

S'il s'agit, comme nous le ferons, de détecter des motifs exceptionnels, on pourra se placer successivement dans plusieurs modèles pour tirer des informations différentes et toutes utiles. On a en effet vu qu'un modèle Mm utilise les comptages des mots de longueur $m+1$ pour estimer ses paramètres. Ainsi, dans l'exemple du texte de Cyrano de Bergerac (voir le paragraphe 5.1), le faible comptage mot `aaa` est surprenant dans un modèle M0 car la lettre `a` est très fréquente, mais pas dans le modèle M1 car la probabilité $\pi(\mathtt{a},\mathtt{a})$ est faible.

Enfin, on peut également penser à des critères liés aux connaissances biologiques que nous avons sur ces séquences pour guider notre choix. Par exemple, il est nécessaire de se placer dans un modèle d'ordre $m \geqslant 2$ si on veut tenir compte de la structure des codons (mots de longueur 3) dans les séquences d'ADN codantes.

3.6 Les chaînes de Markov phasées

On a vu que dans les gènes il convient de regrouper les nucléotides 3 par 3 pour constituer des codons, chaque codon étant associé à un acide aminé (ou un STOP). Il n'y a aucun lieu de penser que la même transition (que ce soit à l'ordre 1 ou à un ordre supérieur) s'ajuste sur des gènes pour prédire un nucléotide en début de codon (on dira "en phase 1"), en milieu de codon ("phase 2") ou en fin de codon ("phase 3"). Sur un jeu de données réelles un tant soit peu conséquent, le plus souvent, un test rejettera fortement cette égalité.

Il convient alors d'introduire des modèles permettant aux matrices de transition de dépendre de la phase : une chaîne de Markov (non homogène !) sera dite phasée si l'on a :

– si i est en phase 1, $\mathbb{P}(X_i = b \mid X_{i-1} = a) = \pi_1(a, b)$,
– si i est en phase 2, $\mathbb{P}(X_i = b \mid X_{i-1} = a) = \pi_2(a, b)$,
– si i est en phase 3, $\mathbb{P}(X_i = b \mid X_{i-1} = a) = \pi_3(a, b)$.

La généralisation à Mm est immédiate. On appellera Mm_3 le modèle markovien d'ordre m et phasé.

Remarque 27. Un modèle markovien phasé sur $\mathcal{A}$ est lui aussi markovien sur un alphabet étendu : notons $\mathcal{A}' = \mathcal{A} \times \{1, 2, 3\}$ l'alphabet indiquant le nucléotide et la phase de X_i.

Exemple 28. Si la première lettre de $X =$ `tccgattgcat` est en phase 1, on lui associe $X' = \mathtt{t}_1\mathtt{c}_2\mathtt{c}_3\mathtt{g}_1\mathtt{a}_2\mathtt{t}_3\mathtt{t}_1\mathtt{g}_2\mathtt{c}_3\mathtt{a}_1\mathtt{t}_2$. Si X est une chaîne de Markov phasée selon les notations ci-dessus, X' est une CM de transition π' avec

$$\pi'(a_i, b_j) = \begin{cases} \pi_j(a, b) & \text{si } j = i + 1 \pmod 3 \\ 0 & \text{sinon} \end{cases}$$

Exemple 29. Les matrices suivantes donnent les comptages des 16 mots uv (dans l'ordre `t,c,a,g`) pour chacune des trois phases chez le virus HIV1

$$\begin{pmatrix} 146 & 97 & 274 & 150 \\ 126 & 96 & 257 & 25 \\ 232 & 108 & 325 & 188 \\ 104 & 69 & 191 & 159 \end{pmatrix} \quad \begin{pmatrix} 32 & 103 & 199 & 174 \\ 83 & 84 & 187 & 16 \\ 141 & 167 & 329 & 410 \\ 69 & 106 & 175 & 170 \end{pmatrix} \quad \begin{pmatrix} 167 & 59 & 71 & 128 \\ 111 & 126 & 207 & 16 \\ 230 & 160 & 304 & 197 \\ 159 & 157 & 272 & 182 \end{pmatrix}$$

On en déduit les estimations $\widehat{\pi_\varphi}$ pour les trois phases $\varphi = 1, 2, 3$ et la vraisemblance $L(\text{M1\_3}) = \sum_\varphi \sum_{a,b} N_\varphi(ab) \log(\widehat{\pi_\varphi}(a, b)) = -9\,908.6$.

On peut tester si M1_3 modélise mieux la séquence que le modèle markovien M1 ; la vraisemblance de celui-ci s'avère être égale à $L(\text{M1}) = -10\,046,1$, de sorte que $2D = 2(L(\text{M1\_3}) - L(\text{M1})) = 275$. Sous l'hypothèse M1, $2D$ suit une loi de χ^2 à $36 - 12 = 24$ degrés de liberté. La valeur 275 est très significative, et on conclut que le modèle phasé représente beaucoup mieux la séquence HIV1 que le modèle non phasé – or, on n'a pas tenu compte de la phase de codage alors que celle-ci varie le long de la séquence : la caractère périodique dû aux codons est tellement marqué qu'il apparaît malgré les décalages de phase.

3.7 Diverses chaînes de Markov parcimonieuses

L'un des défauts essentiels des chaînes de Markov réside dans le fait qu'elles dépendent d'un très grand nombre de paramètres dès que leur mémoire augmente : on a vu que le modèle $\mathcal{M}m$ sur un alphabet de taille s requiert $(k-1) \times k^m$ paramètres. Si l'on traite des nucléotides ($k = 4$), le modèle d'ordre 3 compte déjà 192 paramètres, celui d'ordre 5 en compte plus de 3000 ; pour l'alphabet des 20 acides aminés, si $m = 2$, il y a 7600 paramètres !

Lorsque l'on aura à traiter des séquences biologiques, il sera nécessaire d'estimer ces paramètres. On peut donner un contenu précis à l'affirmation : «un ensemble de données contient une certaine *information* qui devra être partagée en autant qu'il y aura de paramètres à estimer»[10]. Remarquons simplement que, si l'on note $N(W)$ le nombre d'occurrences du mot W dans les séquences observées, il est naturel d'estimer $\pi(a_1 \dots a_m, b)$

[10]Pour une description de l'*information de Fisher* et de la théorie de l'estimation, on pourra se référer à Lehmann et Romano (2005)

par

$$\widehat{\pi}(a_1 \dots a_m, b) = \frac{N(a_1 \dots a_m b)}{N(a_1 \dots a_m \cdot)}$$

où $N(a_1 \dots a_m \cdot) = \sum_c N(a_1 \dots a_m c)$.

Si w est un mot de longueur m, conditionnellement[11] à $N(w)$, $\widehat{\pi}(w, b)$ sera la proportion des occurrences de w qui sont suivies d'un b ; sa variance sera

$$\frac{\pi(w, b) \times (1 - \pi(w, b))}{N(w)}$$

Quand la longueur h de w augmente, $N(w)$ aura bien sûr tendance à décroître et cette variance va croître.

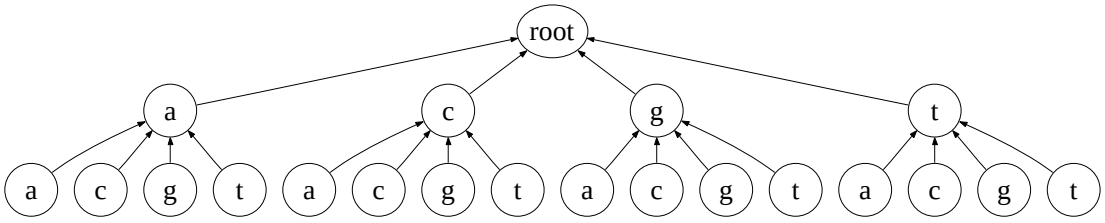


FIG. 3.3 – Représentation d'un modèle markovien (complet) d'ordre 2 par un arbre. Les predicteurs se lisent des feuilles vers la racine.

3.7.1 Les VLMC

C'est pourquoi divers modèles ont été proposés qui partagent deux notions avec les CM : la loi de la séquence se définit récursivement par la données des probabilités conditionnelles $\mathbb{P}(X_t \mid X_s\,,\ s < t)$ et par le fait que celles-ci ne dépendent que d'un nombre réduit, disons m, de lettres X_{t-d} situées avant X_t. Mais on ne considère plus que les k^m mots de longueur m sont tous des prédicteurs différents : on les regroupe en un ensemble de prédicteurs, formant ce que l'on appelle le *contexte*, plus petit : le modèle utilisera donc moins de paramètres (notion de parcimonie).

Reste à choisir un contexte cohérent : à chaque mot $X_{t-m} \dots X_{t-2} X_{t-1}$ de longueur m doit être associé un prédicteur *unique* pour que la loi prédisant X_t soit définie.

Les modèles de Markov à longueur de mémoire variable (VLMC, variable length Markov chains) ont été introduits par Rissanen (1983), puis étudiés par divers auteurs (Buhlmann et Wyner, 1999; Willems et al., 1995; Buhlmann, 2000; Cuturi et Vert, 2005; Talata, 2005; Bourguignon, 2006). L'idée en est que le nombre de lettres précédant la position t nécessaires pour prédire X_t peut varier selon la valeur de ces lettres. Un exemple de tel modèle sera :

- si $X_{t-1} = \mathtt{t}$ ou si $X_{t-1} = \mathtt{c}$, on n'a pas besoin de connaître les $X_s,\ s < t-1$; la prédiction de X_t se fait comme dans le modèle M1,
- si $X_{t-1} = \mathtt{a}$, la prédiction de X_t demande la connaissance de X_{t-2}, mais pas des $X_s,\ s < t-2$; la prédiction de X_t se fait comme dans le modèle M2,
- si $X_{t-1} = \mathtt{g}$ et X_{t-2} vaut $\mathtt{t}$ ou $\mathtt{c}$, la prédiction de X_t se fait à nouveau comme dans le modèle M2,

[11] Pour être rigoureux, il faut préciser ici "dans la séquence privée de sa dernière lettre"

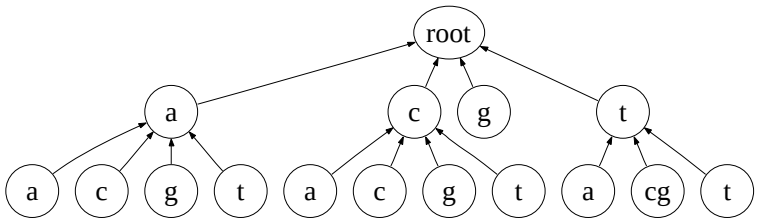


FIG. 3.4 – Représentation d'un VLMC d'ordre 2 par un arbre. Les predicteurs se lisent des feuilles vers la racine.

– dans les autres cas, la prédiction de X_t demande la connaissance de X_{t-2} et X_{t-3}, mais pas des $X_s,\ s < tk-3$; la prédiction de X_t se fait comme dans le modèle M3.

Dans cet exemple, l'information est prise "quand nécessaire" sur les 2 lettres précédant X_k. Mais ce modèle n'utilise que 12 prédicteurs,

```
aa ca ga ta ac cc gc tc g at [cg]t tt
```

contre $4^2 = 16$ dans le modèle M2 complet.

Notons qu'un VLMC est une CM dont l'ordre m est la taille du plus long prédicteur. Il se distingue du modèle complet Mm par le fait que certaines lignes de la matrice de transition sont égales. Dans l'exemple précédent, les lignes associées aux 4 prédicteurs `ag`, `cg`, `gg` et `tg` sont égales ; il en est de même des 2 lignes associées à `[cg]t`.

Un modèle markovien peut être représenté par un arbre : la racine correspond à la lettre à prédire X_t et chaque chemin allant de la racine à une feuille de l'arbre décrit un prédicteur (lu dans le sens décroissant des indices, $X_{t-1}, X_{t-2}, \dots$) (Figure 3.3).

Un VLMC, étant une CM pour laquelle on n'utilise pas toute l'information, sera alors représenté par un arbre dont certaines branches ont été "élaguées". Il y a bijection entre le contexte et l'ensemble des feuilles de l'arbre.

3.7.2 Les PMM

La notion de VLMC sous entend que plus X_{t-d} est loin de X_t (plus t est grand), moins son "influence" sur X_t est grande et donc que l'on peut regrouper en un même prédicteur des contextes partageant un même suffixe ; on peut envisager – et l'expérience montrera que c'est pertinent – de choisir pour prédicteurs (de taille m) des "motifs" obtenus en regroupant plusieurs valeurs de X_{t-d} pour chaque d compris entre 1 et m.

C'est ce qui a conduit Bourguignon (Bourguignon, 2006) à introduire les modèles markoviens parcimonieux (PMM). Ceux ci considèrent l'ensemble $\mathcal{P}$ de toutes les *partitions* possibles de l'alphabet $\mathcal{A}$. Dans le cas des nucléotides, il y a ainsi 15 partitions :

`(t)(c)(a)(g)`	`(tc)(a)(g)`	`(ta)(c)(g)`	`(tg)(a)(c)`	`(ca)(t)(g)`
`(cg)(t)(a)`	`(ag)(t)(c)`	`(tc)(ag)`	`(ta)(cg)`	`(tg)(ac)`
`(tca)(g)`	`(tcg)(a)`	`(tag)(c)`	`(cag)(t)`	`(tcag)`

Notons que le nombre de partitions augmente très vite avec la taille de l'alphabet. Pour $k = 20$, taille de l'alphabet des acides aminés, $|\mathcal{P}| \approx 50\ 10^{12}$. Il conviendra sans doute de regrouper les a.a. en classes fondées sur un critère physico-chimique.

Comme pour les VLMC, la cohérence du contexte résultera de sa construction récursive : l'arbre des prédicteurs sera obtenu en disposant sous chaque noeud une partition de $\mathcal{A}$, jusqu'à atteindre un arbre de profondeur m (Figure 3.4 .

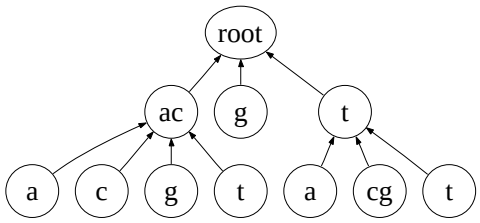


FIG. 3.5 – Représentation d'un PMM d'ordre 2 par un arbre. Les predicteurs se lisent des feuilles vers la racine.

La figure (3.5) représente un PMM d'ordre 2 dont le contexte contient 8 prédicteurs : `a[ac]`, `c[ac]`, `g[ac]`, `t[ac]`, `[acgt]g`, `at`, `[cg]t` et `tt` (la notation `[ ]` indique un "ou" ; par exemple le premier prédicteur `a[ac]` signifie $X_{t-2} =$ `a` et $X_{t-1} =$ `a` ou `c`).

Remarque 30. Dans Zhao et al. (2005), les auteurs introduisent des "VLMC permutés" qui choisissent une permutation des indices $t-1, t-2, \ldots, t-m$ sur laquelle ils construisent un modèle de longueur variable. En un sens, il s'agit d'une solution intermédiaire entre VLMC et PMM.

Exemple 31. Bourguignon a ajusté un modèle PMM de profondeur maximum 3 sur les 149 génomes bactériens complets disponibles. La figure (3.6) représente la différence entre le critère BIC de ce modèle et le critère BIC d'un modèle M3. On constate qu'il est toujours positif (le PMM s'ajuste mieux aux données) et ceci est d'autant plus vrai que le génome bactérien est court. Lorsque l'ordre augmente, on constate une augmentation sensible du gain moyen, même s'il arrive sur quelques cas isolés que l'on obtienne une légère perte.

3.7.3 Les MTD

Une tout autre approche est possible pour restreindre l'ensemble des modèles markoviens d'ordre m fixé et donc réduire le nombre de paramètre. Elle a été introduite par Berchtold (2001); Raftery et Berchtold (2002) et étudiée dernièrement dans Lèbre et Bourguignon (2005).

Elle consiste à associer une loi prédictive de X_t à chaque X_{t-d} et à considérer une combinaison linéaire des lois.

Plus formellement, à chaque distance d sera associée une matrice de transition π_d entre états de $\mathcal{A}$ et le modèle MTD est une CM d'ordre m avec

$$\mathbb{P}(X_t = y \mid (X_{t-m} \ldots X_{t-1}) = (x_m \ldots x_1)) = \sum_{d=1}^{m} \alpha_d \, \pi_d(x_d, y) \tag{3.12}$$

où l'on impose que $\sum \alpha_d$ soit égal à 1.

Notons d'emblée que ce modèle peut être interprété ainsi : pour prédire X_k connaissant le passé, on commence par tirer une distance D selon la loi $\mathbb{P}(D = d) = \alpha_d$; puis on tire X_k selon $\mathbb{P}(X_k = y) = \pi_d(X_{k-d}, y)$. Cette interprétation en fait un modèle de chaîne de Markov cachée (HMM, voir chapitre suivant).

Ce modèle est markovien d'ordre m, mais il ne dépend que de m matrices 1×1, soit $m \times t(t-1)$ paramètres auxquels il faut ajouter m valeurs α_d liées par une contrainte.

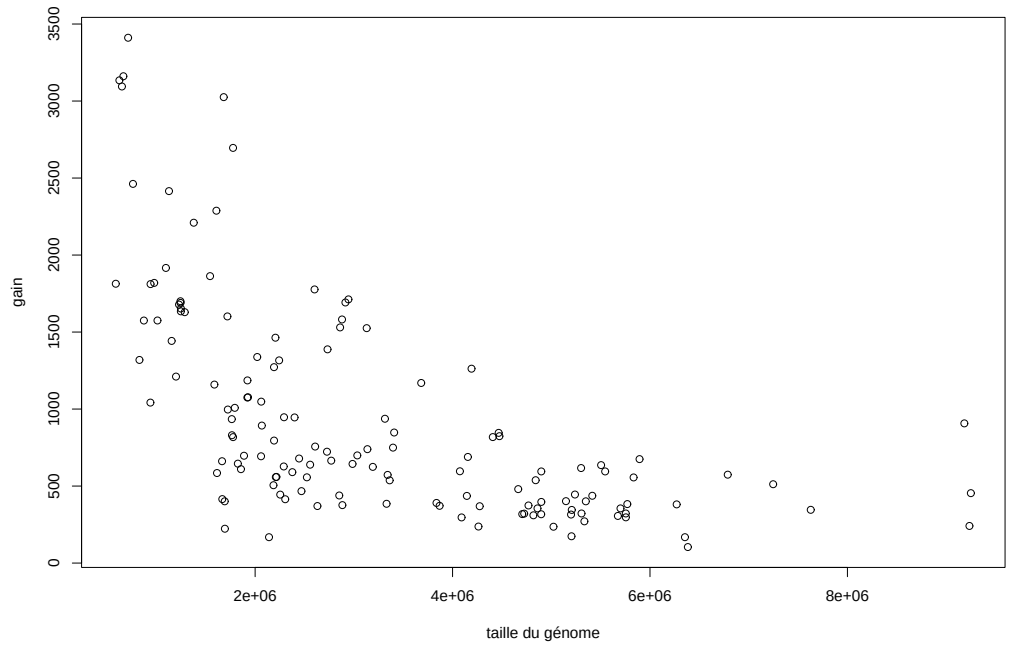


FIG. 3.6 – L'ordonnée de chaque point représente la différence BIC(PMM) - BIC(M) pour l'une des 142 bactéries testées ; en abscisse on a porté taille du génome correspondant. Plus un point est haut, meilleur est le gain du PMM par rapport au modèle markovien classique.

Il y a donc $mt(t-1)+m-1$ paramètres. Mais ces paramètres sont liés : en termes de chaîne d'ordre m, on a immédiatement

$$\begin{aligned}\Delta &= \mathbb{P}(x_m,\ldots,x_d,\ldots,x_1;y)-\mathbb{P}(x_m,\ldots,x'_d,\ldots,x_1;y)\\ &= \alpha_d\,[\pi_d(x,y)-\pi_d(x',y)]\end{aligned}$$

Modifier le prédicteur en une seule position ($x_d \to x'_d$) modifie la probabilité conditionnelle de $\{X_k = y\}$ d'une quantité Δ *qui ne dépend pas des* x_ℓ *pour* $\ell \neq d$. Il est alors montré dans Lèbre et Bourguignon (2005) que ceci diminue le nombre de paramètres effectifs de $k(m-1)$.

En d'autres termes, le modèle défini par (3.12) est sur-paramétré, on ne pourra donc jamais estimer les α_d et les π_d. Néanmoins, il est montré dans Lèbre et Bourguignon (2005) que son estimation (par un algorithme EM, tirant partie de son interprétation HMM) est consistante comme élément de Mm.

Exemple 32. Lèbre a ajusté un modèle MTD sur chaque phase des régions codantes de *Mezorhizobium loti* ($\ell = 148\,983$). Nous donnons ici l'estimation du modèle MTD5 obtenu sur la phase 1 (les prédicteurs X_{t-1} est donc en phase 3, X_{t-2} en phase 2, etc). Les matrices π_φ ci dessous sont pondérées par les poids $\alpha = (0.294\ ,\ 0.470\ ,\ 0.110\ ,\ 0.083\ ,\ 0.043)$.

$$\pi_1 = \begin{pmatrix} 0.485 & 0.186 & 0.036 & 0.293 \\ 0.463 & 0.001 & 0.415 & 0.121 \\ 0.436 & 0.523 & 0.041 & 0 \\ 0 & 0.435 & 0.303 & 0.262 \end{pmatrix} \quad \pi_2 = \begin{pmatrix} 0.121 & 0.294 & 0.466 & 0.119 \\ 0.124 & 0.182 & 0.506 & 0.188 \\ 0.124 & 0.317 & 0.356 & 0.203 \\ 0.099 & 0.231 & 0.506 & 0.164 \end{pmatrix}$$

$$\pi_3 = \begin{pmatrix} 0.132 & 0.447 & 0.306 & 0.115 \\ 0.186 & 0.269 & 0.341 & 0.204 \\ 0.306 & 0.374 & 0.224 & 0.096 \\ 0.302 & 0.477 & 0.084 & 0.137 \end{pmatrix} \quad \pi_4 = \begin{pmatrix} 0.130 & 0.363 & 0.470 & 0.037 \\ 0.332 & 0.316 & 0.295 & 0.057 \\ 0.217 & 0.050 & 0.635 & 0.098 \\ 0.004 & 0.230 & 0.765 & 0.001 \end{pmatrix}$$

$$\pi_5 = \begin{pmatrix} 0.362 & 0.186 & 0.198 & 0.254 \\ 0.216 & 0.082 & 0.443 & 0.259 \\ 0.007 & 0.013 & 0.572 & 0.408 \\ 0.030 & 0.043 & 0.927 & 0 \end{pmatrix}$$

Sa log-vraisemblance vaut $-192\,111$ pour 64 paramètres ; celle du modèle markovien (phasé) d'ordre 5 vaut 185 538, pour 768 paramètres. Le gain de BIC est donc de 94 270 !

3.8 Les chaînes de Markov dérivantes

Nous appelons modèle hétérogène, un modèle markovien dont la matrice de transition est une fonction de la position courante dans la séquence. Avec les modèles phasés, nous avons vu un premier exemple d'un tel modèle (dans ce cas, l'hétérogénéité est périodique). Il est également possible de segmenter les séquences à étudier en fonction de leurs propriétés biologiques (codant, non codant par exemple) ou grâce à des outils comme les chaînes de Markov cachées. Dans ce cas cependant, on se ramène sur chaque segment à l'étude d'un modèle homogène.

Il est également possible d'envisager des modification "continue" du modèle markovien et c'est précisément ce que font les chaînes dérivantes (CMD) introduites par Vergne (2006).

Dérive linéaire On dira que la séquence $X_t, (t = 0, \ldots, \ell)$ est une CMD linéaire s'il existe deux matrices de transition π_0 et π_1 telles que pour tout $t \in [1, \ell]$,

$$\mathbb{P}(X_t = v \mid X_s \; s < t) = \left(\frac{t}{\ell} \, \pi_1 \; + \; (1 - \frac{t}{\ell}) \, \pi_0 \right) (X_{t-1}, v)$$

π_0 et π_1 étant des transitions, il en est bien sûr de même de $\pi_\alpha = \alpha \, \pi_1 + (1 - \alpha) \, \pi_0$ pour tout α de $[0; 1]$.

La vraisemblance d'un tel modèle est facile à écrire mais sa maximisation ne peut se faire analytiquement. Il est préconisé dans (Vergne, 2006) de minimiser la somme des "erreurs de prédiction" du modèle

$$Q(\pi_0, \pi_1) = \sum_{t=1}^{\ell} \sum_{u \in \mathcal{A}} \mathbb{I}\{X_{t-1} = u\} \sum_{v \in \mathcal{A}} \left[\pi_{t/\ell}(u, v) - \mathbb{I}\{X_t = v\} \right]^2$$

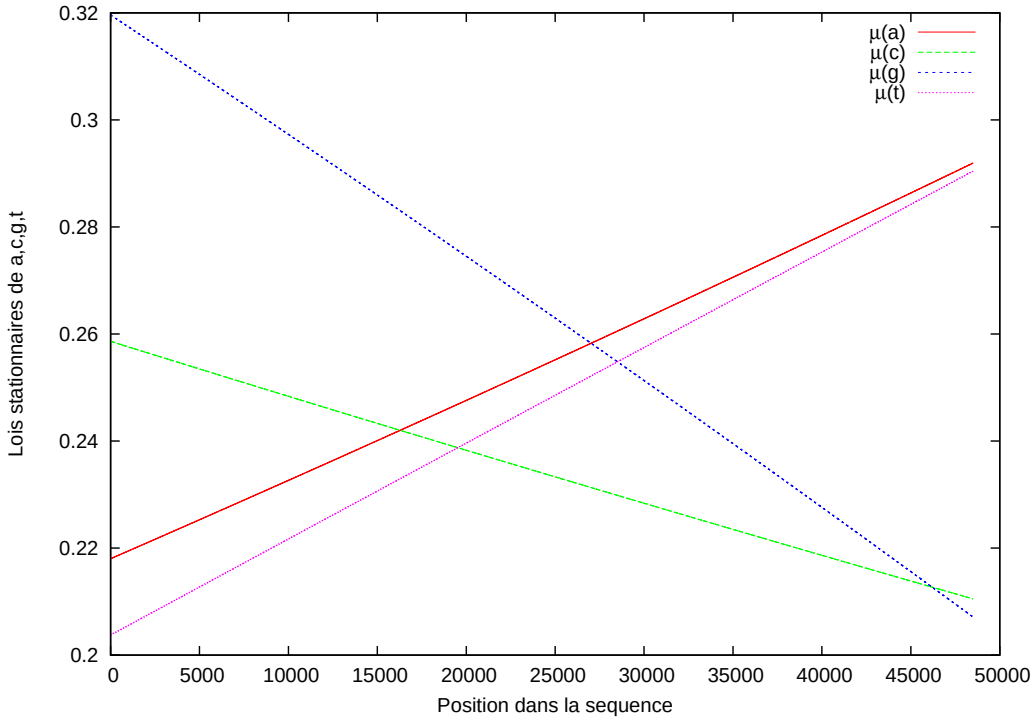


FIG. 3.7 – Ajustement sur le Phage Lambda d'une chaîne de Markov dérivante de degré 1. On a résumé l'information par les quatre droites $\mu_k(u)$ en fonction de la position k pour chacun des nucléotides u.

forme quadratique en les différents paramètres (soumis aux contraintes que les sommes par ligne valent 1). On obtient $\pi_0(x,y) = A_0(u,v)/D(u,v)$ et $\pi_1(u,v) = A_1(u,v)/D(u,v)$, avec

$$D(u,v) = \begin{vmatrix} \mathrm{a} & \mathrm{b} \\ \mathrm{b} & \mathrm{d} \end{vmatrix} \qquad A_0(u,v) = \begin{vmatrix} \mathrm{d} & \mathrm{b} \\ \mathrm{g} & \mathrm{h} \end{vmatrix} \qquad A_1(u,v) = \begin{vmatrix} \mathrm{a} & \mathrm{b} \\ \mathrm{h} & \mathrm{g} \end{vmatrix}$$

où

$$a = \sum \left(1 - \frac{t}{\ell}\right)^2 \qquad b = \sum \frac{t}{\ell}\left(1 - \frac{t}{\ell}\right) \qquad d = \sum \left(\frac{t}{\ell}\right)^2$$

les sommes dans a, b et d étant prises sur les indices t tels que $X_{(t-1)/\ell} = u$, et

$$g = \sum \left(\frac{t}{\ell}\right) \qquad h = \sum \left(1 - \frac{t}{\ell}\right)$$

les sommes dans g et h étant prises sur les indices tels que $X_{(t-1)/\ell} = u$ et $X_{t/\ell} = v$.

Dérive générale On généralise au cas où la transition en position t est, par exemple, un polynôme en $\alpha = t/\ell$. Il est plus intéressant d'introduire une modélisation *spline*, par exemple (spline linéaire) en découpant le segment $[1,\ldots,n]$ en R sous-segments dont on notera les extrémités $T_j, j = 0,\ldots,R$ (avec $T_0 = 1$ et $T_R = \ell$), en considérant $R+1$ transitions π_j et en supposant une dérive linéaire entre π_j et π_{j+1} quand la position t va de T_j à T_{j+1}.

Exemple 33. Vergne a ajusté des modèles de chaînes de Markov dérivantes sur le phage *Lambda* ($\ell = 48\,502$). Nous retiendrons ici uniquement les modèles de degré 1 (dérive

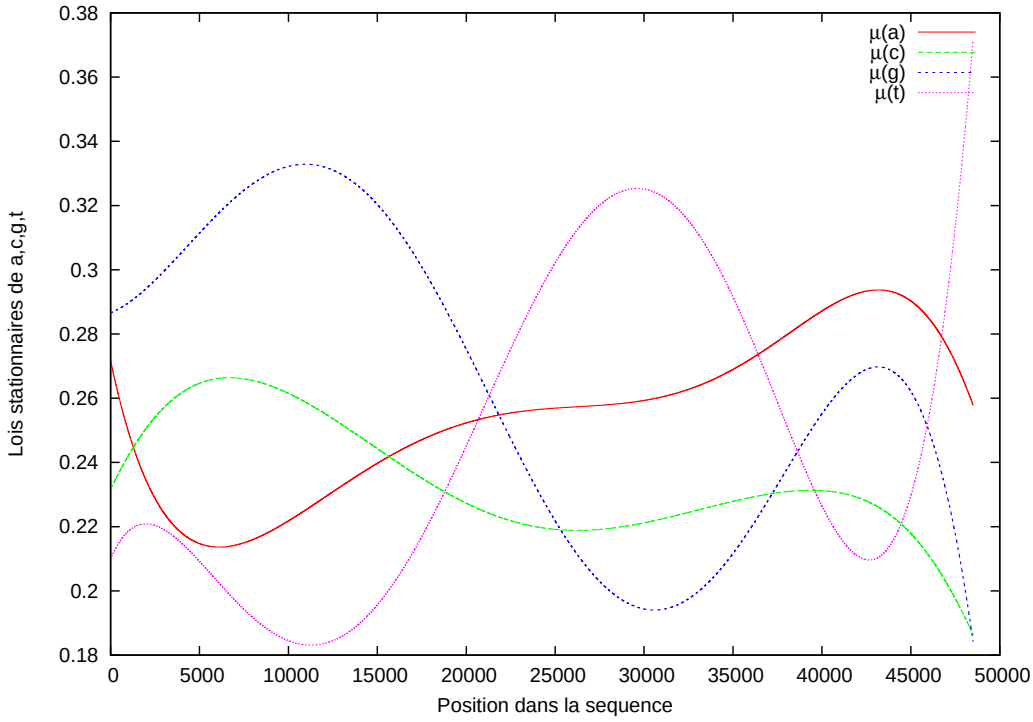


FIG. 3.8 – Ajustement sur le Phage Lambda d'une chaîne de Markov dérivante de degré 5. On a résumé l'information par les quatre courbes $\mu_t(u)$ en fonction de la position t pour chacun des nucléotides u.

linéaire) et 5. Les figures (3.7) et (3.8) donnent une idée de la variabilité de la transition π_t en position t en ne gardant que la loi stationnaire $\mu_t = \mu_t \pi_t$: on a tracé la fréquence prédite en chaque position par le modèle pour les 4 bases.

Il convient bien sûr de comparer la vraisemblance de ces modèles, ce que fait par exemple le critère BIC. Le modèle M1 dépend de 12 paramètres et à chaque fois que le degré de la chaîne dérivante augmente de 1, il convient d'ajouter 12 nouveaux paramètres. De fait, c'est le modèle de degré 5 qui obtient le meilleur critère BIC.

	nb paramètres	Log–vrais	BIC
Markov 1	12	−66 714.19	133 557.75
Dérivant d°1	24	−66 482.59	133 224.02
Dérivant d°5	72	−66 134.50	133 042.83
Dérivant d°8	108	−66 061.09	133 287.43

3.9 Notes bibliographiques

Il existe bien sûr d'innombrables introductions aux Probabilités et/ou aux Statistiques. Citons Sheskin (2003); Lehmann et Romano (2005); Lehmann (2006) ou Johnson (2006). On y trouvera aussi bien la théorie générale de l'inférence (estimation, tests, vraisemblance, information de Fisher, etc.) qu'une description de l'approche non paramétrique (tests non paramétriques, tau de Kendall, etc . . .)

Les processus “à mémoire courte”, attribués à Andrei Markov (1856-1922), ont fait l'objet d'une immense littérature, qu'il n'est pas question de synthétiser ici. Le lecteur se reportera aux ouvrages Karlin et Taylor (1975) ou Norris (1997) pour une introduction ; Cinlar (1975) ou Meyn et Tweedie (1993) permettront d'approfondir la théorie ; Stewart (1994) présente enfin une approche davantage tournée vers leur usage dans des programmes informatiques.

Quelques références concernant Perron-Frobénius : Cinlar (1975); Karlin et Taylor (1975); Meyn et Tweedie (1993) ou Norris (1997).

Chapitre 4
Compter les motifs (G. Nuel)

Dans cette partie nous allons introduire les définitions et notations nécessaires à l'étude des motifs dans les séquences aléatoires (4.1). Nous verrons comment, à un tel motif, on peut associer un automate capable de le "reconnaître" – voire de le compter sur une séquence ; nous montrerons comment le construire et soulignerons à quel point son emploi est efficace pour compter les motifs (4.2) ; enfin, nous présenterons des algorithmes permettant de compter simultanément un grand nombre de motifs (4.3).

4.1 Définitions

4.1.1 Alphabet

On considère un *alphabet* fini $\mathcal{A}$ de cardinal k, dont les éléments sont représentés par des symboles différents. Éventuellement, on choisit pour cet alphabet un ordre arbitraire (par exemple, l'ordre lexico-graphique) qui permet d'identifier chaque élément à un nombre entier. Voici quelques exemples d'alphabets possibles :

Binaire : à cause de sa simplicité, l'alphabet $\mathcal{A} = \{\mathtt{a}, \mathtt{b}\}$ sera souvent utilisé pour servir d'exemple ;

ADN : pour les acides nucléiques on utilisera l'alphabet $\mathcal{A} = \{\mathtt{t}, \mathtt{c}, \mathtt{a}, \mathtt{g}\}$. Bien évidemment, des regroupements sont possibles. Par exemple $\mathcal{A} = \{\mathtt{a} \text{ ou } \mathtt{g}, \mathtt{c} \text{ ou } \mathtt{t}\}$ permet d'étudier l'alternance purines-pyrimidines ;

Protéines : pour les acides aminés, on considère l'alphabet à 20 lettres

$$\mathcal{A} = \{\mathtt{A}, \mathtt{C}, \mathtt{D}, \mathtt{E}, \mathtt{F}, \mathtt{G}, \mathtt{H}, \mathtt{I}, \mathtt{K}, \mathtt{L}, \mathtt{M}, \mathtt{N}, \mathtt{P}, \mathtt{Q}, \mathtt{R}, \mathtt{S}, \mathtt{T}, \mathtt{V}, \mathtt{W}, \mathtt{Y}\}$$

ou toute partition fondée sur des propriétés physico-chimiques ;

Latin : On peut aussi considérer l'alphabet latin complet

$$\mathcal{A} = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}, \mathtt{e}, \mathtt{f}, \mathtt{g}, \mathtt{h}, \mathtt{i}, \mathtt{j}, \mathtt{k}, \mathtt{l}, \mathtt{m}, \mathtt{n}, \mathtt{o}, \mathtt{p}, \mathtt{q}, \mathtt{r}, \mathtt{s}, \mathtt{t}, \mathtt{u}, \mathtt{v}, \mathtt{w}, \mathtt{x}, \mathtt{y}, \mathtt{z}\}$$

ou l'une de ses partitions (ex : $\mathcal{A} = \{\text{consonnes}, \text{voyelles}\}$).

4.1.2 Séquence

Sur un tel alphabet, il est possible de considérer une *séquence* $x = x_1 \dots x_\ell$ ($x_i \in \mathcal{A}$ pour $1 \leqslant i \leqslant \ell$) de longueur ℓ. On notera x_i^j ($i \leqslant j$) pour désigner la sous-séquence $x_i \dots x_j$. On utilisera la même notations mais avec des lettres majuscules pour une séquence aléatoire (une chaîne de Markov par exemple).

4.1.3 Mot

On appelle *mot* de longueur h un élément $w = w_1 \dots w_h \in \mathcal{A}^h$ ($w_i \in \mathcal{A}$ pour $1 \leqslant i \leqslant h$). Il s'agit simplement d'une courte séquence (en général on pourra considérer que h est très petit devant ℓ). Comme pour les séquences, on pourra utiliser la notation w_i^j ($i \leqslant j$) pour désigner le sous-mot $w_i \dots w_j$.

Palindromes

Définition 34. On appellera *inverse-complémentaire* d'une lettre, celle qui lui est associée dans l'appariement entre brins :

$$\overline{\texttt{a}} = \texttt{t} \qquad \overline{\texttt{t}} = \texttt{a} \qquad \overline{\texttt{c}} = \texttt{g} \qquad \overline{\texttt{g}} = \texttt{c}$$

Plus généralement, on appelle inverse-complémentaire d'un mot $w = w_1 w_2 \dots w_h$, et on note $\overline{w}$, le mot obtenu en prenant les lettres de w dans l'ordre inverse et en les remplaçant par leurs inverses complémentaires : $\overline{w} = \overline{w}_h \overline{w}_{h-1} \dots \overline{w}_1$. Par exemple, l'inverse-complémentaire de $w = \texttt{ttcaggcta}$ sera $\overline{w} = \texttt{tagcctgaa}$.

En conséquence si un brin d'ADN porte en une position donnée le mot w, le brin opposé (lu lui aussi dans le sens conventionnel 5'-3') portera le mot inverse-complémentaire $\overline{w}$.

Définition 35. En génomique on appelle *palindrome* un mot w égal à son inverse-complémentaire : $w = \overline{w}$. Donc si un palindrome w figure sur un brin, il figure aussi sur son complémentaire. Exemple : `ttctagaa` est un palindrome.

4.1.4 Motif

On définit un *motif* $\mathcal{W}$ comme un ensemble *fini* de mots $w \in \mathcal{W}$ non inclus les uns dans les autres (cette hypothèse technique n'est pas très restrictive et facilite l'interprétation de cette notion). Afin de définir simplement des motifs complexes, on adopte une série de notations pour décrire la dégénérescence d'un mot donné :

caractère quelconque : le symbole '`.`' à une position donnée signifie que cette position est totalement dégénérée : on peut remplacer ce symbole par n'importe quelle lettre ;

plusieurs choix : lorsque la dégénérescence concerne encore une position donnée mais que seules quelques lettres de l'alphabet sont possibles, on donne la liste de ces lettres encadrée par des crochets. Ainsi, `[ag]` signifie `a` ou `g`.

lettre(s) interdite(s) : on peut également interdire une ou plusieurs lettres à une position donnée en donnant la liste des lettres à éviter entre accolades. Sur l'alphabet des acides nucléiques, `{c}` signifie donc `a`, `g` ou `t`.

répétitions : Faire suivre une position dans le motif d'un (m) (où m est un entier) signifie qu'il faut répéter m le motif . Le symbole $(m-n)$ quant à lui, signifie qu'on le répète un nombre quelconque de fois compris entre m et n. Ainsi, `a(5)` signifie `aaaaa`, `[cg](1-2)` signifie `[cg]` ou bien `[cg][cg]` et enfin, `.(0-5)` signifie 0 à 5 caractères quelconques.

Sur l'alphabet des nucléotides, par exemple, l'ensemble

$$\{\texttt{atagc}, \texttt{atcgc}, \texttt{atggc}, \texttt{attgc}\}$$

correspond au motif

$$\texttt{at.gc} = \texttt{at[acgt]gc} = \texttt{at\{\}gc} = \texttt{at.(1)gc} = \texttt{at.(1-1)gc}$$

4.1.5 Que compter ?

Pour un mot

Il y a, de fait, plusieurs façons de compter le nombre d'occurrences d'un mot dans une séquence. Par exemple dans la séquence

$$x = \texttt{abaaaaaababaaaabbaaaba} \tag{4.1}$$

de longueur $\ell = 22$, combien compte-t-on d'occurrences du mot $w = \texttt{aaa}$ de longueur $h = 3$? A cause de la capacité de w se recouvrir lui même, ses occurrences ont tendance à apparaître par paquets (*clumps* en anglais)[1]. Dans la séquence X on compte 3 de ces paquets : $x_3^8 = \texttt{aaaaaa}$, $x_{12}^{15} = \texttt{aaaa}$ et $x_{18}^{20} = \texttt{aaa}$. À l'intérieur de chacun, on peut compter toutes les occurrences ou seulement les occurrences non chevauchantes. Dans le premier paquet, on compte ainsi 4 occurrences chevauchantes mais seulement 2 occurrences non chevauchantes, et dans le deuxième paquet, on a 2 occurrences ou 1 seule selon le choix fait. Au total ceci conduit à compter, dans x, 7 occurrences (chevauchantes) ou bien 4 occurrences (non chevauchantes). Selon les problèmes considérés, l'un ou l'autre de ces comptages pourra se révéler le plus pertinent. Définissons donc ces deux comptages :

Définition 36 (Occurrences chevauchantes – *overlap* en anglais)**.** On définit le nombre d'*occurrences chevauchantes* du mot w dans la séquence x par

$$N_x^O(w) = \sum_{i=1}^{\ell} \mathbb{I}\{w \text{ se termine en } i\} \tag{4.2}$$

Dans la suite, sauf mention contraire, c'est ce nombre d'occurrences qui sera utilisé (et noté $N_x(w)$).

Définition 37 (Occurrences renouvelantes – *renewal* en anglais –)**.** Deux occurrences distinctes d'un mot sont dites chevauchantes si elles partagent au moins une lettre commune. Une occurrence est dite renouvelante si elle ne chevauche aucune des occurrences renouvelante qui la précède (par conséquent, la première occurrence de la séquence est toujours une occurrence renouvelante). On note $N_x^R(w)$ le nombre d'*occurrences renouvelantes* du mot w dans la séquence x.

Lorsque cela ne prêtera pas à confusion, on pourra, dans ces notations, remplacer simplement la séquence x par sa longueur ℓ ou bien même omettre carrément l'indice ou l'exposant.

Si on reprend l'exemple de la séquence x donnée en (4.1) on constate que $N_x^O(\texttt{aaa}) = 7$ et que $N_x^R(\texttt{aaa}) = 4$.

Pour un motif

Si on considère maintenant un motif $\mathcal{W}$, on peut définir de façon analogue son nombre d'occurrences en remplaçant simplement w par $\mathcal{W}$ dans les définitions 36 et 37. Notons que $N_x^0(\mathcal{W})$ est la somme des $N_x^O(w)$ pour les différents w composant $\mathcal{W}$, mais que la propriété analogue est fausse pour les comptages renouvelants : si $\mathcal{W}$ est composé de deux

[1] on peut aussi compter ces paquets et faire des statistiques sur leur nombre. Voir le paragraphe sur les approximations de Poisson.

mots, soit $\mathcal{W} = \{w, w'\}$, il pourra (aussi) y avoir chevauchement si une occurrence de w chevauche une occurrence de w'.

On pourrait également introduire d'autres comptages faisant, par exemple, intervenir des pondérations différentes pour les différents mots w composant un motif.

4.2 Automates

Toute cette partie fait largement référence à la théorie classique des langages et automates. Nous ne cherchons pas ici à proposer un cours sur ce sujet, mais simplement à introduire les notions qui seront nécessaires à l'étude des motifs dans les séquences biologiques. Nous invitons donc le lecteur plus particulièrement intéressé par cette théorie à consulter un ouvrage de référence sur le sujet comme Hopcroft et al. (2001).

4.2.1 Langages

On appelle *langage* sur un alphabet donné, un ensemble de mots (ou de séquences) sur ce même alphabet. Par exemple $\mathcal{L} = \{\texttt{gctggtgg}\}$ est un langage sur l'alphabet des acides nucléiques réduit au singleton $\texttt{gctggtgg}$; l'ensemble des séquences d'ADN de longueur $\ell \geqslant 100$ est un autre langage (cette fois-ci infini) sur le même alphabet. On désigne par ε le mot (ou la séquence) de longueur nulle.

On définit sur les langages trois opérations dites *régulières* (ou rationnelles) :

addition la *somme* de deux langages est simplement leur réunion. Par exemple $\{\texttt{atc}, \texttt{tca}\} + \{\texttt{tca}, \texttt{ttt}\} = \{\texttt{atc}, \texttt{tca}, \texttt{ttt}\}$. Notons que cette opération est commutative et que $\emptyset$ est son élément neutre.

produit le *produit* $\mathcal{L}_1 \cdot \mathcal{L}_2$ de deux langages est le langage $\{w_1 w_2, w_1 \in \mathcal{L}_1, w_2 \in \mathcal{L}_2\}$ (où $w_1 w_2$ est la concaténation des mots w_1 et w_2). Par extension on désigne par $\mathcal{L}^n$ le langage résultant de n produits successifs de $\mathcal{L}$ avec lui-même ($\mathcal{L}^0 = \{\varepsilon\}$ par convention). Par exemple, $\{\texttt{ab}\} \cdot \{\texttt{abb}, \texttt{bbb}\} = \{\texttt{ababb}, \texttt{abbbb}\}$ et $\{\texttt{ab}\}^3 = \{\texttt{ababab}\}$. Notons que cette opération n'est *pas* commutative et que $\{\varepsilon\}$ est son élément neutre.

clôture de Kleene l'opérateur unaire * de Kleene permet de définir la *clôture* $\mathcal{L}^*$ d'un ensemble $\mathcal{L}$ par $\mathcal{L}^* = \cup_{n \geqslant 0} \mathcal{L}^n$. Par exemple on a $\{\texttt{ab}\}^* = \{\varepsilon, \texttt{ab}, \texttt{abab}, \ldots\}$ et $\mathcal{A}^*$ désigne l'ensemble de tous les mots (ou séquences) sur l'alphabet $\mathcal{A}$.

Définition 38 (langage régulier)**.** Un langage sur un alphabet $\mathcal{A}$ est dit *régulier* (ou rationnel) si il peut être obtenu de façon finie à partir de partie finies de $\mathcal{A}^*$ et d'opérations régulières. La suite finie des opérations permettant d'obtenir le dit langage est appelée son *expression régulière* (ou rationnelle). Par exemple, tout langage fini est régulier, $\mathcal{A}^*$ est régulier. En revanche, le langage

$$\{\texttt{ab}, \texttt{aabb}, \texttt{aaabbb}, \ldots, \underbrace{\texttt{a}\ldots\texttt{a}}_{i \text{ fois}}\underbrace{\texttt{b}\ldots\texttt{b}}_{i \text{ fois}}, \ldots\}$$

n'est pas régulier (attention à ne pas confondre ce langage avec $\texttt{a}^*\texttt{b}^*$). Notons que les langages réguliers (resp. expression régulières) sont des cas particuliers des langages formels (resp. grammaires formelles).

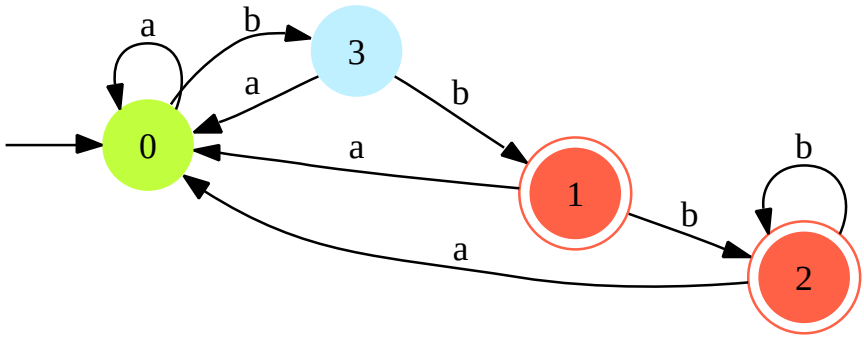


FIG. 4.1 – Représentation graphique de l'automate de l'exemple 41.

4.2.2 Automates Finis Déterministes (AFD)

Définition 39 (AFD)**.** On appelle *Automate Fini Déterministe* (AFD) la donnée d'un quintuplet $(\mathcal{A}, \mathcal{Q}, s, \mathcal{F}, \delta)$ où
- $\mathcal{A}$ est un alphabet fini,
- $\mathcal{Q}$ un ensemble (fini) d'états,
- $s \in \mathcal{Q}$, un état initial,
- $\mathcal{F} \subset \mathcal{Q}$ un ensemble (non vide) d'états, qualifiés de terminaux, et
- $\delta : \mathcal{Q} \times \mathcal{A} \to \mathcal{Q}$ une fonction de transition (qui rende $\mathcal{Q}$ irréductible).

On dit qu'un mot $w = w_1 \dots w_h$ est *accepté* (ou reconnu) par l'automate si il existe une séquence d'états $q_0, q_1, \dots, q_h \in \mathcal{Q}$ telle que $q_0 = s$, $\delta(q_{i-1}, w_i) = q_i$ pour tout $1 \leqslant i \leqslant h$ et $q_h \in \mathcal{F}$ (on dit alors simplement que $\delta(q_0, w) = q_h \in \mathcal{F}$). L'ensemble des mots acceptés par l'automate est appelé son langage.

Graphe associé A un tel AFD on associe naturellement le graphe dont les sommets sont les éléments de $\mathcal{Q}$ et les arêtes sont valuées par les lettres de $\mathcal{A}$ de sorte que si $q = \delta(p, a)$ (où $p, q \in \mathcal{Q}$ et $a \in \mathcal{A}$), on trace une flèche allant p à q et portant le label a.

Il est facile de savoir si un mot $w = w_1 \dots w_h$ (resp. une séquence $x = x_1 \dots x_\ell$) appartient au langage d'un automate, puisqu'il suffit de déterminer si $\delta(s, w) \in \mathcal{F}$ ce qui peut se faire récursivement en $O(h)$ (resp. en $O(\ell)$). Le qualificatif de "final" ou "terminal" pour les éléments de $\mathcal{F}$ se réfère à la présence de la fin d'un motif.

Remarque 40. Notons qu'un AFD vérifiant la définition 39 est classiquement appelé AFD *complet* car les transitions suivant chaque lettre de l'alphabet sont définies pour chaque état. Dans ce document, nous ne considérerons que des AFD complets et omettrons donc l'adjectif correspondant.

Exemple 41. On considère $\mathcal{A} = \{\mathtt{a}, \mathtt{b}\}$, l'espace d'états $\mathcal{Q} = \{0, 1, 2, 3\}$, l'état initial $s = 0$, le sous ensemble d'états terminaux $\mathcal{F} = \{1, 2\}$ et la fonction de transition définie par

$$\delta(0, \mathtt{a}) = 0, \quad \delta(0, \mathtt{b}) = 3, \quad \delta(1, \mathtt{a}) = 0, \quad \delta(1, \mathtt{b}) = 2,$$
$$\delta(2, \mathtt{a}) = 0, \quad \delta(2, \mathtt{b}) = 2, \quad \delta(3, \mathtt{a}) = 0 \text{ et } \delta(3, \mathtt{b}) = 1.$$

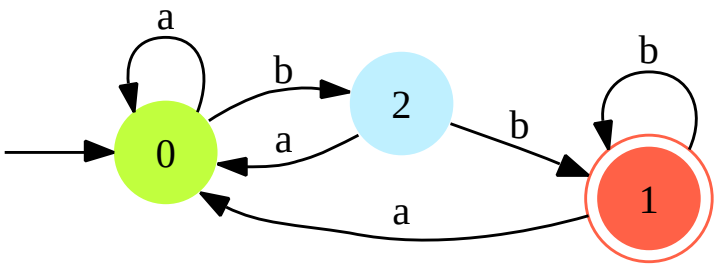


FIG. 4.2 – Automate minimal correspondant au langage $\mathcal{L} = \{\mathtt{a}, \mathtt{b}\}^*\mathtt{bb}$

Le quintuplet $(\mathcal{A}, \mathcal{Q}, s, \mathcal{F}, \delta)$ définit alors un AFD (représenté en figure 4.1). La séquence $x = \mathtt{abbb}$ est-elle reconnue par l'automate ? Pour répondre à cette question, il suffit de déterminer si $\delta(0, \mathtt{abbb}) = \delta(0, \mathtt{bbb}) = \delta(3, \mathtt{bb}) = \delta(1, \mathtt{b}) = 2$ appartient ou non à $\mathcal{F}$; comme c'est le cas, la séquence est reconnue par l'automate. En fait, le langage associé à cet automate est $\mathcal{L} = \{\mathtt{a}, \mathtt{b}\}^*\mathtt{bb}$ (c'est à dire l'ensemble des séquences sur $\mathcal{A}$ se terminant par $\mathtt{bb}$).

Il existe un lien très fort entre les AFD et les langages réguliers comme l'atteste le théorème suivant :

Théorème 42. Pour tout langage régulier $\mathcal{L}$ il existe (à isomorphisme près) un unique plus petit AFD reconnaissant $\mathcal{L}$ et, inversement, tout langage reconnu par un AFD est régulier.

A noter que l'automate de l'exemple 41 (quatre états), s'il reconnaît bien le langage régulier $\mathcal{L} = \{\mathtt{a}, \mathtt{b}\}^*\mathtt{bb}$, n'est pas minimal contrairement à celui de la figure 4.2 (trois états).

4.2.3 Comptages

Nous allons voir ici comment utiliser les automates pour compter le nombre d'occurrences d'un motif $\mathcal{W}$ donné (sur l'alphabet $\mathcal{A}$).

Pour cela on commence par considérer le langage régulier $\mathcal{L} = \mathcal{A}^*\mathcal{W}$ pour lequel il existe (grâce au théorème 42) un unique plus petit AFD $(\mathcal{A}, \mathcal{Q}, s, \mathcal{F}, \delta)$ le reconnaissant. Par définition, on a donc $\delta(s, x) \in \mathcal{F}$ si et seulement si la séquence x se termine par une occurrence de $\mathcal{W}$ ce qui permet d'obtenir le résultat suivant :

Théorème 43. Pour tout motif $\mathcal{W}$ sur $\mathcal{A}$, et pour tout AFD $(\mathcal{A}, \mathcal{Q}, s, \mathcal{F}, \delta)$ reconnaissant le langage $\mathcal{L} = \mathcal{A}^*\mathcal{W}$ alors, si $x = x_1 \dots x_\ell$ est une séquence sur $\mathcal{A}$, la séquence sur $\mathcal{Q}$ définie par

$$y_0 = s \quad \text{et} \quad y_i = \delta(y_{i-1}, x_i) \quad \text{pour tout} \quad 1 \leqslant i \leqslant \ell$$

a la propriété suivante :

$$\mathcal{W} \text{ se termine en position } i \text{ dans } x \iff y_i \in \mathcal{F}.$$

Il suffit donc de construire (en $O(\ell)$) la séquence y à partir de la séquence x et d'y compter le nombre de fois ou y apparaît des éléments de $\mathcal{F}$ pour obtenir le nombre d'occurrences chevauchantes du motif $\mathcal{W}$.

Exemple 44. On se place sur l'alphabet $\mathcal{A} = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$ et on considère le motif $\mathcal{W} =$ `a..ba`. Le plus petit AFD reconnaissant $\mathcal{L} = \mathcal{A}^*\mathcal{W}$ est donné en figure 4.3. Soit $x =$ `bacabaabbacaaba` une séquence de longueur $\ell = 15$ dans laquelle on cherche à localiser les occurrences de $\mathcal{W}$. On commence par construire la séquence y correspondante tel qu'indiqué dans le théorème 43 et on obtient

pos.	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
x	–	b	a	c	a	b	a	a	b	b	a	c	a	a	b	a
y	0	0	11	5	12	4	**1**	3	10	7	**2**	5	12	3	10	12

On obtient donc deux occurrences du motif dans la séquence x la première se terminant en position 6 (avec l'état 1 de l'automate) et la seconde en position 10 (avec l'état 2 de l'automate).

Si l'on s'intéresse maintenant aux occurrences renouvelantes et non plus aux occurrences chevauchantes, la même méthode peut être utilisée à condition cependant de modifier l'AFD de façon à ce qu'il ne reconnaisse que des séquences se terminant par une occurrence du motif qui ne chevauche pas une précédente occurrence. C'est précisément ce que permet la proposition suivante :

Proposition 45. Si $(\mathcal{A}, \mathcal{Q}, s, \mathcal{F}, \delta)$ est un AFD (minimal ou non) reconnaissant le langage $\mathcal{L} = \mathcal{A}^*\mathcal{W}$, où $\mathcal{W}$ est un motif donné, alors l'automate $(\mathcal{A}, \mathcal{Q}, s, \mathcal{F}, \delta')$ reconnaît les séquences se terminant par une occurrence renouvelante de $\mathcal{W}$, avec $\delta' \equiv \delta$ si ce n'est que $\delta'(f, a) = \delta(s, a)$ pour tout $f \in \mathcal{F}$ et $a \in \mathcal{A}$.

Exemple 46. On se place sur l'alphabet $\mathcal{A} = \{\mathtt{a}, \mathtt{b}\}$ et on considère le motif $\mathcal{W} =$ `bb`. Le plus petit AFD reconnaissant $\mathcal{L} = \mathcal{A}^*\mathcal{W}$ est donné en figure 4.2 et la version reconnaissant les occurrences renouvelantes en figure 4.4 : la fonction de transition δ est presque inchangée, seule la transition $\delta(1, \mathtt{b}) = 1$ a été remplacée par $\delta'(1, \mathtt{b}) = 2$. Soit $x =$ `bbbabbabbbababb` une séquence de longueur $\ell = 15$ et dans laquelle on cherche à localiser les occurrences chevauchantes (resp. renouvelantes) de `bb`. Pour cela on utilise l'automate de la figure 4.2 (resp. 4.4) pour produire la séquence y (resp. y') dans laquelle les occurrences du motif correspondent aux apparitions de l'état 1 (seul état final). on obtient le tableau ci-dessous, qui montre qu'il y a 6 occurrences chevauchantes de `bb` dans x mais seulement 4 occurrences renouvelantes.

pos.	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
x	–	b	b	b	a	b	b	a	b	b	b	a	b	a	b	b
y	0	2	**1**	**1**	0	2	**1**	0	2	**1**	**1**	0	2	0	2	**1**
y'	0	2	**1**	2	0	2	**1**	0	2	**1**	2	0	2	0	2	**1**

4.3 Algorithmes

4.3.1 Construction d'automates

Pour toute expression régulière (c'est à dire pour tout langage régulier comme on l'a vu), il est existe donc un AFD minimum associé. Des algorithmes classiques permettent

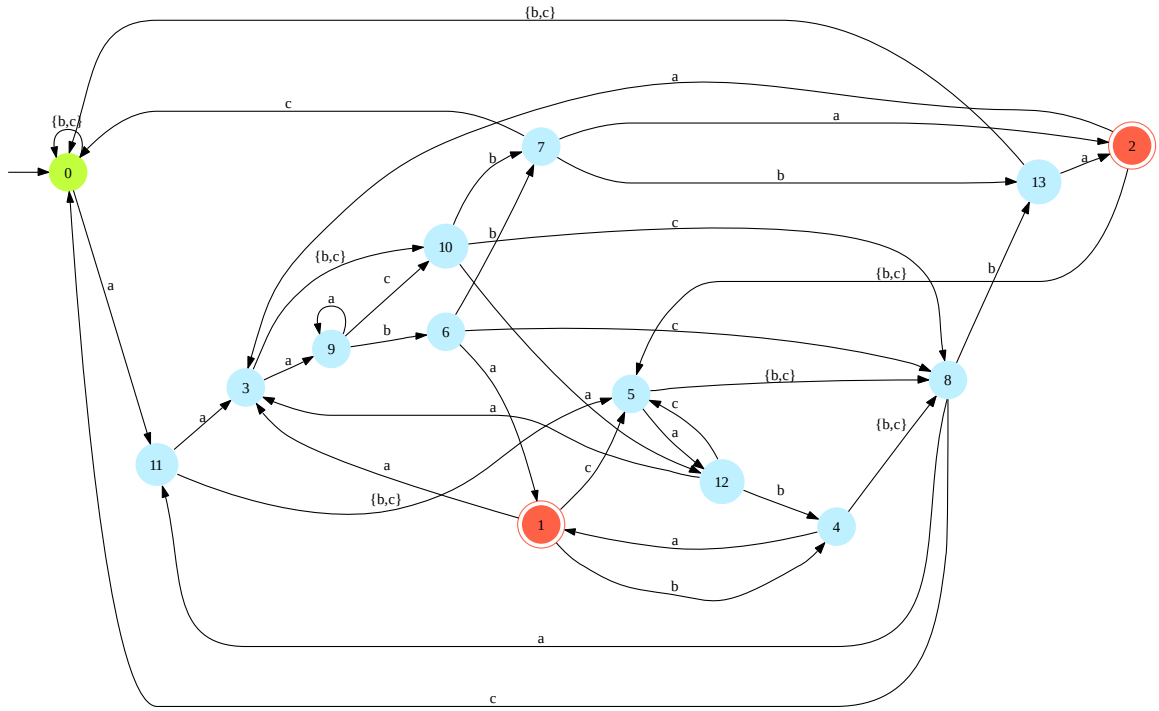


FIG. 4.3 – AFD minimal reconnaissant le langage $\mathcal{L} = \mathcal{A}^*\{\mathtt{a..ba}\}$ avec $\mathcal{A} = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$. On a $\mathcal{Q} = \{0, 1, \ldots, 13\}$ (14 états), $s = 0$, $\mathcal{F} = \{1, 2\}$ et la fonction de transition δ est définie par : $\delta(0, \mathtt{a}) = 11$, $\delta(0, \mathtt{b}) = 0$, $\delta(0, \mathtt{c}) = 0$, $\delta(1, \mathtt{a}) = 3, \ldots$

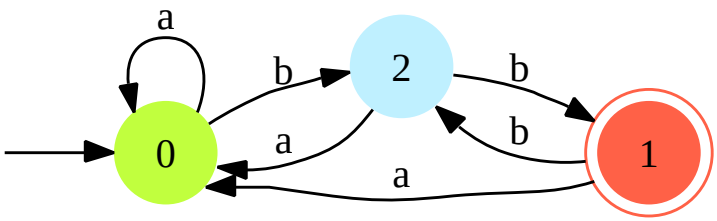


FIG. 4.4 – Automate reconnaissant les séquences sur l'alphabet $\mathcal{A} = \{\mathtt{a}, \mathtt{b}\}$ se terminant par une occurrence non chevauchante de `bb`. Cet automate a été obtenu par application de la proposition 45 à l'automate de la figure 4.2.

d'obtenir cet automate en construisant d'abord un automate fini non déterministe puis d'utiliser un algorithme de déterminisation pour obtenir un AFD qui sera ensuite nettoyé de ses transitions vides (ε-removal) puis minimisé. Le programme FSA implémente l'ensemble de ces méthodes et c'est en grande partie avec cet outil que sont produits les automates de cet ouvrage.

Une telle construction peut néanmoins se révéler très consommatrice de mémoire puisque qu'un espace mémoire maximum de $O(2^Q)$, pour une expression régulière de taille Q, peut être requis au moins dans une étape intermédiaire. Une solution à ce problème est l'utilisation de méthodes de constructions dites "fainéantes" (*lazy construction* en anglais, voir Maout, 2003) dont l'idée est d'utiliser un automate fini non déterministe virtuel pour en déduire l'AFD.

Automate non déterministe

Définition 47 (AFND)**.** On appelle *Automate Fini Non-Déterministe* (AFND) la donnée d'un quintuplet $(\mathcal{A}, \mathcal{Q}, s, \mathcal{F}, \delta)$ où $\mathcal{A}$ est un alphabet fini, $\mathcal{Q}$ un ensemble (fini) d'états, $\mathcal{S} \subset \mathcal{Q}$ un ensemble (non vide) d'états initiaux, $\mathcal{F} \subset \mathcal{Q}$ un ensemble (non vide) d'états terminaux et $\delta : \mathcal{Q} \times \mathcal{A} \cup \varepsilon \to \mathcal{P}(\mathcal{Q})$ une fonction de transition (dans la suite, on se contentera de définir les valeurs non vide de cette fonction de transition). Comme dans le cas des AFD, un mot $w = w_1 \dots w_h$ est reconnu (ou accepté) par l'automate s'il existe un chemin allant de s à un état final utilisant pour les transitions entre états successivement toutes les lettres de w (dans l'ordre) ainsi qu'un nombre quelconque de transitions utilisant ε. Le langage d'un AFND est l'ensemble des mots qu'il reconnaît.

On le voit, la définition d'un AFND est très proche de celle d'un AFD. Il y a seulement deux différences :

- la fonction de transition est à valeurs dans l'ensemble des parties de $\mathcal{Q}$ (rappelons que le vide est bien une partie d'un ensemble) et non plus dans $\mathcal{Q}$ lui-même.
- la fonction de transition est bien sûr définie pour toutes les lettres de $\mathcal{A}$ mais également pour la lettre vide ε.

Remarquons au passage qu'un AFD est un simple cas particulier d'AFND.

On représente graphiquement un AFND sous la forme d'un graphe orienté dont les sommets sont les éléments de $\mathcal{Q}$ et les arêtes sont libellées par des parties de $\mathcal{A} \cup \{\varepsilon\}$. Pour tout $(q, a) \in \mathcal{Q} \times \mathcal{A}$, $\delta(q, a)$ (resp. $\delta(q, \varepsilon)$) est l'ensemble des sommets atteignables depuis q par une arête dont le label contient a (resp. ε).

Proposition 48 (Thompson)**.** Pour tout mot $w = w_1 \dots w_h \in \mathcal{A}^*$, l'AFND $(\mathcal{A}, \mathcal{Q}, s, \mathcal{F}, \delta)$ défini par $\mathcal{Q} = \varepsilon, w_1, w_1w_2, \dots, w_1w_2 \dots w_{h-1}, w$, $s = \varepsilon$ et avec δ définie par $\delta(s, w_1) = \{w_1\}, \delta(w_1, w_2) = \{w_1w_2\}, \dots, \delta(w_1w_2 \dots w_{h-1}, w_h) = \{f\}$, reconnaît le langage $\mathcal{L} = \{w\}$.

De plus, si $A_1 = (\mathcal{A}, \mathcal{Q}_1, s_1, \mathcal{F}_1, \delta_1)$ et $A_2 = (\mathcal{A}, \mathcal{Q}_2, s_2, \mathcal{F}_2, \delta_2)$ sont deux AFND reconnaissant respectivement les langages $\mathcal{L}_1$ et $\mathcal{L}_2$ alors :

addition L'AFND $(\mathcal{A}, \mathcal{Q}, s, \mathcal{F}, \delta)$ défini par $\mathcal{Q} = \{s, f\} \cup \mathcal{Q}_1 \cup \mathcal{Q}_2$, $\mathcal{F} = \{f\}$ et δ définie par $\delta(s, \varepsilon) = \{s_1, s_2\}$ et, pour $i = 1$ ou 2, par $\delta \equiv \delta_i$ sur $\mathcal{Q}_i \times \mathcal{A} \cup \{\varepsilon\}$ à l'exception près que $\delta(f_i, \varepsilon) = \delta_i(f_i, \varepsilon) \cup \{f\}$ pour tout $f_i \in \mathcal{F}_i$, reconnaît le langage $\mathcal{L}_1 + \mathcal{L}_2$.

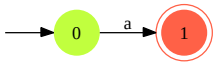
produit L'AFND $(\mathcal{A}, \mathcal{Q}, s, \mathcal{F}, \delta)$ défini par $Q = \mathcal{Q}_1 \cup \mathcal{Q}_2$, $s = s_1$, $\mathcal{F} = \mathcal{F}_2$ et δ définie pour $i = 1$ ou 2 par $\delta \equiv \delta_i$ sur $\mathcal{Q}_i \times \mathcal{A} \cup \{\varepsilon\}$ à l'exception près que $\delta(f_1, \varepsilon) = \delta_1(f_1, \varepsilon) \cup \{s_2\}$, reconnaît le langage $\mathcal{L}_1 \cdot \mathcal{L}_2$.

clôture de Kleene L'AFND $(\mathcal{A}, \mathcal{Q}, s, \mathcal{F}, \delta)$ défini par $Q = \mathcal{Q}_1 \cup \{f\}$, $s = s_1$, $\mathcal{F} = \{f\}$ et δ définie par $\delta \equiv \delta_1$ sur $\mathcal{Q}_1 \times \mathcal{A} \cup \{\varepsilon\}$ à l'exception près que $\delta(s_1, \varepsilon) = \delta_1(s_1, \varepsilon) \cup \{f\}$ et que $\delta(f_1, \varepsilon) = \delta_1(f_1, \varepsilon) \cup \{s_1, f\}$ pour tout $f_1 \in \mathcal{F}_1$, reconnaît le langage $\mathcal{L}_1^*$.

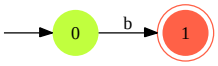
Démonstration. Bien que ce résultat puisse paraître complexe, les constructions qu'il propose sont en fait aussi naturelles qu'élémentaires. Le lecteur pourra aisément s'en convaincre en considérant l'exemple 49. □

Exemple 49. Sur l'alphabet binaire $\mathcal{A} = \{\mathtt{a}, \mathtt{b}\}$ voici quelques exemples des AFND produits par la proposition 48 pour différents langages :

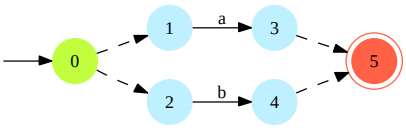
$\mathcal{L}_1 = \{\mathtt{a}\}$ donne



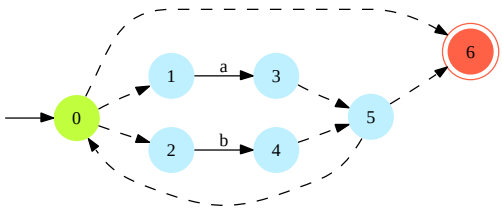
$\mathcal{L}_2 = \{\mathtt{b}\}$ donne



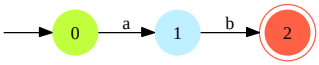
$\mathcal{L}_3 = \mathcal{L}_1 + \mathcal{L}_2 = \mathcal{A}$ donne



$\mathcal{L}_4 = \mathcal{L}_3^* = \mathcal{A}^*$ donne



$\mathcal{L}_5 = \{\mathtt{ab}\}$ donne



$\mathcal{L}_6 = \mathcal{L}_4 \cdot \mathcal{L}_1 \cdot \mathcal{L}_3 \cdot \mathcal{L}_5 = \mathcal{A}^*\{\mathtt{aaab}, \mathtt{abab}\}$

En concaténant les automates précédemment construits on obtient celui de la figure 4.5 :

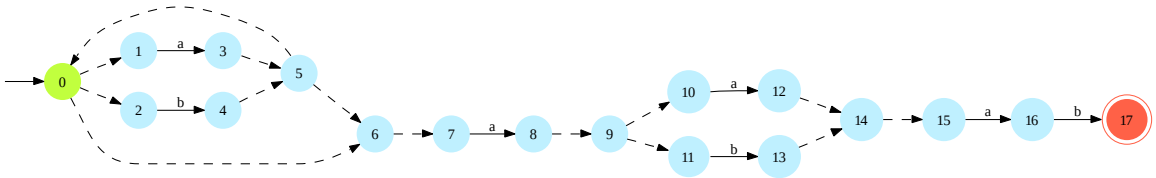
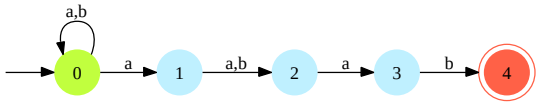


FIG. 4.5 – Représentation graphique d'un AFND reconnaissant $\{\mathtt{a}, \mathtt{b}\}^*\{\mathtt{aaab}, \mathtt{abab}\}$. Les $\varepsilon-$transitions sont en pointillés.

Remarque 50. A noter que si la proposition 48 permet systématiquement de construire un AFND à partir d'une expression régulière, il est souvent possible de réduire sensiblement la taille de cet automate en remplaçant de nombreuses $\varepsilon-$transitions par de simples transitions. Ainsi, l'AFND



reconnaît rigoureusement le même langage que celui de la figure 4.5. De plus, cet automate plus compact est finalement beaucoup plus naturel que le premier.

Corollaire 51. Pour tout langage régulier il existe un AFND reconnaissant ce langage.

Démonstration. D'après définition 38, tout langage régulier sur $\mathcal{A}$ peut être construit à partir des parties finies de $\mathcal{A}^*$ et d'un nombre fini d'opérations régulières (addition, produit et clôture de Kleene). La proposition 48 donnant une construction explicite pour l'ensemble de ces opérations nous fournit donc une preuve constructive. □

Déterminisation

Définition 52 ($\varepsilon-$clôture)**.** Soit $(\mathcal{A}, \mathcal{Q}, s, \mathcal{F}, \delta)$ un AFND, pour tout $\mathcal{S} \subset \mathcal{Q}$ on définit $\varepsilon-$clôture($\mathcal{S}$) comme le plus petit ensemble contenant $\mathcal{S}$ et étant stable par $\varepsilon-$transitions (c'est à dire que $\delta(\varepsilon-\text{clôture}(\mathcal{S}), \varepsilon) = \varepsilon-\text{clôture}(\mathcal{S})$). Plus simplement, il s'agit du sous ensemble dans $\mathcal{Q}$ des états joignables par un nombre quelconque de $\varepsilon-$transitions depuis $\mathcal{S}$.

Proposition 53. Pour tout AFND $(\mathcal{A}, \mathcal{Q}_1, s_1, \mathcal{F}_1, \delta_1)$ reconnaissant un langage régulier donné il existe un AFD $(\mathcal{A}, \mathcal{Q}_2, s_2, \mathcal{F}_2, \delta_2)$ reconnaissant le même langage. De plus $|\mathcal{Q}_2| \leqslant 2^{|\mathcal{Q}_1|}$.

1: $q_0 = \varepsilon-\text{clôture}(s_1) \subset \mathcal{Q}_1$, $\mathcal{Q}_2 = \{q_0\}$, $L_2 = 1$ (cardinal de $\mathcal{Q}_2$) et $i = 0$
2: **while** $i < L_2$ **do**
3: **for all** $a \in \mathcal{A}$ **do**
4: $q = \varepsilon-\text{clôture}\big(\delta_1(q_i, a)\big) \subset \mathcal{Q}_1$
5: **if** $q \notin \mathcal{Q}_2$ **then**
6: ajouter $q_{L_2} = q$ à $\mathcal{Q}_2$ et faire $L_2 = L_2 + 1$
7: $\delta_2(q_i, a) = q$
8: $i = i + 1$
9: $\mathcal{F}_2 = \{q \in \mathcal{Q}_2, q \cap \mathcal{F}_1 \neq \emptyset\}$

ALGORITHME **1:** Déterminisation d'un AFND ; construit à partir de l'AFND $(\mathcal{A}, \mathcal{Q}_1, s_1, \mathcal{F}_1, \delta_1)$ un AFD $(\mathcal{A}, \mathcal{Q}_2, s_2, \mathcal{F}_2, \delta_2)$ reconnaissant le même langage. Complexités (dans le pire des cas) : $O(2^{L_1})$ en espace et $O(2^{L_1})$ en temps où L_1 est le cardinal de $\mathcal{Q}_1$.

Démonstration. La preuve de ce résultat est constructive et utilise l'algorithme 1 dit de déterminisation (également appelé *subset construction* en anglais). $\mathcal{Q}_2$ étant alors, comme on va le voir, un sous-ensemble des parties de $\mathcal{Q}_1$, la majoration est évidente. □

de	vers	$\varepsilon-$clôture(vers)	transition
na	$\{0\}$	$q_0 = \{0,1,2,6,7\}$	na
q_0	$\delta_1(q_0, \mathtt{a}) = \{3,8\}$	$q_1 = \{0,1,2,3,5,7,8,9,10,11\}$	$\delta_2(q_0, \mathtt{a}) = q_1$
q_0	$\delta_1(q_0, \mathtt{b}) = \{4\}$	$q_2 = \{0,1,2,4,5,6,7\}$	$\delta_2(q_0, \mathtt{b}) = q_2$
q_1	$\delta_1(q_1, \mathtt{a}) = \{3,8,12\}$	$q_3 = q_1 \cup \{12,14,15\}$	$\delta_2(q_1, \mathtt{a}) = q_3$
q_1	$\delta_1(q_1, \mathtt{b}) = \{4,13\}$	$q_4 = q_2 \cup \{13,14,15\}$	$\delta_2(q_1, \mathtt{b}) = q_4$
q_2	$\delta_1(q_2, \mathtt{a}) = \{3,8\}$	q_1	$\delta_2(q_2, \mathtt{a}) = q_1$
q_2	$\delta_1(q_2, \mathtt{b}) = \{4\}$	q_2	$\delta_2(q_2, \mathtt{b}) = q_2$
q_3	$\delta_1(q_3, \mathtt{a}) = \{3,8,12,16\}$	$q_5 = q_3 \cup \{16\}$	$\delta_2(q_3, \mathtt{a}) = q_5$
q_3	$\delta_1(q_3, \mathtt{b}) = \{4,13\}$	q_4	$\delta_2(q_3, \mathtt{b}) = q_4$
q_4	$\delta_1(q_4, \mathtt{a}) = \{3,8,16\}$	$q_6 = q_1 \cup \{16\}$	$\delta_2(q_4, \mathtt{a}) = q_6$
q_4	$\delta_1(q_4, \mathtt{b}) = \{4\}$	q_2	$\delta_2(q_4, \mathtt{b}) = q_2$
q_5	$\delta_1(q_5, \mathtt{a}) = \{3,8,12,16\}$	q_5	$\delta_2(q_5, \mathtt{a}) = q_5$
q_5	$\delta_1(q_3, \mathtt{b}) = \{4,13,17\}$	$q_7 = q_4 \cup \{17\}$	$\delta_2(q_5, \mathtt{b}) = q_7$
q_6	$\delta_1(q_6, \mathtt{a}) = \{3,8,12\}$	q_3	$\delta_2(q_6, \mathtt{a}) = q_3$
q_6	$\delta_1(q_6, \mathtt{b}) = \{4,13,17\}$	q_7	$\delta_2(q_6, \mathtt{b}) = q_7$
q_7	$\delta_1(q_7, \mathtt{a}) = \{3,8,16\}$	q_6	$\delta_2(q_7, \mathtt{a}) = q_6$
q_7	$\delta_1(q_7, \mathtt{b}) = \{4\}$	q_2	$\delta_2(q_7, \mathtt{b}) = q_2$

TAB. 4.1 – Déterminisation de l'AFND de la figure 4.5.

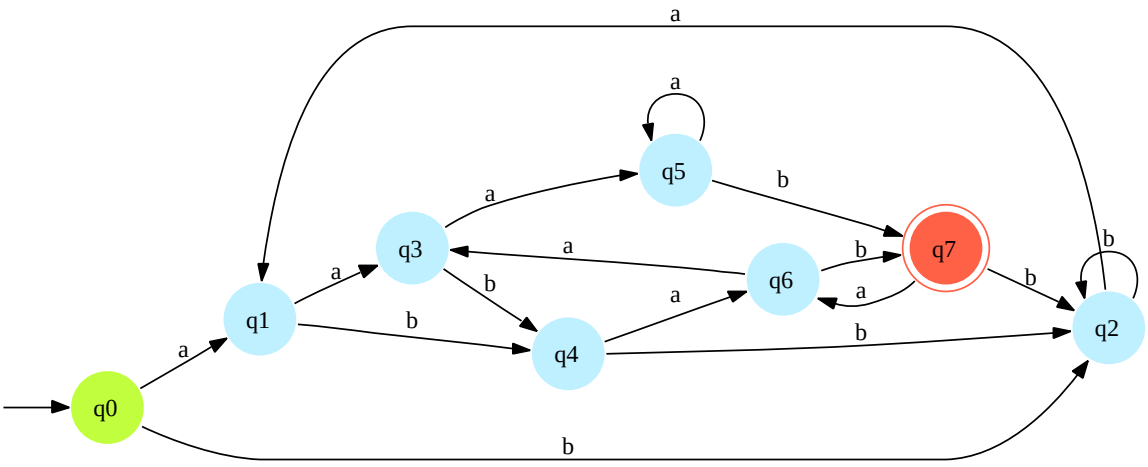


FIG. 4.6 – Représentation graphique de l'AFD de l'exemple 54

Exemple 54. Utilisons l'algorithme 1 pour déterminiser l'AFND de la figure 4.5. On peut voir sur la table 4.1 les différentes itérations de l'algorithme. Comme $\mathcal{F}_1 = \{17\}$, on a finalement $\mathcal{F}_2 = \{q_7\}$. Voir la figure 4.6 pour une représentation graphique de l'AFD obtenu.

S'il est assez simple d'énoncer et de comprendre l'algorithme de minimisation, son implémentation (efficace) peut se révéler délicate. En effet, l'essentiel de l'algorithme est consacré à deux tâches élémentaires : calcul d'image de sous-ensembles d'états par la fonction δ_1 et comparaison de sous-ensembles d'états. Il existe une multitude de stratégies permettant d'optimiser l'exécution de ces deux tâches. Pour la première on peut par

exemple calculer et stocker en mémoire les images des différents états de $\mathcal{Q}_1$ puis se contenter par la suite de faire des unions de ces ensembles. Pour la seconde, on peut penser à garder trace pour chaque sous-ensemble de son nombre d'éléments et de ne réellement comparer deux sous-ensembles que s'ils sont de même taille. Chacune de ces optimisations est plus ou moins bien adaptée au problème en fonction des caractéristiques de l'AFND considéré (nombre moyen d'antécédents par état, nombre d'états, etc ...) et du système sur lequel sont effectués les calculs (voir Leslie, 1995, pour une discussion détaillée sur le sujet).

Minimisation

L'AFD obtenu via l'algorithme de déterminisation n'étant pas nécessairement minimal, il convient de le minimiser. Pour cela, (Hopcroft, 1971) propose un algorithme en $O(n \log n)$ où n est le nombre d'états de l'AFD à minimiser (voir algorithme 2).

1: $p_0 = \mathcal{F}$, $p_1 = \mathcal{Q} \setminus \mathcal{F}$ et $\mathcal{P} = \{p_0, p_1\}$ une partition de $\mathcal{Q}$
2: $n = 2$ (cardinal de $\mathcal{P}$) et $i = 0$
3: **while** $i < n$ **do**
4: **for all** $a \in \mathcal{A}$ **do**
5: $\mathcal{I}_a = \{q \in \mathcal{Q}, \delta(q, a) \in p_i\}$
6: **for all** $p \in \mathcal{P}$ tel que $p \cap \mathcal{I}_a \neq \emptyset$ et $p \cap \mathcal{I}_a \neq p$ **do**
7: // *couper p en deux*
8: $p_n = p \setminus (p \cap \mathcal{I}_a)$
9: $p = (p \cap \mathcal{I}_a)$
10: $n = n + 1$
11: $i = i + 1$
12: s' est l'unique élément de $\mathcal{P}$ contenant s
13: $\mathcal{F}'$ est l'ensemble des éléments de $\mathcal{P}$ qui rencontrent $\mathcal{F}$
14: δ' est le prolongement ensembliste de δ

ALGORITHME **2:** Minimisation d'un AFD ; construit à partir d'un AFD $(\mathcal{A}, \mathcal{Q}, s, \mathcal{F}, \delta)$ un AFD *minimal* $(\mathcal{A}, \mathcal{P}, s', \mathcal{F}', \delta')$ reconnaissant le même langage. Complexités : $O(n)$ en espace et $O(n \log n)$ en temps.

Exemple 55. Minimisons l'AFD de la figure 4.6. On commence par poser $p_0 = \{q_7\}$ et $p_1 = \{q_0, q_1, q_2, q_3, q_4, q_5, q_6\}$. La table 4.2 donne le détail des étapes du partitionnement. On obtient à la fin $\mathcal{P} = \{p_0 = \{q_7\}, p_1 = \{q_0, q_2\}, p_2 = \{q_5\}, p_3 = \{q_3\}, p_4 = \{q_6\}, p_5 = \{q_4\}, p_6 = \{q_1\}\}$. Seuls les états q_0 et q_2 sont finalement fusionnés en un seul et on obtient l'AFD minimal de la figure 4.7.

Heuristique

Si les algorithmes vus précédemment sont (relativement) simples, leurs implémentations efficaces restent assez délicates et il peut donc être intéressant de disposer d'une heuristique de construction plus simple pour les langages de la forme $\mathcal{A}^*\mathcal{W}$ (qui correspondent à la présence du motif $\mathcal{W}$ "quelque part" dans une séquence). Nous présentons ici une telle heuristique qui coïncide avec la solution optimale dans le cas de motifs non dégénérés (et n'est donc pas dans ce cas particulier une heuristique) mais dont la performance décroît rapidement avec la dégénérescence des motifs (voir table 4.3).

n	p_i	$\mathcal{I}_a$	changements
2	$p_0 = \{q_7\}$	$\mathcal{I}_{\mathtt{a}} = \emptyset$	aucun
2	$p_0 = \{q_7\}$	$\mathcal{I}_{\mathtt{b}} = \{q_5, q_6\}$	$p_1 = \{q_0, q_1, q_2, q_3, q_4\}, p_2 = \{q_5, q_6\}$
3	$p_1 = \{q_0, q_1, q_2, q_3, q_4\}$	$\mathcal{I}_{\mathtt{a}} = \{q_3, q_4, q_5, q_7\}$	$p_1 = \{q_0, q_1, q_2\}, p_3 = \{q_3, q_4\}$
4			$p_2 = \{q_5\}, p_4 = \{q_6\}$
5	$p_1 = \{q_0, q_1, q_2\}$	$\mathcal{I}_{\mathtt{b}} = \emptyset$	aucun
5	$p_2 = \{q_5, q_6\}$	$\mathcal{I}_{\mathtt{a}} = \{q_3, q_5\}$	$p_3 = \{q_3\}, p_5 = \{q_4\}$
6	$p_2 = \{q_5, q_6\}$	$\mathcal{I}_{\mathtt{b}} = \emptyset$	aucun
6	$p_3 = \{q_3\}$	$\mathcal{I}_{\mathtt{a}} = \emptyset$	aucun
6	$p_3 = \{q_3\}$	$\mathcal{I}_{\mathtt{b}} = \{q_1, q_3\}$	$p_1 = \{q_0, q_2\}, p_6 = \{q_1\}$
7	$p_4 = \{q_6\}$	$\mathcal{I}_{\mathtt{a}} = \{q_1, q_6\}$	aucun
7	$p_4 = \{q_6\}$	$\mathcal{I}_{\mathtt{b}} = \emptyset$	aucun
7	$p_5 = \{q_4\}$	$\mathcal{I}_{\mathtt{a}} = \{q_0, q_2\}$	aucun
7	$p_5 = \{q_4\}$	$\mathcal{I}_{\mathtt{b}} = \emptyset$	aucun
7	$p_6 = \{q_1\}$	$\mathcal{I}_{\mathtt{a}} = \emptyset$	aucun
7	$p_6 = \{q_1\}$	$\mathcal{I}_{\mathtt{b}} = \{q_0, q_2, q_4, q_7\}$	aucun

TAB. 4.2 – Minimisation de l'AFD de la figure 4.6.

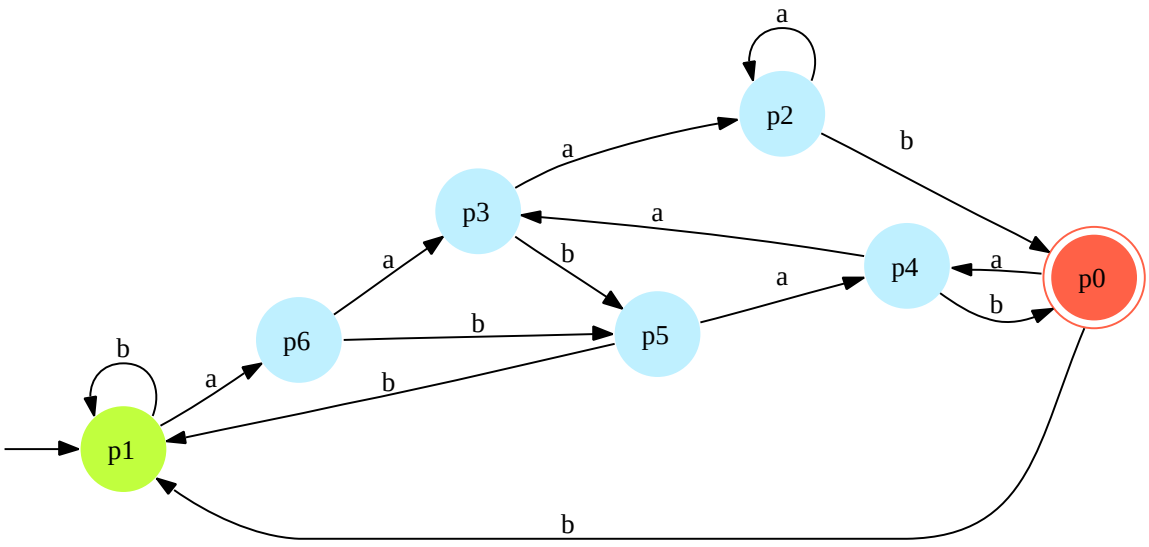


FIG. 4.7 – Représentation graphique de l'AFD minimal de l'exemple 55.

Proposition 56 (heuristique de construction d'AFD)**.** Soit $\mathcal{W}$ un motif quelconque sur l'alphabet $\mathcal{A}$. On considère $\mathcal{Q}$ l'ensemble de tous les préfixes de $\mathcal{W}$ (y compris le préfixe nul ε), on pose $s = \varepsilon$, $\mathcal{F} = \mathcal{W} \subset \mathcal{Q}$ et pour tout $q\mathcal{Q}$ et tout $a\mathcal{A}$ on pose

$$\delta(q, a) = \text{le plus long suffixe de } qa \text{ (concaténation de } q \text{ et } a) \text{ dans } \mathcal{Q}$$

l'AFD $(\mathcal{A}, \mathcal{Q}, s, \mathcal{F}, \delta)$ ainsi construit reconnaît le langage $\mathcal{A}^*\mathcal{W}$ et est minimum (de taille $h + 1$) dans le cas où $\mathcal{W}$ est réduit à un unique mot (de longueur h).

Exemple 57. On se place sur l'alphabet $\mathcal{A} = \{\mathtt{a}, \mathtt{b}\}$ et on considère le motif $\mathcal{W} = \{\mathtt{bb}\}$. En appliquant la proposition 56 on pose $\mathcal{Q} = \{\varepsilon, \mathtt{b}, \mathtt{bb}\}$, $s = \varepsilon$ et on a : $\delta(\varepsilon, \mathtt{a}) = \varepsilon$ (le plus long suffixe de $\mathtt{a}$ appartenant à $\mathcal{Q}$ est en effet bien ε), $\delta(\varepsilon, \mathtt{b}) = \mathtt{b}$, $\delta(\mathtt{b}, \mathtt{a}) = \varepsilon$, $\delta(\mathtt{b}, \mathtt{b}) = \mathtt{bb}$, $\delta(\mathtt{bb}, \mathtt{a}) = \varepsilon$ et $\delta(\mathtt{bb}, \mathtt{b}) = \mathtt{bb}$. On retrouve ainsi exactement l'automate de l'exemple 41 avec les correspondances $0 = \varepsilon$, $1 = \mathtt{bb}$ et $2 = \mathtt{b}$.

Exemple 58. Toujours dans l'alphabet binaire $\mathcal{A} = \{\mathtt{a}, \mathtt{b}\}$, on considère maintenant le motif $\mathcal{W} = \{\mathtt{aaab}, \mathtt{abab}\}$. On se place sur l'espace d'état à 8 éléments

$$\mathcal{Q} = \{\varepsilon, \mathtt{a}, \mathtt{aa}, \mathtt{ab}, \mathtt{aaa}, \mathtt{aba}, \mathtt{aaab}, \mathtt{abab}\},$$

on prend $s = \varepsilon$ pour état initial et $\mathcal{F} = \{\mathtt{aaab}, \mathtt{abab}\}$ pour ensemble d'états finaux.

Au delà des transitions évidentes (ex : $\delta(\mathtt{aa}, \mathtt{b}) = \mathtt{aab}$) il ne reste qu'un petit nombre de transitions à définir. On se contente ici de détailler quelques exemples :

$\delta(\mathtt{ab}, \mathtt{b})$ **:** comme $\mathtt{abb}, \mathtt{bb}, \mathtt{b} \notin \mathcal{Q}$, le seul suffixe de $\mathtt{abb}$ appartenant à $\mathcal{Q}$ est ε, si bien que $\delta(\mathtt{ab}, \mathtt{b}) = \varepsilon$;

$\delta(\mathtt{aba}, \mathtt{a})$ **:** $\mathtt{abaa}, \mathtt{baa} \notin \mathcal{Q}$ mais $\mathtt{aa} \in \mathcal{Q}$, donc $\delta(\mathtt{aba}, \mathtt{a}) = \mathtt{aa}$;

$\delta(\mathtt{aaab}, \mathtt{a})$ **:** $\mathtt{aaaba}, \mathtt{aaba} \notin \mathcal{Q}$ mais $\mathtt{aba} \in \mathcal{Q}$ et donc $\delta(\mathtt{aaab}, \mathtt{a}) = \mathtt{aba}$.

La figure 4.8 donne une représentation graphique de l'automate à 8 états résultant de l'utilisation de l'heuristique. Cette automate est non optimal (l'AFD minimal a 7 états, voir figure 4.7) et a autant d'états que l'automate obtenu par déterminisation (voir figure 4.6) mais est bien différent (nous avons ici deux états finaux, alors que l'automate de la figure 4.6 n'en a qu'un seul).

Il est facile de voir que l'algorithme de minimisation appliqué à l'automate conduit à la fusion des deux états finaux $\mathtt{aaab}$ et $\mathtt{abab}$, on obtient alors bien le même automate qu'en figure 4.7 avec la correspondance suivante : $\{q_0, q_2\} = \varepsilon$, $q_1 = \mathtt{a}$, $q_3 = \mathtt{aa}$, $q_4 = \mathtt{ab}$, $q_5 = \mathtt{aaa}$, $q_6 = \mathtt{aba}$ et $q_7 = \{\mathtt{aaab}, \mathtt{abab}\}$.

Exemple 59. Si on considère maintenant le motif $\mathcal{W} = \{\mathtt{a..ba}\}$ sur l'alphabet $\mathcal{A} = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$ comme dans la figure 4.3, la même proposition donne un automate dont la taille est le nombre de préfixes distincts du motifs : 32. Dans ce cas, on est donc loin de la taille optimale : 13.

4.3.2 Arbres de suffixes

Lorsque l'on s'intéresse à un motif particulier, il est donc possible, après construction de l'AFD correspondant, de compter (et de localiser) en $O(\ell)$ toutes les occurrences de ce motif dans une séquence de longueur ℓ.

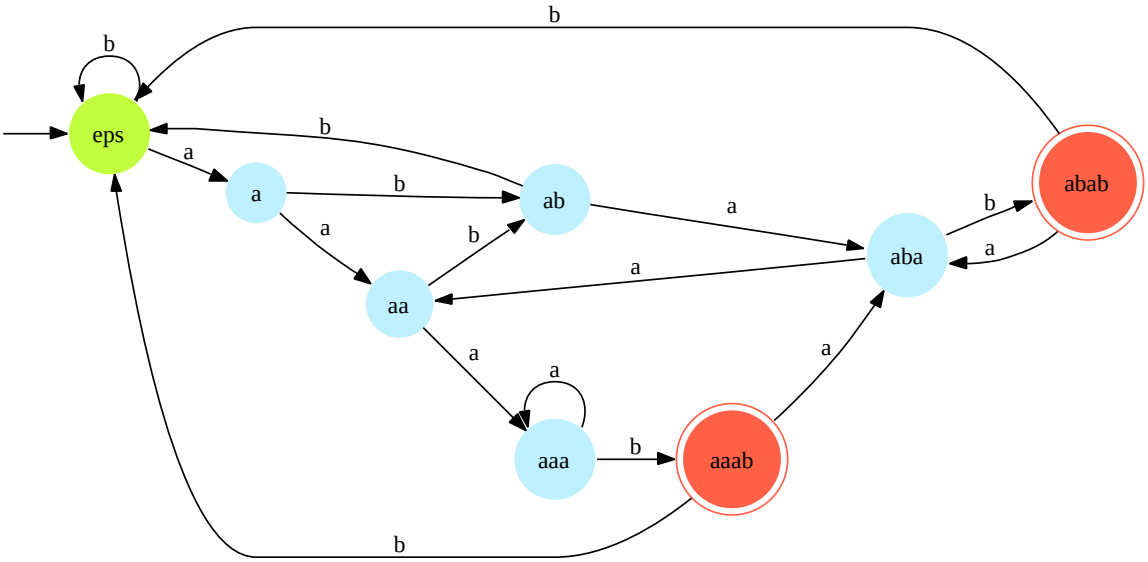


FIG. 4.8 – Représentation graphique de l'AFD de l'exemple 58.

motif	optimal	heuristique
$\{\mathtt{a},\mathtt{b}\}^*\{\mathtt{bb}\}$	3	3
$\{\mathtt{a},\mathtt{b}\}^*\{\mathtt{aba},\mathtt{bab}\}$	7	7
$\{\mathtt{a},\mathtt{b}\}^*\{\mathtt{aba},\mathtt{bba}\}$	4	7
$\{\mathtt{a},\mathtt{b}\}^*\{\mathtt{aaab},\mathtt{abab}\}$	7	8
$\{\mathtt{a},\mathtt{c},\mathtt{g},\mathtt{t}\}^*\{\mathtt{gc.tggtgg}\}$	10	30
$\{\mathtt{a},\mathtt{b},\mathtt{c}\}^*\{\mathtt{a..ba}\}$	13	32
$\{\mathtt{a},\mathtt{b}\}^*\{\mathtt{aba.....bab}\}$	81	158
$\{\mathtt{a},\mathtt{b}\}^*\{\mathtt{aba..........bab}\}$	898	5122

TAB. 4.3 – Taille de l'AFD minimal et taille de l'AFD produit par l'heuristique pour quelques motifs

séquence	a	c	t	g	a	t	#	c	g	g	c	g	a	c	t
position	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14

séquence	t	a	t	c	t	t	a	c	t	g	t	g	c	t	a
position	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29

FIG. 4.9 – Une courte séquence d'ADN composée de deux fragments. Le premier, de longueur 6 et le second de longueur 23. Le caractère '#' est utilisé comme séparateur et les positions sont numérotées à partir de 0.

Si on est amené à regarder un grand nombre de motifs (par exemple tous les mots d'une longueur donnée) il est cependant peu efficace d'utiliser cette approche (elle nécessite la construction et le stockage d'un grand nombre d'automates et/ou de multiples lectures de la même séquence).

Dans ce paragraphe, on considère un texte $T = t_0 \dots t_{\ell-1}$ de longueur ℓ sur l'alphabet (fini) $\mathcal{A}$ (T sera une séquence ou, nous allons le voir, la concaténation de plusieurs séquences). On introduit alors l'ensemble

$$\mathcal{S}(T) = \{T_i = t_i \dots t_{\ell-1}\varepsilon, 0 \leqslant i \leqslant \ell - 1\}$$

des $\ell + 1$ suffixes (incluant $T_\ell = \varepsilon$, le suffixe vide) de T. Comme T_0 est de taille $\ell + 1$, T_1 de taille ℓ, et ainsi de suite, il est clair de $\mathcal{S}(T)$ requiert un espace quadratique avec ℓ. Nous allons voir comment représenter ce même ensemble de manière linéaire au travers d'un *arbre de suffixes* dont nous expliquerons les propriétés et verrons également quelques applications (en particulier en ce qui concerne le comptage de mots).

Définition 60 (arbre de suffixes)**.** On appelle arbre de suffixes (*suffix tree* en anglais) de T, l'arbre $Stree(T)$ dont les branches sont étiquetés par des éléments de $\mathcal{A}^*$ (c'est à dire des textes – éventuellement vides–), dont les feuilles sont $F_0, \dots, F_\ell$, et qui a la propriété suivante : la concaténation $\mathcal{C}(root, F_p)$ des étiquettes rencontrées sur tout chemin allant de la racine (notée *root*) à la feuille F_p, est égale à T_p (le suffixe de T partant de la position p).

Théorème 61 (admis)**.** Il est possible de construire $Stree(T)$ pour un texte de longueur ℓ avec une complexité $O(\ell)$ en espace et en temps.

Remarque 62. Même si l'ordre de grandeur de cette complexité est linéaire avec ℓ, notons que que le coût mémoire (resp. temps) de la construction d'un arbre de suffixes est environ cinquante fois supérieure au coût de stockage (resp. temps de lecture) de la séquence initiale. Il faut également signaler que la compréhension et l'implémentation des algorithmes permettant cette construction (voir fin de chapitre pour les référence précises) est assez délicate et que le recours à une librairie de programmation dédiée comme (libstree) est donc le plus souvent nécessaire.

Exemple 63. La figure 4.9 représente le texte T de longueur 30 formé par la concaténation de deux séquences d'ADN séparées par le caractère '#'. La figure 4.10 représente $Stree(T)$, l'arbre de suffixe de ce texte.

Considérons par exemple le chemin allant de la racine de l'arbre (à l'extrême gauche) jusqu'à la feuille F_{26}. En parcourant ce chemin, on rencontre successivement g, c, et enfin

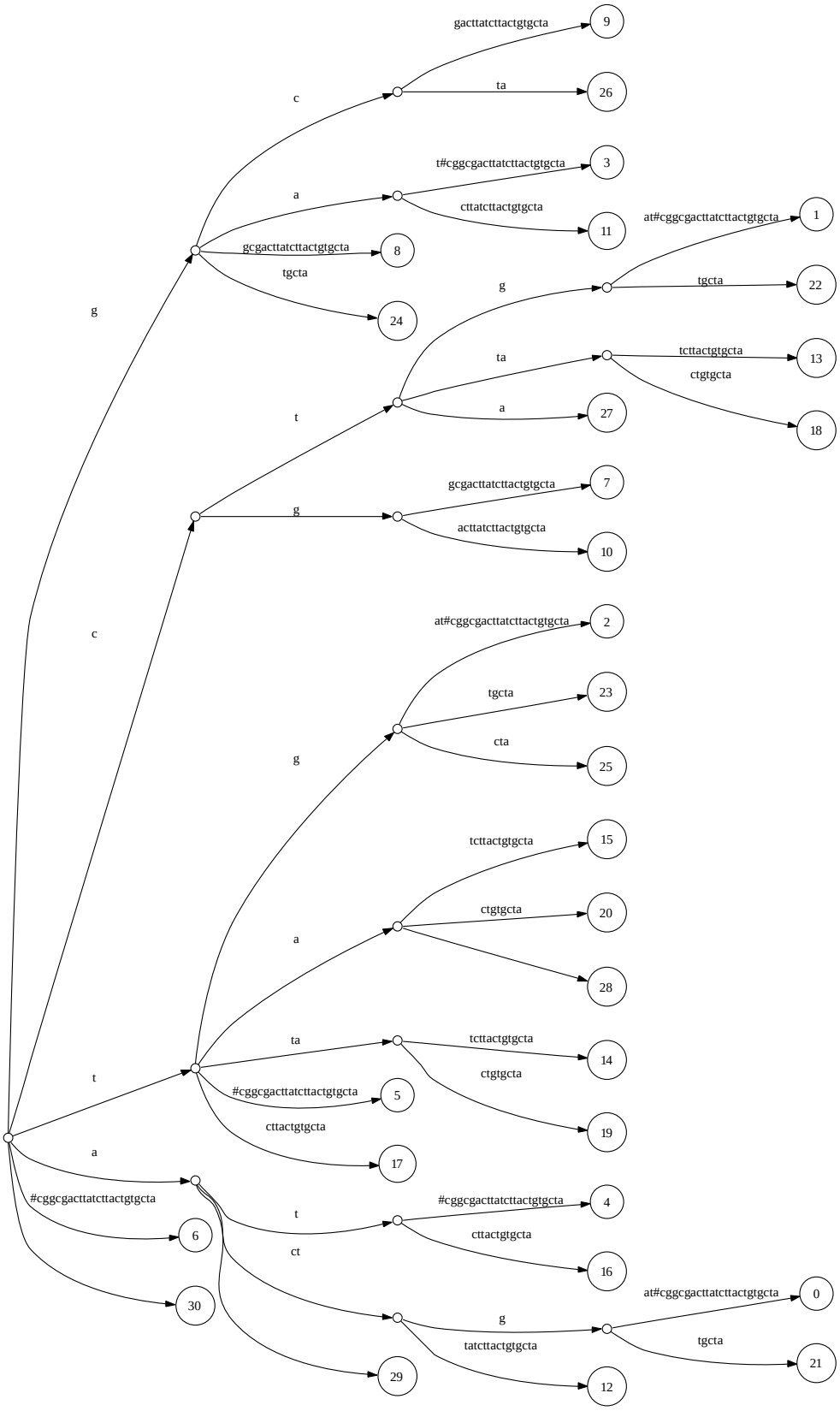


FIG. 4.10 – Arbre de suffixe du texte donnée en figure 4.9. L'arbre est enraciné à gauche.

ta dont la concaténation donne $\mathcal{C}(root, F_{26}) =$ gcta qui est bien égal, comme on le vérifie aisément, à T_{26}, le suffixe partant de la position 26.

Proposition 64. Soit T un texte de longueur ℓ et w un mot de longueur h, alors $\{p, \mathcal{S}_T(p)$ commence par $w\}$ est l'ensemble des positions auxquelles débute une occurrence de w dans T. Si $Stree(T)$ est disponible, il est alors possible de localiser les n occurrences de w en $O(h+n)$.

Démonstration. La complexité $O(h)$ est nécessaire pour trouver le premier noeud interne de $Stree(T)$ correspondant w et $O(n)$ est alors nécessaire pour trouver les n feuilles du sous-arbre partant de ce noeud. □

Exemple 65. Cherchons les occurrences de $w =$ a dans le texte de la figure 4.9. Partant de la racine de l'arbre de la figure 4.10, il suffit d'emprunter l'arête dont l'étiquette commence par a, puis de recenser les feuilles du sous-arbre ainsi délimité. On trouve 6 occurrences de a aux positions (non ordonnées) : 4, 16, 0, 21, 12 et 29.

Considérons maintenant le mot $w =$ ctt. Partant de la racine, on emprunte les arêtes étiquetées c, t puis enfin ta (qui commence bien par t) et on trouve ainsi deux occurrences aux positions 13 et 18. Notons au passage que ces positions correspondent également aux occurrences du mot $w =$ ctta (concaténation des étiquettes des 3 arêtes rencontrées).

Proposition 66 (admise)**.** Quelles que soient F_p et F_q deux feuilles de $Stree(T)$, on note PPAC(F_p, F_q) le Plus Proche Ancêtre Commun de F_p et F_q et on a alors la propriété suivante : $\mathcal{C}\big(root, \text{PPAC}(F_p, F_q)\big)$ est le plus long préfixe commun à T_p et T_q.

Exemple 67. Considérons les feuilles F_0 et F_{12} de l'arbre de la figure 4.10. On trouve $\mathcal{C}\big(root, \text{PPAC}(F_0, F_{12})\big) =$ act qui, comme on le vérifie dans la figure 4.9, est bien la plus longue sous-séquence commune partant de ces deux positions (en effet $t_0t_1t_2 =$ act $= t_{12}t_{13}t_{14}$ et $t_3 =$ g $\neq$ t $= t_{15}$).

Corollaire 68. Soit w un mot de taille h, x une séquence de taille ℓ. Il est possible de localiser (et compter) toutes les occurrences de w avec *au plus r erreurs* dans x en $O(\ell + h + \ell \times r)$.

Démonstration. Voir l'algorithme 3. A noter que la détermination du PPAC de deux feuilles peut être obtenue en temps constant à l'aide d'un calcul préliminaire (non détaillé). □

Exemple 69. Recherchons les occurrences de $w =$ actgat (mot de taille $h = 6$) dans

$$x = \texttt{cggcgacttatcttactgtgcta}$$

(séquence de taille $\ell = 23$) avec au plus $r = 1$ erreur. Comme $T = w\#x$ est justement le texte de la figure 4.9, $Stree(w\#x)$ est l'arbre de la figure 4.10.

- Commençons en $i = 1$:
  - $\mathcal{C}\big(root, \text{PPAC}(F_0, F_7)\big) = \varepsilon$ donc $n = 1$ et $j = 1$;
  - $\mathcal{C}\big(root, \text{PPAC}(F_1, F_8)\big) = \varepsilon$ donc $n = 2$ et $j = 2$;
  - $n > r$ donc pas d'occurrence de w en $i = 1$.
- En $i = 2$:
  - $\mathcal{C}\big(root, \text{PPAC}(F_0, F_8)\big) = \varepsilon$ donc $n = 1$ et $j = 1$;
  - $\mathcal{C}\big(root, \text{PPAC}(F_1, F_9)\big) = \varepsilon$ donc $n = 2$ et $j = 2$;

construire $Stree(w\#x)$ (où # est un caractère séparateur)
for $i = 1 \dots \ell$ **do**
  // une occurrence de w avec moins de r erreurs débute-t-elle en i ?
  $j = 0$ et $n = 0$
  while $j < h$ et $n \leqslant r$ **do**
    $v = \mathcal{C}\big(root, \mathrm{PPAC}(F_j, F_{h+i+j})\big)$
    if $|v| = 0$ **then**
      $n = n + 1$ et $j = j + 1$
    else
      $j = j + |v|$
  if $j \geqslant h$ et $n \leqslant r$ **then**
    une occurrence de w commence en position i avec n erreurs

ALGORITHME **3:** Trouver tous les occurrences du mot w (de taille h) dans la séquence x (de taille ℓ) avec au plus $0 \leqslant r \leqslant h$ erreurs. Complexités : $O(\ell + h)$ en espace et $O(\ell + h + \ell \times r)$ en temps.

  - $n > r$ donc pas d'occurrence de w en $i = 2$.
- En $i = 3$:
  - $\mathcal{C}\big(root, \mathrm{PPAC}(F_0, F_9)\big) = \varepsilon$ donc $n = 1$ et $j = 1$;
  - $\mathcal{C}\big(root, \mathrm{PPAC}(F_1, F_{10})\big) = \mathtt{c}$ donc $n = 1$ et $j = 2$;
  - $\mathcal{C}\big(root, \mathrm{PPAC}(F_2, F_{11})\big) = \varepsilon$ donc $n = 2$ et $j = 3$;
  - $n > r$ donc pas d'occurrence de w en $i = 3$.
- ...
- En $i = 6$:
  - $\mathcal{C}\big(root, \mathrm{PPAC}(F_0, F_{12})\big) = \mathtt{act}$ donc $n = 0$ et $j = 3$;
  - $\mathcal{C}\big(root, \mathrm{PPAC}(F_3, F_{15})\big) = \varepsilon$ donc $n = 1$ et $j = 4$;
  - $\mathcal{C}\big(root, \mathrm{PPAC}(F_4, F_{16})\big) = \mathtt{at}$ donc $n = 1$ et $j = 6$;
  - $j \geqslant h$ et $n \leqslant r$ donc une occurrence de w en $i = 6$ avec $n = 1$ erreur.
- ...
- En $i = 15$:
  - $\mathcal{C}\big(root, \mathrm{PPAC}(F_0, F_{21})\big) = \mathtt{actg}$ donc $n = 0$ et $j = 4$;
  - $\mathcal{C}\big(root, \mathrm{PPAC}(F_4, F_{25})\big) = \varepsilon$ donc $n = 1$ et $j = 5$;
  - $\mathcal{C}\big(root, \mathrm{PPAC}(F_5, F_{26})\big) = \varepsilon$ donc $n = 2$ et $j = 6$;
  - $n > r$ donc pas d'occurrence de w en $i = 15$.
- ...

Si cette application des arbres de suffixes peut sembler n'offrir qu'un gain modeste par rapport à une approche directe (ex : compter le nombre d'erreurs observées en chaque position, ce qui résulte en une complexité $O(\ell \times h)$ pour tester l'ensemble des positions) c'est parce qu'on pense intuitivement d'abord à son usage pour de petites valeurs de h. Les choses sont en effet radicalement différentes si l'on considère des valeurs pour h et ℓ de l'ordre de plusieurs millions où l'approche directe devient très lente voire impossible.

Une utilisation possible de cet algorithme (ou de variantes de celui-ci) est la recherche de répétitions approchées dans un ou plusieurs génomes avec une complexité essentiellement linéaire en la somme des longueurs de ces génomes. Ce sont des stratégies de ce type qui permettent à des programmes comme MUMmer ou MGA d'effectuer des comparaisons massives entre plusieurs génomes de plusieurs millions de lettres en un temps raisonnable.

4.3.3 Arbres de préfixes

Si la puissance des arbres de suffixes rend leur usage attirant pour le comptages de tous les mots d'une longueur h donnée dans une séquence $x = x_1 \dots x_\ell$ (ou un jeu de séquences), il faut bien admettre qu'il existe des méthodes à la fois plus simple et plus efficaces pour ce problème. Nous présentons ici une telle méthode.

Sachant que l'on ne s'intéresse pas au comptage des mots de longueur plus grande que h, il n'est nécessaire ni de collecter, ni de conserver d'information les concernant. On va donc s'intéresser uniquement à l'ensemble

$$\mathcal{W}_h = \{w \in \mathcal{A}^*, w \text{ présent dans } x \text{ et } |w| \leqslant h\}$$

des mots de taille inférieure ou égale h présent au moins une fois dans x. Pour tout $w \in \mathcal{W}_h$ on rappelle qu'on note $N_x(w)$ le nombre d'occurrences (chevauchantes) du mot w dans x. Par convention, $N_x(\varepsilon) = \ell$.

Définition 70 (arbre de préfixes)**.** On appelle *arbre de préfixes* d'ordre h d'une séquence x l'arbre $Ptree_h(x)$ dont les sommets sont les éléments w de $\mathcal{W}_h$ étiquetés par $N_x(w)$, enraciné en ε, dont les arêtes, étiquetées par des éléments de $\mathcal{A}$, sont définies par la fonction de transition :

$$\delta(w, a) = wa \quad \forall w \in \mathcal{W}_h \text{ et } a \in \mathcal{A} \text{ tels que } wa \in \mathcal{W}_h.$$

Remarque 71. Un arbre de préfixes d'ordre h contient donc beaucoup moins d'information qu'un arbre de suffixes. En particulier, seules les nombres d'occurrences sont indiquées (et non plus leurs positions).

Proposition 72. On peut construire $Ptree_h(x)$ en $O(\ell \times h)$.

Démonstration. C'est l'algorithme 4 qui le permet. □

Remarque 73. Signalons au passage que l'algorithme 4 est un algorithme "on-line", ce qui signifie que l'arbre construit à la fin de l'itération i est le l'arbre de préfixe de la séquence $x_1 \dots x_i$.

Exemple 74. Utilisons l'algorithme 4 pour construire sur l'alphabet $\mathcal{A} = \{\mathtt{a}, \mathtt{c}, \mathtt{g}, \mathtt{t}\}$ l'arbre de préfixes d'ordre 3 du texte T donné en figure 4.9.

On initialise l'arbre avec $\mathcal{W}_3 = \mathcal{J} = \{\varepsilon\}$ et $N_T(\varepsilon) = 0$ et on commence à lire successivement les lettres de T.

Requis: $\mathcal{J}$ est un ensemble d'au plus $h+1$ mots
1: $\mathcal{W}_h = \mathcal{J} = \{\varepsilon\}$ et $N_x(\varepsilon) = 0$
2: **for** $i = 1 \dots \ell$ **do**
3: $a = x_i$
4: **if** $a \in \mathcal{A}$ **then**
5: *// la lettre est valide*
6: **for all** $v \in \mathcal{J}$ **do**
7: **if** $|v| < h$ **then**
8: $w = va$
9: **if** $w \in \mathcal{W}_h$ **then**
10: ajouter w à $\mathcal{W}_h$, ajouter l'arête $\delta(v, a) = w$ et faire $N_x(w) = 1$
11: **else**
12: $N_x(w) += 1$
13: $v = w$
14: **else**
15: $v = \varepsilon$ et $N_x(\varepsilon) += 1$
16: **if** $|\mathcal{J}| < h$ **then**
17: ajouter ε à $\mathcal{J}$ et faire $N_x(\varepsilon) += 1$
18: **else**
19: *// la lettre est invalide*
20: $\mathcal{J} = \{\varepsilon\}$

ALGORITHME **4:** Construction du $Ptree_h(x)$ où x est une séquence (ou un jeu de séquences) de longueur ℓ sur l'alphabet $\mathcal{A}$ et où $h \geqslant 1$. Complexités : $O(|\mathcal{W}_h|)$ en espace et $O(\ell \times h)$ en temps.

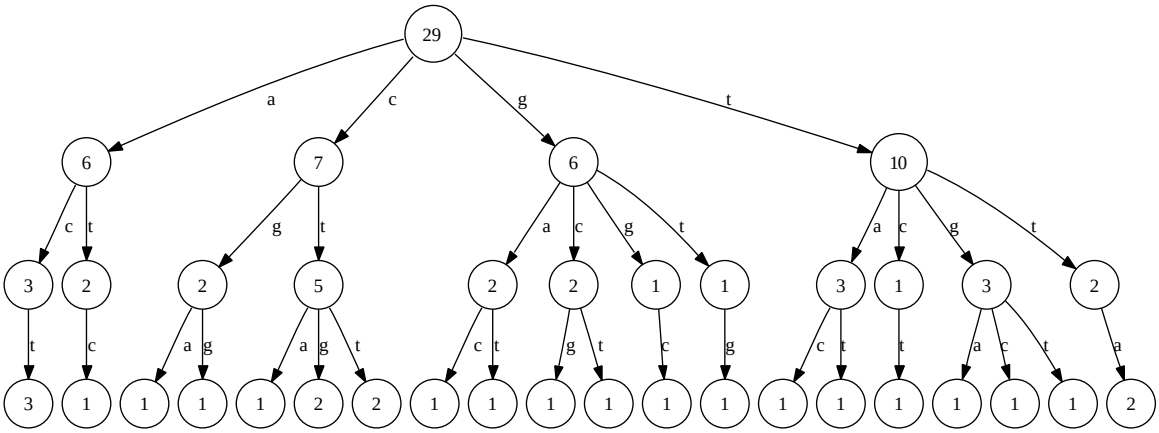


FIG. 4.11 – Arbre de préfixes (profondeur 3) sur l'alphabet ADN du texte de la figure 4.9. La racine est en haut.

- $i = 0$, $a = \texttt{a}$ est un caractère valide ; on ajoute $\texttt{a}$ à $\mathcal{W}_3$ avec $N_T(\texttt{a}) = 1$ et on a $N_T(\varepsilon) = 1$ et $\mathcal{J} = \{\texttt{a}, \varepsilon\}$;
- $i = 1$, $a = \texttt{c}$ est un caractère valide ; on ajoute $\texttt{ac}$ à $\mathcal{W}_3$ avec $N_T(\texttt{ac}) = 1$, on ajoute $\texttt{c}$ à $\mathcal{W}_3$ avec $N_T(\texttt{c}) = 1$ et on a $N_T(\varepsilon) = 2$ et $\mathcal{J} = \{\texttt{ac}, \texttt{c}, \varepsilon\}$;
- $i = 2$, $a = \texttt{t}$ est un caractère valide ; on ajoute $\texttt{act}$ à $\mathcal{W}_3$ avec $N_T(\texttt{act}) = 1$, on ajoute $\texttt{ct}$ à $\mathcal{W}_3$ avec $N_T(\texttt{ct}) = 1$, on ajoute $\texttt{t}$ à $\mathcal{W}_3$ avec $N_T(\texttt{t}) = 1$ et on a $N_T(\varepsilon) = 3$ et $\mathcal{J} = \{\texttt{act}, \texttt{ct}, \texttt{t}, \varepsilon\}$;
- $i = 3$, $a = \texttt{g}$ est un caractère valide ; $N_T(\varepsilon) = 4$, on ajoute $\texttt{ctg}$ à $\mathcal{W}_3$ avec $N_T(\texttt{ctg}) = 1$, on ajoute $\texttt{tg}$ à $\mathcal{W}_3$ avec $N_T(\texttt{tg}) = 1$, on ajoute $\texttt{g}$ à $\mathcal{W}_3$ avec $N_T(\texttt{g}) = 1$ et on a $\mathcal{J} = \{\varepsilon, \texttt{ctg}, \texttt{tg}, \texttt{g}\}$;
- $i = 4$, $a = \texttt{a}$ est un caractère valide ; $N_T(\texttt{a}) = 2$, $N_T(\varepsilon) = 5$, on ajoute $\texttt{tga}$ à $\mathcal{W}_3$ avec $N_T(\texttt{tga}) = 1$, on ajoute $\texttt{ga}$ à $\mathcal{W}_3$ avec $N_T(\texttt{ga}) = 1$ et on a $\mathcal{J} = \{\texttt{a}, \varepsilon, \texttt{tga}, \texttt{ga}\}$;
- $i = 5$, $a = \texttt{t}$ est un caractère valide ; on ajoute $\texttt{at}$ à $\mathcal{W}_3$ avec $N_T(\texttt{at}) = 1$, $N_T(\texttt{t}) = 2$, $N_T(\varepsilon) = 6$, on ajoute $\texttt{gat}$ à $\mathcal{W}_3$ avec $N_T(\texttt{gat}) = 1$ et on a $\mathcal{J} = \{\texttt{at}, \texttt{t}, \varepsilon, \texttt{gat}\}$;
- $i = 6$, $a = \texttt{\#}$ est un caractère invalide on réinitialise $\mathcal{J} = \{\varepsilon\}$
- $i = 7$, $a = \texttt{c}$ est un caractère valide ; $N_T(\texttt{c}) = 2$ et on a $N_T(\varepsilon) = 7$ et $\mathcal{J} = \{\texttt{c}, \varepsilon\}$;
- ...

Au final on obtient l'arbre de la figure 4.11. Pour obtenir le comptage d'un mot (de longueur inférieure ou égale à 3 puisqu'il s'agit d'un arbre d'ordre 3), il suffit de partir de la racine et de suivre les arêtes portant les lettres successives du mot. Si on débouche sur un noeud, alors il contient le comptage du mot, si le noeud n'existe pas, c'est que le mot est de comptage nul. La racine indique le nombre de caractères valides rencontrés. Par exemple, le mot $\texttt{a}$ apparaît 6 fois, le mot $\texttt{ct}$ apparaît 5 fois, le mot $\texttt{ca}$ n'apparaît pas, le mot $\texttt{act}$ apparaît 3 fois, etc ...

La table 4.4 compare le temps de calcul et l'espace mémoire requis pour la construction d'un arbre de suffixes ou d'un arbre de préfixes pour plusieurs génomes. Si, pour de petites séquences, le coût de construction des arbres de suffixes est proche de celui des arbres de préfixes, dès que les longueurs des séquences augmentent, il est à la fois plus lent et, ce qui est plus problématique, nécessite une bien plus grande quantité de mémoire.

Dans cette partie, nous avons donc proposé une méthode efficace permettant le comp-

organisme	taille	arbre	temps (s)	mémoire
Virus HIV	10 kb	*Stree*	0.00	600 Ko
		$Ptree_4$	0.00	1 Ko
		$Ptree_8$	0.00	100 Ko
		$Ptree_{12}$	0.00	250 Ko
Mycoplasma genitalium	580 kb	*Stree*	0.83	39 Mo
		$Ptree_4$	0.04	1 Ko
		$Ptree_8$	0.06	330 Ko
		$Ptree_{12}$	0.35	7 Mo
Escherichia coli	4.6 Mb	*Stree*	8.54	270 Mo
		$Ptree_4$	0.32	16 Ko
		$Ptree_8$	0.53	330 Ko
		$Ptree_{12}$	2.71	40 Mo

TAB. 4.4 – Comparaison du temps de calcul et de l'espace mémoire requis pour la construction d'un arbre de suffixes ou d'un arbre de préfixes (d'ordre 4, 8 et 12) pour plusieurs génomes complets.

tage de tous les mots d'une longueur donnée dans un texte. Il est important de noter que ce point, s'il peut sembler anecdotique à une époque la puissance de calcul des ordinateurs a fait d'énormes progrès, reste tout de même très important lorsque l'on est confronté à des volumes de données importants (comme cela peut facilement être le cas dans l'étude des séquences biologiques) et tant il est facile de mettre en oeuvre des solutions naïves extrêmement peu performantes.

On peut citer par exemple le cas du programme `wordcount` de la populaire suite de logiciels EMBOSS (Rice et al., 2000) dont les performances sont comparées à celle du logiciel SPatt dans la table 4.5. On constate que pour la simple tâche consistant à compter tous les mots de longueur 12 dans un génome de moins de cinq millions de bases, le programme `wordcount` peut prendre jusqu'à plus d'une heure de temps de calcul quand le programme SPatt (utilisant des arbres de préfixes) ne nécessite qu'une poignée de secondes.

h	1	2	3	4	5	6	7	8	9	10	11	12
EMBOSS	na	7.08	8.45	9.41	12	18	47	155	537	1 546	3 111	4 352
SPatt	0.13	0.17	0.20	0.23	0.29	0.35	0.40	0.50	0.80	1.75	3.71	9.86

TAB. 4.5 – Comparaison du temps de calcul (en secondes) nécessaire à de EMBOSS (programme `wordcount`) et SPatt pour compter tous les mots de taille h de *Escherichia coli* K12 (4.6 Mb).

4.4 Notes bibliographiques

Automates

On trouvera avec Hopcroft et al. (2001) un ouvrage de référence sur la théorie des langages et des automates.

La conversion d'une expression régulière en AFND (voir proposition 48) est due à

Thompson (1968). La déterminisation d'un AFND est également appelée *subset construction*. Leslie (1995) propose une revue détaillée de différentes implémentations efficaces. La minimisation d'un AFD (il est parfois fait référence au *partitioning*) est due à Hopcroft (1971). L'heuristique de construction dans le cas d'un simple mot est très classique dans le monde du *string* ou *pattern matching* (Chrochemore et Hancart, 1997).

Les aspects liés à la construction fainéantes (*lazy construction* en anglais) d'automates sont traités dans la thèse de Maout (2003).

Un ensemble de programmes et de librairies de programmation se trouve dans FSA. Pour la visualisation de tout type de graphes, et d'automates en particulier, on pourra voir le programme `dot` du projet Graphviz.

Arbres de Suffixes

Citons trois algorithmes historiques (Weiner, 1973; McCreight, 1976; Ukkonen, 1995) et un article faisant la synthèse de ces différentes approches : Giegerich et Kurtz (1997).

libstree est une librairie en C permettant la création et l'utilisation d'arbres de suffixes.

Parmi les programmes utilisant les arbres de suffixes pour effectuer de l'alignement massif de génomes, donnons les exemples de MUMmer et de MGA.

Chapitre 5

Statistiques de motifs (G. Nuel)

Nous avons vu dans le chapitre précédent comment compter efficacement les occurrences de mots et de motifs dans les séquences biologiques. S'il semble naturel d'utiliser ces comptages pour essayer de mettre en évidence des motifs exceptionnels, la tâche n'est pas aussi aisée qu'elle y parait.

Dans une première partie, nous allons considérer un exemple concret issue de la langue française (5.1). Nous introduirons ensuite le cadre de la modélisation markovienne dans lequel nous allons nous placer (5.2). Nous verrons également avec la notion de PMC (5.3) comment réécrire ce problème complexe afin d'obtenir une approche identique et efficace pour tout motif en utilisant les automates introduits au chapitre précédent. Enfin, nous présenterons en détail un grand nombre d'approches statistiques concurrentes (5.4 à 5.8) et discuterons des avantages et des inconvénients de chacunes d'entre elles (5.9).

5.1 Cyrano de Bergerac

Nous examinons ici l'exemple d'un texte en langue française : l'oeuvre intégrale de "Cyrano de Bergerac", la comédie d'Edmond Rostand. On commence par transformer ce texte en une longue séquence écrite dans l'alphabet latin à 26 lettres en éliminant les espaces et la ponctuation tout en remplaçant tous les caractères accentués par leurs versions sans accents. On obtient alors une séquence de longueur $\ell = 154\,623$ (voir figure 5.1 pour un extrait).

On considère l'ensemble $\mathcal{A}^3 = \{\texttt{aaa}, \texttt{aab}, \texttt{aac}, \ldots\}$ des 26^3 mots de trois lettres possibles et on tente d'y détecter des éléments exceptionnels.

Pour cela on commence par examiner simplement les nombres d'occurrences de tous ces mots dans la séquence : $N(\texttt{aaa}) = 0$, $N(\texttt{aab}) = 0$, $N(\texttt{aac}) = 1$, $\ldots N(\texttt{aaz}) = 0$, $N(\texttt{aba}) = 33, \ldots, N(\texttt{zzx}) = 0$, $N(\texttt{zzy}) = 0$ et $N(\texttt{zzz}) = 0$. On remarque immédiatement qu'une grande majorité des mots de trois lettres ne sont pas présents dans la séquence : 13 428 mots sur les $26^3 = 17\,576$ possibles ont un comptage nul (soit plus de 75% d'entre eux). Cela n'est guère étonnant car la langue française interdit de nombreuses combinaisons de lettres, mais, de ce fait, tous les mots de comptages non nuls ont déjà un caractère exceptionnel.

Voici les 10 premiers mots de cette liste (ordre alphabétique) :

mot	aac	aai	aal	aam	aar	aas	aau	aav	aba	abd
comptage	1	1	4	1	3	1	5	1	33	1

Si certains d'entre eux ne semblent pas faire partie de la langue française, il ne faut pas s'en inquiéter outre mesure. En effet, la séquence étudiée ne comportant pas d'espaces (voir figure 5.1) il est possible que des combinaisons de lettres étranges apparaissent entre plusieurs mots habituellement séparés par des espaces (ex : "...`il se leve et va a carbon de castel-jaloux`..." qui devient "...`ilseleveetv`<u>`aac`</u>`arbondecasteljaloux`...").

```
> Cyrano de Bergerac (154 623 lettres)
edmondrostandcyranodebergeraccomedieheroiqueencinqactesenver
srepresenteeaparissurletheatredelaportesaintmartinledecembre
cestalamedecyranoquejevoulaisdediercepoememaispuisquelleapas
seenvouscoquelincestavousquejelededieerpersonnagescyranodebe
rgeracchristiandeneuvillettecomtedeguicherageneaulebretcarb
ondecasteljalouxlescadetslignieredevalvertunmarquisdeuxiemem
arquistroisiememarquismontfleurybellerosejodeletcuigybrissai
lleunfacheuxunmousquetaireunautreunofficierespagnolunchevaul
egerleportierunbourgeoissonfilsuntirelaineunspectateurungard
ebertrandoulefifrelecapucindeuxmusiciensleslespoeteslespatissie
[...]
isjamaisahtevoilatoilasottisejesaisbienqualafinvousmemettrez
abasnimportejemebatsjemebatsjemebatsilfaitdesmoulinetsimmens
esetsarretehaletantouivousmarracheztoutlelaurieretlarosearra
chezilyamalgrevousquelquechosequejemporteetcesoirquandjentre
raichezdieumonsalutbalaieralargementleseuilbleuquelquechoseq
uesansunplisansunetachejemportemalgrevousilselancelepeehaute
etcestlepeesechappedesesmainsilchancelletombedanslesbrasdele
bretetderagueneauroxanesepenchantsurluietluibaisantlefrontce
stcyranorouvrelesyeuxlareconnaitetditensouriantmonpanacherid
eau
```

FIG. 5.1 – Extrait le la séquence `cyrano` au format FASTA dans l'alphabet latin $\mathcal{A} = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \ldots, \mathtt{x}, \mathtt{y}, \mathtt{z}\}$
.

On peut penser à distinguer entre eux les mots de comptages non nuls en les classant en fonction de leurs nombres d'occurrences :

mot	ant	ran	ent	ous	que	ano	yra	cyr	les	est
comptage	1066	991	973	847	841	823	791	786	780	713

Certains des mots ainsi mis en évidence le doivent vraisemblablement à la fréquence de leur utilisation dans la langue française (`ant`, `ent`, `les`, ...) tandis que d'autres, plus exotiques, le doivent indéniablement à la spécificité du texte "Cyrano de Bergerac" (on pense ici à `ano`, `yra` et `cyr`, trois mots faisant partie du nom `cyrano` par exemple).

On peut s'étonner de constater que cette approche, malgré sa naïveté, met en évidence des propriétés de la langue française mais aussi des spécificités de la séquence étudiée et soulève donc, par la même occasion, l'épineux problème de la distinction de ces deux types d'informations. Sur ce point, on se contentera ici de dire qu'il est évidemment impossible d'effectuer ce distinguo au vu de la séquence seule, sans connaissance préalable particulière : connaissance de la langue française dans notre cas, ou, au minimum comparaison possible avec d'autres textes.

En procédant ainsi, on a implicitement donné le même poids à chacun des mots de trois lettres sans tenir compte de leurs compositions respective en lettres. Il est néanmoins clair que les fréquences d'utilisation des lettres de l'alphabet différent énormément : dans notre séquence, les trois lettres les plus plus répandues sont `e`, `a` et `s` (avec les comptages $n(\mathtt{e}) = 25\ 141$, $n(\mathtt{a}) = 13\ 184$ et $n(\mathtt{s}) = 11\ 753$), tandis que les trois lettres les plus rares

sont `w`, `k` et `z` (avec les comptages $n(\texttt{w}) = 0$, $n(\texttt{k}) = 1$ et $n(\texttt{z}) = 556$), il est donc clair qu'un traitement identique de toutes les lettres n'est pas approprié.

Avec de tels déséquilibres, certaines observations vont, à l'évidence, davantage surprendre que d'autres. Par exemple, on observe que $n(\texttt{aaa}) = n(\texttt{zzz}) = 0$. Si, pour `zzz`, cela n'est guère étonnant car la lettre `z` est très rare, il est assez remarquable que la grande fréquence de la lettre `a` ne permette pas l'apparition du mot `aaa`.

En considérant les fréquences des lettres, il apparaît clairement que le comptage nul du mot `aaa` fait indéniablement de celui-ci un mot exceptionnel. Cette information sans connaissance *a priori* autre que le texte lui-même parvient ici à rendre compte d'un phénomène caractéristique de la langue française : la lettre `a` y est fort rarement répétée et donc *a fortiori*, encore plus rarement triplée.

Cette dernière remarque soulève une nouvelle question : ne pourrait-on essayer de tenir compte de la rareté du mot `aa` avant de déclarer le mot `aaa` exceptionnel ? Plus généralement, y a-t-il un moyen de tenir compte des comptages des mots de deux lettres pour examiner les comptages des mots de trois lettres ?

Nous verrons dans la suite quelle réponse précise on peut apporter à cette question. Pour l'instant, on va se contenter d'un exemple assez parlant avec les mots `aaa` et `qur`. Ces deux mots sont de comptages nuls mais les mots de lettres les composent sont de fréquences très différentes : $n(\texttt{aa}) = 17$ alors que $n(\texttt{qu}) = 1627$ et $n(\texttt{ur}) = 1661$. À la lecture de ces chiffres, on conclut sans mal que, si le comptage nul de `aaa` n'est guère étonnant, celui de `qur` l'est très certainement.

On met ainsi à nouveau en évidence une information concernant la langue française : un `r` ne doit pas (ou bien très rarement) suivre le mot `qu`. Si cette remarque est naturelle dans la mesure où le mot `qur` n'est pas présent dans la séquence, c'est cependant, le caractère exceptionnel de son comptage, compte tenu des fréquences des mots de deux lettres, qui a permis de le remarquer dans la masse des mots de trois lettres de comptages nuls.

Avant de s'atteler aux définitions formelles de ces différentes notions, il est intéressant de considérer un dernier exemple avec les mots `ana` et `din`. Les comptages de ces deux mots sont égaux ($n(\texttt{ana}) = n(\texttt{din}) = 40$) si bien qu'en ne tenant pas compte des fréquences des lettres qui les composent on s'attend à apprécier de manière identique leurs éventuels caractères exceptionnels. Mais en fait, la structure d'*auto-recouvrement* de `ana` facilite l'apparition de ses occurrences alors qu'il n'en est rien pour `din`. En effet, si le mot `ana` est présent dans la séquence, il suffit que le mot de deux lettres `na` soit présent à sa suite pour qu'une nouvelle occurrence du mot apparaisse.

Ces exemples mettent bien en évidence la nature complexe du problème considéré. De plus, il est clair que ces phénomènes gagnent en complexité et en importance avec la longueur des mots examinés (composition des motifs) et dans la mesure où le cardinal de l'alphabet $\mathcal{A}$ est faible, facilitant les auto-recouvrements, deux conditions souvent rencontrées dans l'étude des séquences biologiques.

5.2 Statistique de motifs

Soit $x = x_1 \dots x_\ell$ une séquence biologique observée et $X = X_1 \dots X_\ell$ une séquence aléatoire de même longueur sous un modèle référant choisi (voir 3.1), on notera $\mathbb{P}$ la loi sous ce modèle ; soit $\mathcal{W}$ un motif donné, on notera $N = N_X(\mathcal{W})$.

Définition 75 (Statistique de motifs)**.** On définit la *statistique du motif* $S_X(\mathcal{W}, n)$ par :

$$S_X(\mathcal{W}, n) = \begin{cases} -\log_{10} \mathbb{P}(N_X(\mathcal{W}) \geqslant n) & \text{si } n \geqslant \mathbb{E}[N_X(\mathcal{W})] \\ \log_{10} \mathbb{P}(N_X(\mathcal{W}) \leqslant n) & \text{si } n < \mathbb{E}[N_X(\mathcal{W})] \end{cases}$$

Par convention, on note $S_X(\mathcal{W}) = S_X(\mathcal{W}, N_x(\mathcal{W}))$ et on omettra l'indice X lorsque cela ne prêtera pas à confusion. A noter que, comme pour les comptages, cette définition s'étend naturellement au comptages chevauchants (exposant O) ou renouvelants (exposant R) mais qu'en l'absence d'exposant, c'est au comptage chevauchant que l'on fait référence.

Cette statistique prend donc une valeur positive pour les motifs *sur-représentés* (c'est à dire avec plus d'occurrences que le nombre attendu sous le modèle référant) et négative pour les motifs *sous-représentés*. Dans les deux cas, la statistique représente (au signe près) la degré de significativité de l'observation en échelle logarithmique décimale.

Exemple 76. Prenons pour x le génome complet de *Escherichia coli* K12, `gctggtgg` y apparaît 499 fois et `cttaag` 557 fois. Choisissant pour modèle référant le modèle M1 estimé sur cette même séquence, on a $S(\texttt{gctggtgg}, 499) = 240.81$ ce qui signifie que le motif `gctggtgg` est sur-représenté et que la significativité de cette observation est $10^{-240.81}$. De même, on a $S(\texttt{cttaag}, 557) = -19.73$ ce qui signifie que le mot `cttaag` est sous-représenté et avec une significativité de $10^{-19.73}$.

Comme on le verra plus en détails dans la section 5.8, la théorie des grandes déviations nous assure que cette statistique croit linéairement avec la longueur des séquences considérées (ce qui donne en passant une justification de poids au choix de l'échelle logarithmique). Ainsi, les statistiques obtenues sur des séquences de longueurs différentes ne sont pas comparables sans effectuer une normalisation qui divise $S_X(\mathcal{W})$ par la longueur de X.

Exemple 77. On considère ici la séquence `hiv1` contenant le génome complet du virus HIV de type 1 ($\ell = 9\,718$) et la séquence `10hiv1` obtenue en mettant bout à bout dix séquences `hiv1`.

$\mathcal{W}$	`tagta`	`agtag`	`tctgg`	`catga`	`caaac`	`ctaag`
$S_{\texttt{hiv1}}(\mathcal{W})$	8.168	6.059	4.004	−3.949	−4.856	−5.161
$S_{\texttt{10hiv1}}(\mathcal{W})$	72.86	52.16	32.76	−34.91	−45.41	−48.45
rapport	8.92	8.60	8.18	8.84	9.37	9.39

on a donc bien un ordre de grandeur de 10 pour le rapport entre $S_{\texttt{10hiv1}}/S_{\texttt{hiv1}}$. Bien évidemment, la qualité de cette normalisation est d'autant meilleure que les séquences considérées sont longues.

En effet, si on considère maintenant les séquences `ecoli` et `10ecoli` construites de manière analogue avec le génome complet de *Escherichia coli* K12 ($\ell = 4\,639\,221$) , les rapports obtenus

$\mathcal{W}$	`tagta`	`agtag`	`tctgg`	`catga`	`caaac`	`ctaag`
$S_{\texttt{ecoli}}(\mathcal{W})$	−39.4	−11.0	759	−83.2	−5.84	−319
$S_{\texttt{10ecoli}}(\mathcal{W})$	−382	−112	7571	−818	−49.4	−3173
rapport	9.70	10.2	9.98	9.83	8.46	9.95

sont beaucoup plus proches de 10.

5.3 *Pattern Markov Chain*

On a vu dans la section 4.2 comment on peut utiliser les Automates Finis Déterministes (AFD) pour compter les occurrences de motifs de manière efficace. Le théorème 43 illustre cette propriété en construisant à partir d'une séquence x de lettres, une séquence y auxiliaire d'états de l'automate dans laquelle les occurrences du motif correspondent à celle d'un sous-ensemble d'états. Le but de la présente partie est d'étendre ce résultat aux séquences aléatoires.

5.3.1 Modèle M0

Théorème 78. Soit $X = X_1 \dots X_\ell$ une séquence sur l'alphabet $\mathcal{A}$ sous le modèle M0 de paramètre μ. Si $\mathcal{W}$ est un motif sur $\mathcal{A}$ et $(\mathcal{A}, \mathcal{Q}, s, \mathcal{F}, \delta)$ un AFD reconnaissant le langage $\mathcal{L} = \mathcal{A}^*\mathcal{W}$ alors la séquence $Y = Y_0 \dots Y_\ell$ définie par

$$Y_0 = s \quad \text{et} \quad Y_i = \delta(Y_{i-1}, X_i) \quad \text{pour tout} \quad 1 \leqslant i \leqslant \ell$$

est une chaîne de Markov d'ordre 1 dont la matrice de transition Π vérifie pour tous $p, q \in \mathcal{Q}$

$$\Pi(p,q) = \begin{cases} \mu(a) & \text{si } \delta(p,a) = q \\ 0 & \text{si } q \notin \delta(p, \mathcal{A}) \end{cases}$$

et on a

$$\mathcal{W} \text{ se termine en position } i \text{ dans } X \iff Y_i \in \mathcal{F}$$

Une chaîne de Markov Y d'ordre 1 (ici sur $\mathcal{Q}$) vérifiant cette dernière propriété pour un certain sous-ensemble de ses lettres (ici $\mathcal{F}$) est appelée *Pattern Markov Chain* (PMC).

On peut faire quelques remarques élémentaires concernant la PMC Y construite (comme ci-dessus) via un AFD $(\mathcal{A}, \mathcal{Q}, s, \mathcal{F}, \delta)$:

a) Y est définie sur l'ensemble $\mathcal{Q}$ des états de l'AFD. Par conséquent, le cardinal L de cet ensemble sera minimal si l'AFD considéré l'est également.

b) La matrice de transition Π de Y est creuse par construction : elle a au plus $k \times L$ termes non nuls (k étant le cardinal de l'alphabet $\mathcal{A}$) sur les $L \times L$ termes possibles.

c) On a une décomposition naturelle de Π en $\Pi = P + Q$ avec, pour tous $p, q \in \mathcal{Q}$.

$$P(p,q) = \begin{cases} \Pi(p,q) & \text{si } q \in \mathcal{Q} \setminus \mathcal{F} \\ 0 & \text{sinon} \end{cases} \quad \text{et} \quad Q(p,q) = \begin{cases} \Pi(p,q) & \text{si } q \in \mathcal{F} \\ 0 & \text{sinon} \end{cases}$$

Exemple 79. Considérons par exemple le motif $\mathcal{W} = \{\texttt{abab}, \texttt{baa}\}$ sur l'alphabet binaire $\mathcal{A} = \{\texttt{a}, \texttt{b}\}$. Une représentation graphique du plus petit AFD associé à $\mathcal{W}$ est donnée en figure 5.2. La matrice de transition associée à la PMC Y correspondante est donnée par

$$\Pi = \begin{pmatrix} 0 & 0 & 0 & \mu(\texttt{a}) & 0 & 0 & \mu(\texttt{b}) & 0 \\ 0 & 0 & 0 & \mu(\texttt{a}) & \mu(\texttt{b}) & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \mu(\texttt{a}) & \mu(\texttt{b}) & 0 \\ 0 & 0 & 0 & \mu(\texttt{a}) & \mu(\texttt{b}) & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \mu(\texttt{a}) & \mu(\texttt{b}) & 0 \\ 0 & \mu(\texttt{a})^* & \mu(\texttt{b})^* & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \mu(\texttt{b}) & \mu(\texttt{a}) \\ 0 & \mu(\texttt{a})^* & 0 & 0 & \mu(\texttt{b}) & 0 & 0 & 0 \end{pmatrix}$$

où les termes marqués d'un * appartiennent à Q (il s'agit des transitions se terminant dans un état final) et les autres à P.

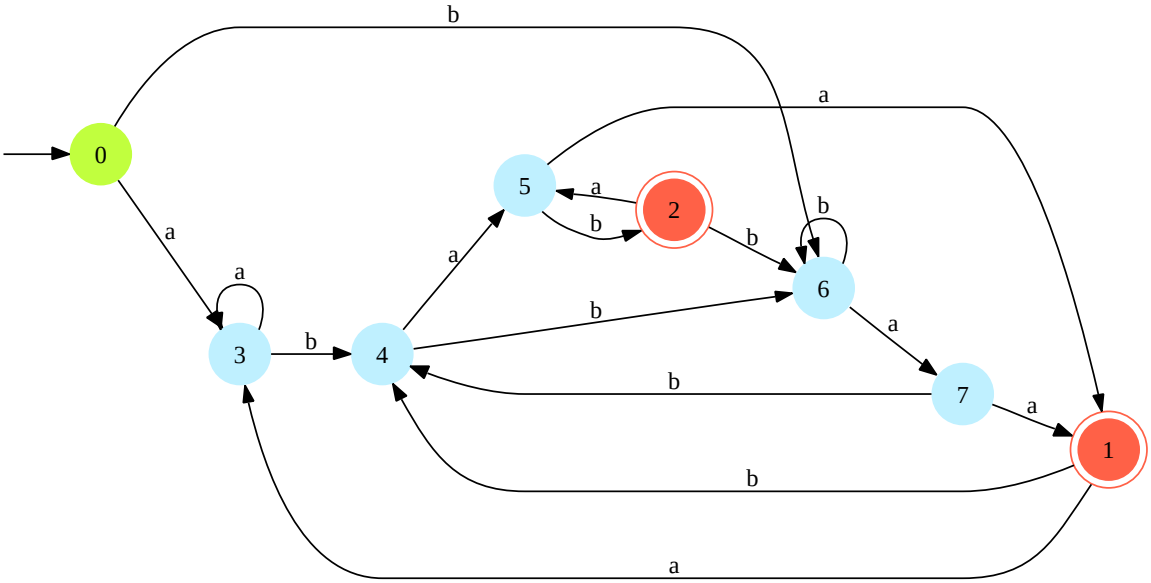


FIG. 5.2 – Représentation graphique du plus petit AFD associé au motif $\mathcal{W} = \{\text{abab}, \text{baa}\}$ défini sur l'alphabet binaire $\mathcal{A} = \{\texttt{a}, \texttt{b}\}$.

5.3.2 Modèle Mm

Avant de pouvoir étendre les résultats précédents au modèle Mm (pour $m \geqslant 1$) il nous faut introduire la notion de m-ambiguïté[1] pour les AFD.

Définition 80. Un AFD $(\mathcal{A}, \mathcal{Q}, s, \mathcal{F}, \delta)$ est *m-ambigu* si et seulement si il existe $q \in \mathcal{Q}$ et $a, b \in \mathcal{A}^m$ tels que $a \neq b$ et $\delta(q, a) = \delta(q, b)$.

Si, pour tout $q \in \mathcal{Q}$ et pour tout $m \geqslant 1$ on introduit la notation

$$\delta^{-m}(q) = \{a \in \mathcal{A}^m, \exists p \in \mathcal{Q},\ \delta(p, a) = q\}$$

on obtient alors qu'un AFD est non m-ambigu si et seulement si, pour tout $q \in \mathcal{Q}$, $\delta^{-m}(q)$ est soit un singleton, soit vide. On note également

$$\Delta^{-1}(q) = \{p \in \mathcal{Q}, \exists a \in \mathcal{A},\ \delta(p, a) = q\}$$

l'ensemble des états permettant d'arriver en un $q \in \mathcal{Q}$ en une transition.

Exemple 81. L'AFD de la figure 4.3 est 1-ambigu. En effet, $\delta^{-1}(5) = \{\texttt{b}, \texttt{c}\}$ n'est pas un singleton. L'AFD de la figure 5.2 est non 1-ambigu, il est même non 2-ambigu, mais en revanche, il est 3-ambigu puisque (par exemple) $\delta^{-3}(6) = \{\texttt{abb}, \texttt{bbb}\}$.

Théorème 82. Soit $X = X_1 \dots X_\ell$ une séquence sur l'alphabet $\mathcal{A}$ sous le modèle Mm ($m \geqslant 1$) de loi initiale μ_0 et de paramètre π. Si $\mathcal{W}$ est un motif sur $\mathcal{A}$ et $(\mathcal{A}, \mathcal{Q}, s, \mathcal{F}, \delta)$ un AFD non m-ambigu reconnaissant le langage $\mathcal{L} = \mathcal{A}^*\mathcal{W}$ alors la séquence $Y = Y_m \dots Y_\ell$ définie par

$$Y_0 = s \quad \text{et} \quad Y_i = \delta(Y_{i-1}, X_i) \quad \text{pour tout} \quad 1 \leqslant i \leqslant \ell$$

[1] Notion à ne pas confondre avec la notion d'ambiguïté pour les AFD qui signifie qu'il existe au moins deux chemins différents permettant de reconnaître le même mot.

est une chaîne de Markov d'ordre 1 dont la matrice de transition Π vérifie pour tous $p, q \in \mathcal{Q}$

$$\Pi(p,q) = \begin{cases} \pi(a_1, \ldots, a_m, b) & \text{si } \delta(p,b) = q \\ 0 & \text{si } q \notin \delta(p, \mathcal{A}) \end{cases}$$

où $a_1 \ldots a_m$ désigne l'unique élément de $\delta^{-m}(p)$. Comme dans le cas M0 on a également

$$\mathcal{W} \text{ se termine en position } i \text{ dans } X \iff Y_i \in \mathcal{F}$$

La chaîne de Markov $Y_m \ldots Y_\ell$ est définie sur l'ensemble

$$\mathcal{Q}' = \{\delta(s,x), x \in \mathcal{A}^*, |x| \geqslant m\} \subset \mathcal{Q}$$

Définition 83. La chaîne de Markov $Y = Y_m, \ldots, Y_\ell$ ainsi associée à l'AFD non m-ambigu reconnaissant un motif $\mathcal{W}$ et à une chaîne de Markov d'ordre m, $X = X_1, \ldots, X_\ell$, est appelée une PMC.

Remarque 84. La PMC Y ainsi construite a les mêmes propriétés que celle du théorème 78 à ceci près qu'elle est définie sur l'ensemble $\mathcal{Q}'$ et non pas sur l'ensemble $\mathcal{Q}$. Par souci de simplicité, on note également L le cardinal de $\mathcal{Q}'$.

Remarque 85. Quel que soit l'ordre de la chaîne de Markov X, la PMC associée est une chaîne de Markov d'ordre 1. Par ailleurs la séquence Y commence par $Y_0 = s, Y_1 \ldots Y_m = \delta(s, x_1 \ldots x_m)$ dès que $X_1 = x_1, \ldots, X_m = x_m$; donc la loi initiale μ de Y est donnée par

$$\mu(\delta(s, x_1 \ldots x_m)) = \mathbb{P}(X_1 = x_1, \ldots, X_m = x_m)$$

Ce peut être – entre autres ! - une loi de Dirac, si $X_1 \ldots X_m$ est fixé, ou la loi associée à la loi stationnaire de X. Mais même dans ce cas *ce ne peut être la loi stationnaire de* Y, ne serait-ce que parce que μ ne charge pas les états finaux de l'AFD.

On est donc capable d'obtenir très facilement la PMC d'un motif dans le cas d'un modèle Mm à condition de disposer d'un AFD *non m-ambigu* reconnaissant ce motif. Un tel AFD peut être produit à partir d'un simple AFD reconnaissant le motif en dupliquant les états m-ambigus jusqu'à ce que toutes les ambiguïtés aient été supprimées ; c'est exactement l'objet du théorème suivant :

Théorème 86. Pour tout $m \geqslant 1$, soit $(\mathcal{A}, \mathcal{Q}, s, \mathcal{F}, \delta)$ un AFD non $(m-1)$-ambigu reconnaissant le langage $\mathcal{L}$. L'algorithme 5 permet d'obtenir un AFD non m-ambigu reconnaissant $\mathcal{L}$. De plus, si l'AFD initial est le plus petit automate non $(m-1)$-ambigu reconnaissant $\mathcal{L}$, l'automate produit est également le plus petit non m-ambigu reconnaissant ce langage.

En partant d'un simple AFD associé à un motif donné, on peut donc utiliser de manière itérative ce théorème pour produire des automates non m-ambigus pour m quelconque.

Exemple 87. On considère le motif `a..ba` sur l'alphabet $\mathcal{A} = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$, on a va appliquer l'algorithme présenté ci-dessus (avec $m = 1$) à l'automate 1-ambigu (14 états) représenté en figure 4.3.

La phase d'initialisation nous donne $\mathcal{Q}_0 = \{0, 1, \ldots, 13\}$, $\mathcal{D}_0 = \{\mathtt{b}, \mathtt{c}\}$, $\mathcal{D}_1 = \{\mathtt{a}\}$, $\ldots$, $\mathcal{D}_{13} = \{\mathtt{b}\}$ et $\mathcal{G}_0 = \{0, 7, 8, 13\}$, $\mathcal{G}_1 = \{4, 6\}$, $\ldots$, $\mathcal{G}_{13} = \{7, 8\}$. Les seuls états présentant des 1-ambiguïtés sont : 0, 5, 8 et 10. Ces quatre cas se traitant de manières très similaires, on se contente ici de donner les détails du déroulement de l'algorithme pour $q = 0$:

Requis: $m \geqslant 1$, $(\mathcal{A}, \mathcal{Q}, s, \mathcal{F}, \delta)$ un AFD non $(m-1)$-ambigu reconnaissant le langage $\mathcal{L}$.
Pour tout $q \in \mathcal{Q}$ on pose $\mathcal{D}_q = \delta^{-m}(q)$ et $\mathcal{G}_q = \Delta^{-1}(q)$.

1: $\mathcal{Q}_0 = \mathcal{Q}$
2: **for all** $q \in \mathcal{Q}_0$ **do**
3: **while** $|\mathcal{D}_q| > 1$ **do**
4: prendre $a = a_1 \dots a_m \in \mathcal{D}_q$
5: ajouter un nouvel état q_a à $\mathcal{Q}$
6: si $q \in \mathcal{F}$, ajouter également q_a à $\mathcal{F}$
7: définir $\mathcal{D}_{q_a} = \{a\}$ et $\mathcal{G}_{q_a} = \emptyset$
8: pour tout $b \in \mathcal{A}$ définir $\delta(q_a, b) = \delta(q, b)$ et ajouter q_a à $\mathcal{G}_{\delta(q,b)}$
9: pour tout $p \in \mathcal{G}_q$, si $\delta(p, a_m) = q$ et si $\delta^{-(m-1)}(p) = a_1 \dots a_{m-1}$ (condition vide si $m = 1$) alors changer $\delta(p, a_m) = q_a$ et ajouter p à $\mathcal{G}_{q_a}$
10: pour tout $p \in \mathcal{G}_q$, si $q \notin \delta(p, \mathcal{A})$ alors enlever q de $\mathcal{G}_q$
11: enlever a de $\mathcal{D}_q$

Fin: $(\mathcal{A}, \mathcal{Q}, s, \mathcal{F}, \delta)$ un AFD non m-ambigu reconnaissant le langage $\mathcal{L}$ et pour tout $q \in \mathcal{Q}$ on a $\delta^{-m}(q) = \mathcal{D}_q$ et $\Delta^{-1}(q) = \mathcal{G}_q$.

ALGORITHME **5:** Lever d'ambiguïté. Cet algorithme transformer un AFD non $(m-1)$-ambigu en un AFD non m-ambigu reconnaissant le même langage.

- soit $a = \mathtt{b} \in \mathcal{D}_0 = \{\mathtt{b}, \mathtt{c}\}$
- on ajoute l'état 14 à $\mathcal{Q}$
- $0 \notin \mathcal{F} = \{1, 2\}$ donc on ne fait rien
- on définit $\mathcal{D}_{14} = \{\mathtt{b}\}$ et $\mathcal{G}_{14} = \emptyset$
- on définit $\delta(14, \mathtt{a}) = 11$, $\delta(14, \mathtt{b}) = \delta(14, \mathtt{c}) = 0$ et on ajoute 0_{b} à $\mathcal{G}_{11}$ ainsi qu'à $\mathcal{G}_0$
- on a $\mathcal{G}_0 = \{0, 14, 7, 8, 13\}$; comme $\delta(0, \mathtt{b}) = \delta(13, \mathtt{b}) = 0$ on remplace ces deux transitions par $\delta(0, \mathtt{b}) = \delta(13, \mathtt{b}) = 14$ et on ajoute 0 et 13 à $\mathcal{G}_{14}$
- comme 0 appartient toujours à $\delta(0, \mathcal{A})$ et à $\delta(0, \mathcal{A})$ on ne fait rien ici
- on enlève $\mathtt{b}$ à $\mathcal{D}_0$ qui est donc désormais un singleton

En procédant de la même façon pour les trois autres états, on obtient finalement l'automate de la figure 5.3 (18 états).

Une nouvelle application du théorème (cette fois ci avec $m = 2$) à partir de cet automate permet d'obtenir (figure 5.4) un automate non 2-ambigu associé au motif (26 états).

On peut utiliser cet automate pour construire une PMC pour ce motif dans le cas d'un modèle M2 de paramètre π. Si on note Π sa matrice de transition on a par exemple $\Pi(12, 4) = \pi(\mathtt{c}, \mathtt{a}, \mathtt{a})$ car $\delta(12, \mathtt{a}) = 4$ et $\delta^{-2}(12) = \{\mathtt{ca}\}$.

Notons qu'il est également possible d'adapter l'heuristique de construction d'AFD de la proposition 56 pour produire un AFD non ambigu :

Proposition 88 (heuristique de construction d'AFD non m-ambigu)**.** Soit $\mathcal{W}$ un motif quelconque sur l'alphabet $\mathcal{A}$ et soit $m \geqslant 1$. On considère $\mathcal{Q}$ la réunion de $\mathcal{A}^0 = \{\varepsilon\}$, $\mathcal{A}^1$, …, $\mathcal{A}^m$ et de l'ensemble de tous les préfixes de $\mathcal{W}$. On pose $s = \varepsilon$, $\mathcal{F} = \mathcal{W} \subset \mathcal{Q}$ et pour tout $q\mathcal{Q}$ et tout $a\mathcal{A}$ on pose

$$\delta(q, a) = \text{le plus long suffixe de } qa \text{ (concaténation de } q \text{ et } a) \text{ dans } \mathcal{Q}.$$

L'AFD $(\mathcal{A}, \mathcal{Q}, s, \mathcal{F}, \delta)$ ainsi construit reconnaît le langage $\mathcal{A}^*\mathcal{W}$.

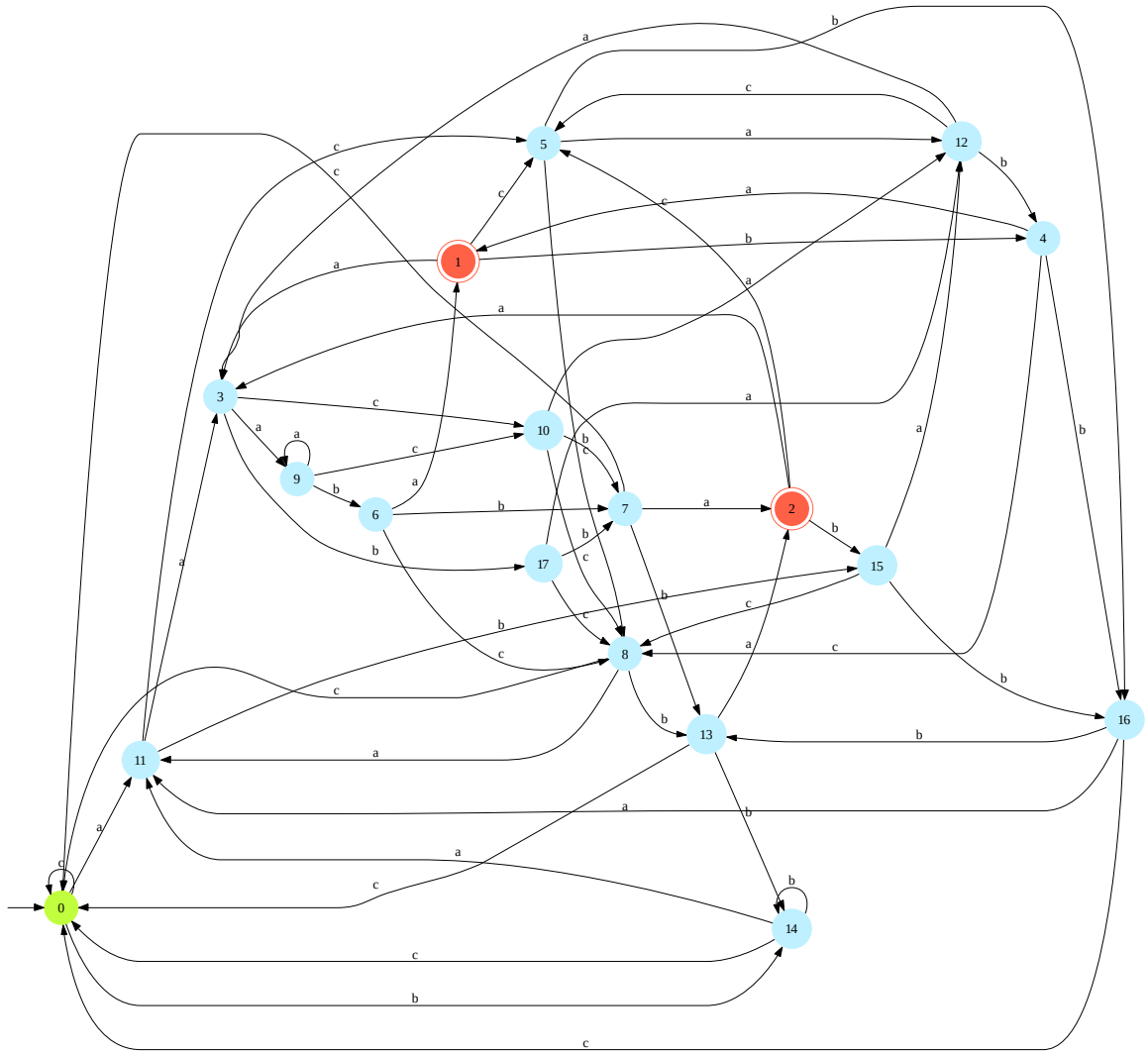


FIG. 5.3 – DFA sans ambiguïté d'ordre 1 reconnaissant le motif `a..ba` sur l'alphabet $\mathcal{A} = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$. Cet automate a 18 états.

Remarque 89. Notons qu'en général, comme dans le cas de la proposition 56, l'AFD non m-ambigu ainsi produit n'est pas optimal. Dans le cas particulier où le motif $\mathcal{W}$ est réduit à un unique mot de taille $h \geqslant m$, il est néanmoins intéressant de constater que l'utilisation de cet automate (non nécessairement optimal) dans le théorème 86 permet d'obtenir une PMC minimale (de taille $L = k^m + h - m$).

Exemple 90. On considère le motif $\mathcal{W} = \{\mathtt{abba}\}$ sur l'alphabet $\mathcal{A} = \{\mathtt{a}, \mathtt{b}\}$ et on se donne $m = 2$. La proposition 88 nous indique de considérer l'ensemble

$$\mathcal{Q} = \{\varepsilon, \mathtt{a}, \mathtt{b}, \mathtt{aa}, \mathtt{ab}, \mathtt{ba}, \mathtt{bb}, \mathtt{abb}, \mathtt{abba}\}$$

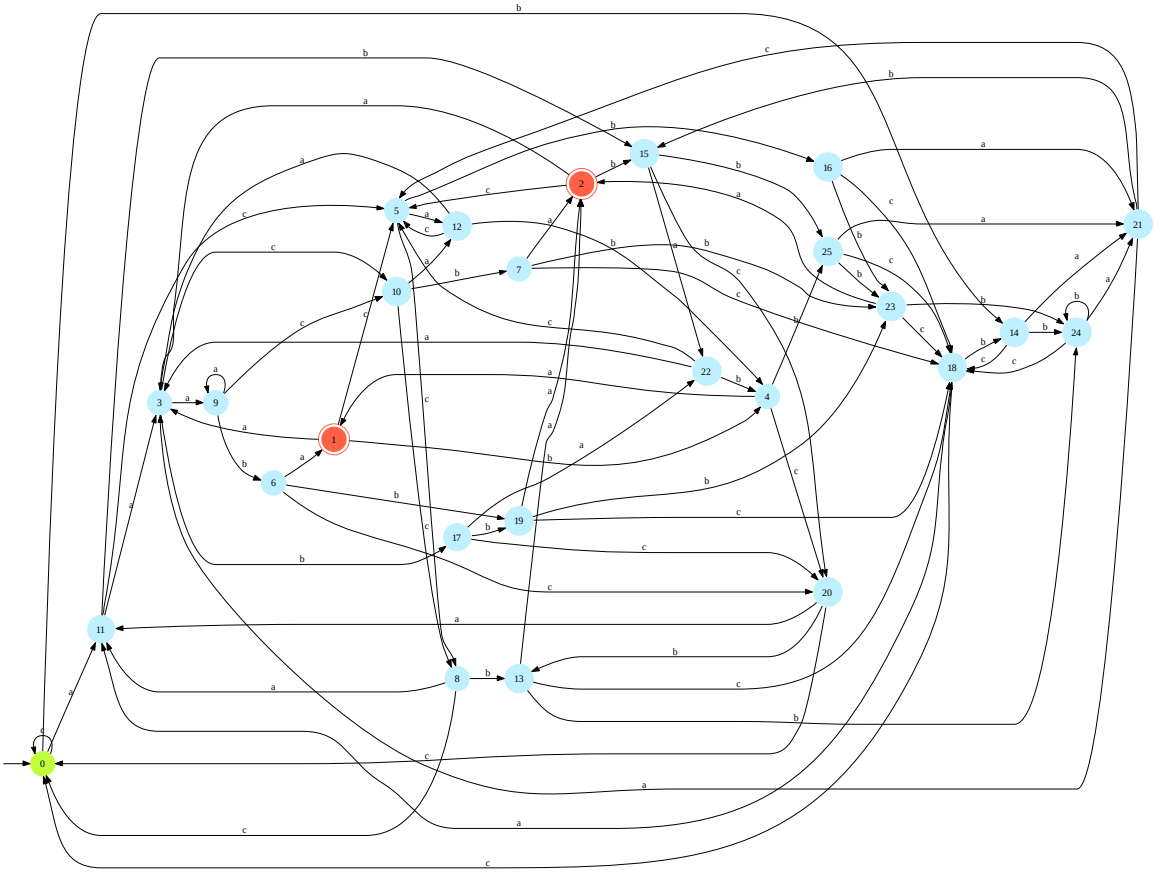


FIG. 5.4 – DFA sans ambiguïté d'ordre 2 reconnaissant le motif `a..ba` sur l'alphabet $\mathcal{A} = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$. Cet automate a 26 états.

et les transitions :

$$\begin{array}{llll}
\delta(\varepsilon, \mathtt{a}) = \mathtt{a} & \delta(\varepsilon, \mathtt{b}) = \mathtt{b} & \delta(\mathtt{a}, \mathtt{a}) = \mathtt{aa} & \delta(\mathtt{a}, \mathtt{b}) = \mathtt{ab} \\
\delta(\mathtt{b}, \mathtt{a}) = \mathtt{ba} & \delta(\mathtt{b}, \mathtt{b}) = \mathtt{bb} & \delta(\mathtt{aa}, \mathtt{a}) = \mathtt{aa} & \delta(\mathtt{aa}, \mathtt{b}) = \mathtt{ab} \\
\delta(\mathtt{ab}, \mathtt{a}) = \mathtt{ba} & \delta(\mathtt{ab}, \mathtt{b}) = \mathtt{abb} & \delta(\mathtt{ba}, \mathtt{a}) = \mathtt{aa} & \delta(\mathtt{ba}, \mathtt{b}) = \mathtt{ab} \\
\delta(\mathtt{bb}, \mathtt{a}) = \mathtt{ba} & \delta(\mathtt{bb}, \mathtt{b}) = \mathtt{bb} & \delta(\mathtt{abb}, \mathtt{a}) = \mathtt{abba} & \delta(\mathtt{abb}, \mathtt{b}) = \mathtt{bb} \\
\multicolumn{2}{c}{\delta(\mathtt{abba}, \mathtt{a}) = \mathtt{aa}} & \multicolumn{2}{c}{\delta(\mathtt{abba}, \mathtt{b}) = \mathtt{ab}}
\end{array}$$

On obtient ainsi l'automate de la figure 5.5 à comparer avec l'automate minimal de la figure 5.6. Les deux automates produisent néanmoins une PMC sur un espace d'état de cardinal 6.

En revanche, si on applique l'heuristique de construction au motif $\mathcal{W} = \{\mathtt{a..ba}\}$ sur l'alphabet $\mathcal{A} = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$ on obtient les cardinaux suivants pour les PMC associées :

m	0	1	2
optimal	13	18	21
heuristique	32	33	36

l'heuristique est donc alors loin de la solution optimale.

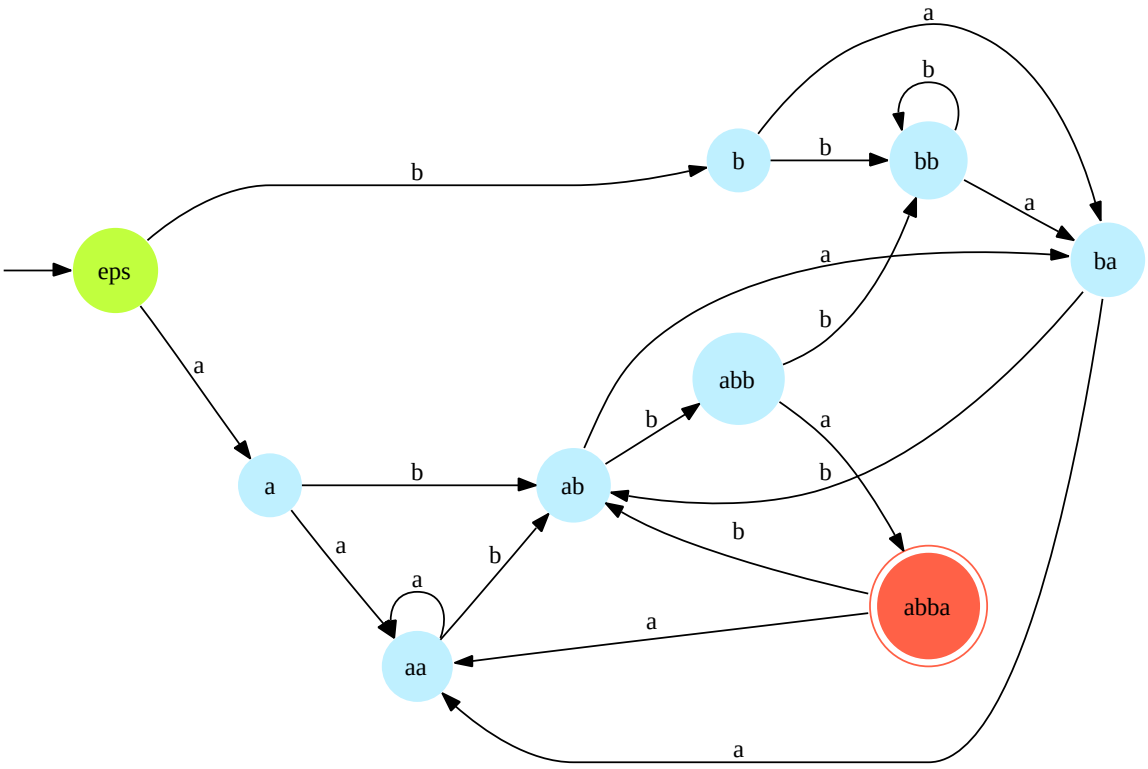


FIG. 5.5 – DFA sans ambiguïté d'ordre 2 reconnaissant le motif `abba` sur l'alphabet $\mathcal{A} = \{\mathtt{a}, \mathtt{b}\}$. Cet automate a 9 états.

5.4 Calculs exacts

Dans cette partie, comme dans la suite, on considère $Y = Y_m \dots Y_\ell$ une PMC associée aux comptages (chevauchants ou renouvelants) d'un motif donné, dans un modèle Mm donné avec $m \geqslant 0$ (le cas M00 est traité comme un cas particulier du cas M0). On note $\Pi = P + Q$ la matrice de transition de Y où Q contient les transitions de comptage (*i. e.* se terminant dans un état de comptage) et P les autres transitions. On note μ_0 la loi initiale de Y qui peut être, selon les cas, une mesure de Dirac (chargeant l'état correspondant au premier mot de m lettres d'une séquence observée), la loi stationnaire du modèle Mm ou bien encore une loi totalement arbitraire.

Exemple 91. Pour illustrer les méthodes proposées, nous allons filer un exemple pratique : on se place dans le modèle M1 sur l'alphabet binaire $\mathcal{A} = \{\mathtt{a}, \mathtt{b}\}$ dont la matrice de transition est

$$\pi = \begin{pmatrix} 0.4 & 0.6 \\ 0.1 & 0.9 \end{pmatrix}$$

La loi stationnaire de ce modèle est $(1/7; 6/7)$ et le module de la seconde valeur propre vaut $\nu = 0.3$.

On s'intéresse aux occurrences chevauchantes du motifs $\mathcal{W} = \{\mathtt{bb}, \mathtt{aba}\}$ dont l'automate minimal (6 états) non 1-ambigu est donné en figure 5.7.

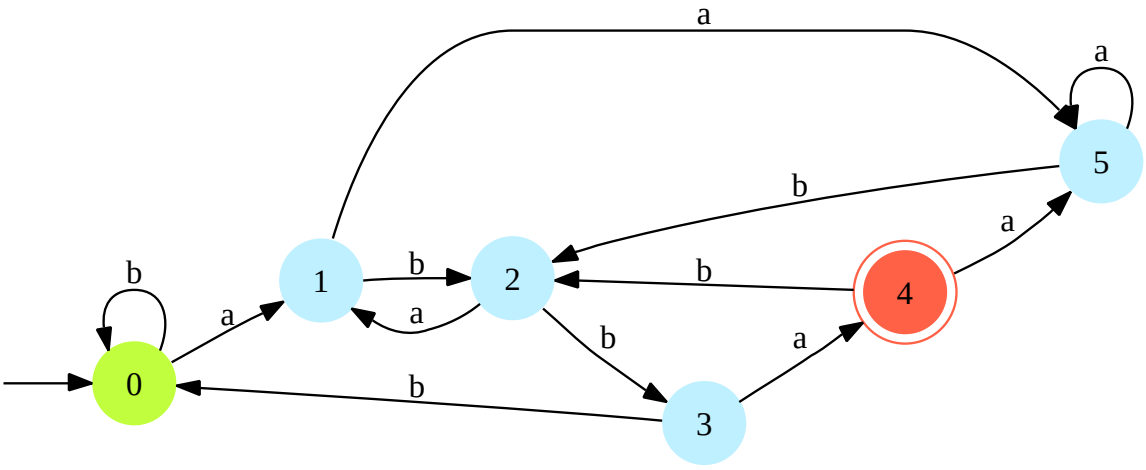


FIG. 5.6 – DFA minimal sans ambiguïté d'ordre 2 reconnaissant le motif `abba` sur l'alphabet $\mathcal{A} = \{\mathtt{a}, \mathtt{b}\}$. Cet automate a 6 états.

La PMC $Y = Y_1 \dots Y_\ell$ associée à ce problème est donc définie sur l'ensemble $\mathcal{Q} = \{1, 2, 3, 4, 5\}$ avec $\mathcal{F} = \{1, 2\}$ (l'état initial 0 n'apparaît pas car $\delta^{-1}(0) = \emptyset$) et dont la matrice de transition $\Pi = P + Q$ est donnée par

$$P = \begin{pmatrix} 0.0 & 0.0 & \mathit{0.1} & 0.0 & 0.0 \\ 0.0 & 0.0 & \mathit{0.4} & \mathit{0.6} & 0.0 \\ 0.0 & 0.0 & \mathit{0.4} & \mathit{0.6} & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & \mathit{0.1} & 0.0 & 0.0 \end{pmatrix} \quad \text{et} \quad Q = \begin{pmatrix} \mathit{0.9} & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ \mathit{0.9} & \mathit{0.1} & 0.0 & 0.0 & 0.0 \\ \mathit{0.9} & 0.0 & 0.0 & 0.0 & 0.0 \end{pmatrix}$$

où les 10 seules transitions non nulles sont indiquées *italique*.

A noter que l'état 5 de notre PMC ne peut être atteint que pour Y_1 et ceci uniquement si $X_1 = \mathtt{b}$. $X_1 = \mathtt{a}$ correspond pour sa part à l'état 3.

5.4.1 *Finite Markov Chain Imbedding*

Définition 92 (FMCI)**.** Pour tout $c \in \mathbb{N}$ on définit la *Finite Markov Chain Imbedding* $Z = Z_m \dots Z_\ell$ de la manière suivante :

$$Z_j = \begin{cases} (Y_j, N_j) & \text{si } N_j < c \\ c_+ & \text{si } N_j \geqslant c \end{cases}$$

où N_j est le nombre d'occurrences du motif dans la séquence $X_1 \dots X_j$.

Comme la PMC Y est une chaîne de Markov d'ordre un, Il est facile de voir que la FMCI Z ainsi construite est aussi une chaîne de Markov. En fait, on se contente de garder la mémoire du nombre d'occurrences observées et on s'arrête de compter lorsque l'on a atteint c occurrences (c'est l'état absorbant c_+).

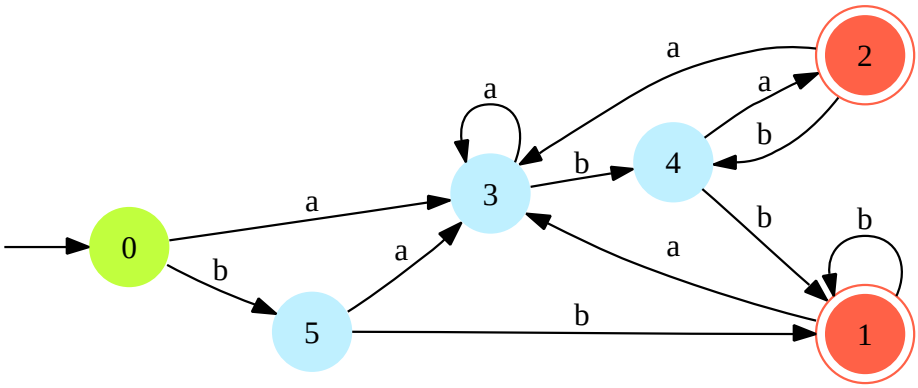


FIG. 5.7 – DFA sans ambiguïté d'ordre 1 reconnaissant les occurrences chevauchantes du motif $\mathcal{W} = \{\mathtt{bb}, \mathtt{aba}\}$ sur l'alphabet binaire $\mathcal{A} = \{\mathtt{a}, \mathtt{b}\}$. Cet automate a 6 états.

Proposition 93. Si on ordonne l'espace d'état de Z par N_j croissants, puis par Y_j et si l'on place c_+ en dernière position (ainsi, on a l'ordre suivant : $(1,0)$, …, $(L,0)$, $(1,1)$, …, $(L,1)$, …) on obtient une matrice de transition définie par blocs :

$$T = \left(\begin{array}{ccc|c} & R & & v \\ \hline 0 & \cdots & 0 & 1 \end{array}\right)$$

où $R = (R_{i,j})_{0 \leqslant i,j < c}$ de dimension $cL \times cL$ est formé de $c \times c$ blocs de taille $L \times L$ et $v = (v_i)_{0 \leqslant i < c}$ de dimension $cL \times 1$ de c blocs de taille $L \times 1$ vérifiant :

$$R_{i,j} = \begin{cases} P & \text{si i=j} \\ Q & \text{si i+1=j} \\ 0 & \text{sinon} \end{cases} \quad \text{et} \quad v_i = \begin{cases} 0 & \text{si } 0 \leqslant i < c-1 \\ \Sigma_Q = \sum_j Q(\cdot, j) & \text{si } i = c-1 \end{cases}$$

Démonstration. Supposons que $z_i = (y_i, n_i)$ avec $y_i \in \mathcal{Q}$ et $0 \leqslant n_i < c$. En lisant la lettre x_{i+1} de la séquence deux cas de figure se présentent.

Dans le premier cas, $y_{i+1} \notin \mathcal{F}$: pas de nouvelle occurrence de motif donc $n_{i+1} = n_i$. On n'a aucune chance d'atteindre l'état final c_+ et la transition de y_i à y_{i+1} (qui s'écrit dans le bloc $R_{n,n}$ avec $n = n_i$) se lit nécessairement dans P.

Dans le deuxième cas, $y_{i+1} \in \mathcal{F}$, on a donc une occurrence du motif que se termine en position $i+1$. La probabilité de transition de y_i à y_{i+1} se lit donc dans la matrice Q. Par ailleurs, on a nécessairement $n_{i+1} = n_i + 1$. Si $n_{i+1} = c$ on a atteint l'état final absorbant c_+ (la transition s'écrit donc dans le bloc v_c de v), sinon, on se trouve dans l'état $z_{i+1} = (y_{i+1}, n_{i+1})$ (la transition s'écrit dans le bloc $R_{n,n+1}$ avec $n = n_i$). □

Exemple 94. Avec $c = 3$ on trouve donc

$$T = \left(\begin{array}{ccc|c} P & Q & 0 & 0 \\ 0 & P & Q & 0 \\ 0 & 0 & P & \Sigma_Q \\ \hline 0 & 0 & 0 & 1 \end{array}\right)$$

où on obtient $\Sigma_Q = (0.9; 0; 0; 1; 0.9)'$ en sommant Q sur les colonnes.

Si on suppose que $N_m = 0$ c'est à dire que les m premières lettres de la séquence X ne contiennent pas d'occurrence, on obtient le résultat suivant :

Théorème 95.

$$\mathbb{P}(N_\ell = i) = M_0 T^{\ell-m} E_i \text{ pour } 0 \leqslant i < c \quad \text{et} \quad \mathbb{P}(N_\ell \geqslant c) = M_0 T^{\ell-m} E_{c_+}$$

avec

$$M_0 = (\mu_0\, 0 \cdots 0 \,|\, 0); \quad E_i = (0 \cdots 0\, 1\, 0 \cdots 0 \,|\, 0)'; \quad \text{et} \quad E_{c_+} = (0 \cdots 0 \,|\, 1)'$$

où le bloc de '1' dans E_i se trouve en $i^{\text{ème}}$ position.

Démonstration. Comme $N_m = 0$, la distribution de Z_m est donnée par le vecteur M_0 défini ci-dessus. Par conséquent, $M_0 T^{\ell-m}$ donne la distribution de Z_ℓ. Pour obtenir un résultat similaire, il ne reste plus qu'à sommer les contributions pour $N_\ell = i$ (L termes) ou bien l'unique terme correspondant à $N_\ell \geqslant c$ (état absorbant c_+). □

Dans les cas plus complexes où N_m n'est pas nécessairement nul, il suffit de modifier la distribution initiale M_0 en conséquences p.

Exemple 96. On considère le cas particulier où $X_1 \equiv \mathtt{a}$, $\ell = 10$ et $c = 3$. Comme $X_1 = \mathtt{a}$ correspond à l'état 3 on a $\mu_0 = (0\,0\,1\,0\,0)$. On effectue les calculs et on trouve :

$$\mathbb{P}(N_{10} = 0) = M_0 T^9 E_0 = 0.00065536 \quad \mathbb{P}(N_{10} = 1) = M_0 T^9 E_1 = 0.006242304$$

$$\mathbb{P}(N_{10} = 2) = M_0 T^9 E_2 = 0.024558336 \quad \mathbb{P}(N_{10} \geqslant 3) = M_0 T^9 E_{3_+} = 0.968544.$$

5.4.2 Algorithmes

Dans cette partie, nous allons voir comment effectuer efficacement ces calculs. Pour cela, on remarque tout d'abord que la matrice T (d'ordre $n = cL + 1$) est très creuse. Comme ce n'est évidemment pas le cas de ses puissances, il serait catastrophique de penser à calculer d'abord $T^{\ell-m}$ (complexité cubique avec n). À la place, on va plutôt tirer avantage de la structure de T en calculant récursivement $M_0 T^{\ell-m}$ à l'aide de $\ell - m$ produit (à gauche) de type vecteur × matrice. Voici l'algorithme résultant de cette idée :

Proposition 97. Soit $u = (u_0 \ldots u_{c-1} \,|\, u_{c_+})$ un vecteur (défini par blocs) de taille $cL+1$.
- *initialisation* : $u_0 = \mu_0$, $u_j = 0$ $(1 \leqslant j < c)$, $u_{c_+} = 0$
- *boucle principale* : pour $i = (m+1) \ldots \ell$:
  - i) $u_{c_+} = u_{c_+} + u_{c-1}\Sigma_Q$
  - ii) pour $j = (c-1) \ldots 1$: $u_j = u_j P + u_{j-1} Q$
  - iii) $u_0 = u_0 P$

– *fin* : u contient $M_0T^{\ell-m}$ (la loi de Z_ℓ)

La complexité de cet algorithme est $O(cL+1)$ en mémoire et $O(kcL\ell)$ en temps (k désignant la taille de l'alphabet).

Démonstration. Il suffit d'écrire le produit par blocs du vecteur x avec la matrice T telle que définie en proposition 93 pour obtenir les étapes i), ii) et iii). Attention cependant à bien les effectuer dans l'ordre indiqué. Pour un i donné, on a donc c opérations de la forme $z = xP + yQ$ à effectuer, chacune étant de complexité $O(kL)$ (c'est le nombre de termes non nuls dans $P+Q$). En répétant ce traitement pour tous les i on obtient la complexité finale. □

Une fois $M_0T^{\ell-m}$ calculé, il ne reste plus qu'à sommer de manière adéquate ses termes (produits scalaires avec les E_i et E_{c_+}) pour obtenir $\mathbb{P}(N_\ell \geqslant c)$, $\mathbb{P}(N_\ell = i)$ avec $1 \leqslant i < c$ ou encore $\mathbb{P}(N_\ell < c)$.

Attention cependant à ne pas commettre l'erreur de calculer $\mathbb{P}(N_\ell < c)$ via la formule

$$\mathbb{P}(N_\ell < c) = 1 - \mathbb{P}(N_\ell \geqslant c).$$

Si une telle chose ne pose aucun problème en arithmétique exacte, il est très maladroit d'y recourir en arithmétique approchée (avec des réels à virgule flottante par exemple). En effet, avec une telle arithmétique, tous les calculs s'obtiennent avec une erreur relative donnée (de l'ordre de 10^{-12} par exemple). Pour le calcul d'un nombre proche de 1, on a donc une erreur absolue du même ordre de grandeur qui se répercute donc sur le résultat en faisant la différence avec 1. Ainsi, un arithmétique approchée, il est possible que $1-(1-10^{-20}) \neq 10^{-20}$ (et donne souvent un résultat aléatoire). Pour éviter ce problème, on calculera donc plutôt $\mathbb{P}(N_\ell < c)$ via la formule :

$$\mathbb{P}(N_\ell < c) = \mathbb{P}(N_\ell = 0) + \ldots + \mathbb{P}(N_\ell = c-1)$$

Exemple 98. On reprend le calcul de l'exemple 96 mais en utilisant cette fois-ci l'algorithme de la proposition 97.

On a $u = (u_0\, u_1\, u_2 \,|\, u_{3_+})$ un vecteur de taille 16 qu'on initialise à $u_0 = \mu_0$, $u_1 = u_2 = 0$ et $u_{3_+} = 0$. $\Sigma_Q = (0.9; 0; 0; 1; 0.9)'$.

– pour $i = 1$ on effectue :

  i) $u_{3_+} = u_{3_+} + u_2\Sigma_Q = 0$

  ii) – pour $j = 3$: $u_2 = u_2P + u_1Q = (0;0;0;0;0)$
  – pour $j = 2$: $u_1 = u_1P + u_0Q = (0;0;0;0;0)$

  iii) $u_0 = u_0P = (0; 0; 0.4; 0.6; 0)$

– pour $i = 2$ on effectue :

  i) $u_{3_+} = u_{3_+} + u_2\Sigma_Q = 0$

  ii) – pour $j = 3$: $u_2 = u_2P + u_1Q = (0;0;0;0;0)$
  – pour $j = 2$: $u_1 = u_1P + u_0Q = (0.54; 0.06; 0; 0; 0)$

  iii) $u_0 = u_0P = (0; 0; 0.16; 0.24; 0)$

– pour $i = 3$ on effectue :

  i) $u_{3_+} = u_{3_+} + u_2\Sigma_Q = 0$

ii) – pour $j = 3$: $u_2 = u_2P + u_1Q = (0.486; 0; 0; 0; 0)$
– pour $j = 2$: $u_1 = u_1P + u_0Q = (0.216; 0.024; 0.078; 0.036; 0)$

iii) $u_0 = u_0P = (0; 0; 0.064; 0.096; 0)$

– pour $i = 4$ on effectue :

i) $u_{3_+} = u_{3_+} + u_2\Sigma_Q = 0.4374$

ii) – pour $j = 3$: $u_2 = u_2P + u_1Q = (0.2268; 0.0036; 0.0486; 0; 0)$
– pour $j = 2$: $u_1 = u_1P + u_0Q = (0.0864; 0.0096; 0.0624; 0.0612; 0)$

iii) $u_0 = u_0P = (0; 0; 0.0256; 0.0384; 0)$

– et ainsi de suite jusqu'à $i = 9$:

i) $u_{3_+} = u_{3_+} + u_2\Sigma_Q = 0.968544$

ii) – pour $j = 3$: $u_2 = u_2P + u_1Q =$
$(0.0077138; 0.0006359; 0.0074788; 0.0087299; 0)$
– pour $j = 2$: $u_1 = u_1P + u_0Q =$
$(0.0008847; 0.0000983; 0.0022364; 0.0030228; 0)$

iii) $u_0 = u_0P = (0; 0; 0.0002621; 0.0003932; 0)$

Développements asymptotiques

On va donner ici des développements asymptotiques pour les calculs de $\mathbb{P}(N_\ell = 0)$, $\mathbb{P}(N_\ell \geqslant 1)$. On peut obtenir des développements similaires pour de plus grands nombres d'occurrences, mais les formules sont alors beaucoup plus complexes et leur intérêt semble plus limité.

Dans toute cette partie, pour tout $i \geqslant m$ on note

$$u^i = \left(u_0^i \mid u_{1_+}^i\right) = M_0 T^{i-m}$$

(ainsi, $u^m = M_0$).

Proposition 99. Pour tout $i \geqslant m$ on a

$$u_0^i = \mu_0 P^{i-m} \quad \text{et} \quad u_{1_+}^{i+1} = u_0^i \Sigma_Q + u_{1_+}^i.$$

Démonstration. C'est une simple application de la proposition 97 avec $c = 1$. □

Le théorème de Perron-Frobénius permet d'obtenir le lemme suivant :

Lemme 100. Si P est diagonalisable et primitive sur son support (hypothèses non restrictives en pratique) alors :

i) $0 < \lambda < 1$ est la valeur propre strictement dominante (en module) de P

ii) $0 \leqslant \nu < \lambda$ est le module de la seconde valeur propre de P

iii) il existe une matrice P_∞ telle que $\forall j \geqslant 1$,

$$d(P^j, \lambda^j P_\infty) = O(\nu^j)$$

.

Dans le cas particulier où $\nu = 0$ on a même égalité entre P^j et $\lambda^j P_\infty$ à partir d'un certain rang.

Exemple 101. Avec notre matrice P on trouve $\lambda = 0.4$, $\nu = 0$ et

$$P_\infty = \begin{pmatrix} 0 & 0 & 0.25 & 0.375 & 0 \\ 0 & 0 & 1 & 1.5 & 0 \\ 0 & 0 & 1 & 1.5 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.25 & 0.375 & 0 \end{pmatrix}$$

et $P^i = \lambda^i P_\infty$ dès que $i \geqslant 2$.

On utilisant ce lemme on obtient donc un corollaire à la proposition 99 :

Corollaire 102. Soit $\alpha \geqslant 1$ alors pour $i \geqslant \alpha + m$ on a

$$u_0^i = \lambda^{i-m}\mu_0\Pi_\infty + O(\nu^\alpha)$$

et pour tout $i \geqslant \alpha + m + 1$

$$u_{1_+}^i = u_{1_+}^{\alpha+m} + \lambda^\alpha \frac{1-\lambda^{i-\alpha-m}}{1-\lambda}\mu_0 P_\infty \Sigma_Q + O(\nu^\alpha)$$

Démonstration. Par simple application du lemme on remplace Π^{i-m} par $\lambda^{i-m}\Pi_\infty$ dans l'expression de u_0^i avec une erreur en $O(\nu^{i-m})$ (et donc, *a fortiori* avec une erreur majorée par $O(\nu^\alpha)$ car $i \geqslant \alpha + m$). Grâce à la relation de récurrence sur u_{1_+} on écrit pour tout $i \geqslant \alpha + m + 1$

$$u_{1_+}^i = u_{1_+}^{\alpha+m} + \sum_{d=\alpha+m}^{i} u_0^d$$

il ne reste plus qu'à sommer les approximations et les erreurs pour obtenir le résultat. □

Exemple 103. Utilisons ce résultat pour calculer $\mathbb{P}(N_\ell = 0)$. Comme on a $\alpha = 2$ on obtient $u_0^\ell = \lambda^{\ell-m}\mu_0 P_\infty$ avec $\lambda = 0.4$ et $\mu_0 P_\infty = (0; 0; 1; 1.5; 0)$. En faisant la somme des termes de u_0^ℓ on obtient donc

$$\mathbb{P}(N_\ell = 0) = 2.5 \times 0.4^{\ell-1}$$

En passant à l'échelle logarithmique on trouve donc

ℓ	125	250	500	1 000
$\log_{10}\mathbb{P}(N_\ell = 0)$	−49.344561	−99.087062	−198.57206	−397.54207

5.4.3 Temps d'attente

Il est également très facile d'utiliser les PMC pour calculer des temps d'attente.

Pour tout $i \geqslant m$ et $q \in \mathcal{Q}$ on définit $\tau_i(q)$ le temps d'attente du premier $Y_j \in \mathcal{F}$ pour $j > i$ sachant que $Y_i = q$; il s'agit donc du temps d'attente de la première occurrence du motif partant de $Y_i = q$.

Proposition 104. Pour tout $q \in \mathcal{Q}$, tout $f \in \mathcal{F}$ et tout $t \in \mathbb{N}$ on a :

$$\mathbb{P}(\tau_i(q) = t) = e_q P^{t-1} Q e'_{\mathcal{F}} \quad \text{et} \quad \mathbb{P}(\tau_i(q) = t, Y_{i+t} = f) = e_q P^{t-1} Q e'_f$$

et on a également la probabilité conditionnelle :

$$\mathbb{P}(Y_{i+t} = f | \tau_i(q) = t) = \frac{e_q P^{t-1} Q e'_f}{e_q P^{t-1} Q e'_{\mathcal{F}}}$$

où e_I désigne pour tout $I \subset \mathcal{Q}$ le vecteur ligne nul partout sauf au positions de I où il est égal à 1.

Démonstration. Par définition, l'événement $\{\tau_i(q) = t\}$ apparaît uniquement si on effectue $t-1$ transitions vers des éléments de $\mathcal{Q} \setminus \mathcal{F}$ (transitions de P) et une dernière transition vers un élément de $\mathcal{F}$ (transition de Q). Si ce dernier élément est f on obtient la loi du couple. La probabilité conditionnelle provient d'une simple application de la formule de Bayes. □

Exemple 105. Considérons le temps d'attente $\tau_1(3)$ (resp. $\tau_1(5)$) de la première occurrence du motif partant de $X_1 = \texttt{a}$ (resp. $X_1 = \texttt{b}$). Une simple application des formules introduites ci-dessus donne la table suivante :

t	1	2	3	4	5	6	7	8
$\mathbb{P}(\tau_1(3) = t)$	0	0.6	0.24	0.096	0.0384	0.01536	0.006144	0.0024576
$\mathbb{P}(\tau_1(5) = t)$	0.9	0	0.06	0.024	0.0096	0.00384	0.001536	0.0006144

Si on calcule la répartition conditionnelle de deux états finaux 1 et 2, les calculs donnent $\mathbb{P}(Y_{1+t} = 1|\tau_1(3) = t) = \mathbb{P}(Y_{1+t} = 1|\tau_1(5) = t) = 0.9$ pour tout $t \geqslant 3$ ce qui signifie que, quel que soit le point de départ, la première occurrence du motif (apparaissant après au moins 3 lettres) est un `bb` (état final 1) dans 90% des cas.

Étudier la répartition des motifs

Pour $1 \leqslant j \leqslant N$, on introduit T_j le temps d'attente entre la $(j-1)^{\text{ème}}$ occurrence (par convention, le début de la séquence si $j = 1$) et la $j^{\text{ème}}$ occurrence. La valeurs de Y_i au point où se termine la $j^{\text{ème}}$ occurrence est noté $F_j \in \mathcal{F}$. On note toujours μ_0 la loi de Y_m.

Supposons qu'on observe n occurrences dans une séquence donnée et l'on note $(t_i, f_i)_{1 \leqslant i \leqslant n}$ les valeurs observées des variables aléatoires (T_i, F_i).

Au lieu de chercher à calculer globalement le degré de significativité correspondant à l'observation n pour la variable aléatoire N, on considère séparément chaque occurrence du motif.

Proposition 106. Pour la première occurrence, on obtient

$$\mathbb{P}(T_1 \leqslant t_1, F_1 = f_1) = \sum_{t \leqslant t_1} \mu_0 P^{t-1} Q e'_{f_1}$$

et pour $j \geqslant 2$ on a

$$\mathbb{P}(T_j \leqslant t_j, F_j = f_j | F_{j-1} = f_{j-1}) = \sum_{t \leqslant t_1} e_{f_{j-1}} P^{t-1} Q e'_{f_j}$$

Démonstration. Il suffit de sommer les bons termes de la forme $\mathbb{P}(\tau_i(q) = t, Y_{i+t} = f)$ pour obtenir le résultat □

En utilisant le fait que $(I-P)^{-1} = \sum_{t>1} P^{t-1}$ on en tire des formules plus compactes (mais nécessitant des calculs matriciels : puissance, inverse) :

Corollaire 107. Pour la première occurrence, on obtient

$$\begin{aligned} \mathbb{P}(T_1 \leqslant t_1, F_1 = f_1) &= \mu_0 (I - P^{t_1})(I-P)^{-1} Q e'_{f_1} \\ \mathbb{P}(T_1 > t_1, F_1 = f_1) &= \mu_0 P^{t_1} (I-P)^{-1} Q e'_{f_1} \end{aligned}$$

et pour $j \geqslant 2$ on a

$$\begin{aligned}\mathbb{P}(T_j \leqslant t_j, F_j = f_j | F_{j-1} &= f_{j-1}) = e_{f_{j-1}}(I - P^{t_j})(I - P)^{-1} Q e'_{f_j} \\ \mathbb{P}(T_j > t_j, F_j = f_j | F_{j-1} = f_{j-1}) &= e_{f_{j-1}} P^{t_j} (I - P)^{-1} Q e'_{f_j}\end{aligned}$$

Remarquons que, pour $t_1 = \ell$, on trouve la significativité de l'événement {aucune occurrence du motif n'est observée sur une séquence de longueur ℓ}.

Exemple 108. On suppose que $\mu_0 = e_3$ (c'est à dire que $X_1 \equiv \mathtt{a}$) et que l'on observe 4 occurrences du motif avec : $(t_1 = 6, f_1 = 1)$, $(t_2 = 1, f_2 = 1)$, $(t_3 = 15, f_3 = 2)$ et $(t_4 = 5, f_4 = 1)$.

On trouve alors :

$$\mathbb{P}(T_1 > 6, F_1 = 1) = 0.01024; \quad \mathbb{P}(T_2 > 1, F_2 = 1 | F_1 = 1) = 0.0909091;$$

$$\mathbb{P}(T_3 > 15, F_3 = 2 | F_2 = 1) = 0.0000067; \quad \mathbb{P}(T_4 > 5, F_4 = 1 | F_3 = 2) = 0.0256.$$

Le fait que les probabilités trouvées soient assez faibles (la troisième en particulier) indique que les occurrences du motif sont ici peu fréquentes.

Simuler la répartition des motifs

On peut également se servir de lois des temps d'attente pour simuler des répartitions de motifs sans utiliser la séquence X. La complexité résultante étant proportionnelle au nombre d'occurrences produites, on est toujours gagnant par rapport à la complexité $O(\ell)$ nécessaire pour simuler X. Le gain sera d'autant plus fort que les occurrences du motif seront rares.

La proposition suivante détaille la démarche à suivre :

Proposition 109. Soit μ_0 la loi de Y_m. La loi de T_1 est donnée par

$$\mathbb{P}(T_1 = t_1) = \mu_0 P^{t_1 - 1} Q e'_{\mathcal{F}}$$

et celle de F_1 par

$$\mathbb{P}(F_1 = f_1 | T_1 = t_1) = \frac{\mu_0 P^{t_1 - 1} Q e'_{f_1}}{\mu_0 P^{t_1 - 1} Q e'_{\mathcal{F}}}$$

Puis, pour tout $j \geqslant 2$ la loi de T_j est

$$\mathbb{P}(T_j = t_j | T_{j-1} = t_{j-1}) = e_{f_{j-1}} P^{t_1 - 1} Q e'_{\mathcal{F}}$$

et celle de F_j par

$$\mathbb{P}(F_j = f_j | T_j = t_j, F_{j-1} = f_{j-1}) = \frac{e_{f_{j-1}} P^{t_j - 1} Q e'_{f_j}}{e_{f_{j-1}} P^{t_j - 1} Q e'_{\mathcal{F}}}$$

On arrête simplement les tirages lorsque $\sum_{j>0} T_j > \ell - m$.

Exemple 110. Nous allons simuler la répartition des occurrences du motif sur une séquence de longueur $\ell = 12$ en utilisant $\mu_0 = e_3$ ($X_1 = \mathtt{a}$).

La loi de T_1 est donnée par la table suivante :

t	1	2	3	4	5	6	7	8
$\mathbb{P}(T_1 \leqslant t)$	0.	0.6	0.84	0.936	0.9744	0.98976	0.995904	0.9983616

On tire $t_1 = 4$.

La loi (conditionnelle) de F_1 est alors donnée par $\mathcal{P}(F_1 = 1) = 0.9$ et $\mathcal{P}(F_1 = 2) = 0.1$. On tire $f_1 = 1$.

La loi (conditionnelle) de f_2 est alors donnée par

t	1	2	3	4	5	6	7	8
$\mathbb{P}(T_2 \leqslant t)$	0.9	0.9	0.96	0.984	0.9936	0.99744	0.998976	0.9995904

On tire $t_2 = 1$.

La loi de F_2 est alors donnée par $\mathcal{P}(F_1 = 1) = 1.0$ et $\mathcal{P}(F_1 = 2) = 0.0$, on a donc $f_2 = 1$.

La loi de T_3 est donc la même loi que T_2. On tire $t_3 = 5$. Les calculs donnent pour F_3 la même loi que pour F_1. On tire $f_3 = 2$.

La loi (conditionnelle) de T_4 est alors donnée par

t	1	2	3	4	5	6	7	8
$\mathbb{P}(T_4 \leqslant t)$	0	0.6	0.84	0.936	0.9744	0.98976	0.995904	0.9983616

On tire $t_4 = 2$.

Comme $t_1 + t_2 + t_3 + t_4 = 12 > 11 = \ell - m$ on arrête le processus et on obtient finalement 3 occurrences du motif : $(t_1 = 4, f_1 = 1)$, $(t_2 = 1, f_2 = 1)$ et $(t_3 = 5, f_3 = 2)$.

Considérations numériques

Les calculs de sommes de la forme

$$S_{t_0}^{t_1}(x, y) = \sum_{t=t_0}^{t_1} xP^tQy' \quad x, y \in \mathbb{R}^L \quad 0 \leqslant t_0 \leqslant t_1 \leqslant \infty$$

se trouvent au coeur de toutes ces méthodes ; comment faire efficacement de tels calculs ?

La première solution (déjà évoquée ci-dessus) consiste à sommer d'abord les puissances de P à l'aide des formule classique de somme des termes d'une suite géométrique. On trouve ainsi

$$S_{t_0}^{t_1}(x, y) = xP^{t_0}(I - P^{t_1 - t_0 + 1})(I - P)^{-1}Qy'$$

avec la convention $P^\infty = 0$ car P est strictement sous-stochastique.

Cette approche faisant intervenir des calculs explicites de produits et d'inverses de matrices, la complexité résultante est en $O(L^2)$ en mémoire et $O(L^3)$ en temps. Pour des valeurs de L relativement petites, cette méthode très simple est donc parfaitement utilisable mais lorsque L devient grand ($L \geqslant 1\,000$ par exemple) elle devient particulièrement inefficace.

Il est alors préférable de tirer parti de la structure creuse des matrices P et Q (au plus kL termes non nul dans $P + Q$) et c'est précisément ce que nous allons voir ici.

Proposition 111. Si on note $0 < \lambda < 1$ la plus grande valeur propre de P alors il existe une matrice P_∞ telle que

$$\lim_{t \to \infty} \frac{P^t}{\lambda^t} = P_\infty$$

et par conséquent, pour α assez grand on peut approcher le calcul de $S_{t_0}^{t_1}(x, y)$ par

$$\sum_{t=t_0}^{\alpha-1} xP^tQy' + \lambda^\alpha \frac{1 - \lambda^{t_1 - \alpha + 1}}{1 - \lambda} xP_\infty Qy'$$

Démonstration. Le résultat sur les puissances de P est une simple application du lemme 100 et la suite est évidente. □

Corollaire 112. Voici un algorithme permettant de calculer $S_{t_0}^{t_1}(x,y)$. Soit u,v et w trois vecteurs (colonne) de taille L.

- *initialisation* :
  - $v = Qy'$ et $w = 0$
  - pour $t = 1 \dots (t_0 - 1)$:
    - $u = v$ et $v = Pv$
    - si $d(v, \lambda u)$ assez petite (pour un λ donné) alors aller à *fin*
- *boucle principale* :
  - $u = v$, $v = Pv$ et $w = w + v$
  - si $v = \lambda u$ alors aller à *fin*
- *fin* :
  - si $t < t_0$ alors
$$w = \lambda^{t_0 - t} \frac{1 - \lambda^{t_1 - t_0 + 1}}{1 - \lambda} v$$
  - si $t_0 \leqslant t < t_1$ alors
$$w = w + \lambda \frac{1 - \lambda^{t_1 - t}}{1 - \lambda} v$$
  - retourner xw

À noter qu'on peut, sur le même principe, écrire un algorithme similaire utilisant des produits à gauche en lieu et place des produits à droite. Son écriture est laissée au lecteur.

Démonstration. Au cours de l'algorithme on a $v = P^t Qy'$, $u = P^{t-1} Qy'$ et $w = \sum_{i=t_0}^{t} P^i Qy'$. Lorsque $v = \lambda u$ on sait que l'on est arrivé au rang à partir duquel P^t/λ^t est constant. Il ne reste plus qu'à sommer les termes restant en utilisant $P^{t+i} Qy' = \lambda^i P^t Qy'$. □

Exemple 113. On va utiliser cet algorithme pour calculer $S_{16}^{\infty}(e_1, e_2)$.

- *initialisation* :
  - $u = Qe_2' = (0; 0; 0; 0.1; 0)'$
  - $t = 1 : u = v$, $v = Pv = (0; 0.06; 0.06; 0; 0)'$
  - $t = 2 : u = v$, $v = Pv = (0.006; 0.024; 0.024; 0; 0.006)'$
  - $t = 2 : u = v$, $v = Pv = 0.4u$
- *fin*
- $3 = t < t_0 = 16$ et $\lambda = 0.4$
$$w = 0.4^{13}/(1 - 0.4)v$$
- le résultat est $e_1 w = 2.684 \times 10^{-8}$

En faisant directement le calcul matriciel $e_1 P^{16} (I - P)^{-1} Qe_2'$ on trouve exactement le même résultat.

De la même façon, nous laissons au lecteur le soin de vérifier qu'on trouve bien $S_2^{\infty}(e_1, e_2) = 0.01$.

5.4.4 Moments

Dans cette partie, on se propose de calculer de manière exacte les deux premiers moments de

$$N_\ell = \sum_{i=m}^{\ell} \mathbb{I}_{Y_i \in \mathcal{F}}$$

où $Y = Y_m \dots Y_\ell$ est la PMC de distribution initiale μ_0 et de matrice de transition $\Pi = P + Q$.

On note μ la distribution stationnaire de Π et ν la valeur absolue de la deuxième valeur propre.

Espérance

Notation 114. Pour tout $a \in \mathcal{Q}$, on note $N_\ell(a)$ le nombre d'occurrences de a dans $Y_m \dots Y_\ell$ et $E_\ell(a) = \mathbb{E}[N_\ell(a)]$ l'espérance de ce comptage.

Proposition 115.

$$E_\ell(a) = \sum_{i=m}^{\ell} \mu_0 \Pi^{i-m} e'_a$$

où e_a est le vecteur indicateur de l'état a. Si de plus $\mu_0 = \mu$ alors $E_\ell(a) = (\ell - m + 1)\mu(a)$.

Démonstration. La première partie de la proposition est évidente car $\mu_0 \Pi^{i-m}$ est la loi de Y_i. La seconde partie utilise simplement le fait que $\mu \Pi^{i-m} = \mu$ pour tout i. □

Si le cas particulier $\mu_0 = \mu$ est intéressant pour sa simplicité, il faut bien avoir conscience qu'il ne se produit pas en pratique car la distribution initiale μ_0 ne peut, par définition, charger que $\delta(s, \mathcal{A}^m)$.

Requis: x un vecteur de taille L et y un réel
1: $x = \mu_0$ et $y = 0$ // *initialisation*
2: **for** $i = m \dots \ell$ **do**
3: $\quad y = y + x \times e'_{\mathcal{F}}$
4: $\quad x = x \times \Pi$
5: $\mathbb{E}(N_\ell) = y$

ALGORITHME **6:** calcul exact de $\mathbb{E}(N_\ell)$. Complexités : $O(L)$ en espace et $O(k \times L \times \ell)$ en temps.

L'algorithme 6 implémente la formule de la proposition 115. l'essentiel des calculs se concentre dans les produits $x \times \Pi$ (ligne 4) qui, en tenant compte de la structure creuse de Π requièrent chacun $k \times L$ opérations résultant ainsi en une complexité $O(k \times L \times \ell)$ en temps.

En utilisant le fait que $|\mu_0 \Pi^j - \mu| = O(\nu^j)$ on peut réduire ce temps de calcul au prix d'une approximation :

Corollaire 116. Pour tout $\alpha \geqslant 0$ et pour tout $\ell \geqslant m + \alpha$

$$E_\ell(a) = \sum_{i=m}^{m+\alpha-1} \mu_0 \Pi^{i-m} e'_a + (\ell - m - \alpha + 1)\mu(a) + O(\nu^\alpha)$$

Démonstration. Comme l'erreur est en $O(\nu^j)$ on peut établir que

$$\sum_{i=m+\alpha}^{\ell} \left(\mu_0 \Pi^{i-m} e'_a - \mu(a)\right) = O\left(\sum_{i=m+\alpha}^{\ell} \nu^{i-m}\right) = O(\nu^\alpha)$$

□

Requis: $\alpha \geqslant 0$, $\ell \geqslant m + \alpha$, μ la loi stationnaire de Π, x un vecteur de taille L et y un réel
1: $x = \mu_0$ et $y = 0$ // *initialisation*
2: **for** $i = m \dots m + \alpha - 1$ **do**
3: // *boucle vide si* $\alpha = 0$
4: $y = y + x \times e'_{\mathcal{F}}$
5: $x = x \times \Pi$
6: $\mathbb{E}(N_\ell) = y + (\ell - m - \alpha + 1)\mu(\mathcal{F})$

ALGORITHME **7:** calcul approché de $\mathbb{E}(N_\ell)$ avec erreur en $O(\nu^\alpha)$. Complexités : $O(L)$ en espace et $O(k \times L \times \alpha)$ en temps.

L'algorithme 7 implémente ce résultat et permet de diviser le temps de calcul par ℓ/α au prix d'une erreur en $O(\nu^\alpha)$.

Exemple 117. Calculons $\mathbb{E}[N_\ell]$ avec $\ell = 100$. On a $\nu = 0.3$ si bien qu'avec $\alpha = 10$ le calcul approché a une erreur (absolue) de l'ordre de grandeur de 5.9×10^{-6}.

On trouve

$$\sum_{i=1}^{10} \mu_0 \Pi^{i-1} e'_{\mathcal{F}} + (100 - 1 - 10 + 1)\mu(\mathcal{F}) = 76.182874$$

alors que

$$\sum_{i=1}^{100} \mu_0 \Pi^{i-1} e'_{\mathcal{F}} = 76.182857$$

soit une erreur absolue de 1.7×10^{-5} (2.2×10^{-7} en erreur relative).

A titre de comparaison, $(100 - 1 + 1)\mu(\mathcal{F}) = 78.0$.

Variance

Notation 118. Pour tout $a, b \in \mathcal{Q}$, on note $N_\ell(a)$ (resp. $N_\ell(b)$) le nombre d'occurrences de a (resp. b) dans $Y_m \dots Y_\ell$ et $E_\ell(a, b) = \mathbb{E}[N_\ell(a)N_\ell(b)]$.

Théorème 119. On a

$$E_\ell(a, b) = \mathbb{I}_{a=b} \times E_\ell(a) + C_\ell(a, b) + C_\ell(b, a) \tag{5.1}$$

avec

$$C_\ell(a, b) = \sum_{i=m}^{\ell-1} (\mu_0 \Pi^{i-m} e'_a) \sum_{j=i+1}^{\ell} (e_a \Pi^{j-i} e'_b). \tag{5.2}$$

Démonstration. Il suffit d'utiliser les décompositions :

$$N_\ell(a) = \sum_{i=m}^{\ell} \mathbb{I}_{Y_i=a} \quad \text{et} \quad N_\ell(b) = \sum_{i=m}^{\ell} \mathbb{I}_{Y_i=b}.$$

pour obtenir ce résultat de manière immédiate. □

Remarque 120. Pour tout $\mathcal{B} \subset \mathcal{Q}$ il est facile de voir que

$$C_\ell(a, \mathcal{B}) = \sum_{b \in \mathcal{B}} C_\ell(a, b)$$

s'obtient en remplaçant simplement e_b par $e_\mathcal{B}$ dans la définition de $C_\ell(a, b)$.

Corollaire 121. On a donc

$$\mathbb{V}(N_\ell) = 2 \sum_{f \in \mathcal{F}} C_\ell(a, \mathcal{F}) + \mathbb{E}(N_\ell) - \mathbb{E}(N_\ell)^2$$

Démonstration. Comme $N_\ell = \sum_{a \in \mathcal{F}} N_\ell(a)$ il est clair que

$$\mathbb{E}(N_\ell^2) = \sum_{a,b \in \mathcal{F}} E_\ell(a, b)$$

En utilisant (5.1) on obtient donc

$$\mathbb{E}(N_\ell^2) = \sum_{a,b \in \mathcal{F}} \mathbb{I}_{a=b} \times E_\ell(a) + 2 \sum_{a,b \in \mathcal{F}} C_\ell(a, b)$$

ce qui donne immédiatement le corollaire. □

Ce résultat est implémenté dans l'algorithme 8 qui, au prix d'une complexité mémoire plus grande (à cause des vecteurs A_f), permet de calculer à la fois l'espérance et la variance du comptage avec la même complexité en temps que pour le calcul de l'espérance seule (algorithme 6).

Requis: x un vecteur de taille L, $(A_f)_{f \in \mathcal{F}}$ F vecteurs de taille $\ell - m + 1$, $(y_f)_{f \in \mathcal{F}}$ F réels, et z, t deux réels.
1: $x = \mu_0$ et $z = 0$
2: **for** $d = 0 \dots \ell - m$ **do**
3: pour $f \in \mathcal{F}$: $A_f(d) = x \times e'_f$, $z+ = A_f(d)$
4: $x = x \times \Pi$
5: *// on a ainsi* $A_f(d) = \mu_0 \Pi^d e'_f$ *pour* $0 \leqslant d \leqslant \ell - m$
6: $x = e'_\mathcal{F}$, $t = 0$ et $y_f = 0$ pour $f \in \mathcal{F}$
7: **for** $i = (\ell - 1) \dots m$ **do**
8: $x = \Pi \times x$
9: pour $f \in \mathcal{F}$: $y_f+ = e_f x$, $t+ = y_f \times A_f(i - m)$
10: $\mathbb{E}(N_\ell) = z$, $\sum_{f \in \mathcal{F}} C_\ell(f, \mathcal{F}) = t$ et donc $\mathbb{V}(N_\ell) = 2t + z - z^2$

ALGORITHME **8:** calcul exact de $\mathbb{V}(N_\ell)$. Complexités : $O(F \times \ell + L)$ en espace et $O(k \times L \times F \times \ell)$ en temps.

Comme dans le cas du calcul de l'espérance, il est cependant possible de réduire de manière très sensible cette complexité en utilisant une approximation pour le calcul de $C_\ell(a, b)$:

Proposition 122. Pour tout $\ell \geqslant m+\alpha$ on a

$$\begin{aligned} C_\ell(a,b) = & \sum_{d=1}^{\min(\ell-m-\alpha,\alpha-1)} (\ell-m-\alpha+1-d)\mu(a)(e_a\Pi^d e'_b) \\ & + \mathbb{I}_{\{\ell\geqslant m+2\alpha\}} \times \frac{(\ell-m-2\alpha+1)(\ell-m-2\alpha+2)}{2}\mu(a)\mu(b) \\ & + \sum_{i=m}^{m+\alpha-1} (\mu_0\Pi^{i-m}e'_a) \sum_{j=i+1}^{\ell} \begin{cases} e_a\Pi^{j-i}e'_b & \text{si } j-i\leqslant\alpha-1 \\ \mu(b) & \text{si } j-i\geqslant\alpha \end{cases} \\ & + (\ell-m-\alpha+1)O(\nu^\alpha) \end{aligned}$$

Démonstration. On part simplement de l'équation (5.2) et du fait que $\ell \geqslant m+\alpha$ pour décomposer la somme sur $m \leqslant i \leqslant \ell-1$ en deux morceaux : le premier pour $m \leqslant i \leqslant m+\alpha-1$ et le second (éventuellement vide) pour $m+\alpha \leqslant i \leqslant \ell-1$. On obtient ainsi $C_\ell(a,b) = A + B$ avec

$$\begin{aligned} A &= \sum_{i=m}^{m+\alpha-1} (\mu_0\Pi^{i-m}e'_a) \sum_{j=i+1}^{\ell} (e_a\Pi^{j-i}e'_b) \\ &= \sum_{i=m}^{m+\alpha-1} (\mu_0\Pi^{i-m}e'_a) \sum_{j=i+1}^{\ell} \begin{cases} e_a\Pi^{j-i}e'_b & \text{si } j-i\leqslant\alpha-1 \\ \mu(b) & \text{si } j-i\geqslant\alpha \end{cases} + \alpha O(\nu^\alpha) \end{aligned}$$

et

$$B = \sum_{i=m+\alpha}^{\ell-1} (\mu_0\Pi^{i-m}e'_a) \sum_{j=i+1}^{\ell} (e_a\Pi^{j-i}e'_b) = \sum_{d=1}^{\ell-m-\alpha} (e_a\Pi^d e'_b) \sum_{i=m+\alpha}^{\ell-d} (\mu_0\Pi^{i-m}e'_a)$$

or comme $\mu_0\Pi^{i-m}e'_a = \mu(a) + O(\nu^{i-m})$ on obtient que

$$\begin{aligned} B &= \sum_{d=1}^{\ell-m-\alpha} (\ell-m-\alpha+1-d)\mu(a)(e_a\Pi^d e'_b) + O(\nu^\alpha) \times \sum_{d=1}^{\ell-m-\alpha} (e_a\Pi^d e'_b) \\ &= \sum_{d=1}^{\ell-m-\alpha} (\ell-m-\alpha+1-d)\mu(a)(e_a\Pi^d e'_b) + (\ell-m-\alpha)O(\nu^\alpha) \end{aligned}$$

Si on suppose maintenant que $\ell \geqslant m+2\alpha$ on peut aller un peu plus loin en écrivant

$$\begin{aligned} B = \sum_{d=1}^{\alpha-1} (\ell-m-\alpha+1-d)\mu(a)(e_a\Pi^d e'_b) + & \sum_{d=\alpha}^{\ell-m-\alpha} (\ell-m-\alpha+1-d)\mu(a)\mu(b) \\ & + \sum_{d=\alpha}^{\ell-m-\alpha} (\ell-m-\alpha+1-d)\mu(a)O(\nu^d) + (\ell-m-\alpha)O(\nu^\alpha) \end{aligned}$$

et la proposition est ainsi démontrée. □

L'algorithme 9 utilise ce résultat pour réduire sensiblement l'espace requis ainsi que pour diviser le temps de calcul par ℓ/α par rapport à l'algorithme 8, au prix d'une erreur en $O(\ell\nu^\alpha)$.

Requis: $\alpha \geqslant 0$, $\ell \geqslant m + \alpha$, μ la loi stationnaire de Π, x un vecteur de taille L, $(A_f)_{f\in\mathcal{F}}$ et $(B_f)_{f\in\mathcal{F}}$ F vecteurs de taille α, $(y_f)_{f\in\mathcal{F}}$ F réels, et z, t deux réels

1: $x = \mu_0$ et $z = 0$
2: **for** $d = 0 \dots (\alpha - 1)$ **do**
3: pour $f \in \mathcal{F} : A_f(d) = x \times e'_f$, $z+ = A_f(d)$
4: $x = x \times \Pi$
5: $z+ = (\ell - m - \alpha + 1)\mu(\mathcal{F})$
6: $x = e'_{\mathcal{F}}$
7: **for** $d = 0 \dots (\alpha - 1)$ **do**
8: pour $f \in \mathcal{F} : B_f(d) = e_f \times x$
9: $x = \Pi \times x$
10: *// on a ainsi* $A_f(d) = \mu_0\Pi^d e'_f$ *et* $B_f(d) = e_f\Pi^d e'_{\mathcal{F}}$ *pour* $0 \leqslant d < \alpha$
11: $t = 0$ et $y_f = 0$ pour $f \in \mathcal{F}$
12: **for** $d = 1 \dots \min(\ell - m - \alpha, \alpha - 1)$ **do**
13: pour $f \in \mathcal{F} : y_f + B_f(d)$, $t+ = (\ell - m - \alpha + 1 - d)\mu(f)B_f(d)$
14: **if** $\ell \geqslant m + 2\alpha$ **then**
15: pour $a \in \mathcal{F} : y_f+ = (\ell - m - 2\alpha + 1)\mu(\mathcal{F})$
16: $t+ = (\ell - m - 2\alpha + 1)(\ell - m - 2\alpha + 2)\mu(\mathcal{F})^2/2$
17: **for** $i = (m + \alpha - 1) \dots m$ **do**
18: **for** $f \in \mathcal{F}$ **do**
19: si $\ell - i < \alpha$, $y_f+ = B_f(\ell - i)$ sinon $y_f+ = \mu(\mathcal{F})$
20: $t+ = y_f A_f(i - m)$
21: $\mathbb{E}(N_\ell) = z$, $\sum_{f\in\mathcal{F}} C_\ell(f, \mathcal{F}) = t$ et donc $\mathbb{V}(N_\ell) = 2t + z - z^2$

ALGORITHME **9:** calcul approché de $\mathbb{V}(N_\ell)$ avec erreur en $O(\ell\nu^\alpha)$. Complexités : $O(F \times \alpha + L)$ en espace et $O(k \times L \times F \times \alpha)$ en temps.

Exemple 123. On va calculer l'espérance et la variance du comptage pour $\ell = 100$ en utilisant l'algorithme approché avec $\alpha = 5$.

On calcule tout d'abord la loi stationnaire et la valeur absolue de la seconde valeur propre et on obtient :

$$\mu = (0.7714286\ ;\ 0.0085714\,;\ 0.1342857\,;\ 0.0857143;\ 0) \quad \text{et} \quad \nu = 0.3$$

si bien que $\mu(\mathcal{F} = \{1, 2\}) = 0.78$.

Dans la phase d'initialisation, on commence par calculer $A_1(d)$ et $A_2(d)$ pour $0 \leqslant d < \alpha$:

- $x = \mu_0 = (0; 0; 1; 0; 0)$ donc $A_1(0) = 0$ et $A_2(0) = 0$;
- $x = x \times \Pi = (0; 0; 0.4; 0.6; 0)$ donc $A_1(1) = 0$ et $A_2(1) = 0$;
- $x = x \times \Pi = (0.54; 0.06; 0.16; 0.24; 0)$ donc $A_1(2) = 0.54$ et $A_2(2) = 0.06$;
- $x = x \times \Pi = (0.702; 0.024; 0.142; 0.132; 0)$ donc $A_1(3) = 0.702$ et $A_2(3) = 0.024$;
- $x = x \times \Pi = (0.7506; 0.0132; 0.1366; 0.0996; 0)$ donc $A_1(4) = 0.7506$ et $A_2(4) = 0.0132$;

de sorte que

d	0	1	2	3	4
$A_1(d)$	0	0	0.54	0.702	0.7506
$A_2(d)$	0	0	0.06	0.024	0.0132

Au passage, on a calculé $z = \sum_{d=0}^{\alpha-1}(A_1(d) + A_2(d)) = 2.0898$ et on obtient donc finalement $z = z + (100 - 1 - 5 + 1)\mu(\mathcal{F}) = 76.1898$.

De même que pour les vecteurs A_f on va maintenant calculer les vecteurs B_f :

- $x = e'_{\mathcal{F}} = (1; 1; 0; 0; 0)'$ donc $B_1(0) = 1$ et $B_2(0) = 1$;
- $x = x \times \Pi = (0.9; 0; 0; 1; 0.9)'$ donc $B_1(1) = 0.9$ et $B_2(1) = 0$;
- $x = x \times \Pi = (0.81; 0.6; 0.6; 0.81; 0.81)$ donc $B_1(2) = 0.81$ et $B_2(2) = 0.6$;
- $x = x \times \Pi = (0.789; 0.726; 0.726; 0.789; 0.789)$ donc $B_1(3) = 0.789$ et $B_2(3) = 0.726$;
- $x = x \times \Pi = (0.7827; 0.7638; 0.7638; 0.7827; 0.7827)$ donc $B_1(4) = 0.7827$ et $B_2(4) = 0.7638$;

si bien que que

d	0	1	2	3	4
$B_1(d)$	1	0.9	0.81	0.789	0.7827
$B_2(d)$	1	0	0.6	0.726	0.7638

Les calculs préliminaires étant maintenant terminés, on aborde la partie principale de l'algorithme :

- pour $d = 1 \ldots 4$:
  - $d = 1$:
    - $y_1 = y_1 + 0.9 = 0.9$, $t = t + (95 - 1)\mu(1) \times 0.9 = 65.262857$
    - $y_2 = y_2 + 0 = 0$ et $t = t + (95 - 1)\mu(2) \times 0 = 65.262857$
  - $d = 2$:
    - $y_1 = y_1 + 0.81 = 1.71$, $t = t + (95 - 2)\mu(1) \times 0.81 = 123.37457$
    - $y_2 = y_2 + 0.6 = 0.6$ et $t = t + (95 - 2)\mu(2) \times 0.6 = 123.85286$
  - $d = 3$:
    - $y_1 = y_1 + 0.789 = 2.499$, $t = t + (95 - 3)\mu(1) \times 0.789 = 179.84931$
    - $y_2 = y_2 + 0.726 = 1.326$ et $t = t + (95 - 3)\mu(2) \times 0.726 = 180.42182$
  - $d = 4$:
    - $y_1 = y_1 + 0.7827 = 3.2817$, $t = t + (95 - 4)\mu(1) \times 0.7827 = 235.36736$
    - $y_2 = y_2 + 0.7638 = 2.0898$ et $t = t + (95 - 4)\mu(2) \times 0.7638 = 235.96312$

- comme $\ell = 100 \geqslant 11 = m + 2\alpha$
  - $y_1 = y_1 + 90\mu(\mathcal{F}) = 73.4817$ et $y_2 = 72.2898$
  - $t = t + 90 \times 91/2 \times \mu(\mathcal{F})^2 = 2727.3611$
- pour $i = 5 \dots 1$:
  - $i = 5$:
    - $y_1 = y_1 + \mu(\mathcal{F}) = 74.2617$, $t = t + y_1 \times 0.7506 = 2783.102$
    - $y_2 = y_2 + \mu(\mathcal{F}) = 73.0698$, $t = t + y_1 \times 0.0132 = 2784.0665$
  - $i = 4$:
    - $y_1 = y_1 + \mu(\mathcal{F}) = 75.0417$, $t = t + y_1 \times 0.702 = 2836.7457$
    - $y_2 = y_2 + \mu(\mathcal{F}) = 73.8498$, $t = t + y_1 \times 0.024 = 2838.5181$
  - $i = 3$:
    - $y_1 = y_1 + \mu(\mathcal{F}) = 75.8217$, $t = t + y_1 \times 0.54 = 2879.4619$
    - $y_2 = y_2 + \mu(\mathcal{F}) = 74.6298$, $t = t + y_1 \times 0.06 = 2883.9396$
  - $i = 2$ et $i = 2$ n'apportant rien de plus car $A_1(0) = A_2(0) = A_1(1) = A_2(1) = 0$.

Au final, on obtient donc :

$$\mathbb{E}(N_{100}) = z + O(\nu^\alpha) = 76.1898 + O(0.00243)$$

et

$$\mathbb{V}(N_{100}) = 2t + z - z^2 + 95 * O(\nu^\alpha) = 39.183474 + O(0.23085)$$

Le calcul complet donne pour sa part

$$\mathbb{E}(N_{100}) = 76.182857 \quad \text{et} \quad \mathbb{V}(N_{100}) = 39.366465;$$

l'ordre de grandeur des erreurs est donc bien respecté.

Voici un tableau donnant les erreurs attendues et observées sur le calcul de la variance pour différentes valeurs de α :

α	5	6	7	8	9	10
att.	0.2308500	0.0685260	0.0203391	0.0060361	0.0017912	0.0005314
obs.	0.1829911	0.0576265	0.0181109	0.0056806	0.0017784	0.0005558

On peut remarquer au passage que la variance est rapidement déterminée avec une grande précision : erreur relative de l'ordre de 10^{-5} pour $\alpha = 10$. Si l'erreur absolue croit linéairement avec ℓ, il en est approximativement de même pour la variance si bien que l'erreur relative avec $\alpha = 10$ reste ici du même ordre de grandeur pour $\ell = 1\,000$ ou $\ell = 10\,000$.

5.4.5 Lois jointes

On suppose ici qu'on dispose d'une partition $\mathcal{F} = \mathcal{F}_1 \cup \mathcal{F}_2$ et que l'on cherche à calculer la loi jointe de $N_\ell(\mathcal{F}_1)$ et $N_\ell(\mathcal{F}_2)$.

On fixe d'abord $c, d \geqslant 0$ et introduit la FMCI $Z = Z_m \dots Z_\ell$ définie par

$$Z_j = \begin{cases} (N_j(\mathcal{F}_2), Y_j, N_j(\mathcal{F}_1)) & \text{si } N_j(\mathcal{F}_2) \leqslant d \text{ et } N_j(\mathcal{F}_1) < c \\ (N_j(\mathcal{F}_2), Y_j, c_+) & \text{si } N_j(\mathcal{F}_2) \leqslant d \text{ et } N_j(\mathcal{F}_1) \geqslant c \\ f & \text{si } N_j(\mathcal{F}_2) > d \end{cases}$$

l'état f étant absorbant.

Pour fixer les idées, on suppose à partir de maintenant que $c = 3$ et $d = 2$.

Proposition 124. Si on ordonne les états de Z par $N_j(\mathcal{F}_2)$ croissants, puis par Y_j croissants puis par $N_j(\mathcal{F}_1)$ croissants (la valeur c_+ étant la plus grande) et que l'on place l'état absorbant f en dernière position on obtient une matrice de transition T définie par blocs de taille $(c+1)L$:

$$T = \left(\begin{array}{ccc|c} R & S & 0 & 0 \\ 0 & R & S & 0 \\ 0 & 0 & R & \Sigma_S \\ \hline 0 & 0 & 0 & 1 \end{array}\right)$$

Σ_S désignant la somme (en colonne) de la matrice S et où les matrices R et S sont définies par blocs de taille L :

$$R = \begin{pmatrix} P & Q_1 & 0 & 0 \\ 0 & P & Q_1 & 0 \\ 0 & 0 & P & Q_1 \\ 0 & 0 & 0 & P+Q_1 \end{pmatrix} \quad \text{et} \quad S = \begin{pmatrix} Q_2 & 0 & 0 & 0 \\ 0 & Q_2 & 0 & 0 \\ 0 & 0 & Q_2 & 0 \\ 0 & 0 & 0 & Q_2 \end{pmatrix}$$

avec $Q = Q_1 + Q_2$ avec Q_1 (resp. Q_2) contenant uniquement les transitions se terminant en $\mathcal{F}_1$ (resp. $\mathcal{F}_2$).

Démonstration. La preuve de ce résultat étant très similaire à celle de la proposition 93, elle est laissée au lecteur. □

Partant de M_0 la distribution de Z_m on peut donc obtenir la distribution de Z_ℓ en $O(\ell dcL)$ et en tirer (par exemple) la probabilité $\mathbb{P}(N_\ell(\mathcal{F}_2) = d, N_\ell(\mathcal{F}_2) \geqslant c)$ en sommant les bonnes contributions.

Exemple 125. On prend $\mathcal{F}_1 = \{1\}$ et $\mathcal{F}_2 = \{2\}$ et on cherche à calculer la loi de $N_\ell(1)$ conditionnellement à $N_\ell(2) = 2$ partant de $\mu_0 = (0; 0; 1; 0; 0)$ pour $\ell = 10$, $\ell = 15$ ou $\ell = 20$.

Dans ces trois cas, on part du vecteur M_0 de taille 61 dont les coordonnées sont toutes nulles sauf la troisième qui vaut 1. Puis on calcule $M_0 T^{\ell-m}$ dont on extrait et somme les termes correspondant à $N_\ell(2) = 2$.

Dans le cas $\ell = 10$ on trouve par exemple $\mathbb{P}(N_{10}(2) = 2, N_{10}(1) = 0) = 0.0016036$. Il ne reste plus qu'à diviser ces probabilités par $\mathbb{P}(N_{10}(2) = 2)$ (obtenu en sommant sur les valeurs de $N_{10}(1)$) pour obtenir les probabilités conditionnelles recherchées :

probabilité	$n = 0$	$n = 1$	$n = 2$	$n = 3_+$
$\mathbb{P}(N_{10}(1) = n \mid N_{10}(2) = 2)$	0.1422986	0.2428889	0.2794602	0.3353523
$\mathbb{P}(N_{15}(1) = n \mid N_{15}(2) = 2)$	0.0031818	0.0165962	0.0478208	0.9324012
$\mathbb{P}(N_{20}(1) = n \mid N_{20}(2) = 2)$	0.0000496	0.0004537	0.0021800	0.9973166

Les calculs de lois jointes ou conditionnelles sont donc également possibles avec cette approche mais cela pose plusieurs problèmes :

1) la complexité (en temps mais également, et c'est plus grave, en mémoire) est proportionnelle au produit des comptages observés et peut donc rapidement exploser, y compris si l'on veut conditionner non plus par rapport à un comptage, mais par rapport à deux, trois ou même plus ;
2) il est nécessaire qu'un partitionnement de $\mathcal{F}$ soit compatible avec le conditionnement souhaité. Cela peut donc nécessiter le développement d'automates sous la contrainte de compatibilité avec celui-ci.

5.4.6 Plusieurs séquences

Jusqu'à présent on a considéré le cas d'une seule séquence $X = X_1 \dots X_\ell$. Supposons maintenant que l'on s'intéresse à s séquences $X^1 = X_1^1 \dots X_{\ell_1}^1, \dots, X^s = X_1^s \dots X_{\ell_s}^s$.

Une approximation couramment utilisée consiste à concaténer ces s séquences en une seule de taille $\ell = \ell_1 + \dots \ell_s$ (avec d'éventuelles pénalisations liées au nombre de séquences et à la taille du motif) mais cela n'est pas satisfaisant dans la mesure où l'on perd ainsi l'influence de la distribution initiale de chaque début de séquence.

Avec les approches exactes proposées dans cette partie, il est très facile de traiter ce problème. Comme dans le cas d'une seule séquence, on se donne un $c \geqslant 1$ fixé et on introduit pour tout $1 \leqslant \sigma \leqslant s$ la PMC $Y^\sigma = Y_m^\sigma \dots Y_{\ell_\sigma}^\sigma$ (de loi initiale μ_0^σ) et la FMCI associée :

$$Z_j^\sigma = \begin{cases} (Y_j^\sigma, N_{\ell_1+\dots+\ell_{\sigma-1}+j}) & \text{si } N_{\ell_1+\dots+\ell_{\sigma-1}+j} < c \\ c_+ & \text{sinon} \end{cases}$$

où

$$N_{\ell_1+\dots+\ell_{\sigma-1}+j} = N_{\ell_1}^1 + \dots + N_{\ell_{\sigma-1}}^{\sigma-1} + N_j^\sigma$$

est le nombre total d'occurrences observées dans les $\sigma - 1$ premières séquences et dans la séquence σ jusqu'à la position j.

On adapte alors la proposition 97 pour obtenir la

Proposition 126. Soit $u = (u_0 \dots u_{c-1} \,|u_{c_+})$ un vecteur (défini par blocs) de taille $cL+1$.

- *initialisation* : $u_0 = \mu_0^1$, $u_j = 0$ $(1 \leqslant j < c)$, $u_{c_+} = 0$
- *boucle principale* : pour $\sigma = 1 \dots s$
  - pour $0 \leqslant j < c$, $u_j = \mathrm{sum}(u_j) \times \mu_0^p$ et u_{c_+} inchangé
  - pour $i = (m+1) \dots \ell_\sigma$:
    - i) $u_{c_+} = u_{c_+} + u_{c-1}\Sigma_Q$
    - ii) pour $j = (c-1) \dots 1$: $u_j = u_j P + u_{j-1} Q$
    - iii) $u_0 = u_0 P$
  - u contient la loi de Z_ℓ^σ
- *fin* : u contient la loi de Z_ℓ^s

La complexité de cet algorithme est $O(cL+1)$ en mémoire et $O(kcL\ell)$ en temps avec $\ell = \ell_1 + \dots \ell_s$.

Démonstration. La preuve est très similaire à celle de la proposition 97. La seule difficulté provient du fait qu'il faut, à chaque nouvelle séquence, tenir compte de la distribution de N obtenue jusqu'alors et s'en servir pour initialiser u (c'est à cela que servent les transformations $u_j = \mathrm{sum}(u_j) \times \mu_0^p$). □

Bien que cela ne semble pas évident au premier abord, il convient de noter que le résultat de ce calcul ne dépend pas de l'ordre dans lequel sont considérées les séquences.

Exemple 127. Considérons le cas de $s = 3$ séquences de longueurs $\ell_1 = 10$, $\ell_2 = 5$ et $\ell_3 = 15$. Chacune de ces séquences commence par un `a` si bien que $\mu_0^1 = \mu_0^2 = \mu_0^3 = (0; 0; 1; 0; 0)$.

En menant l'algorithme proposé jusqu'au bout on trouve

$$\mathbb{P}(N_{\ell_1} + N_{\ell_2} + N_{\ell_3} \leqslant 2) = 1.165 \times 10^{-7}$$

alors que

ℓ	25	26	27	28	29	30
$\mathbb{P}(N_\ell \leqslant 2) \times 10^{-7}$	6.230	1.187	0.5152	0.223	0.0962	0.04139

On constate que la présence de a au début de chaque séquence favorise moins l'apparition d'occurrences du motif. D'autre part, ce tableau montre que la solution exacte ne correspond pas à un problème à une séquence de longueur choisie (même si $\ell = 26$ donne ici un résultat assez proche de la vraie valeur).

On peut également s'intéresser à l'espérance et la variance exacte du comptage total $N_{\ell_1+\ldots+\ell_s} = N_{\ell_1} + \ldots + N_{\ell_s}$ ce qui est très facile grâce à l'indépendance de ces variables aléatoires. Il suffit donc de calculer séparément $\mathbb{E}[N_{\ell_\sigma}]$ et $\mathbb{V}[N_{\ell_\sigma}]$ pour $1 \leqslant \sigma \leqslant s$ puis de les sommer :

$$\mathbb{E}[N_{\ell_1+\ldots+\ell_s}] = \sum_{\sigma=1}^{s} \mathbb{E}[N_{\ell_\sigma}] \quad \text{et} \quad \mathbb{V}[N_{\ell_1+\ldots+\ell_s}] = \sum_{\sigma=1}^{s} \mathbb{V}[N_{\ell_\sigma}]$$

Du point de vue algorithmique, on peut se contenter d'appliquer l'algorithme 9 successivement à chacune des séquences. Dans ce cas cependant, on va être amené à effectuer plusieurs fois le calcul des termes $A_f(d) = \mu_0 \Pi^d e'_f$ et $B_f(d) = e_f \Pi^d e'_{\mathcal{F}}$. Il sera donc plus efficace d'effectuer ces calculs préliminaires avant d'exécuter une version de l'algorithme 9 allégée du calcul des termes A et B.

Exemple 128. On reprend les trois séquences de l'exemple 127 pour lesquelles on va calculer l'espérance et la variance du comptage. Voici ce que l'on obtient :
- $\ell_1 = 10 : \mathbb{E}[N_{\ell_1}] = 5.983$ et $\mathbb{V}[N_{\ell_1}] = 2.909$
- $\ell_2 = 5 : \mathbb{E}[N_{\ell_2}] = 2.090$ et $\mathbb{V}[N_{\ell_2}] = 0.905$
- $\ell_1 = 15 : \mathbb{E}[N_{\ell_3}] = 9.883$ et $\mathbb{V}[N_{\ell_3}] = 4.934$

si bien qu'au final on trouve :

$$\mathbb{E}[N_{\ell_1} + N_{\ell_2} + N_{\ell_3}] = 17.956 \quad \text{et} \quad \mathbb{V}[N_{\ell_1} + N_{\ell_2} + N_{\ell_3}] = 8.748$$

5.4.7 Modèle hétérogène

Jusqu'à présent, on a supposé l'homogénéité des modèles considérés, c'est à dire que la matrice de transition $\Pi = P + Q$ reste constante tout au long de la séquence. Que se passe-t-il si on suppose que $\Pi_i = P_i + Q_i$ dépend de la position i dans la séquence c'est à dire que pour tout $a, b \in \mathcal{Q}$ et tout $m < i \leqslant \ell$ on a (voir le paragraphe 3.8)

$$\mathbb{P}(Y_i = b | Y_{i-1} = a) = \Pi_i(a, b)$$

En fait cela ne change pas grand chose car il suffit de remplacer Π (resp. P ou Q) par Π_i (resp. P_i ou Q_i) dans les formules. Par exemple la proposition 97 devient

Proposition 129. Soit $u = (u_0 \ \ldots \ u_{c-1} \,|u_{c_+})$ un vecteur (défini par blocs) de taille $cL+1$.
- *initialisation* : $u_0 = \mu_0$, $u_j = 0$ $(1 \leqslant j < c)$, $u_{c_+} = 0$
- *boucle principale* : pour $i = (m+1) \ldots \ell$:
  - i) $u_{c_+} = u_{c_+} + u_{c-1} \Sigma_{Q_i}$
  - ii) pour $j = (c-1) \ldots 1 : u_j = u_j P_i + u_{j-1} Q_i$

iii) $u_0 = u_0 P_i$

– *fin* : u contient $M_0 \prod_{i=m+1}^{\ell} T_i$ (la loi de Z_ℓ)

Elle garde rigoureusement la même complexité que précédemment.

De même on peut sans mal étendre les résultats concernant les temps d'attentes ou bien les moments de comptages au cas hétérogène. Il faut cependant bien avoir conscience qu'il n'est alors plus possible d'utiliser la moindre approximation asymptotique car les produits de la forme $\prod_{i=1}^{j} P_i$ (même chose avec Π_i) n'ont aucune raison de converger (à la différence de P^j).

5.5 Approximations gaussiennes

5.5.1 Cas Markov

On a vu dans la section 5.4.4 comment calculer l'espérance et la variance exacte de N_ℓ.

Proposition 130 (TLC)**.** Soit $Y = Y_m \dots Y_\ell$ une chaîne de Markov d'ordre irréductible de 1 sur $\mathcal{Q}$, de loi initiale μ_0 et de matrice de transition $\Pi = P + Q$. Pour tout $\mathcal{F} \subset \mathcal{Q}$ on a alors

$$\frac{N_\ell(\mathcal{F}) - \mathbb{E}[N_\ell(\mathcal{F})]}{\sqrt{\mathbb{V}[N_\ell(\mathcal{F})]}} \overset{\ell\to\infty}{\longrightarrow} \mathcal{N}(0,1) \quad \text{p.s.}$$

Démonstration. Les chaînes de Markov irréductibles étant des processus fortement mélangeants, les théorèmes classiques de limite centrale s'appliquent. □

Corollaire 131. Si on note μ_ℓ et σ_ℓ^2 l'espérance et la variance de N_ℓ on trouve pour tout $n \geqslant 0$ donc que

$$\mathbb{P}(N_\ell \leqslant n) \simeq \mathbb{P}\left(\mathcal{N}(0,1) \leqslant \frac{n-\mu_\ell}{\sigma_\ell}\right) \quad \text{et} \quad \mathbb{P}(N_\ell \geqslant n) \simeq \mathbb{P}\left(\mathcal{N}(0,1) \geqslant \frac{n-\mu_\ell}{\sigma_\ell}\right)$$

Il faut cependant bien avoir conscience que cette approximation (comme toutes les approximations de type TLC) est d'autant plus fausse que l'on s'éloigne du centre de la distribution.

Exemple 132. Dans l'exemple 123 on a trouvé que $\mu_{100} = 76.182857$ et $\sigma_{100} = 6.274267$ qui vont donc être ici les paramètres de notre approximation gaussienne.

On peut voir sur la figure 5.8 que la distribution exacte et la gaussienne correspondante sont très proches l'une de l'autre. Il est également possible se servir de l'approximation gaussienne pour calculer des valeurs approchées de la fonction de répartition :

n	74	75	76	77
$\mathbb{P}(N_{100} \leqslant n)$	0.3844425	0.4452114	0.5078552	0.5709010
$\mathbb{P}(\mathcal{N}(\mu_{100}, \sigma_{100}^2) \leqslant n)$	0.3639553	0.4252325	0.4883749	0.5518106

Les valeurs approchées sont très proches des valeurs exactes (erreur relative de l'ordre de 5%). Mais si on s'intéresse à une partie plus éloignée du centre de la distribution, la qualité des approximations se dégrade fortement (erreur relative allant jusqu'à 10^5) :

n	0	1	2	3
$\mathbb{P}(N_{100} \leqslant n)$	1.0×10^{-39}	1.2×10^{-37}	7.0×10^{-36}	2.7×10^{-34}
$\mathbb{P}(\mathcal{N}(\mu_{100}, \sigma_{100}^2) \leqslant n)$	3.2×10^{-34}	2.2×10^{-33}	1.5×10^{-32}	9.7×10^{-32}

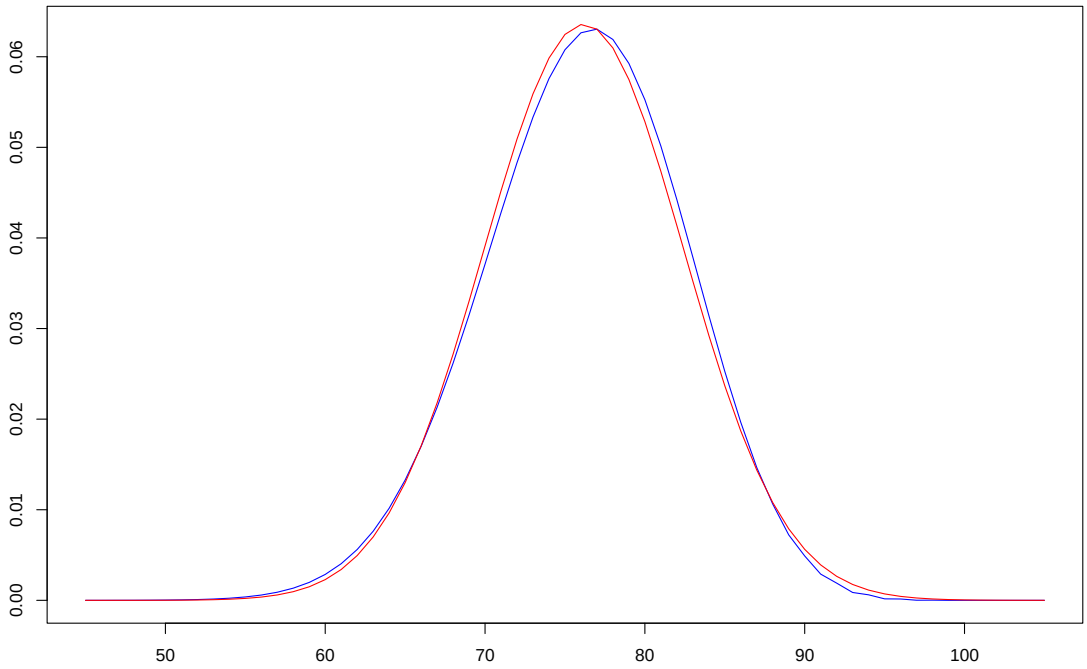


FIG. 5.8 – Loi exacte (en bleu) de N_{100} et gaussienne de moyenne $\mu_{100} = 76.182857$ et d'écart-type $\sigma_{100} = 6.274267$ (en rouge).

5.5.2 Lois jointes

Il est facile d'obtenir l'approximation gaussienne de la loi jointe des comptages de deux mots en calculant leurs espérances $\mu_\ell^{(1)}$ et $\mu_\ell^{(2)}$ et la matrice de covariance Σ. Néanmoins, comme pour la loi jointe exacte, il est nécessaire de construire une PMC pour laquelle l'ensemble des états finaux se découpe en deux parties disjointes, chacune étant associée à l'un des mots. Dans ce cadre, $(N_\ell^{(2)}|N_\ell^{(1)} = n)_1$ est la v.a. gaussienne d'espérance $\mu_\ell^{(2)} + \Sigma_{12}\Sigma_{11}^{-1}(n_1 - \mu_\ell^{(1)})$ et de variance $\Sigma_{22} - \Sigma_{21}\Sigma_{11}^{-1}\Sigma_{12}$, ce qui permet d"évaluer $\mathbb{P}(N_\ell^{(2)} \geq n|N_\ell^{(1)} = n_1)$.

Ceci se généralise bien sûr à plus de deux mots.

5.5.3 Loi de $(\mathbf{N}_m, \mathbf{N}_{m+1})$

Soit $X_1^\ell = X_1 \dots X_\ell$ une chaîne de Markov d'ordre m sur l'alphabet $\mathcal{A}$ de taille k, de paramètre π et de loi initiale μ_0. Soit $\mathbf{N}_m$ (resp. $\mathbf{N}_{m+1}$) le vecteur des comptages aléatoires de tous les mots de longueur m (resp. $m+1$). On se propose dans cette partie de déterminer l'espérance $\mathbf{M}$ et la matrice de covariance $\mathbf{\Sigma}$ de $(\mathbf{N}_m, \mathbf{N}_{m+1})$.

Pour chaque $v, w \in \mathcal{A}^m \cup \mathcal{A}^{m+1}$, il est pour cela nécessaire de disposer d'une PMC non m-ambiguë comptant séparément les occurrences de v et w. Plutôt que de construire une telle PMC pour chaque couple possible, on va créer une unique PMC que l'on pourra utiliser pour l'ensemble des couples.

On considère donc $Y_m^\ell = Y_m \dots Y_\ell$ la chaîne de Markov d'ordre 1 sur $\mathcal{Q} = \mathcal{A}^m \cup \mathcal{A}^{m+1}$, dont la loi initiale charge uniquement $\mathcal{A}^m$ avec la loi μ_0 et dont la matrice de transition Π est non nulle uniquement pour les termes de la forme

$$\Pi(aa_1 \dots a_m, a_1 \dots a_m b) = \pi(a_1 \dots a_m, b) \quad \forall a \in \{\varepsilon\} \cup \mathcal{A}, \forall b \in \mathcal{A} \tag{5.3}$$

On suppose sans perte de généralité que $\mathcal{A} = \{0, \dots, k-1\}$ ce qui permet d'établir une bijection naturelle entre $\mathcal{A}^m$ et $\{0, \dots, k^m - 1\}$ en associant $w(i) = w_1(i) \dots w_m(i) \in \mathcal{A}^m$ à tout $i = \sum_{d=1}^m w_d(i) k^{m-d}$ et on note alors

$$\pi(i, a) = \mathbb{P}\big(X_{m+1} = a \,|\, X_1 \dots X_m = w_1(i) \dots w_m(i)\big)$$

pour tout $a \in \mathcal{A}$. On étend ensuite cette bijection à $\mathcal{Q} = \mathcal{A}^m \cup \mathcal{A}^{m+1}$ en définissant $w(i + pk^m) = (p-1).w(i)$ pour $0 \leqslant i < k^m$ et $1 \leqslant p \leqslant k$. On identifie alors $\mathcal{A}^m$ et $\mathcal{Q}$ avec les ensembles avec lesquels ils sont en bijection pour obtenir le résultat suivant :

Proposition 133. Les seuls termes non nuls de Π sont de la forme :

$$\Pi(i + pk^m, k^m + ki + b) = \pi(i, b) \quad 0 \leqslant i < k^m,\ 0 \leqslant p \leqslant k,\ 0 \leqslant b < k$$

Démonstration. On part simplement de (5.3) en considérant $i = \sum_{d=1}^m a_d k^{m-d}$. On a alors $k(i - a_1 k^{m-1}) = \sum_{d=1}^{m-1} a_{d+1} k^{m-d}$ si bien que

$$w(i) = a_1 \dots a_m \quad \text{et} \quad w\big(k(i - a_1 k^{m-1})\big) = a_2 \dots a_m b$$

le résultat est alors évident. □

On en déduit l'algorithme 10 qui permet d'effectuer le produit d'un vecteur par Π sans même avoir à stocker Π de manière explicite.

Requis: x, y deux vecteurs de taille $k^m + k^{m+1}$
1: $y = 0$
2: **for** $i = 0 \dots k^m - 1$, $b = 0 \dots k - 1$, $p = 0 \dots k$ **do**
3: $\quad y(k^m + ki + b) += x(i + pk^m) \times \pi(i, b)$

ALGORITHME **10:** Calcul de $y = x \times \Pi$. Complexité $O(k^{m+1})$ en espace et en temps.

Remarque 134. On peut modifier l'algorithme 10 pour qu'il effectue le calcul de $y = \Pi \times x$ en remplaçant simplement la ligne 3 par $y(i + pk^m) += x(k^m + ki + b) \times \pi(i, b)$.

Exemple 135. On considère une chaîne de Markov d'ordre $m = 2$ sur l'alphabet binaire $\mathcal{A} = \{\mathrm{a}, \mathrm{b}\}$ ($k = 2$). On numérote 0 la lettre a et 1 la lettre b. La bijection w donne alors

$$\begin{array}{llll} w(0) = \mathrm{aa} & w(1) = \mathrm{ab} & w(2) = \mathrm{ba} & w(3) = \mathrm{bb} \\ w(4) = \mathrm{aaa} & w(5) = \mathrm{aab} & w(6) = \mathrm{aba} & w(7) = \mathrm{abb} \\ w(8) = \mathrm{baa} & w(9) = \mathrm{bab} & w(10) = \mathrm{bba} & w(11) = \mathrm{bbb} \end{array}$$

et les seuls termes non nuls de la matrice de transition sont :

$$\left\{ \begin{array}{l} \Pi(0,4) = \Pi(4,4) = \Pi(8,4) = \pi(\mathrm{a}|\mathrm{aa}) = \pi(0,0) \\ \Pi(0,5) = \Pi(4,5) = \Pi(8,5) = \pi(\mathrm{b}|\mathrm{aa}) = \pi(0,1) \\ \Pi(1,6) = \Pi(5,6) = \Pi(9,6) = \pi(\mathrm{a}|\mathrm{ab}) = \pi(1,0) \\ \Pi(1,7) = \Pi(5,7) = \Pi(9,7) = \pi(\mathrm{b}|\mathrm{ab}) = \pi(1,1) \\ \Pi(2,8) = \Pi(6,8) = \Pi(10,8) = \pi(\mathrm{a}|\mathrm{ba}) = \pi(2,0) \\ \Pi(2,9) = \Pi(6,9) = \Pi(10,9) = \pi(\mathrm{b}|\mathrm{ba}) = \pi(2,1) \\ \Pi(3,10) = \Pi(7,10) = \Pi(11,10) = \pi(\mathrm{a}|\mathrm{bb}) = \pi(3,0) \\ \Pi(3,11) = \Pi(7,11) = \Pi(11,11) = \pi(\mathrm{b}|\mathrm{bb}) = \pi(3,1) \end{array} \right.$$

Toujours grâce à la bijection w on obtient :

Proposition 136. Pour tout $v \in \mathcal{A}^m$ et pour tout $a \in \mathcal{A}$ on a

$$\mathbf{N}_m(v) = \sum_{p=0}^{k} N_\ell(i + pk^m) \quad \text{et} \quad \mathbf{N}_{m+1}(av) = N_\ell(i + (a+1)k^m)$$

où $0 \leqslant i < k^m$ est tel que $w(i) = v$ et, pour tout $j \in \mathcal{Q}$, $N_\ell(j)$ est le nombre d'occurrences de j dans Y_m^ℓ.

Démonstration. Par construction de $Y_m^\ell = Y_m \dots Y_\ell$, $Y_m \in \mathcal{A}^m$ et $Y_j \in \mathcal{A}^{m+1}$ pour tout $j > m$. Ainsi une occurrence de v se termine en j si et seulement si $Y_j = v$ (ce qui ne peut se produire que si $j = m$) ou bien $Y_j = av$ pour un certain $a \in \mathcal{A}$ (ce n'est possible que pour $j > m$). □

Une application de ce résultat avec les notations E_ℓ et C_ℓ de la section 5.4.4 donne alors :

Corollaire 137. Pour tous $v, w \in \mathcal{A}^m$ et $a, b \in \mathcal{A}$, soit $0 \leqslant i, j < k^m$ tels que $w(i) = v$ et $w(j) = w$ alors

$$\mathbb{M}(v) = \sum_{p=0}^{k} E_\ell(i + pk^m) \quad \text{et} \quad \mathbf{M}(av) = E_\ell(i + (a+1)k^m)$$

ainsi que

$$\mathbf{\Sigma}(v,w) = \mathbb{I}_{v=w}\mathbf{M}(v) + \sum_{p=0}^{k}\sum_{q=0}^{k} C_\ell(i+pk^m, j+qk^m) + \sum_{p=0}^{k}\sum_{q=0}^{k} C_\ell(j+qk^m, i+pk^m) - \mathbf{M}(v)\mathbf{M}(w)$$

$$\mathbf{\Sigma}(av,w) = \mathbb{I}_{v=w}\mathbf{M}(av) + \sum_{q=0}^{k} C_\ell(i+(a+1)k^m, j+qk^m) + \sum_{q=0}^{k} C_\ell(j+qk^m, i+(a+1)k^m) - \mathbf{M}(av)\mathbf{M}(w)$$

$$\mathbf{\Sigma}(v,bw) = \mathbb{I}_{v=w}\mathbf{M}(bw) + \sum_{p=0}^{k} C_\ell(i+pk^m, j+(b+1)k^m) + \sum_{q=0}^{k} C_\ell(j+(b+1)k^m, i+pk^m) - \mathbf{M}(v)\mathbf{M}(bw)$$

$$\mathbf{\Sigma}(av,bw) = \mathbb{I}_{av=bw}\mathbf{M}(av) + C_\ell(i+(a+1)k^m, j+(b+1)k^m) + C_\ell(j+(b+1)k^m, i+(a+1)k^m) - \mathbf{M}(av)\mathbf{M}(bw)$$

L'algorithme 11 utilise le corollaire 116 et la proposition 122 pour calculer des valeurs approchées de $E_\ell(i+pk^m)$ et $C_\ell(i+pk^m, j+qk^m)$ pour tous $0 \leqslant i,j < k^m$ et tous $0 \leqslant p,q \leqslant k$ afin de combiner ces quantités comme indiqué dans le corollaire 137 pour obtenir $\mathbf{M}$ et $\mathbf{\Sigma}$.

Remarque 138. On peut de manière similaire obtenir un algorithme exact en utilisant la proposition 115 et le théorème 119 mais ce point est laissé au lecteur.

Exemple 139. On applique l'algorithme 11 à l'exemple 135 avec $\mu_0 = e_0$ (car $X_1 \ldots X_2 =$ aa), $\ell = 10\,000$, $\alpha = 41$ et

$$\begin{cases} \pi(\mathrm{a}|\mathrm{aa}) = 0.4 & \pi(\mathrm{b}|\mathrm{aa}) = 0.6 \\ \pi(\mathrm{a}|\mathrm{ab}) = 0.1 & \pi(\mathrm{b}|\mathrm{ab}) = 0.9 \\ \pi(\mathrm{a}|\mathrm{ba}) = 0.3 & \pi(\mathrm{b}|\mathrm{ba}) = 0.7 \\ \pi(\mathrm{a}|\mathrm{bb}) = 0.6 & \pi(\mathrm{b}|\mathrm{bb}) = 0.4 \end{cases}$$

On trouve alors : $\mu\left(\mathcal{A}^m\right) = 0$ et

$$\mu\left(\mathcal{A}^{m+1}\right) = \begin{pmatrix} 0.05 & 0.075 & 0.025 & 0.225 & 0.075 & 0.175 & 0.225 & 0.150 \end{pmatrix}$$

$\mathbf{M} = (\mathbf{M}_m, \mathbf{M}_{m+1})$ et $\mathbf{\Sigma} = \begin{pmatrix} \mathbf{\Sigma}_{m,m} & \mathbf{\Sigma}_{m,m+1} \\ \mathbf{\Sigma}_{m+1,m} & \mathbf{\Sigma}_{m+1,m+1} \end{pmatrix}$ avec

Requis: $\alpha \geqslant 0$, $\ell \geqslant m+\alpha$, $x, y, A_0, \ldots, A_{\alpha-1}, B_0, \ldots, B_{\alpha-1}$, $2\alpha+2$ vecteurs de taille $L = k^m + k^{m+1}$, μ la loi stationnaire de Π

$\mathbf{\Sigma} = 0$, $A_0 = 0$, $\forall 0 \leqslant i < k^m$, $A_0(i) = \mu_0(i)$ et $\mathbf{M} = A_0$
for $d = 1 \ldots \alpha - 1$ **do**
  $A_d = A_{d-1} \times \Pi$ et $\mathbf{M}+ = A_d$
$\mathbf{M}+ = (\ell - m - \alpha + 1) \times \mu$
for $i = 0 \ldots k^m - 1$ **do**
  for $p = 1 \ldots k$ **do**
    $\mathbf{M}(i)+ = \mathbf{M}(i + pk^m)$
// le calcul de $\mathbf{M}$ *est maintenant terminé*
for $j = 0 \ldots k^m - 1$ et $q = 0 \ldots k$ **do**
  $j_0 = j + qk^m$, $x = y = 0$ et $B_0 = e'_{j_0}$
  for $d = 1 \ldots \min(\ell - m - \alpha, \alpha - 1)$ **do**
    $B_d = \Pi \times B_{d-1}$, $y+ = B_d$ et $x+ = (\ell - m - \alpha + 1 - d)B_d$
  if $\ell \geqslant m + 2\alpha$ **then**
    $x+ = (\ell - m - 2\alpha + 1)(\ell - m - 2\alpha + 2)\mu(j_0)/2$
    $y+ = (\ell - m - 2\alpha + 1)\mu(j_0)$
  $x\times = \mu$ *// produit terme à terme*
  for $i = m + \alpha - 1 \ldots m$ **do**
    if $\ell - i < \alpha$ **then**
      $y+ = B_{\ell - i}$
    else
      $y+ = \mu(j_0)$
    $x+ = y \times A_{i-m}$ *// produit terme à terme*
  // $x(\cdot)$ contient maintenant $C_\ell(\cdot, j_0)$
  for $i = 0 \ldots k^m - 1$ et $p = 0 \ldots k$ **do**
    $i_0 = i + pk^m$, $\mathbf{\Sigma}(i,j)+ = x(i_0)$ et $\mathbf{\Sigma}(j,i)+ = x(i_0)$
    if $p > 0$ **then**
      $\mathbf{\Sigma}(i_0,j)+ = x(i_0)$ et $\mathbf{\Sigma}(j,i_0)+ = x(i_0)$
    if $q > 0$ **then**
      $\mathbf{\Sigma}(i,j_0)+ = x(i_0)$ et $\mathbf{\Sigma}(j_0,i)+ = x(i_0)$
    if $pq > 0$ **then**
      $\mathbf{\Sigma}(i_0,j_0)+ = x(i_0)$ et $\mathbf{\Sigma}(j_0,i_0)+ = x(i_0)$
    $\mathbf{\Sigma}(i_0,j_0)- = \mathbf{M}(i_0)\mathbf{M}(j_0)$
    if i=j **then**
      if $q = 0$ **then**
        $\mathbf{\Sigma}(i_0,j_0)+ = \mathbf{M}(i_0)$
      else if $p = 0$ ou $p = q$ **then**
        $\mathbf{\Sigma}(i_0,j_0)+ = \mathbf{M}(j_0)$

ALGORITHME **11:** Calcul approché de $\mathbf{M}$ (erreur de l'ordre de ν^α) et de $\mathbf{\Sigma}$ (erreur de l'ordre de $(\ell - m - \alpha + 1)\nu^\alpha$) où ν est la valeur absolue de la seconde valeur propre de Π. Complexités $O(\alpha k^{m+1})$ en espace et $O(\alpha k^{2m+2})$ en temps.

$$\mathbf{M}_m = \begin{pmatrix} 1251.03 & 2499.78 & 2499.15 & 3749.04 \end{pmatrix}$$

$$\mathbf{M}_{m+1} = \begin{pmatrix} 500.36 & 750.55 & 249.95 & 2249.57 & 749.67 & 1749.23 & 2249.20 & 1499.47 \end{pmatrix}$$

$$\mathbf{\Sigma}_{m,m} = \begin{pmatrix} 1803.25 & -403.68 & -403.61 & -995.95 \\ -403.68 & 338.55 & 338.43 & -273.31 \\ -403.61 & 338.43 & 338.55 & -273.37 \\ -995.95 & -273.31 & -273.37 & 1542.63 \end{pmatrix}$$

$\mathbf{\Sigma}_{m+1,m+1} =$

$$\begin{pmatrix} 938.92 & 220.07 & -9.91 & -276.64 & 220.13 & -506.62 & -276.61 & -309.34 \\ 220.07 & 424.01 & 16.41 & -133.50 & 423.99 & -541.07 & -133.50 & -276.40 \\ -9.91 & 16.41 & 247.09 & -119.52 & 16.41 & 111.17 & -119.50 & -142.14 \\ -276.64 & -133.50 & -119.52 & 330.46 & -133.54 & 344.37 & 330.35 & -341.99 \\ 220.13 & 423.99 & 16.41 & -133.54 & 424.07 & -541.12 & -133.50 & -276.43 \\ -506.62 & -541.07 & 111.17 & 344.37 & -541.12 & 996.68 & 344.36 & -207.77 \\ -276.61 & -133.50 & -119.50 & 330.35 & -133.50 & 344.36 & 330.47 & -342.07 \\ -309.34 & -276.40 & -142.14 & -341.99 & -276.43 & -207.77 & -342.07 & 1896.15 \end{pmatrix}$$

$\mathbf{\Sigma}_{m,m+1} = {}^t\mathbf{\Sigma}_{m+1,m} =$

$$\begin{pmatrix} 1159.05 & 644.06 & 6.50 & -410.18 & 644.20 & -1047.74 & -410.11 & -585.78 \\ -286.55 & -117.06 & 127.57 & 210.87 & -117.13 & 455.61 & 210.86 & -484.18 \\ -286.52 & -117.10 & 127.58 & 210.83 & -117.09 & 455.53 & 210.96 & -484.20 \\ -585.98 & -409.90 & -261.65 & -11.52 & -409.98 & 136.60 & -11.72 & 1554.16 \end{pmatrix}$$

5.5.4 Approche fondée sur les martingales

Cette approche considère le nombre d'occurrences d'un mot W *conditionnellement* au nombre d'occurrences de W^-, le mot W privé de sa dernière lettre. Elle peut être appliquée dans deux objectifs :

1) mesurer le caractère d'exceptionnalité de W ; on notera que l'on s'appuie sur une statistique assez pauvre, seulement fondée sur la comparaison des fréquences des quatre mots de la forme W^-u ;
2) comparer les caractères d'exceptionnalité de plusieurs mots emboîtes les uns dans les autres. Un phénomène perturbe souvent l'analyse des résultats : si un mot W est "très fréquent" dans une séquence, tous les mots Wu obtenus en ajoutant une lettre, u, à la suite de W auront tendance à être aussi "très fréquents". On fera bien sûr la même remarque pour les mots de la forme uW. Dit autrement, si W est très fréquent, il en est de même de son préfixe W^- et son suffixe ${}^-W$ (W privé de sa première lettre). On observe donc des paquets de mots liés par des relations d'inclusion, et qui sont tous "très fréquents" (ou de même "très rares"). Se pose donc le problème "quelle est la longueur du mot qui contribue le plus à cette exceptionnalité parmi plusieurs mots emboîtes".

Considérons donc une séquence S de longueur n, réalisation en régime stationnaire d'une chaîne M1, dont la transition sera notée π et la loi stationnaire μ. Pour un mot $W = w_1 w_2 ... w_h$,

$$Z_k = \mathbb{I}\{W \text{ finit en position } k\} = \mathbb{I}\{X_{k-h+1} ... X_k = w_1 ... w_h\}.$$

s'écrit

$$Z_k = \mathbb{I}\{W^- \text{ finit en position } k-1 \quad \text{et} \quad X_k = w_h\}.$$

où $W^- = w_1 w_2 ... w_{h-1}$; si $\mathcal{F}_{k-1}$ est la tribu engendrée par $\{X_j \; j < k\}$ on a

$$\mathbb{E}(Z_k \mid \mathcal{F}_{k-1}) = \mathbb{I}\{W^- \text{ finit en } k-1\} \; \pi(w_{h-1}, w_h)$$

et l'on peut introduire le "compensateur" $\Delta_k = Z_k - \mathbb{E}(Z_k \mid \mathcal{F}_{k-1})$ et la martingale $M_n = \sum_{k=1}^{n} \Delta_k$. Du Théorème de la Limite Centrale pour martingales dû à Rebolledo (1980), on peut alors déduire (voir Prum et al., 1995) le théorème :

Théorème 140.

$$\frac{1}{\sqrt{n}} \left[N(W) - N(W^-) \frac{N(w_{h-1}w_h)}{N(w_{h-1})} \right] \xrightarrow{\mathcal{L}} \mathcal{N}\big(0, Kp(1-p)\big)$$

quand $n \to \infty$ avec

$$p = \pi(w_{h-1}, w_h) \qquad \text{et} \qquad K = \mu(W^-) \left(1 - \frac{\mu(W^-)}{\mu(w_{h-1})} \right).$$

5.5.5 Modèle "shuffle"

Parallèlement à cette modélisation markovienne, une autre modélisation s'est révélée donner des résultats très simples (dont on montre d'ailleurs qu'ils sont équivalents à ceux du modèle markovien quand la taille de la séquence tend vers l'infini). Elle est fondée sur l'étude de l'ensemble des séquences "partageant" les comptes de mots de longueur $m+1$ (la statistique exhaustive du modèle Mm) et n'a donc de sens *que* pour les mots (ou motifs) de longueur $h \geqslant m+2$. En outre son usage est aisé essentiellement pour le modèle d'ordre 1, et c'est dans ce cadre que nous le présenterons ici.

Le fait que la matrice de transition π de la chaîne de Markov modélisant la séquence n'est jamais connue et doit donc être estimée sur les données, implique qu'un résultat donnant l'espérance (la variance, la loi) du nombre d'occurrences d'un mot W dans ce modèle doit toujours être combiné avec une procédure de "plug-in", dans laquelle on remplace π par $\hat{\pi}$ – ce qui n'est pas sans conséquences sur ces paramètres. Cette approche *shuffle* évite par construction ce problème (voir la discussion au paragraphe 5.6.1).

Étant donnée une séquence X, cette approche (Cowan, 1991; Prum et al., 1995) consiste à considérer toutes les séquences qui donneraient la même estimation de π,

$$\hat{\pi}(a,b) = \frac{N_X(ab)}{N_X(a.)} \quad \text{où, comme toujours,} \quad N_X(a.) = \sum_b N_X(ab)$$

Ce sont donc les séquences pour lesquelles tous les mots de deux lettres ont mêmes nombres d'occurrences que dans X.

De fait, on va aussi fixer la première lettre X_1 [2], afin de pouvoir utiliser le théorème suivant :

[2] S'il existe a tel que $N_X(a.) = N_X(.a) + 1$, alors $X_1 = a$ et il existera b tel que $N_X(b.) = N_X(.b) - 1$ et b sera la dernière lettre de la séquence. Si pour tout a, $N_X(a.) = N_X(.a)$, se donner X_1 équivaut à se donner X_n.

Théorème 141. (Whittle, 1955) Étant donnée une séquence $X = X_1 \dots X_n$, notons $\mathcal{S}$ la statistique composée de X_1 et de la statistique $N_2 = \{N_X(uv), u, v \in \mathcal{A}\}$. Il nous sera utile de "ranger" les termes de N_2 dans une matrice $k \times k$, que nous noterons aussi N_2. L'ensemble (lui aussi noté $\mathcal{S}$) des séquences partageant cette statistique a pour cardinal

$$\sharp\, \mathcal{S} = K(X)H(X) \qquad \text{où} \qquad K(X) = \prod_a \frac{N_X(a.))!}{\prod_b N_X(ab)!}$$

tandis que $H(X)$ est le cofacteur du terme (d, d) de la matrice $M = I - \hat{\pi}$, où $d = X_n$ est la dernière lettre de X.

Remarque 142. Dans le modèle M1, la probabilité de la séquence X est égale à $\mu(X_1) \prod \pi(a, b)^{N_X(ab)}$. En conséquence toutes les séquences appartenant à $\mathcal{S}$ ont même probabilité : la chaîne de Markov de transition $\hat{\pi}$ induit sur $\mathcal{S}$ la loi uniforme. Travailler dans le modèle "shuffle", c'est aussi travailler dans le modèle markovien conditionnellement aux comptages $N_X(ab)$ et après plug-in.

Espérance

Cowan a introduit la transformation $\mathcal{T}_k$ qui consiste, si une occurrence de W commence en position k, à la remplacer par le mot de deux lettres $\alpha\omega$ où $\alpha = w_1$ est la première lettre de W et $\omega = w_h$ sa dernière lettre. On dira que l'on a "vidé" cette occurrence de W de son intérieur (figure 5.9).

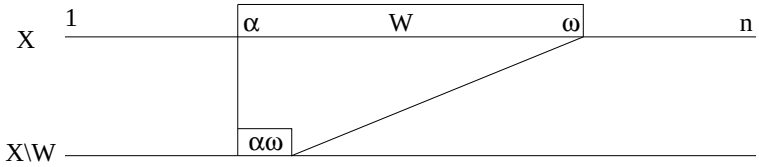


FIG. 5.9 – La transformation de Cowan

Notons $X\backslash W_k$ la séquence obtenue. Clairement

$$N_{X\backslash W_k}(ab) = N_X(ab) - w_{ab}$$

où

$$\begin{aligned} w_{ab} &= N_W(ab) \qquad \text{si} \quad ab \neq \alpha\omega \\ w_{\alpha\omega} &= N_W(\alpha\omega) - 1 \end{aligned}$$

Notons $\mathcal{S}\backslash W_k$ l'ensemble des séquences X' telles que $X'_1 = X_1$ et, pour tout (a, b), $N_{X'}(ab) = N_{X\backslash W_k}(uv)$). La transformation $\mathcal{T}_k$ établit une bijection entre

- l'ensemble des séquences de $\mathcal{S}$ telles que le mot commençant en position k soit W et
- l'ensemble des séquences de $\mathcal{S}\backslash W_k$ telles que le mot commençant en position k soit $\alpha\omega$.

Les cardinaux de ces deux ensembles, disons respectivement $\beta_k(W, \mathcal{S})$ et $\beta_k(\alpha\omega, \mathcal{S}\backslash W)$,

sont égaux. Comme on a des lois uniformes,

$$\begin{aligned}\mathbb{E}(N_X(W)) &= \frac{1}{\sharp\,\mathcal{S}}\sum_{X\in\mathcal{S}}\sharp\,\{k,\text{ tels que } W \text{ commence en position } k \text{ dans } X)\\ &= \frac{1}{\sharp\,\mathcal{S}}\sum_{k}\sum_{X} k,\text{ tels que } W \text{ commence en position } k \text{ dans } X)\\ &= \frac{1}{\sharp\,\mathcal{S}}\sum_{k}\beta_k(W,\mathcal{S}).\end{aligned}$$

Bien sûr, le même raisonnement appliqué à $\mathcal{S}\backslash W$ et au mot $\alpha\omega$ donne

$$\mathbb{E}(N_{X\backslash W}(\alpha\omega)) = \frac{1}{\sharp\,\mathcal{S}\backslash W}\sum_{k}\beta_k(\alpha\omega,\mathcal{S}\backslash W).$$

De $\beta_i(W,\mathcal{S}) = \beta_i(\alpha\omega,\mathcal{S}\backslash W)$, on déduit donc

$$\mathbb{E}(N_X(W)\,|\,\mathcal{S}) = \frac{\sharp\,\mathcal{S}\backslash W}{\sharp\,\mathcal{S}}\,\mathbb{E}(N_{X\backslash W}(\alpha\omega)\,|\,\mathcal{S}\backslash W)$$

Mais $(N_{X\backslash W}(\alpha\omega)$ est une constante ! On a donc le théorème :

Théorème 143. (Cowan, 1991)

$$\mathbb{E}(N_X(W)|\mathcal{S}) = \frac{\sharp\,\mathcal{S}\backslash W}{\sharp\,\mathcal{S}}\,N_{X\backslash W}(\alpha\omega)$$

Il reste à remarquer que $\dfrac{\sharp\,\mathcal{S}\backslash W}{\sharp\,\mathcal{S}}$ est le produit de

- $\dfrac{H(X\backslash W)}{H(X)}$, quotient de deux déterminants 3×3,
- et $\dfrac{K(X\backslash W)}{K(X)}$; c'est le quotient de termes factoriels gigantesques, mais une simplification évidente intervient, de sorte que seuls restent un nombre de facteurs inférieur à $2h$. Montrons ceci sur un exemple :

Exemple 144. Pour le virus HIV1, on a (dans l'ordre `a,c,g,t`)

$N_2 = \begin{pmatrix} 1112 & 561 & 1024 & 713\\ 795 & 413 & 95 & 470\\ 820 & 457 & 661 & 432\\ 684 & 342 & 590 & 548\end{pmatrix}$, donc les $N(a.)$ valent : $\begin{pmatrix} 3410\\ 1773\\ 2370\\ 2164\end{pmatrix}$

$K(X)$ est le produit de quatre termes :

$$K(X) = \frac{3410!}{1112!\;561!\;1024!\;713!}\quad\cdots\quad\frac{2164!}{684!\;342!\;590!\;548!}$$

La formule de Stirling indique que $K(X)$ est de l'ordre de 10 à la puissance 5627.

Choisissons $W =$ `gatc`. Il y a 33 occurrences de W dans la séquence de HIV. Remplacer W par $\alpha\omega =$ `gc` ne modifie aucun $N(ab)$ sauf $N($`ga`$)$, $N($`at`$)$, $N($`tc`$)$ (chacun décroit de 1) et $N($`gc`$)$ (qui croit de 1). Donc $N($`a.`$)$ et $N($`t.`$)$ décroissent chacun de 1, les autres sommes

par ligne restant invariantes. Dans $K(\mathcal{S}\backslash W)/K(\mathcal{S})$ ces sommes par lignes interviennent sous la forme

$$\frac{3409!\ 1773!\ 2370!\ 2163!}{3410!\ 1773!\ 2370!\ 2164!} = \frac{1}{3410 \times 2164}$$

On vérifiera que le terme associé aux $N(ab)$ se réduit à $820 \times 713 \times 342/458$, ce qui donne finalement à $K(\mathcal{S}\backslash W)/K(\mathcal{S})$ la valeur $0,05916$. Un calcul élémentaire donne $H(X) = 0.3215289$ et $H(X\backslash W) = 0.32147721$. Le rapport des cardinaux de Whittle vaut 0.05915 et $\mathbb{E}N(\texttt{gatc}) = 0.05915 \times (451 + 1) = 27.092$.

Variance

Un raisonnement similaire, transformant deux occurrences de W en $\alpha\omega$ permet le calcul de la variance. Mais il est nécessaire de tenir éventuellement compte du fait que deux occurrences de W peuvent se recouvrir.

Notation 145. Soit W un mot de longueur h et W' un mot de longueur h'. Posons

$$\delta(W, W'; d) = \mathbb{I}\{W \text{ en position } k \text{ et } W' \text{ en position } k + d \text{ sont compatibles}\}$$

et, quand $\delta(W, W'; d) = 1$, notons $W^{(d)}W'$ le mot de longueur $d + h'$ obtenu quand W et W' se chevauchent sur $h - d$ positions. Alors

Théorème 146. La variance de $N_W(X)$ conditionnellement à $\mathcal{S}$ se déduit de

$$\begin{aligned}
\mathbb{E}(N_X(W)^2 \mid \mathcal{S}) &= N_{X\backslash W}(\alpha\omega)\ \frac{\sharp\, \mathcal{S}\backslash W}{\sharp\, \mathcal{S}} \\
&+ N_{X\backslash W\backslash W}(\alpha\omega)[\ N_{X\backslash W\backslash W}(\alpha\omega) - 1]\ \frac{\sharp\, \mathcal{S}\backslash W\backslash W}{\sharp\, \mathcal{S}} \\
&+ 2\ \sum_{d=1}^{h-2} \delta(W, W; d)\ N_{X\backslash W^{(d)}W}(\alpha\omega)\ \frac{\sharp\, \mathcal{S}\backslash W^{(d)}W}{\sharp\, \mathcal{S}}.
\end{aligned}$$

De même, si $W \neq W'$, $cov(N_X(W), N_X(W')|\mathcal{S})$ se déduit de

$$\begin{aligned}
\mathbb{E}(N_X(W) \times N_X(W') \mid \mathcal{S}) &= N_{X\backslash W\backslash W'}(\alpha\omega)[\ N_{X\backslash W\backslash W'}(\alpha\omega) - \delta(\alpha\omega, \alpha'\omega'; 0)] \\
&\qquad \times \frac{\sharp\, \mathcal{S}\backslash W\backslash W'}{\sharp\, \mathcal{S}} \\
&+ \sum_{d=2-h'}^{h-2} \delta(W, W'; d)\ N_{X\backslash W^{(d)}W'}(\alpha\omega)\ \frac{\sharp\, \mathcal{S}\backslash W^{(d)}W'}{\sharp\, \mathcal{S}}.
\end{aligned}$$

Approximation gaussienne

Il est assez facile de montrer (Prum et al., 1995) que, quand W est fixé et $n \to \infty$, $Z(W) = \dfrac{N_W - \mathbb{E}(N_X(W)|\mathcal{S})}{Var(N_X(W)|\ \mathcal{S})^{1/2}}$ converge en loi vers une gaussienne centrée réduite. Issue d'un TCL, cette approximation sera rapidement de bonne qualité pour peu que l'on ne soit pas dans les queues de la gaussiennes, où il conviendra de mettre en oeuvre d'autres outils.

Exemple 147. Dans l'exemple du HIV1, on trouve $\mathbb{V}(N(\texttt{gatc})) = 23.583$. La "gaussienne" centrée réduite $Z = (N(\texttt{gatc}) - \mathbb{E}(N(\texttt{gatc})))\ \mathbb{V}(N(\texttt{gatc}))^{-1/2}$ se réalise donc en $z = (33 - 27.092)/\sqrt{23.583} = 1.215$, ce qui n'est pas significatif aux niveaux usuels.

Dans la partie "centrale" de la gaussienne, on pourra comme précédemment mesurer le caractère d'exceptionnalité d'un mot 'trop fréquent' W à partir de la p-valeur $p = \mathbb{P}(\mathcal{N}(0;1) \geq z)$ où z est la valeur observée de la statistique $Z(W)$.

Notons enfin que ceci se généralise aisément à un "motif", $\mathcal{W} = (W_1, W_2, \ldots, W_r)$: si l'on pose $N((\mathcal{W})) = \sum \mathcal{N}(\mathcal{W}_\rangle)$, le théorème précédent permet d'obtenir l'espérance et la variance de $N(\mathcal{W})$ [3].

5.6 Approximations binomiales

On rappelle que pour tout $m \leqslant i \leqslant \ell$ on a

$$\mathbb{P}(Y_i \in \mathcal{F}) = \mu_0 \Pi^{i-m} e'_{\mathcal{F}} = \mu(\mathcal{F}) + O(\nu^{i-m})$$

où μ est la loi stationnaire de Π et ν le module de la seconde valeur propre en module.

Proposition 148. Si l'on traite les Y_i comme s'ils étaient indépendants (ce qui est évidemment faux) et si l'on approche $\mathbb{P}(Y_i \in \mathcal{F})$ par $\mu(\mathcal{F})$, on obtient

$$\mathcal{L}(N_\ell) = \mathcal{L}\left(\sum_{i=m}^{\ell} Y_i\right) \simeq \mathcal{B}\left(\ell - m + 1, \mu(\mathcal{F})\right)$$

Si cette approximation est très grossière, on verra en section 5.9 qu'elle est cependant assez performante en pratique.

Exemple 149. On se place sur l'alphabet $\mathcal{A} = \{\texttt{a}, \texttt{c}, \texttt{g}, \texttt{t}\}$ sur lequel on considère le modèle markovien d'ordre $m = 1$ dont la matrice de transition est :

$$\pi = \begin{pmatrix} 0.45 & 0.16 & 0.10 & 0.29 \\ 0.11 & 0.41 & 0.22 & 0.26 \\ 0.40 & 0.28 & 0.11 & 0.21 \\ 0.18 & 0.23 & 0.45 & 0.14 \end{pmatrix}$$

On considère une séquence de longueur $\ell = 1\,000$ dont la première lettre est un $\texttt{a}$ et on s'intéresse au motif $\mathcal{W}_1 = \{\texttt{cc.tg}\}$ dont l'automate non 1-ambigu minimal (12 états) est donné en figure 5.10. Cet automate n'a qu'un seul état final ($\mathcal{F} = \{7\}$) et en cherchant la loi stationnaire de la PMC associée on trouve $\mu(\mathcal{F}) = 1.0982353 \times 10^{-2}$.

On peut voir en figure 5.11 une comparaison de la loi exacte et de l'approximation binomiale. Comme dans le cas des approximations gaussiennes on peut évidemment se servir de cette approximation pour calculer des valeurs approchées de la fonction de répartition (ou de la fonction de survie). On trouve par exemple

n	0	1	9	10
$\mathbb{P}(N_{1\,000} \leqslant n)$	1.01×10^{-5}	1.32×10^{-4}	0.338	0.464
$\mathbb{P}(\mathcal{B}(1000, \mu(\mathcal{F}) \leqslant n)$	1.60×10^{-5}	1.94×10^{-4}	0.341	0.461

[3] on peut bien sûr introduire des "poids" γ_i et considérer $N(\mathcal{W}) = \sum \gamma_i N(W_i)$.

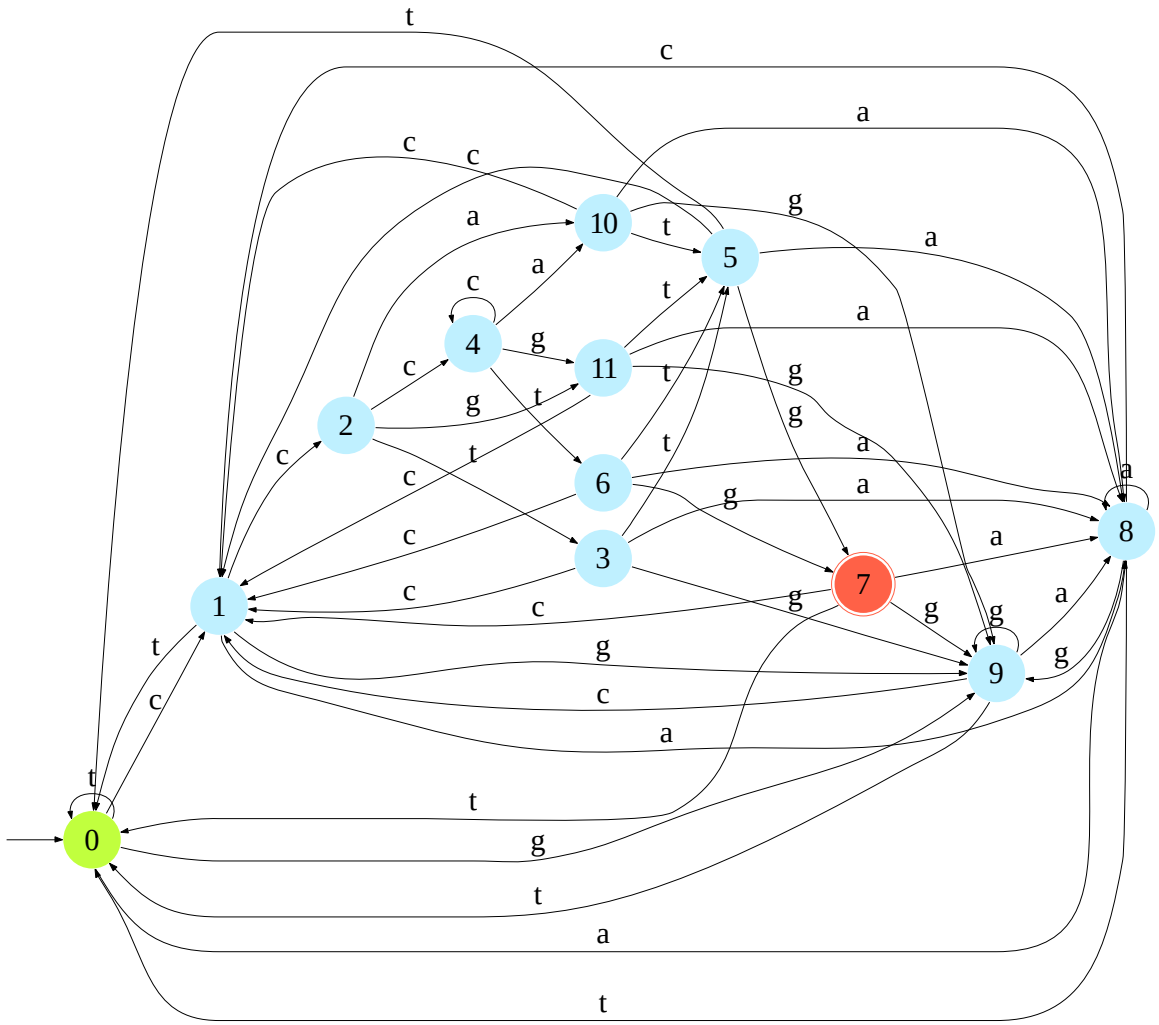


FIG. 5.10 – Représentation graphique du plus petit AFD non 1-ambigu associé au motif $\mathcal{W}_1 = \{\texttt{cc.tg}\}$ défini sur l'alphabet $\mathcal{A} = \{\texttt{a}, \texttt{c}, \texttt{g}, \texttt{t}\}$. Cet automate a 12 états.

ou encore

n	10	11	44	45
$\mathbb{P}(N_{1000} \geqslant n)$	0.662	0.536	$1.23e{-16}$	$2.03e{-17}$
$\mathbb{P}(\mathcal{B}(1000, \mu(\mathcal{F})) \geqslant n)$	0.659	0.539	$3.02e{-14}$	$7.08e{-15}$
$\mathbb{P}(\mathcal{N}(\mu_{1000}, \sigma^2_{1000}) \geqslant n)$	0.615	0.490	$2.68e{-26}$	$8.58e{-28}$

A la différences des approximations gaussiennes, les valeurs numériques restent ici assez proches les unes des autres, y compris dans les queues de la distribution.

Exemple 150. Dans le même cadre que celui de l'exemple 149 on considère maintenant le motif $\mathcal{W}_2 = \{\texttt{ccccc}\}$ dont l'automate non 1-ambigu minimal (8 états) est donné en figure 5.12.

Cet automate n'a qu'un seul état final ($\mathcal{F} = \{5\}$) et en cherchant la loi stationnaire de la PMC associée on trouve $\mu(\mathcal{F}) = 7.607566 \times 10^{-3}$.

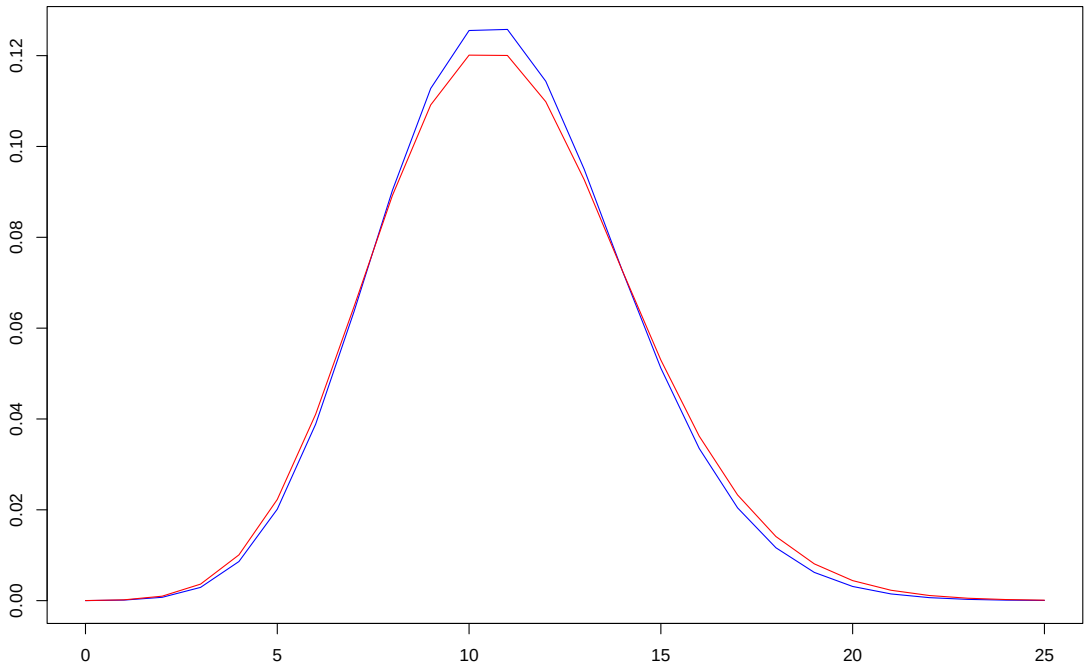


FIG. 5.11 – Loi exacte (en bleu) de $N_{1\,000}(\mathcal{W}_1)$ et binomiale de paramètres $\ell - m + 1 = 1\,000$ et $\mu(\mathcal{F}) = 1.0982353 \times 10^{-2}$ (en rouge).

La figure 5.13 compare la distribution exacte à l'approximation binomiale. Cette fois, les deux distributions diffèrent sensiblement. Nous verrons plus tard que cela est dû à la structure du motif $\mathcal{W}_2$ qui peut facilement se recouvrir lui-même.

En effet, si on considère maintenant les occurrences renouvelantes du même motif (qui perdent donc toute capacité à se recouvrir elles-mêmes). On obtient un nouvel automate (toujours à 8 états et avec un seul état final) dont la PMC associée donne $\mu(\mathcal{F}) = 4.541075 \times 10^{-3}$. La figure 5.14 compare la distribution théorique à l'approximation binomiale correspondante. Comme on peut le constater ces deux distributions sont très proches.

Exemple 151. Toujours dans le cadre que celui des exemples 149 et 150 on considère enfin le motif $\mathcal{W}_3 = \mathcal{W}_1 + \mathcal{W}_2 = \{\texttt{cc.tg|ccccc}\}$ dont l'automate non 1-ambigu minimal (14 états) est donné en figure 5.16 (vu sa complexité, nous l'avons cette fois-ci représenté par les valeurs de sa fonction de transition δ).

Cet automate a deux états finaux ($\mathcal{F} = \{9, 10\}$) et en cherchant la loi stationnaire de la PMC associée on trouve $\mu(\mathcal{F}) = 1.8589919 \times 10^{-2}$.

La figure 5.15 compare la distribution exacte à l'approximation binomiale. Comme dans l'exemple 150, les deux distributions diffèrent sensiblement même si la différence est ici légèrement moins marquée (car la propension du motif à se recouvrir lui même est moins forte que dans l'exemple précédent).

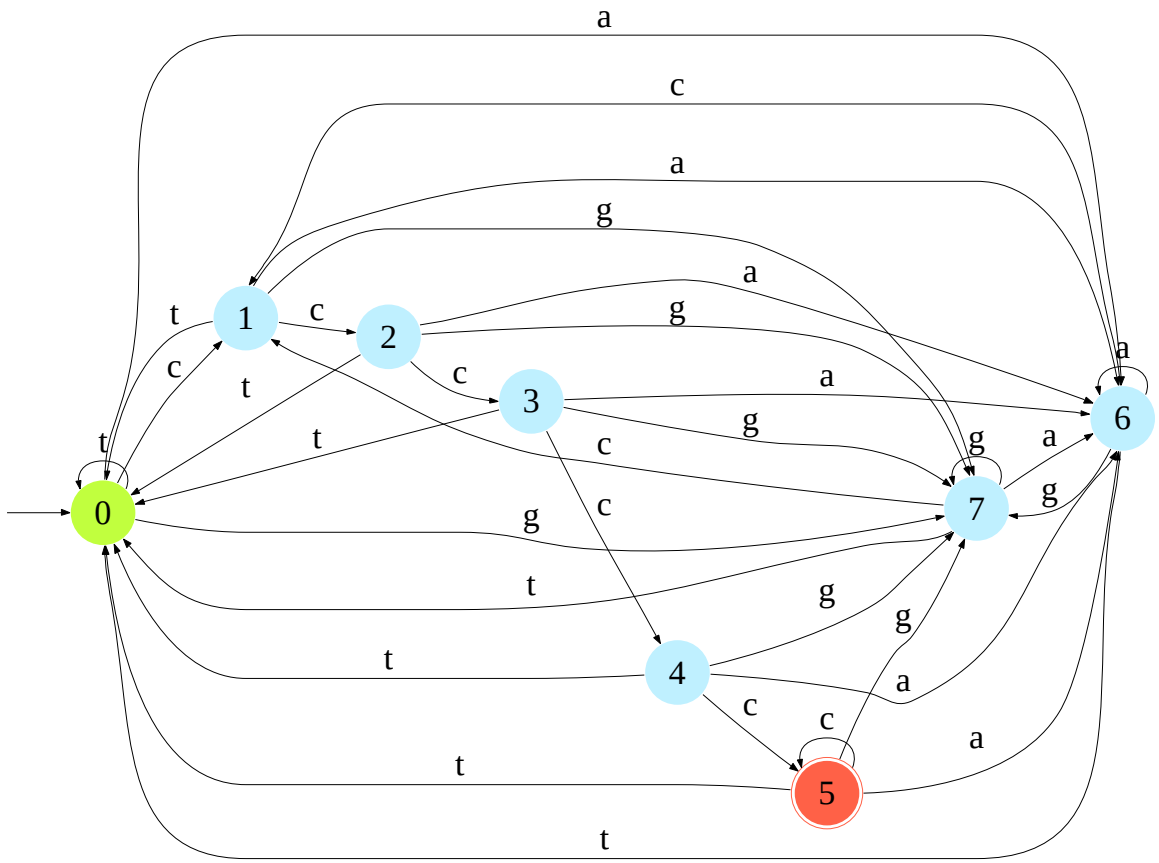


FIG. 5.12 – Représentation graphique du plus petit AFD non 1-ambigu associé au motif $\mathcal{W}_2 = \{\texttt{ccccc}\}$ défini sur l'alphabet $\mathcal{A} = \{\texttt{a}, \texttt{c}, \texttt{g}, \texttt{t}\}$. Cet automate a 8 états.

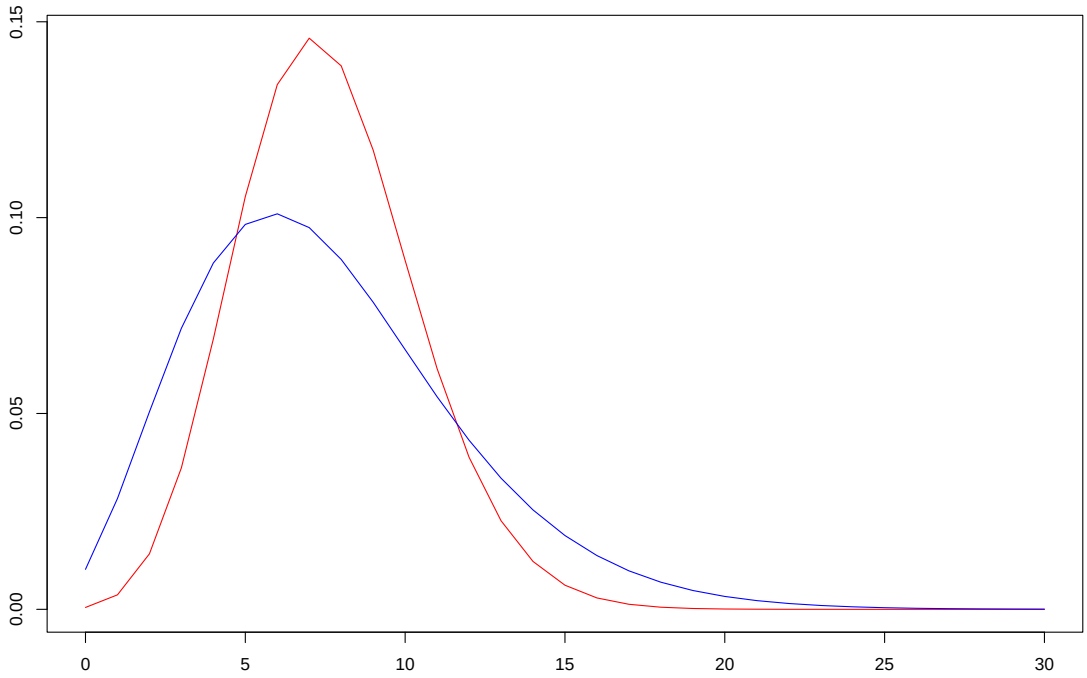


FIG. 5.13 – Loi exacte (en bleu) de $N_{1\,000}(\mathcal{W}_2)$ et binomiale de paramètres $\ell - m + 1 = 1\,000$ et $\mu(\mathcal{F}) = 4,541075 \times 10^{-3}$ (en rouge).

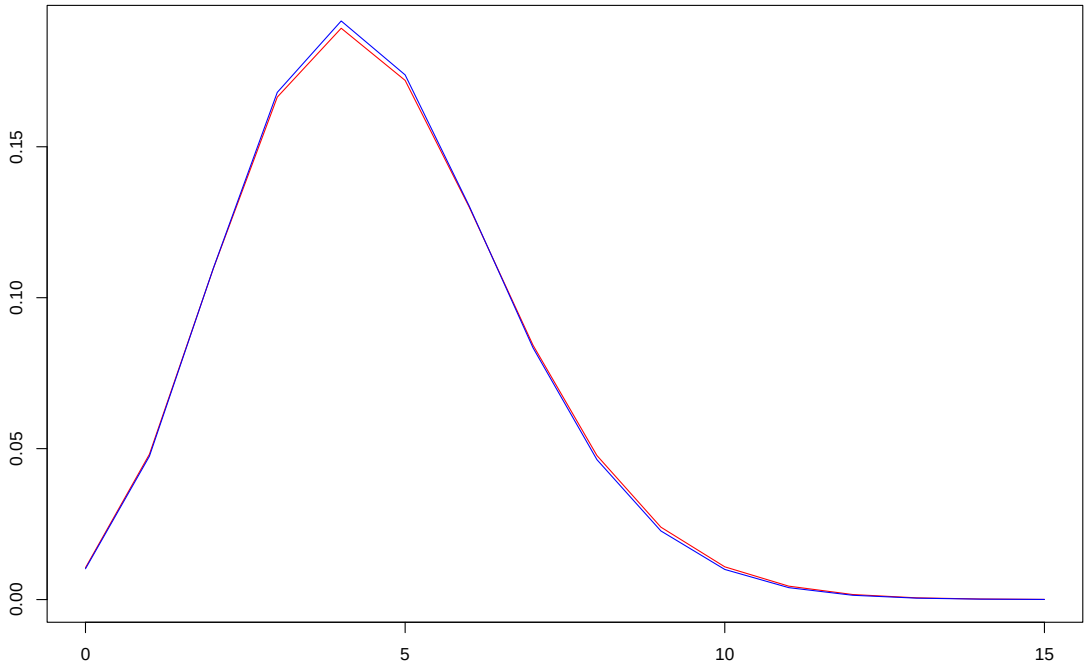


FIG. 5.14 – Loi exacte (en bleu) de $N^R_{1\,000}(\mathcal{W}_2)$ (comptages renouvelants) et binomiale de paramètres $\ell - m + 1 = 1\,000$ et $\mu(\mathcal{F}) = 7.607566 \times 10^{-3}$ (en rouge).

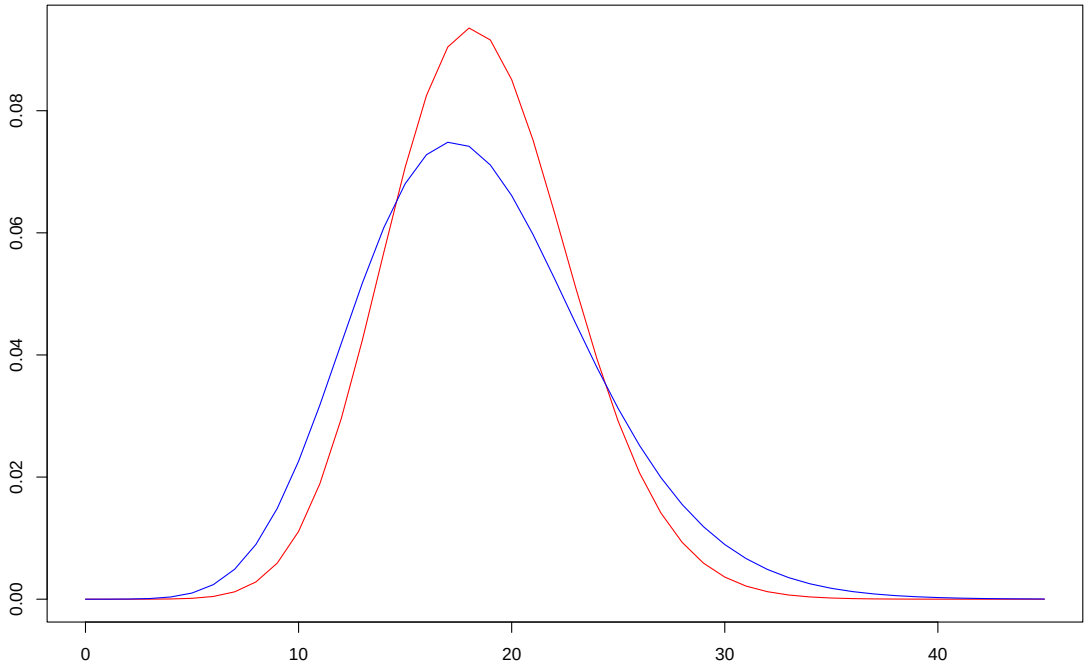


FIG. 5.15 – Loi exacte (en bleu) de $N_{1\,000}(\mathcal{W}_3)$ et binomiale de paramètres $\ell - m + 1 = 1\,000$ et $\mu(\mathcal{F}) = 1.8589919 \times 10^{-2}$ (en rouge).

p	$\delta(p, \mathtt{a})$	$\delta(p, \mathtt{c})$	$\delta(p, \mathtt{g})$	$\delta(p, \mathtt{t})$
0	10	1	11	0
1	10	2	11	0
2	12	4	13	3
3	10	1	11	5
4	12	6	13	7
5	10	1	8	0
6	12	9	13	7
7	10	1	8	5
8	10	1	11	0
9	12	9	13	7
10	10	1	11	0
11	10	1	11	0
12	10	1	11	5
13	10	1	11	5

FIG. 5.16 – Représentation du plus petit AFD non 1-ambigu associé au motif $\mathcal{W}_3 = \mathcal{W}_1 + \mathcal{W}_2 = \{\mathtt{cc.tg|ccccc}\}$ défini sur l'alphabet $\mathcal{A} = \{\mathtt{a}, \mathtt{c}, \mathtt{g}, \mathtt{t}\}$. $s = 0$ et $\mathcal{F} = \{8, 9\}$. Cet automate a 14 états.

5.6.1 Prise en compte de l'estimation des paramètres

Jusqu'à présent, on a supposé que le paramètre π de la chaîne de de Markov X était connu. En pratique cependant, ce paramètre doit être estimé sur une séquence ou sur un jeu de séquences. Il est donc naturel de se demander quel peut être l'effet de cette variabilité sur le calcul des statistiques de motifs.

Pour répondre à cette question, on se donne $m \geqslant 0$ et on considère $\chi = \chi_1 \dots \chi_n$ une chaîne de Markov de paramètre π défini par

$$\pi(w,a) = \mathbb{P}(\chi_{m+1} = a | \chi_1 \dots \chi_m = w) \quad \forall (w,a) \in \mathcal{A}^m \times \mathcal{A}$$

et on s'intéresse à $\widehat{\pi}$ l'estimateur par maximum de vraisemblance sur la séquence χ :

$$\widehat{\pi}(w,a) = \frac{\mathbf{N}_{m+1}(wa)}{\sum_{b \in \mathcal{A}} \mathbf{N}_{m+1}(wb)} \quad \forall (w,a) \in \mathcal{A}^m \times \mathcal{A}$$

où $\mathbf{N}_{m+1}$ est le vecteur (aléatoire) des fréquences de tous les mots de longueur $m+1$ dans la séquence χ. Notons que l'estimateur $\widehat{\mu}$ par maximum de vraisemblance de μ, la loi stationnaire de χ, est alors défini comme une fonction de $\widehat{\pi}$ (il est proportionnel au vecteur propre à gauche associé à la valeur propre 1 de la matrice de transition).

La loi de $\widehat{\mu}$ étant de ce fait assez complexe, on préfère introduire les estimateurs suivants :

$$\mu_{\mathbf{N}}(w) = \frac{\mathbf{N}_m(w)}{n-m+1} \quad \text{et} \quad \pi_{\mathbf{N}}(w,a) = \frac{\mathbf{N}_{m+1}(wa)}{\mathbf{N}_m(w)} \quad \forall (w,a) \in \mathcal{A}^m \times \mathcal{A}$$

où $\mathbf{N} = (\mathbf{N}_m, \mathbf{N}_{m+1})$. Notons que ces estimateurs sont asymptotiquement équivalents à ceux du maximum de vraisemblance.

Considérons maintenant $W = w_1 \dots w_h$ ($w_i \in \mathcal{A}$) un mot de longueur $h \geqslant m-1$ et on introduit

$$P(\mathbf{N}) = \mu_{\mathbf{N}}(w_1 \dots w_m) \times \pi_{\mathbf{N}}(w_1 \dots w_m, w_{m+1}) \times \dots \times \pi_{\mathbf{N}}(w_{h-m} \dots w_{h-1}, w_h)$$

la probabilité pour W d'apparaître à une position donnée dans une séquence markovienne d'ordre m de paramètres $(\mu_{\mathbf{N}}, \pi_{\mathbf{N}})$.

Pour tout $\ell \geqslant h$ et pour tout $0 \leqslant N_{\text{obs}} \leqslant \ell - h + 1$, on a alors

$$F(\mathbf{N}) \triangleq \mathbb{P}\left(\mathcal{B}(\ell - h + 1, P(\mathbf{N})) \geqslant N_{\text{obs}}\right) = \frac{\beta(P(\mathbf{N}), N_{\text{obs}}, \ell - h - N_{\text{obs}} + 2)}{\beta(N_{\text{obs}}, \ell - h - N_{\text{obs}} + 2)}$$

où les fonctions beta (complète et incomplète) sont définies pour tous $a, b > 0$ et tout $x \in [0,1]$ par

$$\beta(a,b) = \int_0^1 t^{a-1}(1-t)^{b-1} dt \quad \text{et} \quad \beta(x,a,b) = \int_0^x t^{a-1}(1-t)^{b-1} dt$$

la fonction beta incomplète est de plus différentiable en x et vérifie

$$\frac{\partial \beta(x,a,b)}{\partial x} = x^{a-1}(1-x)^{b-1} \tag{5.4}$$

(voir Press et al., 1997, chapitre 6, pour plus de détails sur les fonction beta et leurs relations avec la binomiale).

En remplaçant $\mu_{\mathbf{N}}$ et $\pi_{\mathbf{N}}$ par leurs expressions dans $P(\mathbf{N})$ on obtient

$$P(\mathbf{N}) = \frac{1}{n-m+1} \prod_{w \in \mathcal{A}^m} \frac{\prod_{a \in \mathcal{A}} \mathbf{N}_{m+1}(wa)^{A_{m+1}(wa)}}{\mathbf{N}_m(w)^{A_m(w)}} \tag{5.5}$$

où $A_{m+1}(wa)$ compte les occurrences du mot wa dans $W = w_1 \dots w_h$ et $A_m(w)$ celles du mot w in $w_2 \dots w_{h-1}$. Remarquons que dans le cas particulier où $h = m-1$, tous les $A_m(w)$ sont alors nuls et on obtient simplement $(n-m+1) \times P(\mathbf{N}) = \mathbf{N}_{m+1}(W)$.

Proposition 152. On peut approcher la loi de $\log_{10} F(\mathbf{N})$ par $\mathcal{N}(\nu, \sigma^2)$ avec

$$\nu = \log_{10} \mathbb{P}\left(\mathcal{B}(\ell - h + 1, p) \geqslant N_{\text{obs}}\right) \quad \text{et} \quad \sigma = \frac{\sqrt{{}^t\nabla F(\mathbf{M}) \times \mathbf{\Sigma} \times \nabla F^+(\mathbf{M})}}{\ln(10) F^+(\mathbf{M})}$$

où p est la probabilité qu'a le mot W d'apparaître dans une chaîne de Markov d'ordre m de paramètres (μ, π) et où $\mathbf{M}$ et $\mathbf{\Sigma}$ sont (respectivement) l'espérance et la matrice de covariance (toutes les deux calculées en section 5.5.3) du vecteur $\mathbf{N}$.

Démonstration. Un simple développement de Taylor d'ordre 1 [4] au voisinage de $\mathbf{M}$ donne

$$\log_{10} F(\mathbf{N}) \simeq \log_{10} F(\mathbf{M}) + \frac{{}^t(\mathbf{N} - \mathbf{M}) \nabla F(\mathbf{M})}{\log(10) F(\mathbf{M})}$$

Comme $P(\mathcal{M}) \simeq p$ on peut approcher l'espérance de $\log_{10} F(\mathbf{N})$ par ν et l'expression de la variance σ^2 s'obtient alors de manière immédiate. □

Corollaire 153. On a $\sigma = Q\sqrt{{}^t\mathbf{G} \times \mathbf{\Sigma} \times \mathbf{G}}$ avec

$$Q = \frac{p^{N_{\text{obs}}} (1-p)^{\ell - h - N_{\text{obs}} + 1}}{\log(10) \beta(p, N_{\text{obs}}, \ell - h - N_{\text{obs}} + 2)}$$

et ${}^t\mathbf{G} = \left[\begin{array}{cc} {}^t\mathbf{G}_m & {}^t\mathbf{G}_{m+1} \end{array} \right]$ où

$$\mathbf{G}_m(w) = -\frac{A_m(w)}{\mathbf{M}_m(w)} \quad \text{et} \quad \mathbf{G}_{m+1}(wa) = \frac{A_m(wa)}{\mathbf{M}_{m+1}(wa)}$$

pour tout $w \in \mathcal{A}^m$ et tout $a \in \mathcal{A}$.

Démonstration. Pour tout $(w, a) \in \mathcal{A}^m \times \mathcal{A}$ on dérive (5.5) et on trouve

$$\frac{\partial P(\mathbf{N})}{\partial \mathbf{N}_m(w)} = -\frac{A_m(w)}{\mathbf{N}_m(w)} \times P(\mathbf{N}) \quad \text{et} \quad \frac{\partial P(\mathbf{N})}{\partial \mathbf{N}_{m+1}(w)} = \frac{A_{m+1}(wa)}{\mathbf{N}_{m+1}(wa)} \times P(\mathbf{N})$$

si bien que $\nabla P(\mathbf{M}) = \mathbf{G}$. Il ne reste alors plus qu'à dériver F en utilisant (5.4) pour obtenir le corollaire. □

Exemple 154. On se place sur l'alphabet binaire $\mathcal{A} = \{\mathtt{a}, \mathtt{b}\}$ et on considère le modèle markovien d'ordre $m = 2$ défini dans l'exemple 139. On suppose que $n = 10\,000$ et on s'intéresse au mot $W = \mathtt{aaabbb}$ de taille $h = 6$.

[4] montrer la normalité asymptotique de $Z = f(X)$ à partir de celle de X et d'un développement de Taylor s'appelle la Δ-méthode (Lehmann, 2006).

Les termes non nuls de A_2 et A_3 sont :

$$A_2(\mathtt{aa}) = A_2(\mathtt{ab}) = A_2(\mathtt{bb}) = A_3(\mathtt{aaa}) = A_3(\mathtt{aab}) = A_3(\mathtt{abb}) = A_3(\mathtt{bbb}) = 1$$

si bien que les seuls termes non nuls de $\mathbf{G}$ sont finalement

$$G_2(\mathtt{aa}) = -\frac{1}{1251.03}; \quad G_2(\mathtt{ab}) = -\frac{1}{2499.78}; \quad G_2(\mathtt{bb}) = -\frac{1}{3749.04};$$

$$G_3(\mathtt{aaa}) = \frac{1}{500.36}; \; G_3(\mathtt{aab}) = \frac{1}{750.55}; \; G_3(\mathtt{abb}) = \frac{1}{2249.57}; \; G_3(\mathtt{bbb}) = \frac{1}{1499.47}.$$

Par ailleurs, on a $p = \mu(\mathtt{aa})\pi(\mathtt{a}|\mathtt{aa})\pi(\mathtt{b}|\mathtt{aa})\pi(\mathtt{b}|\mathtt{ab})\pi(\mathtt{b}|\mathtt{bb}) = 0.0108$ si bien qu'en fixant $\ell = n = 10\,000$ et $N_{\text{obs}} = 200$ on trouve

$$\nu = \log_{10} \mathbb{P}(\mathcal{B}(9\,995, 0.0108) \leqslant 200) = -14.99117$$

On a également

$$\log Q = 200 \times \log(0.018) + 9\,795 \times \log(0.9892) - \log\log(10) \\ - \log \beta(0.018, 200, 9\,796) = 3.7111921$$

donc $Q = 40.902536$. Comme on a par ailleurs ${}^t\mathbf{G} \times \mathbf{\Sigma} \times \mathbf{G} = 0.0019664$ on trouve finalement

$$\sigma = 1.8137689$$

ce qui constitue une variabilité conséquente autour de la moyenne.

Pour plus de détails sur cette méthode et sur les conséquences de cette sensitivité pour les études de motifs on se reportera à Nuel (2006e). Notons que les résultats présentés dans cet article semble indiquer un manque flagrant de robustesse pour les statistiques de motifs si bien que l'usage de modèles markovien d'ordres importants (ex : $m \geqslant 5$ pour $n \simeq 10^6$, ou $m \geqslant 4$ pour $n \simeq 10^5$) est fortement déconseillé.

5.7 Approximations de Poisson composées

5.7.1 Mots recouvrants

Nous avons déjà vu dans les discussions précédentes que le fait qu'un mot (ou un motif) soit recouvrant ou non est important pour les lois relatives au nombre de ses occurrences. Précisons cette notion d'abord pour un *mot* w, puis pour un motif $\mathcal{W}$ quelconque – et l'on constatera que la généralisation n'est pas évidente.

Définition 155. Si w et w' sont deux mots (de longueurs notées h et h'), on dit que w' recouvre w si les r dernières lettres de w sont les r premières lettres de w' pour un $r \geq 1$.

La conséquence immédiate est qu'il est possible d'avoir sur une séquence une occurrence de w en une position i et une occurrence de w' en position $i+p$ avec $p = h - r + 1$. Si $w = w'$, on parlera de mot *autorecouvrant*.

Par exemple `tctaggt` et `ggtaaa` sont recouvrants (avec $r = 3$) ; on peut observer sur une séquence `tctaggtaaa` (décalage de $p = 4$).

Un mot $w = w_1 \dots w_h$ sera autorecouvrant si et seulement si il est périodique c'est à dire si $\forall i,\ w_{i+p} = w_i$ dès que $i + p \leq h$. Bien sûr, si w admet la période p, il admet aussi la période $2p$ (si $2p < h$), $3p$ (si $3p < h$), etc ...

Exemple 156. aabaabaa est périodique, de période $p = 3$. Il admet aussi la période 6.

Structure d'un mot périodique : Si w est périodique et que p est la plus petite période, w répétera un certain nombre de fois – disons c fois – son préfixe de taille p, le mot D. Si la longueur h du mot n'est pas un multiple de p, il y aura un "suffixe reste", le mot R : w s'écrira $D(c)R$.

R est bien sûr un préfixe de w (tout comme de D). Il peut arriver que R lui même soit périodique et de période p_2 avec $1 \leq p_2 < p$.

Dans notre exemple, $D =$ aab se répète $c = 2$ fois, puis est suivi du reste $R =$ aa qui est lui même périodique et de période $p_2 = 1$.

Il en résulte que si, dans une séquence, une occurrence de w commence en une position que l'on numérotera 0, une autre occurrence peut aussi commencer en positions $p, \ldots, cp$ mais aussi en $cp + p_2$, voire $cp + 2p_2$, voire $cp + 3p_2$, etc, si ces longueurs sont inférieures à h.

Dans notre exemple on peut faire commencer une nouvelle occurrence de w après $p = 3$ ou $2p = 6$ positions, mais aussi après $cp + p_2 = 2p + 1 = 7$ positions. (aabaabaaabaabaa).

Notons que ce processus peut se reproduire, R s'écrivant comme c_2 répétitions de son préfixe de taille p_2, suivi d'un reste R_2, qui peut lui même être périodique, etc. Il sera facile au lecteur de formaliser ceci par une définition récursive. Contentons nous de donner un exemple un peu plus complexe : dans le mot suivant, nous avons souligné les positions où peut commencer une nouvelle occurrence.

aaabaaabaaabaacdaaabaaabaaabaacdggaaabaaabaaabaacdaaabaaabaaabaa

Les périodes sont $p = 34$ puis $34 + 16 = 50$ puis $50 + 4, 50 + 8, 50 + 12 = 62$ et enfin $62 + 1$.

Toujours de façon récursive, on dira qu'une période n'est pas principale si elle est conséquence des périodes précédemment trouvées (p.ex. si p est la plus petite période de w, $2p, 3p, \ldots$ ne seront pas principales). Les périodes principales sont donc celles obtenues du fait qu'un reste est lui-même périodique (avec les notations ci dessus, $cp + p_2$ sera principale [mais non pas $cp + 2p_2, cp + 3p_2, \ldots$] !).

Occurrences par paquets Il apparaît très vite qu'une occurrence d'un mot périodique w "facilite" une seconde apparition, qui chevauche la première. Si $w =$ aabaabaa commence en position i, pour avoir une autre occurrence en position $i+3$, il suffit que les trois lettres suivantes soient baa [5]. Les occurrences de w vont donc apparaître par paquets, c'est à dire par familles dans lesquelles chaque occurrence recouvre l'occurrence précédente : aabaabaabaabaabaabaabaa est un exemple de paquet de w, contenant 4 occurrences (leurs premières lettres ont été soulignées).

Une occurrence $w^{(1)}$ de w étant fixée, on peut noter θ la probabilité pour que (au moins) une autre occurrence de w la recouvre. Le nombre H d'occurrences dans un paquet se révèle alors géométrique : si $w^{(1)}$ ne recouvre pas une occurrence précédente, le paquet se réduit à cette unique occurrence avec probabilité $1 - \theta$; avec probabilité θ, le paquet se prolonge ; si $w^{(2)}$ est *la première* occurrence de w chevauchant $w^{(1)}$, le même raisonnement montre que le paquet s'arrête là avec probabilté $1 - \theta$ [donc $\mathbb{P}(H = 2) = \theta(1 - \theta)$ et $\mathbb{P}(H > 2) = \theta^2$]. Poursuivant ainsi, on obtient

$$\mathbb{P}(H = n) = (1 - \theta)\ \theta^n \tag{5.6}$$

[5] dans M00 observer w en une position fixée a pour probabilité $1/4^8 \approx 1/65000$, alors que compléter aabaa en w a une probabilité de $1/4^3 = 1/64$

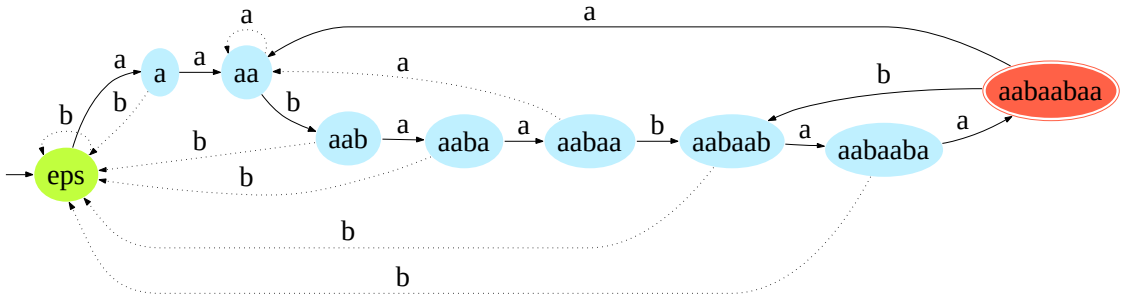


FIG. 5.17 – Représentation de l'AFD associé au mot $w =$ aabaabaa. On a représenté en trait plein les flèches correspondant à une obtention directe du mot à partir de s et les deux flèches issues de w : elles ne pointent pas sur eps

Calcul de θ Une occurrence $w^{(1)}$ de w peut être recouverte par plus d'une occurrence ultérieures. Mais il est évident que *la première* nouvelle occurrence de w qui recouvre $w^{(1)}$ ne peut commencer qu'en une période *principale*. Notons $R(p)$ la probabilité conditionnelle à une occurrence en position i qu'il y ait aussi une occurrence commençant en $i+p$. Le calcul de $R(p)$ est élémentaire dans tout modèle markovien.

Exemple 157. Toujours si $w =$ aabaabaa, $R(3)$ est la probabilité de voir baa derrière une occurrence. Dans M1, on aura $R(3) = \pi(\mathtt{a,b})\,\pi(\mathtt{b,a})\,\pi(\mathtt{a,a})$.

Les événements {le premier mot recouvrant $w^{(1)}$ commence en période principale p} étant incompatibles, on a

$$\theta = \sum R(p) \quad \text{somme prise sur les périodes principales} \tag{5.7}$$

Ce sera exactement la formule 5.8 (exprimée en terme de Π et non de π) définissant le coefficient d'autorecouvrement.

Loi de Poisson géométrique Nous ne développerons pas le résultat connu, qui dit que pour des mots rares w (en un sens à préciser) mais recouvrants, les occurrences de w ne recouvrant pas une occurrence (les premières occurrences dans chaque paquet) apparaissent asymptotiquement selon un processus de Poisson [6]. On en déduit que, dans ces conditions, $N_\ell(w)$ suit un type particulier de loi de Poisson composées, les lois de Poisson géométriques : nous retrouverons ce résultat dans un cadre plus général (voir 5.7.5).

Exemple du traitement par AFD La figure (5.17) représente l'AFD associé au mot $w =$aabaabaa. Comme ce mot est de longueur $h = 8$, toute boucle permettant d'aller de w à w en $h-1 = 7$ transitions ou moins est un recouvrement du mot.

La boucle correspondant à la période 3, $\delta(w, \mathtt{baa})$ est de longueur 3 : ($w \xrightarrow{\mathtt{b}}$ aabaab $\xrightarrow{\mathtt{a}}$ aabaaba $\xrightarrow{\mathtt{a}} w$) ; on vérifie que celle relative a la période 7 a bien pour longueur 7.

Ces boucles dans le graphe ne passent pas par $\mathcal{F} = w$ car elles correspondent aux périodes principales. Quant à la période non principale 6, elle correspond bien sûr à

[6] dont le taux λ' se calcule facilement en écrivant que l'espérance du nombre (total) d'occurrences sur une séquence est le produit de l'espérance du nombre de paquets par θ, qui est (aussi) l'espérance du nombre H d'occurrences par paquet.

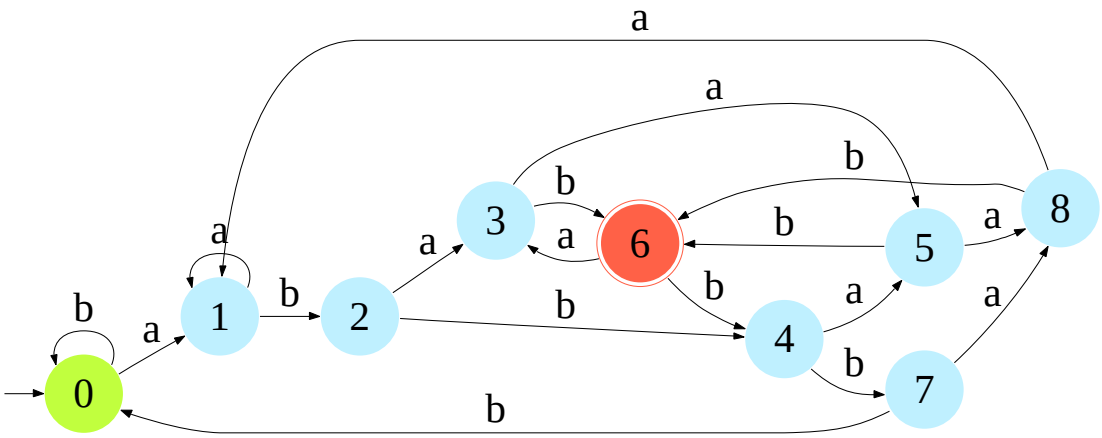


FIG. 5.18 – Représentation de l'AFD associé au motif $W =$ `ab.(0-2)ab`.

parcourir deux fois celle de longueur 3 (la définition 158 distingue un recouvrement non principal d'un recouvrement principal par le fait que le chemin des $\delta(i, x_n)$ rencontre ou non $\mathcal{F}$.)

Notons qu'aucune boucle utilisant 8 transition ou plus ne correspond à un recouvrement. En effet, pour une telle boucle, on serait arrivé dans l'état final aussi bien en partant de w que d'un autre état ; recouvrement correspond à la possibilité d'obtenir "plus facilement" w quand on part d'une de ses occurrences que "ex nihilo". La définition 158, appliquée au cas présent, demandera qu'il y ait une boucle partant de w et arrivant en w ($\delta(w, x) = w$), mais telle que le chemin défini par la même suite de lettres partant de l'état initial (noté s) n'amène pas en w (soit $\delta(s, x) \neq w$).

Cas de motifs La généralisation aux motifs n'est en aucun cas évidente : la table 5.2 (page 155) montre que le nombre d'états finaux pour un seul motif peut être très élevé et la notion de période d'un motif peut être délicate à définir lorsque la longueur des mots composant ce motif varie.

Présentons rapidement ce dernier point avec le motif $\mathcal{W} =$ `ab.(0-2)ab`, dont l'AFD est représenté en figure (5.18). Il est autorecouvrant, mais il n'y a pas de période p (au sens {une occurrence en position 0} $\Rightarrow$ {une occurrence est possible en position p}). On vérifie aisément sur la figure 5.18 qu'il y a six boucles $w \longrightarrow w$: ce sont les $\delta(w, x)$ pour $x =$ `ab` ($g = 0$), `aab`, `bab` ($g = 1$) et `aaab`, `baab` et `bbab` ($g = 2$) – on a noté g le nombre de points dans la seconde occurrence –, et que aucun des x de cette liste ne produit w ex nihilo (c'est à dire que la condition $\delta(s, x) \neq w$ est bien respectée).

5.7.2 Matrice d'auto-recouvrement

Définition 158. Fixons $i, j \in \mathcal{F}$ .Pour tout $x = x_1 \dots x_m \in \mathcal{A}^m$ on note $q_0 = i$ et $q_t = \delta(q_{t-1}, x_t)$ pour $1 \leqslant t \leqslant m$. On dit que x est un *recouvrement* de i par j si et seulement si $q_m = \delta(i, x) = j$ et $\delta(s, x) \neq j$. Si de plus $\{q_1, \dots, q_{m-1}\} \cap \mathcal{F} = \emptyset$ alors x est un recouvrement *principal* de i par j. On appelle *coefficient de recouvrement* de i par j

et l'on note $A_{i,j}$ la quantité

$$A_{i,j} = \sum_{x \in \mathcal{P}_{i,j}} \prod_{t=1}^{m} \Pi(q_{t-1}, q_t) \tag{5.8}$$

où $\mathcal{P}_{i,j}$ désigne l'ensemble des recouvrements principaux de i par j.

Proposition 159. Pour tout $i, j \in \mathcal{F}$, $A_{i,j}$ est calculable récursivement à l'aide de la relation :

$$A_{i,j} = \sum_{a \in \mathcal{A}} \Pi\big(i, \delta(i,a)\big) F_j\big(\delta(i,a), \delta(s,a)\big)$$

où $F_j(p,q)$ est définie pour tout $p, q \in \mathcal{Q}$ par

$$F_j(p,q) = \begin{cases} 0 & \text{si } p = q \text{ ou si } p \in \mathcal{F} \setminus \{j\} \\ 1 & \text{si } p = j \\ \sum_{a \in \mathcal{A}} \Pi\big(p, \delta(p,a)\big) F_j\big(\delta(p,a), \delta(q,a)\big) & \text{sinon} \end{cases}$$

Démonstration. On considère tous les chemins partant de i jusqu'à ce qu'on atteigne j (condition terminale $p = j$) ou que le chemin n'a plus aucune chance d'être un chevauchement de i par j (condition terminale $p = q$) ou d'être principal (condition terminale $p \in \mathcal{F} \setminus \{j\}$). □

Exemple 160. Appliquons la proposition 159. En reprenant l'AFD de la figure 5.16 on a $\mathcal{F} = \{8, 9\}$.

Commençons par le calcul de $A_{8,8}$. On a

$$A_{8,8} = \Pi(8,10)F_8(10,10) + \Pi(8,1)F_8(1,1) + \Pi(8,11)F_8(11,11) + \Pi(8,0)F_8(0,0)$$

Comme tous les $F_8(p,p)$ sont nuls, on obtient $A_{8,8} = 0$. Notons que la même décomposition en remplaçant F_8 par F_9 donne $A_{8,9} = 0$. A partir de maintenant, on va omettre directement tous les termes de la forme $F(p,p)$.

On considère ensuite le calcul de $A_{9,8}$. On a

$$A_{9,8} = \Pi(9,12)F_8(12,10) + \Pi(9,9)F_8(9,1) + \Pi(9,13)F_8(13,11) + \Pi(9,7)F_8(7,0)$$

et on a $F_8(12,10) = \Pi(12,5)F_8(5,0)$, $F_8(13,11) = \Pi(13,5)F_8(5,0)$, $F_8(7,0) = \Pi(7,8)F_8(8,11) + \Pi(7,5)F_8(5,0)$ et $F_8(5,0) = \Pi(5,8)F_8(8,11)$. En sommant toutes les contributions on obtient donc

$$\begin{aligned} A_{9,8} &= \Pi(9,12,5,8) + \Pi(9,13,5,8) + \Pi(9,7,8) + \Pi(9,7,5,0) \\ &= \pi(\texttt{atg}|\texttt{c}) + \pi(\texttt{gtg}|\texttt{c}) + \pi(\texttt{tg}|\texttt{c}) + \pi(\texttt{ttg}|\texttt{c}) \\ &= 0.014355 + 0.02079 + 0.117 + 0.01638 \\ &= 0.168525 \end{aligned}$$

Et avec la même décomposition, on trouve également $A_{9,9} = \Pi(9,9) = \pi(\texttt{c}|\texttt{c}) = 0.41$.

Remarque 161. Si un automates permet de compter les occurrences renouvelantes d'un motif, on sait (voir proposition 45) qu'alors $\delta(f,a) = \delta(s,a)$ pour tout $f \in \mathcal{F}$ et tout $a \in \mathcal{A}$. Il est donc clair que dans un tel cas $A_{i,j} = 0$ pour tous $i, j \in \mathcal{F}$.

5.7.3 Résultat principal

Définition 162 (loi de Poisson composée)**.** Soit $(\lambda_k)_{k\geqslant 1}$ une suite réelle telle que $\sum_{k\geqslant 1}\lambda_k = \lambda < \infty$; on dit que

$$N = \sum_{m=1}^{M} K_m \sim \mathcal{CP}\big((\lambda_k)_{k\geqslant 1}\big)$$

suit une *loi de Poisson composée* de paramètres $(\lambda_k)_{k\geqslant 1}$, si $M \sim \mathcal{P}(\lambda)$ est indépendant des K_m qui sont identiquement distribués comme K, avec

$$\mathbb{P}(M=m) = e^{-\lambda}\frac{\lambda^m}{m!} \quad \text{et} \quad \mathbb{P}(K=k) = \frac{\lambda_k}{\lambda}$$

Si de plus $\mathbb{E}[K^2] < \infty$ alors

$$\mathbb{E}[N] = \lambda\,\mathbb{E}[K] \quad \text{et} \quad \mathbb{V}[N] = \lambda\,\mathbb{E}[K^2]$$

Théorème 163. Pour tout $\ell \geqslant m$ on a

$$\mathcal{L}(N_\ell) \simeq \mathcal{CP}\big((\lambda_k)_{k\geqslant 1}\big)$$

avec

$$\lambda_k = (\ell - m + 1)\mathbf{M}\mathbf{A}^{k-1}(\mathbf{I}-\mathbf{A})^2\mathbf{1}$$

où $\mathbf{M} = \big(\mu(f)\big)_{f\in\mathcal{F}}$, $\mathbf{A} = (A_{i,j})_{i,j\in\mathcal{F}}$, $\mathbf{I}$ est la matrice identité et $\mathbf{1}$ est un vecteur colonne de 1.

Démonstration. Voir Roquain et Schbath (2006) pour la preuve de ce résultat dans le cas d'une matrice d'auto-recouvrement définie pour motif. Ce résultat est ici étendu (sans preuve) à la matrice d'auto-recouvrement définie pour un automate. □

Corollaire 164. Si de plus $\mathbf{I}-\mathbf{A}$ est inversible alors :

$$\lambda = (\ell - m + 1)\mathbf{M}(\mathbf{I}-\mathbf{A})\mathbf{1} \quad \text{et} \quad \mathbb{E}[K] = \frac{(\ell - m + 1)\mathbf{M}\mathbf{1}}{\lambda}$$

Démonstration. Ce résultat utilise simplement les séries de matrices :

$$\sum_{k\geqslant 1}\mathbf{A}^{k-1} = (\mathbf{I}-\mathbf{A})^{-1} \quad \text{et} \quad \sum_{k\geqslant 1} k\mathbf{A}^{k-1} = (\mathbf{I}-\mathbf{A})^{-2}$$

□

5.7.4 Cas Poisson

Proposition 165. Si $\mathbf{A} = 0$ alors, pour tout $\ell \geqslant m$ on a $\mathcal{L}(N_\ell) \simeq \mathcal{P}(\lambda)$ avec $\lambda = (\ell - m + 1)\mu(\mathcal{F})$.

Démonstration. On applique le théorème 163 avec $\mathbf{A} = 0$ et on trouve $\lambda_1 = (\ell - m + 1)\mathbf{M}\mathbf{1} = \ell\mu(\mathcal{F})$ et $\lambda_k = 0$ si $k \geqslant 2$. Ainsi, $\lambda = (\ell - m + 1)\mu(\mathcal{F})$ et la variable aléatoire K (dans la définition de la loi de Poisson composée) prend toujours la valeur 1. Le résultat est alors évident. □

Remarque 166. D'après la proposition 148 on a

$$\mathcal{L}(N_\ell) \simeq \mathcal{B}\left(\ell - m + 1, \mu(\mathcal{F})\right)$$

donc si ℓ est grand devant $\mu(\mathcal{F})$ la distribution obtenue est très proches de $\mathcal{P}(\ell\mu(\mathcal{F}))$. Ainsi, les résultats obtenus avec une approximation binomiale ou de Poisson sont en général très similaires.

Remarque 167. Le calcul de la fonction de répartition ou de la fonction de survie d'une loi de Poisson s'effectue de manière classique à l'aide de la fonction Gamma incomplète (évaluée via une série ou bien via un fraction continue). Le lecteur intéressé se reportera aux ouvrages de références sur le sujet citées au paragraphe 5.10.

5.7.5 Cas Poisson géométrique

Définition 168 (loi de Poisson géométrique)**.** Si $N \sim \mathcal{CP}\big((\lambda_k)_{k\geqslant 1}\big)$ avec

$$\lambda_k = \lambda(1-\theta)^{k-1}\theta \quad \forall k \geqslant 1$$

avec $\lambda > 0$ et $0 < \theta \leqslant 1$ alors $N \sim \mathcal{GP}(\lambda, \theta)$ suit une loi de Poisson géométrique de paramètre λ pour la partie Poisson et de paramètre θ pour la partie géométrique. De plus on a

$$\mathbb{E}[N] = \frac{\lambda}{\theta} \quad \text{et} \quad \mathbb{V}[N] = \frac{\lambda(2-\theta)}{\theta^2}$$

Proposition 169. Si $\mathbf{A} = (1-\theta)\mathbf{I}$ alors, pour tout $\ell \geqslant m$ on a $\mathcal{L}(N_\ell) \simeq \mathcal{GP}(\lambda, \theta)$ avec $\lambda = (\ell - m + 1)\mu(\mathcal{F})\theta$.

Démonstration. On applique le théorème 163 avec $\mathbf{A} = (1-\theta)\mathbf{I}$ ce qui donne $\lambda_k = (\ell - m + 1)\mu(\mathcal{F})(1-\theta)^{k-1}\theta^2$, donc $\lambda = (\ell - m + 1)\mu(\mathcal{F})\theta$ si bien finalement que $\lambda_k = \lambda(1-\theta)^{k-1}\theta$. □

Exemple 170. Reprenons le motif de l'exemple 150. L'état final 9 de l'automate 5.16 correspondant clairement au motif $\mathcal{W}_2$ on a $\mathbf{A} = A_{9,9} = 0.41$ (comme cela a été calculé dans l'exemple 160) et donc $\theta = 0.59$. On a par ailleurs calculé $\mu(\mathcal{F}) = 7.607566 \times 10^{-3}$ si bien que

$$\mathcal{L}(N_{1000}) \simeq \mathcal{GP}(\lambda = 4.4884639, \theta = 0.59)$$

Afin de pouvoir utiliser ce résultat, il est nécessaire d'être capable de calculer efficacement la distribution, et les fonctions de répartition ou de survie d'une loi de Poisson géométrique. C'est précisément ce point qui est abordé dans la suite de cette partie.

A partir de maintenant, on suppose que $N \sim \mathcal{GP}(\lambda, \theta)$ avec $\lambda > 0$ et $0 < \theta < 1$ (le cas dégénéré $\theta = 1$ correspondant simplement au cas Poisson).

Proposition 171. On a $\mathbb{P}(N = 0) = e^{-\lambda}$ et pour tout $n \geqslant 1$ on a

$$\mathbb{P}(N = n) = \sum_{m=1}^{n} e^{-\lambda}\frac{\lambda^m}{m!}(1-\theta)^{n-m}\theta^m C_{n-1}^{m-1}$$

où C_{n-1}^{m-1} désigne le coefficient binomial.

Démonstration. D'après la définition 162

$$\begin{aligned}\mathbb{P}(N=n) &= \sum_{m=1}^{n} e^{-\lambda}\frac{\lambda^m}{k!}\sum_{k_1,\ldots,k_m\in\mathbb{N}^*}\mathbb{I}_{\{k_1+\ldots+k_m=n\}}\frac{\lambda_{k_1}\times\ldots\times\lambda_{k_m}}{\lambda^m}\\ &= \sum_{m=1}^{n} e^{-\lambda}\frac{\lambda^m}{k!}\sum_{k_1,\ldots,k_m\in\mathbb{N}^*}\mathbb{I}_{\{k_1+\ldots+k_m=n\}}(1-\theta)^{n-m}\theta^m\end{aligned}$$

Reste à compter le nombre de façons de totaliser n avec une somme de m entiers au moins égaux à un (et en tenant compte de l'ordre de ces entiers). Pour cela, considérons simplement l'écriture $n = 1+1+1\cdots+1$ (n termes !) ; cette expression contient $n-1$ signes $+$; on obtient une solution du problème posé en choisissant $m-1$ de ces signes et en les remplaçant disons par le mot "et", ce qui est possible de C_{n-1}^{m-1}façons, chacune donnant une solution différente.

Par exemple pour $5 = 1+1+1+1+1$ et $m = 3$, il y a 6 façons d'effacer 2 signes $+$. Un choix est (1 et $1+1$ et $1+1$), soit la décomposition en (1,2,2).

□

Corollaire 172 (récurrence directe)**.** On a

$$\mathbb{P}(N=0) = e^{-\lambda} \quad \text{et} \quad \mathbb{P}(N=1) = e^{-\lambda}(1-\theta)z \quad \text{avec} \quad z = \frac{\lambda\theta}{1-\theta}$$

et, pour tout $n \geqslant 2$

$$\mathbb{P}(N=n) = \frac{(2n-2+z)}{n}(1-\theta)\mathbb{P}(N=n-1) + \frac{(2-n)}{n}(1-\theta)^2\mathbb{P}(N=n-2)$$

Démonstration. En utilisant la définition 13.1.2 de Abramowitz et Stegun (1972) on montre aisément que

$$\mathbb{P}(N=n) = e^{-\lambda}(1-\theta)^n z M(-n+1,2,-z)$$

où M désigne fonction confluente hypergeométrique de Kummer (parfois également notée ${}_1F_1$). Grâce à (13.1.2 Abramowitz et Stegun, 1972), on a $e^{-z}M(n+1,2,z) = M(-n+1,2,-z)$ et on sait également (13.4.1 Abramowitz et Stegun, 1972) que $\forall a,b,\in\mathbb{R}$ on a

$$M(a+1,b,z) = \frac{1}{a}\left[(2a-b+z)M(a,b,z) + (b-a)M(a-1,b,z)\right]$$

On remplace fait simplement $a = n$ et $b = 2$ pour obtenir le corollaire. □

Exemple 173. On s'intéresse à la loi de de N avec $\lambda = 4.4884639$ et $\theta = 0.59$. On a $\mathbb{P}(N=0) = \exp(-\lambda) = 0.01123789$, $z = (\lambda\times\theta)/(1-\theta) = 6.459009$ et $\mathbb{P}(N=1) = \exp(-\lambda)\times\lambda\times\theta = 0.02976012$.

– la relation de récurrence avec $n = 2$ donne :

$$\mathbb{P}(N=2) = 1.73409685\times\mathbb{P}(N=1) + 0\times\mathbb{P}(N=0) = 0.05160693$$

– puis avec $n = 3$:

$$\mathbb{P}(N=3) = 1.42939790\times\mathbb{P}(N=2) - 0.05603333\times\mathbb{P}(N=1) = 0.07209927$$

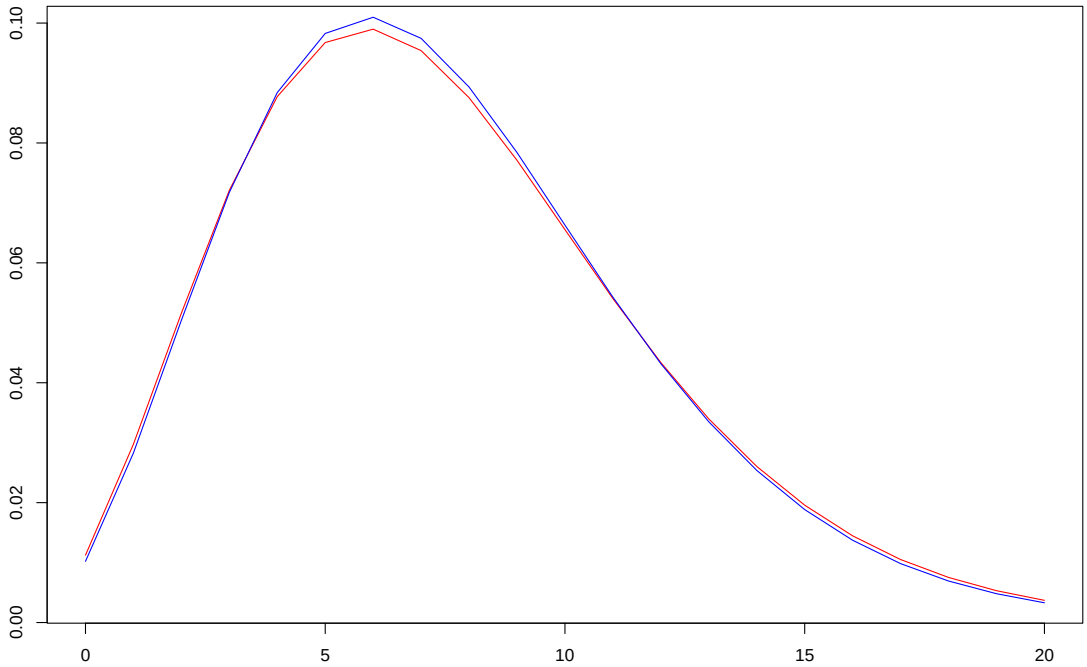


FIG. 5.19 – Loi exacte (en bleu) de $N_{1\,000}(\mathcal{W}_2)$ et Poisson géométrique de paramètres $\lambda = 4.4884639$ et $\theta = 0.59$ (en rouge).

– $n = 4$:

$$\mathbb{P}(N = 4) = 1.2770484 \times \mathbb{P}(N = 3) - 0.0840500 \times \mathbb{P}(N = 2) = 0.0877367$$

– et ainsi de suite jusqu'à $n = 10$:

$$\mathbb{P}(N = 10) = 1.00281937 \times \mathbb{P}(N = 9) - 0.13448 \times \mathbb{P}(N = 8) = 0.06554734$$

La figure 5.19 compare la distribution exacte de $N_{1000}(\mathcal{W}_2)$ à l'approximation par une loi de Poisson géométrique. A la différence de l'approximation binomiale (voir figure 5.13) on a ici une très bonne adéquation des deux distributions ; la structure d'auto-recouvrement du motif a bien été prise en compte.

En en sommant les termes de la distribution on peut également obtenir les valeurs de la fonction de répartition que l'on peut comparer au valeurs exactes :

n	0	1	2	3
$\mathbb{P}(N_{1\,000} \leqslant n)$	0.01021174	0.03849789	0.08891733	0.1606732
$\mathbb{P}(\mathcal{GP}(\lambda, \theta) \leqslant n)$	0.01123789	0.04099801	0.09260494	0.1647042

Il est donc possible de calculer la loi ($\mathbb{P}(N = n)$) ou la fonction de répartition ($\mathbb{P}(N \geqslant n)$) d'une loi de Poisson géométrique avec une complexité linéaire en temps et constante en mémoire en utilisant cette relation de récurrence. Le calcul de la fonction de survie ($\mathbb{P}(N > n)$) pose cependant des problèmes spécifiques. Pour des raisons numériques, il n'est pas recommandé d'effectuer son calcul en utilisant la formule $\mathbb{P}(N > n) = 1 - \mathbb{P}(N \leqslant n)$ car on récupère ainsi une erreur *absolue* (et non relative) égale à la précision des calculs. Une meilleur solution consiste à sommer les termes $\mathbb{P}(N = i)$ à partir de $i \geqslant n$ jusqu'à ce que la convergence numérique soit atteinte. Si une telle approche est souvent efficace en pratique elle n'est pas totalement satisfaisante dans la mesure ou rien n'assure que la convergence numérique corresponde à la convergence théorique. Voici pourquoi nous nous proposons ici de donner une expression directe de la queue de la distribution à travers la proposition suivante :

Proposition 174. On a $\mathbb{P}(N \geqslant 1) = 1$ et, pour tout $n \geqslant 1$ on a

$$\mathbb{P}(N \geqslant n) = \sum_{m=1}^{n} H(n, m) + \sum_{m>n} e^{-\lambda} \frac{\lambda^m}{m!}$$

avec

$$H(n, m) = e^{-\lambda} \frac{(\lambda\theta)^m}{m!} (1 - \theta)^{n-m} C_{n-1}^{m-1} F(1, n, n - m + 1, 1 - \theta)$$

où la fonction F (parfois également notée ${}_2F_1$) désigne la série hypergéométrique de Gauss (voir définition A.S.15.1.1).

Démonstration. Pour tout $n \geqslant 1$ on a

$$\mathbb{P}(N \geqslant n) = \sum_{i=n}^{\infty} \sum_{m=1}^{i} e^{-\lambda} \frac{\lambda^m}{m!} (1 - \theta)^{i-m} \theta^m C_{i-1}^{m-1}$$

En inversant permutant les deux sommes on obtient

$$\mathbb{P}(N \geqslant n) = \sum_{m=1}^{n} e^{-\lambda} \frac{\lambda^m}{m!} \sum_{i=n}^{\infty} (1 - \theta)^{i-m} C_{i-1}^{m-1} + \sum_{m>n} e^{-\lambda} \frac{\lambda^m}{m!} \sum_{i=m}^{\infty} (1 - \theta)^{i-m} C_{i-1}^{m-1}$$

or

$$\sum_{i=m}^{\infty}(1-\theta)^{i-m}C_{i-1}^{m-1}=\theta^{-m}$$

et

$$\sum_{i=n}^{\infty}(1-\theta)^{i-m}C_{i-1}^{m-1}=(1-\theta)^{n-m}C_{n-1}^{m-1}F(1,n,n-m+1,1-\theta)$$

d'où le résultat. □

Corollaire 175 (récurrence complémentaire)**.** Pour tout $n\geqslant 1$ on a

$$H(n,1)=e^{-\lambda}\lambda(1-\theta)^{n-1}\quad\text{et}\quad H(n,2)=\lambda\frac{(n-2)\theta+1}{2(1-\theta)}H(n,1)$$

ainsi que la relation de récurrence suivante pour tout $3\leqslant m\leqslant n$

$$H(n,m)=z\frac{n-m+1-(n-2m+2)(1-\theta)}{\theta m(m-1)}H(n,m-1)\\+z^2\frac{(n-m+1)(2-m)(1-\theta)}{\theta m(m-1)^2(m-2)}H(n,m-2)$$

avec $z=\lambda\theta/(1-\theta)$.

Démonstration. Grâce à A.S.15.2.12 on sait que pour tous a,b,c,x on a

$$c(c-1)(z-1)F(a,b,c-1,x)+c(c-1-(2c-a-b-1)x)F(a,b,c,x)\\+(c-a)(c-b)xF(a,b,c+1,x)=0$$

En prenant $a=1$, $b=n$, $c=n-m+2$ et $x=(1-\theta)$ on obtient

$$(n-m+1)(n-m+2)(1-\theta)F(1,n,n-m+1,1-\theta)=\\+(n-m+2)(n-m+1-(n-2m+2)(1-\theta))F(1,n,n-m+2,1-\theta)\\+(n-m+1)(2-m)(1-\theta)F(1,n,n-m+3,1-\theta)$$

dont on tire aisément la relation de récurrence sur $H(n,m)$. □

Exemple 176. Toujours avec $\lambda=4.4884639$ et $\theta=0.59$, calculons $\mathbb{P}(N\geqslant 21)$.

On a

$$H(21,1)=\exp(-\lambda)\times\lambda\times(1-\theta)^{21}=9.087821\times10^{-10}$$

et

$$H(21,2)=\lambda\times\frac{19\times\theta+1}{2(1-\theta)}H(21,1)=66.83432\times H(11,1)=6.073784\times10^{-8}$$

Puis on utilise la récurrence

– pour $m=3$ on a :

$$H(21,3)=21.94968\times H(21,2)-45.90246\times H(11,1)=1.291461\times10^{-6}$$

– $m=4$:

$$H(21,4)=10.81063\times H(21,3)-14.49551\times H(21,2)=1.308108\times10^{-5}$$

– et ainsi de suite jusqu'à $m = 21$:

$$H(21, 21) = 0.2291150 \times H(21, 20) - 0.003451313 \times H(11, 9) = 1.087410 \times 10^{-8}$$

En en sommant ces termes au fur et à mesure des calculs on trouve

$$\sum_{m=1}^{21} H(n, m) = 0.007711767$$

Par ailleurs, on peut utiliser la fonction gamma incomplète comme en section 5.7.4 pour obtenir

$$\sum_{m>21} e^{-\lambda} \frac{\lambda^m}{m!} = 2.750010 \times 10^{-9}$$

En sommant ces deux résultats on obtient donc

$$\mathbb{P}(N \geqslant 21) = 0.007711767 + 2.750010 \times 10^{-9} = 0.00771177$$

à comparer à la valeur exacte : $\mathbb{P}(N_{1000} \geqslant 21) = 0.006471458$.

A titre de vérification, on calcule également (récurrence directe)

$$1 - \mathbb{P}(N \leqslant 20) = 1 - 0.9922882 = 0.00771177$$

ce qui donne bien le même résultat.

On dispose donc ainsi d'un algorithme linéaire en temps et constant en espace permettant le calcul de la fonction de survie d'une loi de Poisson géométrique.

5.7.6 Cas général

Lorsqu'il n'est possible ni de se ramener au cas d'une loi de Poisson, ni même au cas d'une loi de Poisson géométrique, il est alors nécessaire de calculer les fonctions de répartition et de survie d'une loi de Poisson composée. Nous expliquons ici comment on procède.

On commence par introduire une réécriture de la loi de Poisson composée :

Proposition 177. La loi de $N \sim \mathcal{CP}((\lambda_k)_{k\in\mathbb{N}^*})$ avec $\sum_{k=1}^{\infty} \lambda_k = \lambda$ et $p_k = \lambda_k/\lambda$ est donnée pour tout $n \geqslant 1$ par

$$\mathbb{P}(N = n) = \sum_{(k_1,\ldots,k_n)\in\mathcal{K}_n} e^{-\lambda} \lambda^{k_1+k_2+\ldots+k_n} \frac{p_1^{k_1} p_2^{k_2} \ldots p_n^{k_n}}{k_1! k_2! \ldots k_n!}$$

avec

$$\mathcal{K}_n = \{(k_1, k_2, \ldots, k_n) \in \mathbb{N}^n, 1k_1 + 2k_2 + \ldots + nk_n = n\}$$

Démonstration. Partant de la définition 162, il est clair que si $M = m$ alors $(K_1, \ldots, K_m)$ est distribué selon la loi multinomiale $\mathcal{M}(m, p_1, \ldots, p_n)$. Il suffit alors de considérer les combinaisons dont la somme fait n via l'ensemble $\mathcal{K}_n$ pour obtenir le résultat. □

Corollaire 178. Avec les même notations que dans la proposition 177 on a $\mathbb{P}(N=n) = D(n,n)$ pour tout $n \geqslant 1$, D étant défini par

$$D(i,j) \triangleq \sum_{(k_1,\ldots,k_i)\in\mathcal{K}_{i,j}} e^{-\lambda}\lambda^{k_1+\ldots+k_i}\frac{p_1^{k_1}\ldots p_i^{k_i}}{k_1!\ldots k_i!} \quad \text{avec} \quad i \geqslant 1 \quad \text{et} \quad j \geqslant 0$$

où

$$\mathcal{K}_{i,j} = \{(k_1,\ldots,k_i) \in \mathbb{N}^i, 1k_1 + \ldots + ik_i = j\}$$

De plus, pour tout $i \geqslant 1$ on a la relation de récurrence suivante :

$$D(i,j) = \sum_{k=0}^{E(j/i)} \frac{\lambda^k p_i^k}{k!} D(i-1, j-ik)$$

avec, par convention, $D(0,0) = e^{-\lambda}$ et, pour tout $j \geqslant 1$, $D(0,j) = 0$. Dans le cas particulier où $i = 1$ on obtient ainsi

$$D(1,j) = e^{-\lambda}\frac{\lambda^j p_1^j}{j!}$$

Démonstration. Le seul point délicat est la relation de récurrence qui s'obtient en effectuant une somme sur les différentes valeurs possibles pour k_i. □

Exemple 179. On reprend le motif de l'exemple 151 pour lequel on a calculé la matrice d'auto-recouvrement dans l'exemple 160 :

$$\mathbf{A} = \begin{pmatrix} 0 & 0 \\ 0.168525 & 0.41 \end{pmatrix}$$

En utilisant la PMC de l'automate correspondant on trouve également $\mathbf{M} = (0.0109824; 0.0076076)$ et donc $\mu(\mathcal{F}) = \mathbf{M1} = 0.0185899$.

On trouve $\lambda = \ell\mathbf{M}(\mathbf{I}-\mathbf{A})\mathbf{1} = 14.188813$ et les premiers termes de la distribution de K ($\mathbb{E}[K] = 0.168525$) sont donnés par

i	1	2	3	4	5
$\mathbb{P}(K=i)$	0.8169896	0.1079761	0.0442702	0.0181508	0.0074418

En appliquant le corollaire 178 on obtient facilement la distribution de N :

n	0	1	2	3	4
$10^4 \times \mathbb{P}(N=n)$	0.0068846	0.0798067	0.4731116	1.9139577	5.9484901

La figure 5.20 compare la distribution exacte à l'approximation par la loi de Poisson composée. On constate une bien meilleure ressemblance entre les deux distribution que dans le cas de l'approximation binomiale (voir figure 5.15).

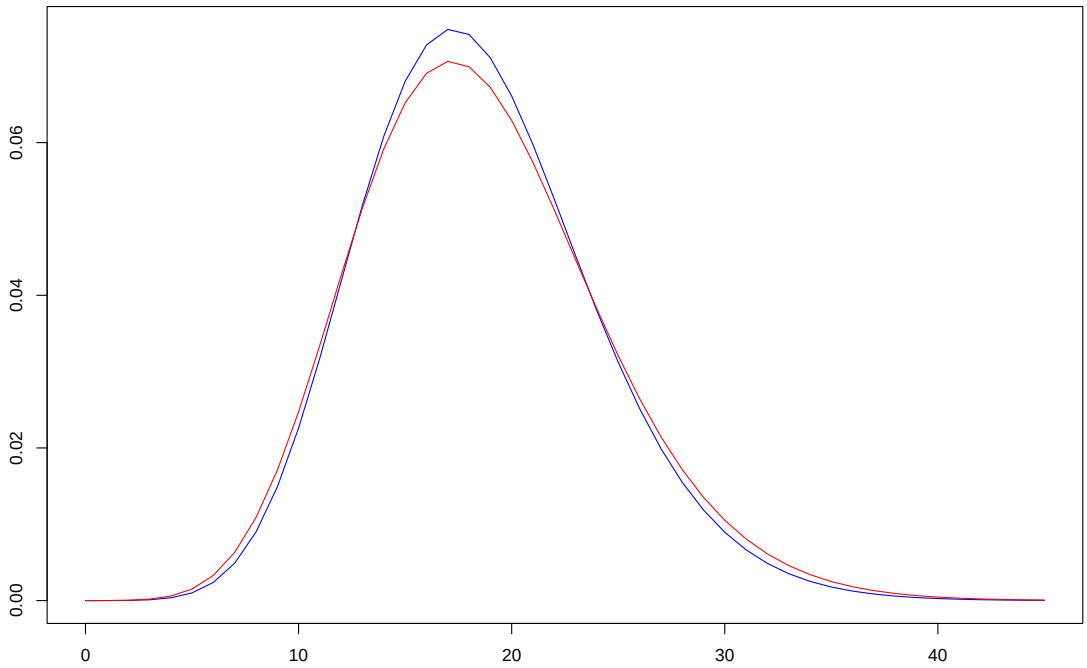


FIG. 5.20 – Loi exacte (en bleu) de $N_{1\,000}(\mathcal{W}_3)$ et Poisson composée (en rouge).

5.8 Grandes déviations

5.8.1 Introduction

Soit $(X_i)_{1\leqslant i\leqslant n}$ est un n-échantillon de la variable aléatoire réelle X dont les deux premiers moments sont finis (on note $\mathbb{E}[X] = m$ et $\mathbb{E}[X] = \sigma^2$). La loi des grands nombre affirme que si $S_n = X_1 + \ldots + X_n$ alors le rapport S_n/n tend presque sûrement vers μ quand n tend vers l'infini ; la moyenne empirique tend vers l'espérance. De plus, le Théorème de la Limite Centrale (TLC) indique que la loi de S_n est asymptotiquement gaussienne. Ainsi, les déviations de S_n autour de sa moyenne $n\mu$ sont asymptotiquement de l'ordre de grandeur de $\sigma\sqrt{n}$ (l'écart-type de S_n).

Ainsi, pour tout $a > 0$, $\mathbb{P}(S_n - \mu n \geqslant a\sqrt{n})$ tend vers une constante lorsque n tend vers l'infini. Si l'on s'intéresse maintenant aux probabilités de la forme $\mathbb{P}(S_n - \mu n \geqslant an)$, on considère alors des déviations de l'ordre de grandeur de n (et non plus de $\sqrt{n}$) et on entre dans le cadre de la théorie *grandes déviations* qui affirme que ces probabilités décroissent à vitesse exponentielle.

Pour fixer les idées, considérons un exemple simple : X est une variable de Bernoulli de paramètre $p = \mathbb{P}(X = 1)$ ($\mathbb{P}(X = 0) = 1 - p$). On a, $\mu = p$ et $\sigma = \sqrt{p(1-p)}$.

Comme $S_n \sim \mathcal{B}(n, p)$ on a

$$\mathbb{P}(S_n - pn \geqslant an) = \mathbb{P}(\mathcal{B}(n, p) \geqslant (\mu + a)n)$$

En utilisant le TLC on obtient

$$\mathbb{P}(S_n - pn \geqslant an) \simeq \mathbb{P}\left(\mathcal{N}(0, 1) \geqslant \frac{a\sqrt{n}}{\sigma}\right)$$

Si on considère maintenant le théorème de Cramer-Chernov on obtient

$$\mathbb{P}(S_n - pn \geqslant an) \simeq e^{-nI(p+a)}$$

avec

$$I(p+a) = (p+a)\log\left(\frac{p+a}{p}\right) + (1-p-a)\log\left(\frac{1-p-a}{1-p}\right)$$

Exemple 180. Considérons le cas où $p = 0.01$, $n = 10\,000$ et $a = 0.0001$. On a

$$P = \mathbb{P}(S_{10\,000} - 100 \geqslant 1) = \mathbb{P}(\mathcal{B}(10\,000, 0.01) \geqslant 101) = 0.4734375$$

L'approximation gaussienne donne

$$P_{\text{TLC}} = \mathbb{P}\left(\mathcal{N}(0,1) \geqslant 0.1005038\right) = 0.4599722$$

et les grandes déviations donnent

$$P_{\text{GD}} = e^{-10\,000 \times 5.033923 \times 10^{-7}} = 0.9949787$$

Comparons les mêmes quantité pour quelques valeurs de a :

a	0.0001	0.001	0.01	0.02	0.03	0.04	0.05
$-\log_{10} P$	0.325	0.771	18.2	58.6	114	181	257
$-\log_{10} P_{\text{TLC}}$	0.337	0.803	23.3	89.4	199	353	550
$-\log_{10} P_{\text{GD}}$	0.00219	0.212	17.0	57.2	113	180	255

Si, lorsque l'on considère de petites valeurs de a, l'approximation gaussienne donne de bon résultats tandis que l'approximation par les grandes déviations est de mauvaise qualité, la tendance s'inverse lorsque a augmente. Ainsi, pour $a = 0.05$ par exemple, l'approximation gaussienne donne $P_{\text{TLC}} \simeq 10^{-550}$ alors que la valeur exacte est $P \simeq 10^{-257}$!

Si le TLC est particulièrement adapté à l'étude du centre de la distribution, les approximations de types grandes déviations sont connues pour être de bien meilleure qualité lorsque l'on s'intéresse à la queue de la distribution (c'est à dire aux petites p-values).

Partant de ce constat, l'idée de recourir aux grandes déviations pour le calcul de statistiques de motifs semble particulièrement intéressante ; c'est précisément ce que nous allons faire ici.

5.8.2 Niveau 1

Il existe des résultats de grandes déviations de différents niveaux selon que l'on s'intéresse à la distribution de la moyenne de variables aléatoires (niveau 1) ou à la loi empirique de ces variables (niveau 2). Nous allons ici nous concentrer sur le premier niveau, traitant succinctement le second dans la section 5.8.3.

Comme précédemment, on considère la PMC $Y = Y_m \dots Y_\ell$ de matrice de transition $\Pi = P + Q$ et on s'intéresse à la loi de N_ℓ.

Pour tout $\theta \in \mathbb{R}$, on introduit d'abord $\rho(\theta)$ la plus grande valeur propre réelle de la matrice $P + e^\theta Q$ (son existence est assurée par l'irréductibilité de la PMC et le théorème de Perron-Frobénius). On pose ensuite $\Lambda(\theta) = \log \rho(\theta)$ et on considère sa duale de Legendre Λ^* définie pour tout $a \in \mathbb{R}$ par

$$\Lambda^*(a) = \sup_{\theta \in \mathbb{R}} \{\theta.a - \Lambda(\theta)\}$$

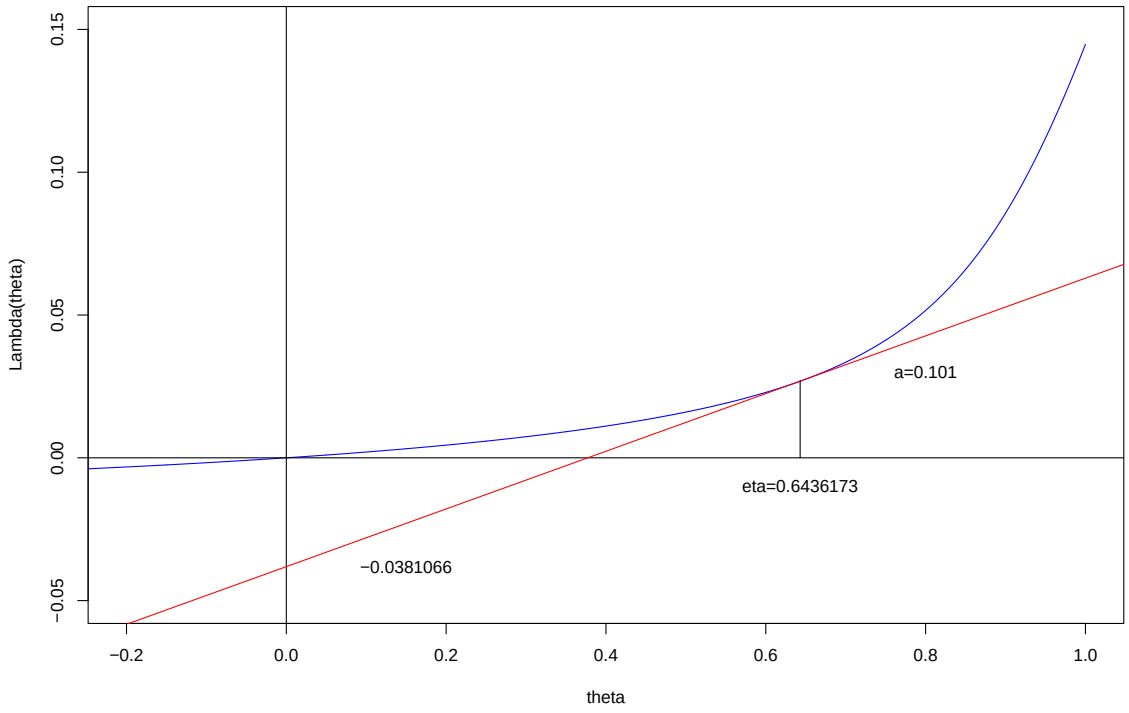


FIG. 5.21 – Courbe $y = \Lambda(\theta)$ (en bleu). La tangente à la courbe de pente $a = 0.101$ est matérialisée en rouge. Le point de contact se trouve à l'abscisse $\eta = 0.6436173$. En considérant l'intersection de cette tangente avec l'axe des ordonnées on trouve $\Lambda^*(a) = -(-0.0381066) = 0.0381066$.

Remarque 181. Une simple étude de fonction montre que $\Lambda^*(a) = \eta.a - \lambda(\eta)$ pour η vérifiant $\Lambda'(\eta) = a$. Ainsi une abscisse pour de la duale de Legendre correspond à une pente pour la fonction initiale.

Exemple 182. La figure 5.21 nous propose une construction géométrique de $\Lambda^*(a)$ à partir de la courbe de la fonction Λ. On cherche tout d'abord le point η tel que $\Lambda'(\eta) = a$ puis l'intersection de la tangente en ce point avec l'axe des ordonnées donne alors immédiatement l'opposé de $\Lambda^*(a)$.

Théorème 183 (Cramer-Chernov)**.** Pour tout $a \geqslant 0$ on a

$$\lim_{\ell \to +\infty} \frac{1}{\ell - m + 1} \log \mathbb{P}\left(\frac{N_\ell}{\ell - m + 1} \geqslant a \right) = -\Lambda^*(a) \quad \text{si } a \geqslant \frac{\mathbb{E}[N_\ell]}{\ell - m + 1}$$

et

$$\lim_{\ell \to +\infty} \frac{1}{\ell - m + 1} \log \mathbb{P}\left(\frac{N_\ell}{\ell - m + 1} \leqslant a \right) = -\Lambda^*(a) \quad \text{si } a \leqslant \frac{\mathbb{E}[N_\ell]}{\ell - m + 1}$$

Remarque 184. Si $a = 0$ il est facile de voir qu'on a alors $\Lambda^*(0) = -\log \lambda$ où λ est la plus grande valeur propre de la matrice P. Le théorème de Cramer-Chernov s'écrit donc dans ce cas :

$$\lim_{\ell \to +\infty} \frac{1}{\ell - m + 1} \log \mathbb{P}\left(N_\ell = 0 \right) = \lambda$$

ce qui est cohérent avec le développement asymptotique du corollaire 102 qui indique que $\mathbb{P}(N_\ell = 0) \simeq (\mu_0 \Pi_\infty \mathbf{1}) \times \lambda^{\ell-m}$

Exemple 185. On reprend le motif de l'exemple 151 ($m1$ et $\ell = 1\,000$). Voici ce que donne le résultat de Cramer-Chernov pour quelques valeurs de a :

n	0	1	100	101
$a = n/(\ell - m + 1)$	0.0	1.0×10^{-3}	1.0×10^{-2}	1.01×10^{-2}
$10^2 \times \Lambda^*(a)$	1.5234953	1.1640148	3.7464082	3.8106638
$e^{-(\ell-m+1)\Lambda^*(a)}$	2.42×10^{-7}	8.81×10^{-6}	5.37×10^{-17}	2.82×10^{-17}
$\mathbb{P}(N_\ell \leqslant n)$	2.62×10^{-7}	3.7×10^{-6}	nd	nd
$\mathbb{P}(N_\ell \geqslant n)$	nd	nd	1.70×10^{-18}	8.85×10^{-19}

Les ordres de grandeurs sont corrects, mais il y des différences sensibles. Notons que dans le cas particulier où $n = 101$, la figure 5.21 montre la construction géométrique de $\Lambda^*(a)$.

Le théorème de Cramer-Chernov nous renseigne donc sur la vitesse de décroissance exponentielle de la probabilité considérée, mais on peut obtenir un résultat plus précis avec le :

Théorème 186 (Bahadur-Rao)**.** Pour tout $a \geqslant \mathbb{E}[N_\ell]/(\ell - m + 1)$ (resp. $\leqslant$) non nul, on a

$$\mathbb{P}(N_\ell \geqslant (\ell - m + 1)a) \sim C(a) \times e^{-(\ell-m+1)\Lambda^*(a)} \quad \text{(resp. } \leqslant\text{)}$$

asymptotiquement lorsque ℓ tend vers l'infini avec

$$C(a) = \frac{1}{(1 - e^{-|\eta|}) \sqrt{2\pi(\ell - m + 1)\Lambda''(\eta)}}$$

où η vérifie $\Lambda'(\eta) = a$.

Remarque 187. Attention, Dans le cas où $a = 0$ il est clair que $\eta = 0$ si bien que le terme $C(0)$ n'est pas défini. On se ramène donc dans ce cas à Cramer-Chernov (voir remarque 184).

Exemple 188. On poursuit 185 en considérant maintenant le théorème de Bahadur et Rao. La table suivante compare les valeurs exactes de la statistique de motif (p-values en échelle log-décimale) aux approximations de grandes déviations :

n	0	1	100	101
exact	-6.58	-5.43	17.77	18.05
Cramer-Chernov	-6.62	-5.06	16.27	16.55
Bahadur-Rao	nd	-5.42	17.68	17.96

Dans le cas où $a = 0$ la correction de Bahadur et Rao n'est pas applicable mais, comme cela est expliqué dans la remarque 184, le résultat de Cramer-Chernov est très proche de la valeur exacte. Pour $a > 0$ on constate une bien meilleure qualité de l'approximation utilisant la correction de Bahadur et Rao.

Calculs numériques

Si les résultats de grandes déviations présentés ci-avant sont intéressants, leur mise en oeuvre pratique peut se révéler délicate. En effet, le calcul de $\Lambda^*(a)$ nécessite la maximisation d'une fonction dont chaque évaluation requiert le calcul de la plus grande valeur propre d'une matrice d'ordre L. Il est donc nécessaire à la fois d'être capable de calculer cette valeur propre de la manière efficace et d'effectuer la maximisation avec aussi peu que possible d'évaluations de la fonction.

Pour ce deuxième point, il est classique de recourir à l'algorithme de Brent qui mélange l'optimisation par la méthode du nombre d'or (*golden section* en anglais) et des approximations paraboliques. En ce qui concerne le calcul de $\rho(\theta)$, on peut calculer l'ensemble des valeurs propres de $P + e^{\theta}Q$ avec une complexité $O(L^3)$ en temps et $O(L^2)$ en espace en utilisant les algorithmes classiques d'algèbre linéaire (réduction sous une forme de Hessenberg puis décomposition QR). Il est également possible que calculer uniquement la plus grande valeurs propre par l'algorithme d'Arnoldi (*implicitly restarted*) avec une complexité $O(k \times L)$ (c'est le nombre de termes non nuls de la matrice) en temps et en espace. Si la complexité de la seconde approche est plus faible que la première, il faut néanmoins remarquer qu'en pratique, l'approche classique reste plus efficace pour de petites valeurs de L (ex : $L \leqslant 20$).

Une fois $\Lambda^*(a) = \eta.a - \Lambda(\eta)$ déterminé (avec $\Lambda'(\eta) = a$), l'approximation de Bahadur et Rao requiert le calcul de $\Lambda''(\eta)$. Comme on ne dispose pas de formule explicite pour la fonction Λ, il n'y a malheureusement pas de formule analytique permettant ce calcul. Pour résoudre ce problème nous proposons d'utiliser une approximation de Chebyshev d'ordre 10 sur l'intervalle $[\eta - 1, \eta + 1]$ pour la fonction Λ puis de dériver deux fois cette approximation pour obtenir $\Lambda''(\eta)$. Cette approche requérant 11 évaluations de Λ elle résulte en une complexité $O(k \times L)$ en temps comme en espace et s'effectue en pratique plus rapidement que la détermination de $\Lambda^*(a)$.

5.8.3 Niveau 2

Avec les grandes déviations de niveau 1 on s'intéresse à la loi d'une moyenne (ici, le nombre moyen d'occurrences du motif). En niveau 2, on considère la distribution de la loi empirique, c'est à dire de la loi jointe des comptages des lettres dans la séquence. Dans le cas d'une chaîne de Markov $Y = Y_m \dots Y_\ell$ il est plus naturel de s'intéresser à la distribution empirique des fréquences c'est à dire à D_ℓ^2 définie par

$$D_\ell^2 = \left(\begin{array}{ccccc} \frac{N_\ell(11)}{\ell - m} & \dots & \frac{N_\ell(ij)}{\ell - m} & \dots & \frac{N_\ell(LL)}{\ell - m} \end{array} \right)$$

où $N_\ell(ij)$ désigne le nombre d'occurrence du mot ij dans la séquence Y. La somme de tous ces comptages faisant $\ell - m$ (longueur de la séquence moins un), il est clair que $D_\ell^2 \in \mathcal{M}_1^+(\mathcal{Q}^2)$ l'ensemble des loi de probabilité sur $\mathcal{Q}^2$.

Théorème 189 (Grandes Déviations pour les fréquences)**.** Pour tout $\Gamma \subset \mathcal{M}_1^+(\mathcal{Q}^2)$ on a

$$-\inf_{\overset{\circ}{\Gamma}} I \leqslant \liminf_{\ell \to +\infty} \frac{1}{\ell - m} \log \mathbb{P}(D_\ell^2 \in \Gamma) \leqslant \limsup_{\ell \to +\infty} \frac{1}{\ell - m} \log \mathbb{P}(D_\ell^2 \in \Gamma) \leqslant -\inf_{\overline{\Gamma}} I$$

où la fonction de taux $I : \Gamma \to \mathbb{R}$ est infinie sauf si

$$\gamma \in \mathcal{S} = \left\{ \gamma \in \mathcal{M}_1^+\left(\mathcal{Q}^2\right), \forall i \in \mathcal{Q}, \sum_{j \in \mathcal{Q}} \gamma(ij) = \sum_{j \in \mathcal{Q}} \gamma(ji) = \overline{\gamma}(i) \right\}$$

auquel cas

$$I(\gamma) = H\left(\gamma|\overline{\gamma}\otimes\Pi\right) = \sum_{i,j\in\mathcal{Q}} \gamma(ij)\log\left(\frac{\gamma(ij)}{\overline{\gamma}(i)\Pi(i,j)}\right)$$

où Π est la matrice de transition de Y.

Remarque 190. Avec le niveau 1 (théorème de Cramer-Chernov par exemple), il est nécessaire de maximiser une fonction dont les évaluations sont complexes (la fonction Λ) dans un espace simple (la droite réel) alors que dans le niveau 2, on doit minimiser une fonction simple (une entropie relative) dans un espace complexe (le sous-ensemble contraint $\mathcal{S}\subset\mathbb{R}^{L^2}$). Ce "transfert" de la difficulté entre les niveaux 1 et 2 est tout à fait classique dans la théorie des grandes déviations.

Remarque 191. La fonction I est convexe (car c'est une entropie relative) et son minimum est atteint en $\gamma_0 = \mu\otimes\Pi$ où $I(\gamma_0)=0$, μ étant la loi stationnaire de Y. En effet $\sum_j \mu(i)\Pi(i,j) = \mu(i)$ car Π est stochastique et $\sum_j \mu(j)\Pi(j,i) = \mu(i)$ car $\mu\Pi=\mu$. On a donc $\overline{\gamma_0}\otimes\Pi = \mu$ d'où le résultat.

Corollaire 192. Soit $v\in\mathcal{Q}^2$ et soit $a\geqslant 0$ tel que $a\geqslant\gamma_0(v)$ (resp. $\leqslant$) alors

$$\lim_{\ell\to+\infty}\frac{1}{\ell-m}\log\mathbb{P}\left(\frac{N_\ell(v)}{\ell-m}\geqslant a\right) = \inf\left\{I(\gamma),\gamma\in\mathcal{S},\gamma(v)=a\right\}\quad(\text{resp. }\leqslant)$$

Démonstration. On applique le théorème 189 avec $\Gamma = \left\{\gamma\in\mathcal{M}_1^+\left(\mathcal{Q}^2\right),\gamma(v)\geqslant a\right\}$ (resp. $\leqslant$). Supposons que l'infimum de I sur Γ est atteint en un certain γ_1 (qui est nécessairement un élément de $\mathcal{S}$). On a donc $\gamma_1\geqslant a\geqslant\gamma_0$ si bien qu'il existe $c\in[0,1]$ tel que $a = c\gamma_1(v)+(1-c)\gamma_0(v)$. On pose $\gamma_2 = c\gamma_1+(1-c)\gamma_0$ et, en utilisant la convexité de I on obtient

$$I(\gamma_2)\leqslant cI(\gamma_1)+(1-c)I(\gamma_2) = cI(\gamma_1)$$

si bien que $I(\gamma_1)\leqslant I(\gamma_2)\leqslant cI(\gamma_1)$ donc $c=1$ c'est à dire que $\gamma_1(v)=a$. L'infimum est donc atteint sur le bord du domaine Γ. On conclut par un argument de continuité pour la borne inférieure du théorème 189. □

Remarque 193. Le corollaire reste valable si on considère une famille de mots. Par exemple, si v et w sont deux éléments de $\mathcal{Q}^2$ et $a\geqslant 0$ tels que $a\leqslant\gamma_0(v)+\gamma_0(w)$ alors l'infimum de I sur l'ensemble $\Gamma = \left\{\gamma\in\mathcal{M}_1^+\left(\mathcal{Q}^2\right),\gamma(v)+\gamma(w)\geqslant a\right\}$ est encore atteint sur le bord du domaine. On utilisera ainsi typiquement ce corollaire avec l'ensemble des mots de la forme $\mathcal{Q}\times\mathcal{F}$ de sorte que la somme des comptages donne la fréquence du motif initialement considéré.

Mise en oeuvre pratique

Pour utiliser le corollaire 192, il est donc nécessaire de minimiser la fonction I sur $\{\gamma\in\mathcal{S},\gamma(v)=a\}$. Pour cela, on commence par introduire $F\subset E=\mathbb{R}^{L^2}$ le sous espace vectoriel défini par les équations suivantes :

$$\begin{cases}\sum_{j\in\mathcal{Q}}\gamma(ij) = \sum_{j\in\mathcal{Q}}\gamma(ji) & \forall i\in\mathcal{Q}\\ \sum_{i,j\in\mathcal{Q}}\gamma(ij)=1\\ \gamma(v)=a\end{cases}$$

Il suffit ensuite de minimiser I sur l'intersection de F^+ de F avec le demi-espace $E^+=\mathbb{R}^{+L^2}$. Soit P la projection orthogonale de E sur F (on peut l'obtenir avec une

complexité $O(L^3)$ en temps et $O(L^2)$ en mémoire) et soit $\gamma_1 \in \mathcal{F}^+$ (on peut par exemple prendre le vecteur des fréquences observées dans une séquence donnée). Comme on a

$$\inf_{\gamma \in F^+} I(\gamma) = \inf_{\nu \in E, P(\nu) \in F^+} I \circ P(\nu)$$

on effectue une minimisation sans contrainte de $I \circ P$ via une descente du gradient conjuguée en partant d'un point de départ ν_1 vérifiant la contrainte $P(\nu_1) \in F^+$ (on peut pour cela se servir de γ_1) et en effectuant toutes les minimisation unidimensionnelles sur des segments vérifiant la contrainte.

5.9 Comparaison des méthodes

On dispose donc d'un grands nombre de méthodes concurrentes permettant de calculer la statistique d'exceptionnalité d'un motif. Chacune de ces méthodes présente ses propres avantages et inconvénients si bien qu'il n'est en général pas facile de choisir celle que l'on va utiliser. L'objectif de cette partie est justement de faire le point sur leurs qualités respectives.

On propose tout d'abord de faire (section 5.9.1 la synthèse de leurs complexités respectives et de présenter quelques exemples concrets de motifs. Nous comparons ensuite les approches markovienne et *shuffle* pour les approximations gaussiennes (section 5.9.2) ainsi que les performances des grandes déviations précises ou non (section 5.9.3). Enfin, l'efficacité des méthodes sont comparées spécifiquement dans les queues de distributions (section 5.9.4) et sur des cas réels (section 5.9.5) avant de conclure (5.9.6).

Dans toute cette partie, c'est le logiciel SPatt (version 1.2.2, compilé avec gcc 3.4.1) qui est utilisé pour les calculs. La machine utilisé est une station de travail équipée d'un processeur intel P4 cadencé à 2.8 Gz et muni de 512 Mo de mémoire sous le systême d'exploitation linux 2.6.8.1.

Deux critères sont utilisés dans les comparaisons :

a) l'erreur relative définie comme le rapport

$$\left| \frac{\text{valeur} - \text{référence}}{\text{référence}} \right|$$

et le plus souvent exprimée en % ;

b) le tau de Kendall [7] défini comme le taux d'accord de rangs entre une liste observée et une liste de référence. Un tau de Kendall de 100% correspond à des rangs identiques dans les deux listes et -100% à des rangs parfaitement inversés. Si l'on note p la la probabilité que deux couples pris au hasard soit ordonnés de la même façon dans les deux listes, on a $\tau = 2p - 1$.

5.9.1 Complexités

La table 5.1 propose une synthèse des complexités en temps et en espace pour les différentes méthodes proposées. On peut voir que la complexité kL (k est la taille de

[7] Si des individus i ($1 \leq i \leq n$ sont classés en rang $R(i)$ pour un premier critère et en rang $S(i)$ pour un second, on pose $D(i,j) = +1$ si les ordres $R(i)$-$R(j)$ et $S(i) - S(j)$ coïncident, $D(i,j) = -1$ sinon. Le tau de Kendall est la moyenne des $D(i,j)$: $\tau = 2/(n(n-1)) \sum D(i,j)$ (voir Sheskin (2003) ou Lehmann (2006)).

méthode	temps	espace
exacte	$O(knL\ell)$	$O(nL)$
gaussienne exacte	$O(kLF\ell)$	$O(kL + F\ell)$
gaussienne approchée*	$O(kLF\alpha)$	$O(kL + F\alpha)$
binomiale	$O(kL + \log n)$	$O(kL)$
Poisson	$O(kL + \log n)$	$O(kL)$
géometrique Poisson**	$O(kL + n)$	$O(kL)$
Poisson composée**	$O(kL + F^3 + n^3)$	$O(kL + F^2 + n^2)$
grandes déviations	$O(kL)$	$O(kL)$
grandes déviations précises	$O(kL)$	$O(kL)$

TAB. 5.1 – Complexités en temps et en espace pour le calcul de la statistique d'un motif en utilisant différentes méthodes. ℓ est la longueur de la séquence, k la taille de l'alphabet, m l'ordre du modèle markovien, L le cardinal de l'espace d'état de la PMC, F le nombre d'état finaux, α le rang à partir duquel la PMC est considérée comme stationnaire (en général $a \leqslant 100$). (*) avec une erreur en $O(\ell\nu^\alpha)$ où ν est la seconde valeur propre (en module) de Π, la matrice de transition de la PMC. (**) . Les complexités indiquées n'incluent pas le calcul de la matrice d'auto-recouvrement.

l'alphabet et L le cardinal de l'espace d'état de la PMC) qui correspond au nombre de termes non nuls de la matrice de transition de la PMC est toujours présente dans les complexités. Ce chiffre correspond en fait au coût du produit de cette matrice avec un vecteur (opération élémentaire de l'ensemble des méthodes proposées) dans le cas de la complexité temporelle et au coût de stockage de cette matrice dans le cas de la complexité en espace. Les calculs exacts (p-values ou moments) multiplient les complexités par le facteur ℓ (longueur de la séquence) ce qui est bien naturel puisqu'on ne peut dans ce cas se permettre aucune approximation. En contrepartie, les complexités de ces méthodes restent identiques si l'on considère des modèles hétérogènes (c'est à dire dont les paramètres dépendent de la position dans la séquence). Pour le calcul de p-values, apparaît en plus le facteur n (nombre d'occurrences du motif). Au final, les méthodes exactes, calculs de p-values en particulier, sont difficilement utilisables pour de très longues séquences et/ou (mais cela va souvent de paire) de grands nombres d'occurrences. Il faut cependant remarquer que les complexités obtenues restent raisonnable en espace et linéaires avec l'ensemble des paramètres considérés en temps. Ainsi obtenir la valeur d'une p-value sur une séquence 100 fois plus longue ne demande "que" 100 fois plus de temps (pour le même motif et même nombre d'occurrence).

Pour le calcul des moments (et de l'approximation gaussienne qui va avec), l'utilisation d'une approximation permet d'accélérer énormément les calculs par rapport à l'approche exacte au prix d'une erreur le plus souvent négligeable en pratique. Pour les approches binomiales et Poisson, le calcul de la loi stationnaire de la PMC est d'abord nécessaire (complexité kL en utilisant des produits successifs, ou mieux, l'algorithme d'Arnoldi) auquel s'ajoute le calcul de fonction de répartition ou de la fonction de survie de la loi correspondante afin d'obtenir la p-value. Cette dernière tâche peut être particulièrement simple et rapide dans le cas des approximations binomiale ou de Poisson, un peu plus complexe dans le cas géométrique Poisson, et franchement lourde (voir inutilisable) dans le cas général d'une loi de Poisson composée. En prime, toutes les approximations de Poisson nécessitent le calcul préalable de la matrice d'auto-recouvrement ce qui peut prendre un temps considérable pour des motifs complexes (quand F, le nombre d'états

motif	alphabet	$m = 0$	$m = 1$	$m = 2$
`bb|aba`	binaire	8(2)	7(2)	7(2)
`abab|baa`	binaire	6(2)	5(2)	5(2)
`a.(2)ba`	binaire	14(2)	14(2)	14(2)
`gctggtgg|ccaccagc`	ADN	17(2)	18(2)	28(2)
`g.tggtgg`	ADN	11(1)	15(1)	25(1)
`g.tggtgg|ccacca.c`	ADN	24(4)	27(4)	37(5)
`cctgg.(0-12)cctgg`	ADN	66(1)	92(1)	226(1)
`cctgg.(5-12)cctgg`	ADN	127(6)	173(6)	381(6)
`ttgaca.(16-18)tataat`	ADN	1 059 (43)	1 365 (43)	2 715 (43)
`atgca.(11)att.(12)gcatt`	ADN	2 220 (84)	2 871 (84)	5 645 (84)
`PROSITE1`	protéine	21(1)	96(2)	513(4)
`PROSITE2`	protéine	329(30)	1 393 (78)	10 688 (633)
`PROSITE3`	protéine	16 064(163)	66 620 (163)	333 324 (163)

TAB. 5.2 – Pour chaque motif, le cardinal L de l'espace d'états de la PMC est donné, ainsi que le nombre F d'états finaux (entre parenthèses). Les motifs PROSITE sont `PROSITE1=W.WH.CH.H[YN]HS[MI][DE]`, `PROSITE2=[LIVMF]GE.[GAS][LIVM].(5-11)R[STAQ]A.[LIVMA].[STACV]` et `PROSITE3=CG(2).(4-7)G.(3)C.(4-5)C.(3-5)[NHGS].[FYWMI].(2)QC`

finaux, est grand). L'approche Poisson composée est donc certainement la seule de ces approches qui sera rédhibitoire dans un certain nombre de situations courantes (ex : F ou n grands). Les grandes déviations enfin, gardent une complexité proportionnelle à kL grâce à l'algorithme d'Arnoldi. Comme un grand nombre de calculs de valeurs propres est nécessaire pour effectuer les minimisations (algorithme de Brent), il faut cependant bien avoir conscience que les calculs peuvent être beaucoup plus longs que dans le cas des autres approximations (où un seul de ces calculs est le plus souvent nécessaire). Les grandes déviations précises ont la même complexité que les grandes déviations simples mais, du fait d'évaluations de valeurs propres additionnelles (approximation de Chebyshev) requièrent en pratique environ trois fois plus de temps.

On comprend bien que le cardinal L est un point critique pour toutes ces complexités. Afin de mieux appréhender l'influence de ce paramètre, nous donnons la valeurs dans la table 5.2 pour quelques exemples de motifs et quelques valeurs de m l'ordre markovien considéré.

Pour un motif donné, il est clair que la taille de l'automate associé croit avec m. Il est néanmoins possible que la PMC associée soit définie sur un ensemble de cardinal légèrement plus petit (comme cela est expliqué en remarque 84). C'est exactement ce qui se passe avec les exemples pris dans l'alphabet binaire. En revanche, dans tous les autres cas, on retombe sur un comportement plus naturel où L croit avec m. A noter que dans le pire des cas (c'est à dire si tous les états de l'automate présentent une ambiguïté), L doit être mutiplié par k lorsque m augmente de 1. Dans les exemples présentés ici, on est bien loin d'une telle croissance le facteur étant au pire égale à 1.0 dans le cas de l'alphabet binaire ($k = 2$), 2.2 dans le cas de l'ADN ($k = 4$) et 5.0 dans le cas des protéines ($k = 20$).

En ce qui concerne la valeurs de F, est elle parfois exactement égale à la cardinalité du motif (c'est à dire le nombre de mots non dégénérés que contient le motif) mais est le plus souvent bien inférieure. Par exemple, la cardinalité du motif `cctgg.(0-12)cctgg`

méthode	$h=2$	$h=3$	$h=4$	$h=5$	$h=6$	$h=7$	$h=8$
exacte	20.66	15.63	17.07	19.31	25.02	43.68	145.77
gaussienne approchée	0.00	0.00	0.00	0.01	0.03	0.11	0.42
binomiale	0.00	0.00	0.00	0.01	0.03	0.09	0.33
géométrique Poisson*	0.00	0.00	0.01	0.01	0.04	0.012	0.46
grandes déviations	0.02	0.07	0.42	2.22	9.34	27.16	67.27
grandes déviations précises	0.06	0.31	1.82	9.33	38.25	100.97	199.41

TAB. 5.3 – Complexités-temps empiriques (exprimées en secondes) pour le traitement de tous les mots de longueur h du génome HIV type 1 ($\ell = 9\,718$) pour un modèle d'ordre $m = 1$ avec différentes méthodes. On a donc $L = h + 3$, 4^h mots traités dont la somme des nombres d'occurrences vaut $\ell - (h-1)$. (*) tous les mots non recouvrants sont traités avec une approximation de Poisson.

est $\simeq 2.2 \times 10^7$ alors que $F = 1$, la cardinalité du motif `PROSITE3` est $\simeq 10^{30}$ alors que $F = 163$. C'est évidemment là que s'exprime toute la puissance de l'approche du problème des motifs par les automates. Remarquons que la valeur de F peut rester stable ou augmenter avec m mais que ce comportement varie fortement en fonction des motifs considérés.

Donnons enfin avec la table 5.3, le temps de calcul empirique pour obtenir les statistiques de 4^h mots sur un génome donné dans un modèle d'ordre $m = 1$. On sait que dans un tel cas, $L = h + 3$ et $F = 1$ pour chaque mot. La somme des occurrences de tous ces mots étant évidemment de l'ordre de grandeur de n, on s'attend à obtenir un temps de calcul stable pour la méthode exacte. Ce n'est cependant pas ce que l'on observe ici et cela est dû à deux phénomènes. Le premier (assez marginal) est lié au fait que les calculs exacts dans le cas où le nombre d'occurrences n est nul s'affranchissent de la longueur ℓ de la séquence grâce au développement asymptotique introduit en proposition 99. Le second phénomène est lié à l'augmentation linéaire de L mais surtout à la croissance exponentielle du nombre de mots à traiter. Ainsi, pré-traitement de chacun de ces mots (incluant notamment la construction de la PMC qui lui est associée et l'initialisation de l'espace mémoire) finit par prendre le pas sur le temps de calcul proprement dit. Les approches gaussienne (approchée) et binomiale sont très rapides et proposent des temps de calculs très similaires. Du fait du calcul linéaire avec n de la p-value dans le cas de l'approximation par des lois de Poisson géométriques, le temps de calcul est légèrement plus long avec cette méthode. De plus, un grand nombre des mots considérés ne sont pas recouvrants et sont donc en fait traités avec simple approximation de Poisson (plus rapide). On observe néanmoins une augmentation linéaire avec ℓ du temps de calcul pour l'approche géométrique Poisson. Comme dans le cas exact, le temps de calcul des grandes déviations croit avec le nombre de mots considérés. Comme cela à été dit plus haut, les grandes déviations précises requièrent environ trois fois plus de temps de calcul et finissent même, pour $h \geqslant 6$ par dépasser le temps de calcul de la méthode exacte. Il faut néanmoins remarquer que si ce derniers temps croit linéairement avec ℓ, il n'en est rien pour les grandes déviations qui restent donc concurrentielles dès lors que des séquences assez longues (ex : $\ell \geqslant 10^5$) sont considérées.

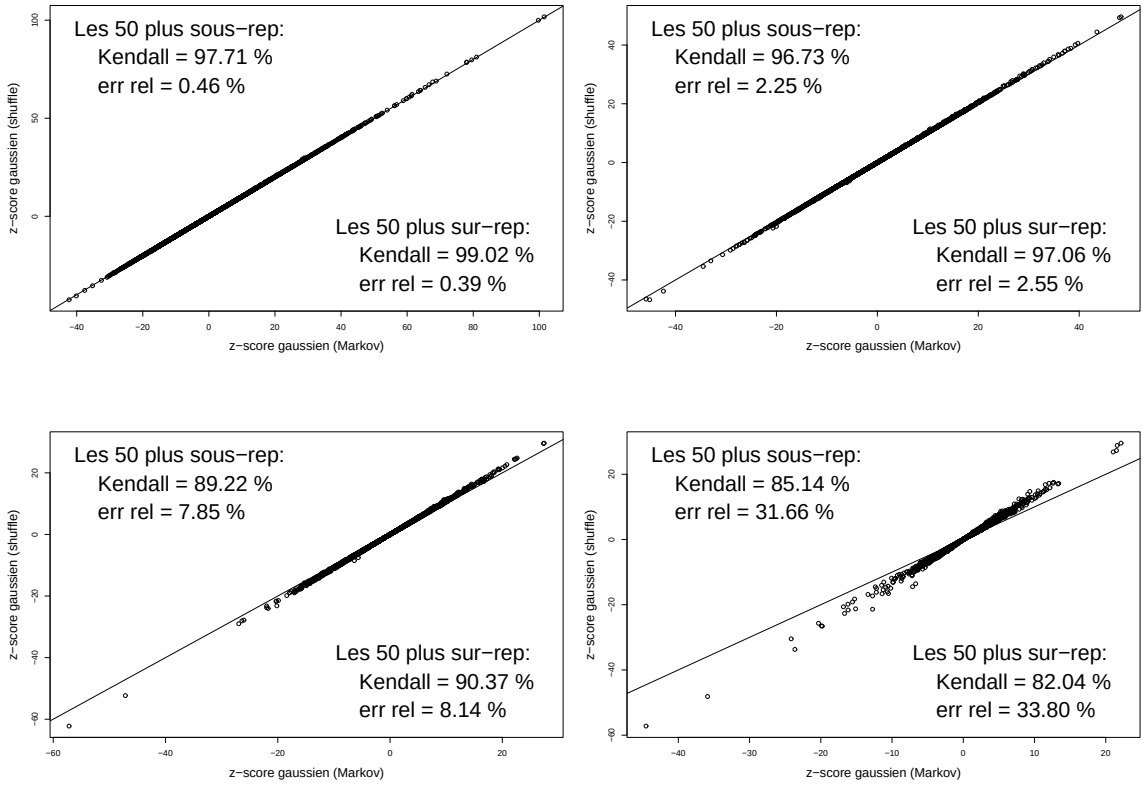


FIG. 5.22 – Z-scores de tous les 4 096 mots de longueur $h = 6$ sur le génome complet de *Escherichia coli* K12 ($\ell = 4\,639\,675$). Un modèle Markvovien est utilisé en abscisse et le modèle shuffle correspondant est utilisé en ordonnée. De gauche à droite et de haut en bas sont successivement considérés $m = 1$, 2, 3 et 4. Sur chaque graphe, l'erreur relative moyenne ainsi que le tau de Kendall sont calculés pour les 50 mots les plus sur- (resp. sous-) représentés.

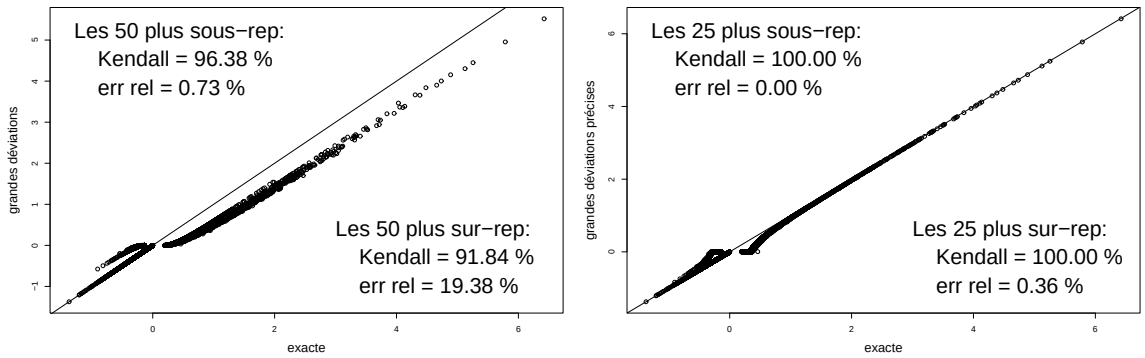


FIG. 5.23 – Statistiques de motifs de tous les 16 384 mots de longueur $h = 7$ sur le génome complet de HIV type 1 ($\ell = 9\,718$) en utilisant un modèle markovien d'ordre $m = 1$. La méthode de référence (exacte, 44.13s de temps-calcul) est comparée (sur le graphe de gauche) aux grandes déviations (27.13s de temps-calcul) et (sur le graphe de droite) aux grandes déviations précises (101.13s de temps-calcul). Sur chaque graphe, l'erreur relative moyenne ainsi le tau de Kendall sont calculés pour les mots les plus sur- (resp. sous-) représentés.

5.9.2 Comparaison Markov *vs* ***shuffle***

Dans le cadre des approximations gaussiennes, nous avons introduit la modélisation *shuffle* des séquences qui consiste à mélanger une séquence observée en conservant les comptages de tous les mots de longueur $m+1$ pour un modèle d'ordre m. Un tel modèle est évidemment proche du modèle markovien d'ordre m dont les paramètres ont été estimés sur la séquence observée qui conserve également les comptages de mots de longueur $m+1$ mais seulement en *en moyenne*.

Étant capables de calculer les z-scores de comptages dans ces deux modèles il est naturel de les comparer afin de mieux comprendre les conséquences de leurs similitudes en ce qui concerne les statistiques de motifs. La figure 5.22 effectue cette comparaison pour différentes valeurs de m. Si la ressemblance est très forte (tant pour les valeurs que pour les rangs) lorsque m est petit ($m = 1$ et $m = 2$), un biais apparaît clairement lorsque m augmente : les z-scores markoviens sont plus faibles que les z-scores *shuffle*. Cela est lié au fait que le modèle *shuffle* a pour support un espace d'états plus grand que le modèle Markov (le premier est limité aux séquences dont les fréquences des mots de longueur $m + 1$ tandis que le second couvre l'ensemble des séquences possibles de longueur ℓ) ce qui donne une plus grande variance dans le cas Markov. Évidemment, plus les séquences considérées seront courtes et plus ce phénomène sera sensible.

5.9.3 Grandes déviations précises

Dans la section 5.8 on a introduit deux approximations de type grandes déviations. Si la première s'appuie simplement sur le théorème de Cramer-Chernov, la seconde propose un résultat plus précis à l'aide de celui de Bahadur-Rao. On a vu que la seconde demandait environ trois fois plus de temps que la première. La question est de savoir si le gain en précision justifie-t-il ce surcoût ?

n	0	5	10	15	20	25	30	35	40
gaussienne	33	30	28	28	28	29	30	32	34
binomiale	5.0	5.3	5.6	5.8	5.9	6.2	6.3	6.5	6.6
Poisson composée	3.4	3.7	3.9	4.0	4.1	4.2	4.3	4.4	4.5
grandes déviations	7e-4	2.8	4.9	8.3	11.2	11.7	11.0	9.6	8.5
g. d. précises	7e-4	0.05	0.2	2.2	5.7	4.8	3.9	2.2	0.4
exceptionnalité moy.	-66.9	-58.4	-52.4	-47.4	-43.1	-39.2	-35.8	-32.6	-29.7

TAB. 5.4 – Erreur relative moyenne (exprimée en %) pour la statistique (sous le modèle markovien d'ordre $m = 1$ estimé sur le génome du virus HIV type 1) de tous les 256 mots de taille $h = 3$ en supposant qu'ils apparaissent n fois dans une séquence de longueur $\ell = 9\,718$. La méthode exacte est utilisée comme référence. A titre indicatif, la dernière ligne donne comme "exceptionnalité moyenne" la valeur moyenne sur tout les mots de longueur 3 de la statistique $S(w, n)$.

La figure 5.9.3 compare les résultats des deux approches à la référence (calculs exacts). Avant d'aller plus avant, il convient de remarquer que l'on a considéré ici une courte séquence ($\ell = 9\,718$) afin de faciliter les calculs exacts. Une telle situation ne favorise cependant pas les grandes déviations car les événements observés ne se trouvent pas très loin dans les queues des distributions (qui constituent le cadre naturel pour cette théorie) et les performances numériques présentées ici ne sont pas représentatives des performances réelles des grandes déviations dans les extrêmes.

Sur le graphe de gauche (grandes déviations simples), on observe un biais systématique (pour les mots sur- et sous-représentés) ainsi qu'un groupe de mots sous-représentés en parfaite correspondance avec la référence. Ce groupe est en fait composé des mots de fréquences nulles pour lesquels on sait que l'approximation des grandes déviations est par construction extrêmement proche du résultat exact (voir remarque 184). En passant au graphe de droite, on peut constater la précision de la correction apportée par le théorème de Bahadur-Rao. Dès que l'on se place dans les queues de distribution (à gauche ou à droite) on obtient un accord de rang parfait et les valeurs elles-mêmes sont extrêmement proches (voir égales pour les mots sur-représentés). Lorsque l'on se rapproche du centre de la distribution, la qualité de l'approximation se dégrade progressivement (mais moins vite que sur le graphe de gauche). Il semble donc y avoir un intérêt considérable à utiliser les approximations précises plutôt que les simples.

5.9.4 Cas extrêmes

Dans cette partie, nous proposons d'examiner le comportement dans les extrêmes sur une séquence courte (sur laquelle les calculs exacts sont rapides) en considérant des comptages observés n arbitrairement faibles ou élevés (selon les cas).

On considère d'abord les mots sous-représentés avec les tables 5.4 (pour l'erreur relative) et 5.5 (pour le tau de Kendall). On fixe $h = 3$ de sorte que le nombre attendu de chacun des mots de longueur h dans la séquence HIV type 1 est de l'ordre de grandeur de 150. Ainsi, les valeurs de n indiquées dans ces tables placent les événements correspondants dans l'extrême gauche des distributions (mots sous-représentés). C'est d'ailleurs bien ce que l'on observe avec la dernière ligne des tableaux qui donne la valeur moyenne des statistiques de mots ; plus n est faible, plus la valeur absolue de cette moyenne est

n	0	5	10	15	20	25	30	35	40
gaussienne	95.1	95.3	95.6	96.0	96.2	96.4	96.4	96.7	96.8
binomiale	94.4	94.6	94.9	94.9	95.2	95.4	95.7	96.1	96.2
Poisson composée	99.2	99.3	99.2	99.4	99.2	99.4	99.3	99.2	99.5
grandes déviations	100	100	100	100	100	100	100	100	100
g. d. précises	100	100	100	100	100	100	100	100	100
exceptionnalité moy.	-66.9	-58.4	-52.4	-47.4	-43.1	-39.2	-35.8	-32.6	-29.7

TAB. 5.5 – Tau de Kendall moyen (exprimée en %) pour la statistique (sous le modèle markovien d'ordre $m = 1$ estimé sur le génome du virus HIV type 1) de tous les 256 mots de taille $h = 3$ en supposant qu'ils apparaissent n fois dans une séquence de longueur $\ell = 9\,718$. La méthode exacte est utilisée comme référence. A titre indicatif, la dernière ligne donne comme "exceptionnalité moyenne" la valeur moyenne sur tout les mots de longueur 3 de la statistique $S(w, n)$.

n	20	40	60	80	100	120	140	160	180
gaussienne	93	183	267	343	415	484	551	615	678
binomiale	0.9	1.6	2.1	2.5	2.9	3.2	3.5	3.7	4.0
Poisson composée	0.6	1.2	1.6	1.9	2.2	2.5	2.7	3.0	3.2
grandes déviations	34	10.2	4.8	3.0	2.1	1.6	1.3	1.1	0.9
g. d. précises	10.5	0.7	0.02	5e-3	2e-3	1e-3	8e-4	9e-4	9e-4
exceptionnalité moy.	5.8	19.3	37.2	58.0	81.1	106.0	132.5	160.3	189.2

TAB. 5.6 – Erreur relative moyenne (exprimée en %) pour la statistique (sous le modèle markovien d'ordre $m = 1$ estimé sur le génome du virus HIV type 1) de tous les 1 024 mots de taille $h = 5$ en supposant qu'ils apparaissent n fois dans une séquence de longueur $\ell = 9\,718$. La méthode exacte est utilisée comme référence. A titre indicatif, la dernière ligne donne comme "exceptionnalité moyenne" la valeur moyenne sur tout les mots de longueur 5 de la statistique $S(w, n)$.

grande.

En terme d'erreur relative (table 5.4) les approximations gaussiennes sont les plus mauvaises (environ 30%). Ce résultat n'a rien d'une surprise dans la mesure où ces approximations bien adaptées au centre de la distribution dont on s'est ici écarté à dessein. Les approximations binomiale et de Poisson composée se placent respectivement autour de 6% et 4%. Pour les grandes déviations, on observe une tendance à la baisse lorsque l'on avance dans la queue de la distribution, ce qui n'est pas une surprise. La correction des grandes déviations précises améliorent sensiblement le résultat.

En ce qui concerne les rangs (voir table 5.5), les approximations gaussiennes et binomiales sont a peu près à égalité (95% d'accord de rang) tandis que les approximations de Poisson composée se rapprochent de la perfection (plus de 99%) qui est atteinte par les grandes déviations (100% avec ou sans la correction).

Pour les mots sur-représentés (tables 5.6 et 5.7), on fixe $h = 5$ de sorte que le nombre attendu de chacun des mots de longueur h dans la séquence est de l'ordre de grandeur de 10. Les valeurs de n indiquées dans ces tables placent ainsi les événements correspondants dans l'extrême droite des distributions (mots sur-représentés).

Les résultats observés sont cohérents avec le cas des mots sous-représentés : la qualité

n	20	40	60	80	100	120	140	160	180
gaussienne	99.8	99.3	98.7	98.1	97.6	97.0	96.5	95.9	95.4
binomiale	99.5	98.9	98.1	97.5	97.0	96.4	95.8	95.2	94.7
Poisson composée	100	99.9	99.9	99.8	99.8	99.7	99.7	99.6	99.5
grandes déviations	100	100	100	100	100	100	100	100	100
g. d. précises	99.9	100	100	100	100	100	100	100	100
exceptionnalité moy.	5.8	19.3	37.2	58.0	81.1	106.0	132.5	160.3	189.2

TAB. 5.7 – Tau de Kendall moyen (exprimée en %) pour la statistique (sous le modèle markovien d'ordre $m = 1$ estimé sur le génome du virus HIV type 1) de tous les 1 024 mots de taille $h = 5$ en supposant qu'ils apparaissent n fois dans une séquence de longueur $\ell = 9\,718$. La méthode exacte est utilisée comme référence. A titre indicatif, la dernière ligne donne comme "exceptionnalité moyenne" la valeur moyenne sur tout les mots de longueur 5 de la statistique $S(w, n)$.

des approximations gaussiennes décroît lorsque la statistique moyenne augmente (avec des valeurs catastrophiques pour l'erreur relative qui s'expliquent également par le fait qu'on a considéré des mots dont le nombre attendu est faible) et les grandes déviations précises qui surclassent toutes les autres méthodes dès que l'on entre dans la queue de la distribution. Au niveau des rangs, comme dans le cas sous-représenté, approximation gaussienne et binomiale font jeu égal tandis que les grandes déviations donne un résultat quasi parfait. Les approximations utilisant des lois de Poisson composées donnent également de bons résultats mais il faut noter que leurs qualités se dégradent avec l'exceptionnalité des événements correspondants.

Les grandes déviations précises surclassant toujours les grandes déviations classiques, seules les première seront désormais considérées.

5.9.5 Cas réels

Commençons par les mots de taille $h = 4$ chez le virus HIV type 1 (voir figure 5.24). Les approximations gaussiennes diffèrent sensiblement de la référence, en particulier pour les mots les plus sur-représentés où l'erreur relative est élevée (30%) et l'accord de rang inférieur à 90%. En passant aux approximations binomiales les choses s'améliorent de manière importante en ce qui concerne l'erreur relative, mais l'accord de rang reste à peu près du même ordre de grandeur. On obtient une légère amélioration sur ce front en passant aux approximations de Poisson composées mais c'est clairement avec les grandes déviations que les meilleurs résultats sont obtenus (100% d'accord de rang et erreur relative inférieure au pourcent).

Passons maintenant aux mots de taille $h = 7$ sur la même séquence (5.25). Cette fois-ci, les résultats sont encore plus catastrophiques pour les approximations gaussiennes dont les résultats dans les extrêmes (mot sur-représentés notamment) sont complètement faux [8]. Ce résultat est en partie expliqué par le faible nombre attendu de ces mots mais également par le très mauvais comportement des approximation de type TLC dans les queues de distributions. En passant aux approximations binomiales, on obtient de biens meilleurs résultats et les quelques mots (chevauchants) pour lesquels les calculs manquent

[8] les différentes "traînées" visibles sur le graphe correspondent à des nombres d'observations différents : $n = 1$ pour le plus à gauche, $n = 2$ pour le suivant, etc.

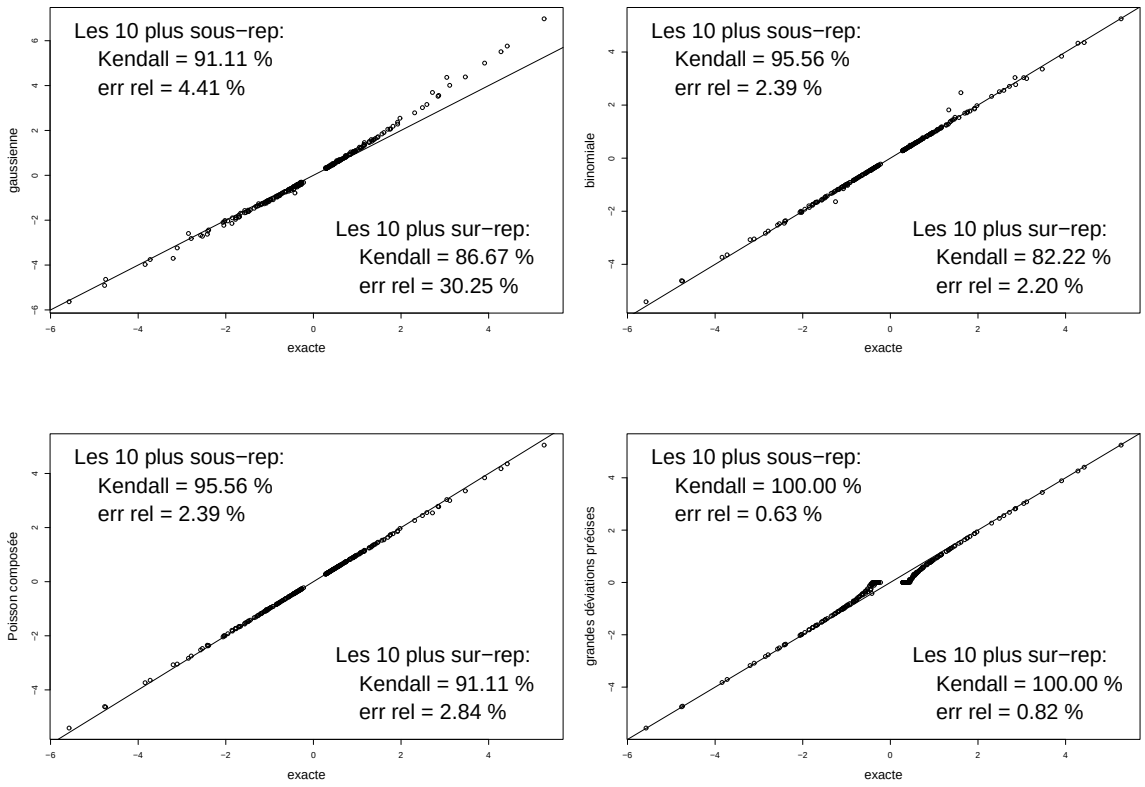


FIG. 5.24 – Statistiques de motifs de tous les 256 mots de longueur $h = 4$ sur le génome complet de HIV type 1 ($\ell = 9\,718$) en utilisant un modèle markovien d'ordre $m = 1$. La méthode de référence (exacte, 17.07 s de temps-calcul) est comparée (de gauche à droite et de haut en bas) à l'approximation gaussienne (temps-calcul négligeable), à l'approximation binomiale (temps-calcul négligeable), à l'approximation de Poisson composée (0.01 s de temps-calcul) et aux grandes déviations précises (1.83 s de temps-calcul). Sur chaque graphe, l'erreur relative moyenne ainsi le tau de Kendall sont calculés pour les 50 mots les plus sur- (resp. sous-) représentés.

de précision sont bien traités par les approximations de Poisson composées. Avec les grandes déviations, on obtient une qualité similaire dans les extrêmes.

Les grandes déviations donnant le résultat le plus fiable à la fois en terme d'erreur relative et de rangs dans les extrêmes, c'est cette méthode que nous retiendrons désormais comme référence lorsque les calculs exacts ne sont pas possibles (par manque de temps).

Considérons maintenant une séquence plus longue avec le génome complet de *Bacillus subtilis* ($4.2Mb$). Pour les mots de longueur $h = 4$ (figure 5.26) et $h = 7$ (figure 5.27) on obtient des résultats similaires. Les approximations gaussiennes sont biaisées dans les extrêmes avec pour conséquence un manque de précision aussi bien en terme d'erreur relative que de rangs. Les approximations binomiales, bien que très simples se comportent assez bien du point de vue des deux critères. Les quelques erreurs rencontrées avec les approximation binomiales (qui concernent les mots chevauchants) sont très bien corrigées par les approximations de Poisson composées dont les performances se rapprochent de la perfection.

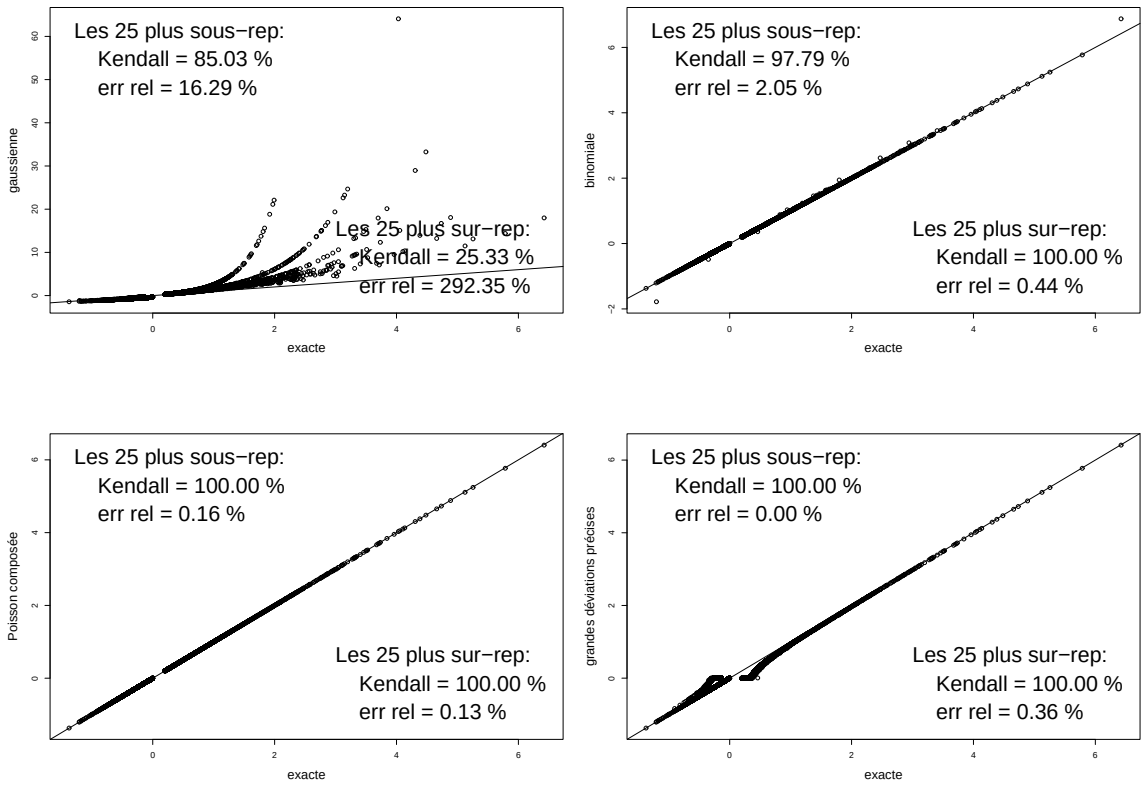


FIG. 5.25 – Statistiques de motifs de tous les 16 384 mots de longueur $h = 7$ sur le génome complet de HIV type 1 ($\ell = 9\,718$) en utilisant un modèle markovien d'ordre $m = 1$. La méthode de référence (exacte, 43.93 s de temps-calcul) est comparée (de gauche à droite et de haut en bas) à l'approximation gaussienne (0.10 s de temps-calcul), à l'approximation binomiale (0.09 s de temps-calcul), à l'approximation de Poisson composée (0.12 s de temps-calcul) et aux grandes déviations précises (101.07s de temps-calcul). Sur chaque graphe, l'erreur relative moyenne ainsi le tau de Kendall sont calculés pour les 50 mots les plus sur- (resp. sous-) représentés.

5.9.6 Conclusions

Les approximations gaussiennes et binomiale sont certainement les plus rapides à mettre en oeuvre. Elles ont dans certains cas des performances similaires (en terme de rang notamment) mais les approximations gaussiennes manquent totalement de précision lorsque le nombre attendu du motif est faible (disons inférieur à 20) et ne sont pas du tout fiables dans les extrêmes. A l'inverse, l'approximation binomiale se révèle d'une performance étonnante qui, alliée à la rapidité de calcul, fait certainement d'elle une très bonne heuristique pour le calcul de statistique de motifs.

Les approximations de Poisson composées sont globalement très fiables mais ont une complexité qui peut se révéler insurmontable dans le cas le plus général (c'est à dire ni Poisson, ni géométrique Poisson). De ce fait, nous déconseillons leur usage dans le cas où le nombre F d'état finaux de la PMC est plus grand que un.

Les grandes déviations demandent un temps de calcul sensiblement plus long (parfois même plus long que celui de l'approche exacte sur de courtes séquences) mais ce temps

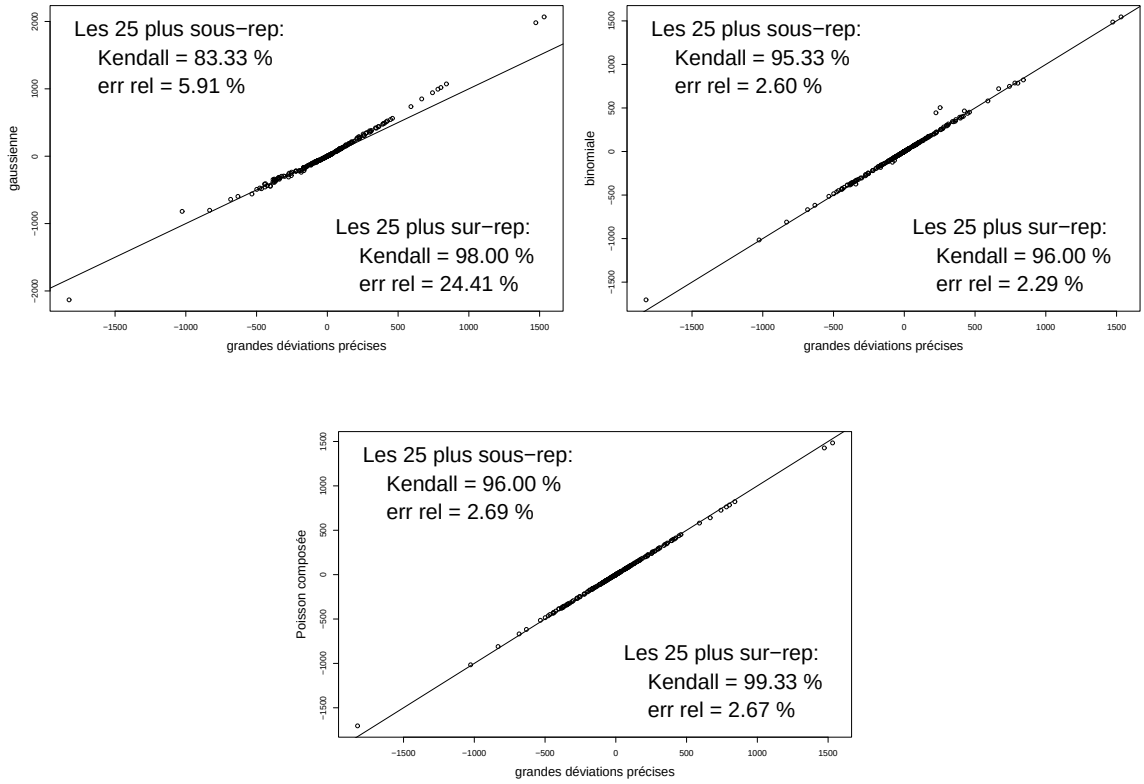


FIG. 5.26 – Statistiques de motifs de tous les 256 mots de longueur $h = 4$ sur le génome complet de *Bacillus subtilis* ($\ell = 4\,214\,630$) en utilisant un modèle markovien d'ordre $m = 1$. La méthode de référence (grandes déviations précises, 2.12 s de temps-calcul) est comparée (de gauche à droite et de haut en bas) à l'approximation gaussienne (0.28 s de temps-calcul), à l'approximation binomiale (0.29 s de temps-calcul) et à l'approximation de Poisson composée (0.59 s de temps-calcul). Sur chaque graphe, l'erreur relative moyenne ainsi le tau de Kendall sont calculés pour les 50 mots les plus sur- (resp. sous-) représentés.

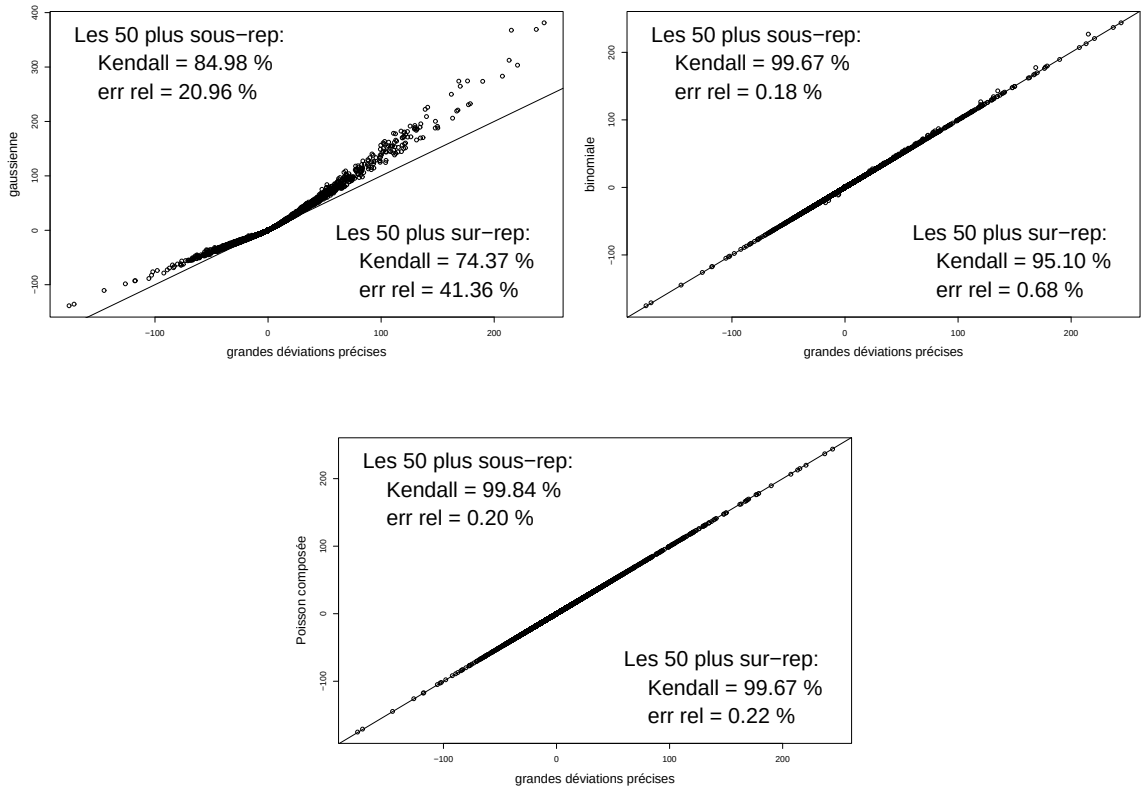


FIG. 5.27 – Statistiques de motifs de tous les 16 384 mots de longueur $h = 7$ sur le génome complet de *Bacillus subtilis* ($\ell = 4\,214\,630$) en utilisant un modèle markovien d'ordre $m = 1$. La méthode de référence (grandes déviations précises, 283.05 s de temps-calcul) est comparée (de gauche à droite et de haut en bas) à l'approximation gaussienne (0.52 s de temps-calcul), à l'approximation binomiale (0.54 s de temps-calcul) et à l'approximation de Poisson composée (0.93 s de temps-calcul). Sur chaque graphe, l'erreur relative moyenne ainsi le tau de Kendall sont calculés pour les 50 mots les plus sur- (resp. sous-) représentés.

ne dépend que du produit kL. Les performances de cette approche sont clairement les meilleures dès lors que l'on s'intéresse aux événements de la queue de distribution (qui sont par essence les événements d'intérêts ce qui fait certainement d'elle la meilleure alternative au calcul exact lorsque celui-ci est trop long à mettre en oeuvre.

5.10 Notes bibliographiques

PMC

Si l'utilisation des automates finis déterministes est connue depuis bien longtemps en *pattern matching*, l'idée de s'en servir pour étudier la distribution des motifs est, elle, beaucoup plus récente. Stefanov et Pakes (1997, 1999) ont les premiers proposé une méthode permettant d'obtenir la fonction génératrice du comptage d'un motif via des résultats sur les familles exponentielles. Cette méthode étudiant pas à pas la formation

des motifs ne fait cependant qu'une utilisation *implicite* des automates. Nicodème et al. (2002) franchissent ensuite un cap important avec une méthode similaire qui identifie cette fois-ci de manière explicite le rôle clef des automates. Peu de temps après, Crochemore et Stefanov (2003) associent à leur tour le processus de formation des motifs à l'automate qui lui correspond.

Dans tous les cas cependant, la notion de PMC, bien qu'implicite, n'est pas mise en avant et les auteurs se contentent de l'utiliser pour obtenir des fonctions génératrices. Notons que, pour cela, il est nécessaire de passer par des calculs formels qui peuvent être lourds (ex : résolution formelle d'un système linéaire d'équations), si bien que l'utilisation de ces méthodes reste finalement assez limitée en pratique.

Nuel (2006b) propose alors d'étendre ces résultats en introduisant la notion de PMC et en l'appliquant à de nombreuses approches statistiques. Ce chapitre reprend un grand nombre des résultats de cet article. Remarquons au passage que, si la notion d'automate non m-ambigus est déjà présente dans Nicodème et al. (2002) (bien qu'elle n'apparaisse alors pas sous ce nom), les auteurs, à la différence de Nuel (2006b), ne donnent aucun algorithme explicite pour sa construction : ils se contentent d'indiquer qu'il suffit de dupliquer les états ambigus jusqu'à résolution du problème.

Calculs exacts

Dans le cadre de l'étude des occurrences de motifs, la technique des FMCI a été introduite par Fu et Koutras (1994) (voir Lou, 1996; Fu et Lou, 2003, pour plus de détails). Bien que cette méthode soit extrêmement puissante et élégante, elle nécessite le calcul de un ou plusieurs termes d'une puissance de la matrice de transition associée. L'approche la plus directe de ce calcul étant en général impossible, Fu (1996) souligne l'importance de considérer ce calcul de manière récursive.

Nuel (2006c) reprend cette idée et propose un algorithme permettant de traiter efficacement les calculs pour une FMCI quelconque. En considérant de plus la structure particulière des FMCI de motifs et son lien avec la notion de PMC, on obtient un algorithme particulièrement efficace pour le calcul exact de la distribution du nombre d'occurrences. Les développements asymptotiques proposés dans ce document sont également généralisés dans cet article.

Kleffe et Borodovsky (1992) sont les premiers à donner les formules exactes pour l'espérance et la variance d'un comptage dans une chaîne de Markov. Nuel (2006d) reprend ces formules en y ajoutant une estimation de l'erreur commise en approximant les puissances de la matrice de transition par la loi stationnaire. Comme cela est proposé dans Nuel (2006b), ces résultats sont ici étendus aux PMC.

Approximation gaussiennes

Le calcul de l'espérance conditionnelle d'un comptage est dû à Cowan (1991), et c'est à Prum et al. (1995) que revient la variance conditionnelle et la convergence vers une loi gaussienne (cas *shuffle*, équivalence dans le cas Markov). Le logiciel RMES implémente ces formules et permet le calcul de z-scores.

Approximation binomiales

Si la paternité de ces approximations est difficile à établir, on peut dire que van Helden et al. (1998) sont les premiers à l'utiliser de manière intensive (en validant par simulation

cette approche). Nuel (2005) propose une implémentation efficace de ces approximations et compare ses performances aux approximations gaussiennes et aux grandes déviations, confirmant leur étonnante qualité. Nuel (2006e) considère le problème de la sensibilité des ces approximations à l'estimation des paramètres et propose l'approche par delta-méthode.

Approximations de Poisson composées

Chrysaphinou et Papastavridis (1988) introduisent d'abord l'approximation par une loi de Poisson géométrique dans le cas d'une séquence indépendante ($m = 0$). Arratia et al. (1990) utilisent les approximations de Chen-Stein pour approcher la loi du comptage d'un motif dans le cas Markov par une loi de Poisson composée. Schbath (1995) étudie ensuite cette idée en détails et, plus récemment, Robin (2002) souligne que la loi de Poisson composée est en fait une simple loi de Poisson géométrique lorsque l'on s'intéresse à un mot non dégénéré. Nuel (2006a) s'intéresse justement de près à cette distribution en proposant une récurrence qui permet d'accélérer le calcul de sa fonction de répartition. Le résultat concernant la fonction de survie d'une loi de Poisson géométrique ainsi que la formule dans le cas général sont originaux. L'extension de la matrice de covariance aux PMC est due à Nuel (2006b) et le résultat donnant les paramètres de la loi de Poisson composée à partir cette matrice revient à Roquain et Schbath (2006).

Grandes déviations

Den Hollender (2000) est un très bon ouvrage d'introduction à la théorie des Grandes Déviations dont nous conseillons la lecture. Il contient notamment le résultat de Cramer-Chernov et ceux concernant le niveau 2 dans le cas markovien. Pour le théorème de Bahadur-Rao, on pourra se reporter à Kontoyiannis et Meyn (2003b,a).

L'utilisation de ces résultats pour l'étude des occurrences de motifs a été étudiée en détail par Nuel (2001, 2004) (pour le niveau 1) et par Pudlo (2004) (pour le niveau 2). L'argument de convexité est également dû à Pudlo.

Comparaisons des méthodes

Reinert et al. (2000, 2005) proposent une revue très complète des différentes approches statistiques au problème des motifs. Ces deux articles ne s'intéressent cependant que de manière marginale à la mise en oeuvre pratique de ces méthodes et discutent leurs performances relatives du seul point de vue théorique. Robin et Schbath (2001) proposent pour leur part une comparaison moins complète mais plus pratique, même si l'on peut regretter que cette comparaison se concentre sur la distance entre les loi exactes et approchées sans s'intéresser aux comportements spécifiques de ces approximations dans les queues de distribution. Nuel (2006d) étend et complète cette étude en considérant un plus large panel d'approximations et en s'intéressant de très près à leurs comportements dans les extrêmes. Le paragraphe 5.9 de cet ouvrage reprend pour l'essentiel la méthodologie et les résultats de cet article.

Troisième partie

Motifs

Chapitre 6

SPA : Simple web tool to assess statistical significance of DNA patterns

Application note de Richard et Nuel publiée en 2003 dans *Nucleic Acids Research*. Il y est présenté une web application (aujourd'hui obsolète) permettant de calculer des statistiques de motifs soit par une approche exacte, soit par les grandes déviations.

Nucleic Acids Research, 2003, Vol. 31, No. 13 ***3679–3681***
DOI: 10.1093/nar/gkg613

SPA: simple web tool to assess statistical significance of DNA patterns

H. Richard and G. Nuel*

Laboratoire Statistique et Genome, CNRS, INRA, Genopole, Université d'Evry Val d'Essone, 523 place des terrasses, 91000 Evry, France

Received February 17, 2003; Revised April 4, 2003; Accepted April 14, 2003

ABSTRACT

Many statistical methods and programs are available to compute the significance of a given DNA pattern in a genome sequence. In this paper, after outlining the mathematical background of this problem, we present SPA (Statistic for PAtterns), an expert system with a simple web interface designed to be applied to two of these methods (large deviation approximations and exact computations using simple recurrences). A few results are presented, leading to a comparison between the two methods and to a simple decision rule in the choice of that to be used. Finally, future developments of SPA are discussed. This tool is available at the following address: http://stat.genopole.cnrs.fr/SPA/.

INTRODUCTION

Since the appearance of large scale genome sequencing projects, automatic extraction of biologically meaningful information has become a key issue in genetic research. One way to assess the functional role of an oligomer would be to evaluate its degree of significance with respect to a random sequence model. Markov chain models of order m, preserving counts of $m + 1$ letter words, and thus modelling neighbouring letter interactions, have been extensively used in the study of biological sequences (1,2). For example, under-represented palindromic words in bacterial genomes have been shown to be closely correlated to restriction enzyme recognition sites (3,4). On the other hand, careful analysis of over-represented words has been helpful in the detection of transcription factor binding sites (5,6) and in the analysis of polyadenylation signals (7,8) in yeast ORF flanking regions.

Efficient methods in estimating the statistic of a pattern in random texts have been developed during the past years, from Gaussian or Poisson approximations (9–11), generating functions (12–14), to simple recurrence formulae (15). Recently large deviation approaches (16) allowed good estimations of extreme statistics. Today, except for a web site for the analysis of ORF upstream regions (17), there is no general and reliable tool to help a biologist in ranking oligonucleotides of interest. Here we present a simple application which allows the Markovian analysis of a set of words with a chosen fixed length. Two methods are carried out to achieve this goal: exact computation based on recurrence formulae and large deviation approximations. To achieve a good trade-off between speed and precision, we developed an expert system which selects the best method for each word. After a brief mathematical introduction and a presentation of the web interface, we will detail the way the software chooses between the two methods.

MATERIAL AND METHODS

Mathematical background

Markov models and probabilities of words. We consider a sequence X over a finite size alphabet $\mathcal{A}$ (for example $\mathcal{A} = \{\mathtt{a},\mathtt{c},\mathtt{g},\mathtt{t}\}$ for DNA sequences) assumed to be generated from an order m Markov model.

For a given word W of length h, we denote as $N(W)$ its number of (possibly overlapping) occurrences in the sequence X. A pattern $\mathcal{W}$ is defined as the finite family of words $\{W_1, \ldots, W_r\}$. Its number of occurrences is simply calculated as:

$$N_{\mathcal{W}} = N(W_1) + \cdots + N(W_r)$$

We now consider an observed sequence x (for example, a DNA sequence) and we say that a pattern $\mathcal{W}$ is over- (respectively under-) represented if:

$$n_{\mathcal{W}} \geq \mathbf{E}[N_{\mathcal{W}}] (\text{resp. } n_{\mathcal{W}} < \mathbf{E}[N_{\mathcal{W}}])$$

where $n_{\mathcal{W}}$ is the number of occurrences of $\mathcal{W}$ in the observed sequence and $\mathbf{E}[N_{\mathcal{W}}]$ is the mathematical expectation of $N_{\mathcal{W}}$.

z-score. For a pattern $\mathcal{W}$, we define its z-score $Z_{\mathcal{W}}$ according to:

$$\mathbf{P}(\mathcal{N}(0,1) \geq Z_{\mathcal{W}}) = \mathbf{P}(N_{\mathcal{W}} \geq n_{\mathcal{W}})$$

if $\mathcal{W}$ is over-represented (i.e. is observed more than expected). Otherwise, according to:

$$\mathbf{P}(\mathcal{N}(0,1) \geq Z_{\mathcal{W}}) = \mathbf{P}(N_{\mathcal{W}} < n_{\mathcal{W}})$$

if $\mathcal{W}$ is under-represented.

*To whom correspondence should be addressed. Tel: +33 1 60 87 88 01; Fax: +33 1 60 87 38 09; Email: nuel@genopole.cnrs.fr

Published by Oxford University Press 2003

Table 1. Complete description of patterns for different string definitions

pattern definition	list of words	length	size
{[at]t[cg]}	{atc,atg,ttc,ttg}	3	4
{acgtaa}{a[cg]taa}	{acgtaa,actaa,agtaa}	6	3
{g.tggtgg}	{gatggtgg,gctggtgg,ggtggtgg,gttggtgg}	8	4

In both cases, $\mathcal{N}(0, 1)$ denotes a Gaussian distribution, and **P** is the probability symbol. Using this convention, Z_W will be positive (respectively negative) for over- (respectively under-) represented patterns. The use of this statistic enables the representation of the p-value in a Gaussian scale, albeit this choice does not mean we use a Gaussian approximation for the computation of that p-value. For example, we can say that a pattern with Z_W larger than 2.96 (the Gaussian quantile for a probability of 0.05), is *significantly* over-represented at a 5% level.

Robin and Daudin (15) have proposed a method based on generative series and simple recurrences to compute the exact value of Z_W. Nuel (16) also proposed the use of a large deviation approach to get an asymptotic approximation of the same z-score. In the following, we will denote by Z_W^{SR} the z-scores computed with the simple recurrence method and by Z_W^{LD} those obtained with the large deviation.

Web interface

We have developed a web interface called SPA (Statistics for PAtterns) whose purpose is to provide a simple access to the patterns z-score computed through these two methods [simple recurrence (SR) and large deviation (LD)].

A SPA request consists of: (i) a set of sequences (in FASTA format); (ii) a list of patterns or a fixed word length when all words are selected; and (iii) the order of the Markov model. The parameter of this model is estimated on the set of sequences using the maximum likelihood. Pattern counts are also obtained on this set of sequences. Explicit selection of the method is possible, but nevertheless it is recommended to leave this parameter on auto since it selects the most suitable computation for each word.

The size n of the chosen sequence (or the size of the longest sequence) is directly related to the efficiency of the different methods. As the LD method proposes an asymptotic approximation, its results will obviously be more reliable as n grows. On the other side, time complexity of the SR approach is in $O(n^2)$, so it is quite clear that high values for n are not recommended here. In practice, SR method will be limited to sequences smaller than 100 kb ($n = 10^5$).

Each pattern must be described with a braced string using the following conventions: . means any letter and [ ] can be used to describe several possible letters for a single position. Union of patterns can be done by concatenation of their string definitions. Table 1 illustrates some examples of syntax and the resulting patterns. The length h of a pattern will be defined as the length of the longest word in it. The size r of a pattern will be the number of words it contains. For the LD method, h will be a critical parameter as memory and time complexity are in $O(k^h)$ where k is the size of the alphabet (usually four for DNA sequences). In practice, LD will be limited to patterns of size smaller than 10. For the SR method, there is no such limitations concerning h, but as time complexity grows with r^2, very large patterns are generally to be avoided.

Table 2. z-scores computed through the large deviation for restriction sites and *chi* patterns with various *Markov* model orders

restriction sites (and their inverses)				
organism pattern	$M0$	$M1$	$M2$	$M3$
B. subtilis {cgcg}	-9.5	-39.5	-20.4	–
B. subtilis {ggatcc}	-21.6	-23.5	-29.2	-23.1
E. coli {ccgcgg}	-18.1	-30.0	-38.2	-29.5
E. coli {ggtctc}{gagacc}	-18.1	-30.0	-38.2	-29.5
chi (and their inverses)				
B subtilis {ccgct}{agcgg}	$+72.2$	$+59.0$	$+24.7$	$+5.1$
E. coli {gctggtgg}{ccaccagc}	$+46.0$	$+47.0$	$+28.2$	$+15.5$
H. influenzae {g.tggtgg}{ccacca.c}	$+21.0$	$+20.3$	$+9.7$	$+6.0$

As said in the introduction, a Markov model of order m preserves the counts of length $m + 1$ words. Therefore, m will be limited to $h - 2$, due to the fact that the estimation is done on sequences. Obviously, the quality of the estimation depends on the total length l of the provided sequences. Practically, l should be a $O(4^m)$ kb to give the user a satisfying estimate. Furthermore, memory and computational complexity inherent to *Markov* models limit the maximal order to four with SR, and six and LD.

RESULTS AND DISCUSSION

Interpreting results

Here, we will present two short examples, the first one showing an academic test on biologically known patterns and the second one detailing a sample output.

Table 2 shows Z_W^{LD} statistics for patterns known to be restriction enzyme recognition sites and *chi* patterns in three bacterial genomes. The statistic is given for orders from 0 to 3. As expected, all of these patterns have very high significance; restriction sites are largely under-represented (p-value close to 10^{-341} for example with a z-score of -39.5) and *chi* patterns are largely over-represented (p-value around 10^{-1134} with a z-score of $+72.2$). We can also see that this z-score has a great dependence with the order of the chosen model (which is not a surprise). Therefore, it is quite clear that a user who wants to rank a set of patterns should be aware of this and perform at least two different computations for two different orders.

In Figure 1, we can see a sample of ranked output. For more convenience, two tables, listing respectively the most over- and under-represented words are joined in the output. For each pattern is indicated its number of occurrences, the method used (LD or SR), the computed z-score and the rank. As we observe here highly significant patterns, the large deviation provide the best results and that is why there is no use of SR methods in that output. Many patterns among the over-represented ones are closely related to gctggtgg the *chi* of *Escherichia coli* (gctggt, ctggtg, ggtggt) or to its inverse ccaccagc (ccagca, ccagcg). For under-represented patterns, we can see the restriction site ccgccg well ranked (rank 15) as well as many palindromes (ggcgcc, gcatgc, ...). Finally, we can observe that there is a great difference between that statistical ranking and the frequencies ranking. This can show the value of performing this kind of statistical analysis rather than simple frequency comparison.

most over-represented patterns					most under-represented patterns				
pattern	occ	method	z-score	rank	pattern	occ	method	z-score	rank
(cagcag)	93	LD	+9.5442	1	(ggcgcc)	3	LD	-8.4266	1
(ggcgat)	116	LD	+9.4530	2	(gcatgc)	10	LD	-7.8237	2
(cgacag)	101	LD	+9.1654	3	(cggccg)	5	LD	-7.2316	3
(tgctgg)	108	LD	+9.1289	4	(cacgtg)	1	LD	-6.8251	4
(ctggaa)	82	LD	+8.8292	5	(cttgga)	1	LD	-5.8768	5
(gccagc)	103	LD	+8.6101	6	(gccgcc)	13	LD	-5.7656	6
(gctgaa)	87	LD	+8.2518	7	(cgcatg)	17	LD	-5.6996	7
(gctggt)	86	LD	+8.2514	8	(ttggaa)	5	LD	-5.6643	8
(ttccag)	70	LD	+8.2217	9	(tgctag)	0	LD	-5.5425	9
(ccagca)	82	LD	+8.1862	10	(catgcg)	18	LD	-5.4966	10
(ctggtg)	78	LD	+7.9393	11	(attgga)	4	LD	-5.4918	11
(ccagcg)	91	LD	+7.9265	12	(gggacc)	0	LD	-5.4492	12
(atccag)	64	LD	+7.8878	13	(cttggg)	3	LD	-5.4259	13
(ctggga)	88	LD	+7.7296	14	(gggccc)	3	LD	-5.3794	14
(ggtggt)	72	LD	+7.5941	15	(ccgcgg)	13	LD	-5.1858	15
(gctgat)	87	LD	+7.5327	16	(cgcttg)	14	LD	+7.1675	16
(atcgcc)	89	LD	+7.5161	17	(ctagca)	0	LD	+7.1105	17
(cagccg)	92	LD	+7.5027	18	(ctagcg)	1	LD	+7.0681	18

Figure 1. Sample output from SPA for the following request: all words of length $h=6$ in the first 100 kb of the complete genome of *E.coli* with a Markov model of order 1.

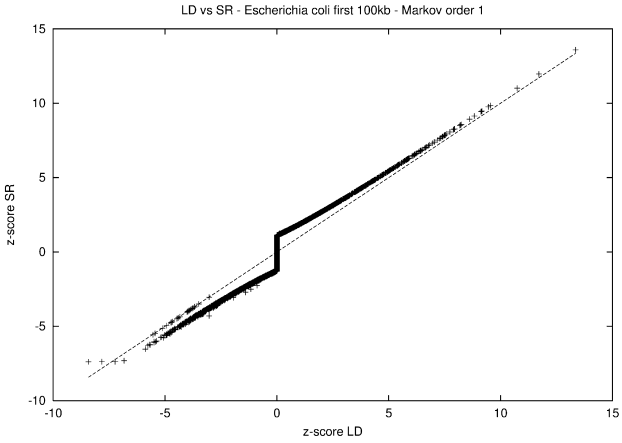


Figure 2. Comparison of the z-scores obtained through large deviation (LD) and simple recurrence (SR) methods for all words of length $h=6$ on the first 100 kb of *E.coli* genome for Markov model of order $m=1$.

Methods comparison

Here we want to take into account the limitations highlighted in the section on web interface, as well as the accuracy of the different approaches, in order to help the users to make a choice between the two methods.

As we need to compute our results with both the methods, we have chosen to work with a not too long sequence consisting of the first 100 kb of *E.coli.* All the 4096 words of length six are studied and the results are plotted in Figure 2.

Overall, the two methods appear to give similar results. As expected, the large deviation fail to give correct answers for words with a high *p*-value (above 1% which means $|z\text{-}score| \leq 2.3$). We also know that the LD z-score gets more accurate as the $|z\text{-}score|$ grows and that is exactly what we observe. When the z-score >5, SR and LD do equally well, and the ranking is preserved; it is a reliable region for LD. For negative *z*-scores, very small probability sums induce computational errors with the SR method; this explains the four words present on the extreme left of Figure 2. As we prefer ranking preservation rather than exact results, we decided to cut the reliable region for SR at −2.5.

For a given pattern on a sequence whose length n is in SR requirement ($n \leq 100$ kb, we propose to do the following: first we perform an evaluation using LD and if the z-score is in the range [−2.5; +5] another computation using SR is then performed.

CONCLUSION

The value of using statistical significance rather than frequencies to rank patterns has been shown many times. The usual problem is that there are many different methods and programs to compute those statistics. With our user friendly web interface, we provide a simple way to access two of these methods (large deviation approximations and exact computations using simple recurrence) and we also facilitate the choice with an empirical decision rule.

In the near future, we plan to integrate more methods in SPA (Gaussian and compound Poisson approximations, generative series) to improve both speed and accuracy of the tool. We also plan to use exact computation of the expectation value and the variance of N_W provided by RMES (10) to make an easier choice between the different methods.

REFERENCES

1. Waterman,M.S. (1995) *Introduction to Computational Biology.* Chapman and Hall, Cambridge University Press, Cambridge, UK.
2. Ewens,W.J. and Grant,G.R. (2001) *Statistical Methods in Bioinformatics*. Springer Verlag, New York.
3. Gelfand,M.S. and Koonin,E.V. (1997) Avoidance of palindromic words in bacterial and archaeal genomes: a close connection with restriction enzymes [published erratum appears in *Nucleic Acids Res.* 1997 **25**, 5135–5136]. *Nucleic Acids Res.*, **25**, 2430–2439.
4. Karlin,S., Burge,C. and Campbell,A.M. (1992) Statistical analysis of counts and distributions of restriction sites in dna sequences. *Nucleic Acids Res.*, **20**, 1363–1370.
5. Hampson,S., Kibler,D. and Baldi,P. (2002) Distribution patterns of over-represented k-mers in non-coding yeast DNA. *Bioinformatics*, **18**, 513–528.
6. Brazma,A., Jonassen,I., Vilo,J. and Ukkonen,E. (1998) Predicting gene regulatory elements *in silico* on a genomic scale. *Genome Res.*, **8**, 1202–1215.
7. van Helden,J., del Olmo,M. and Perez-Ortin,J.E. (2000) Statistical analysis of yeast genomic downstream sequences reveals putative polyadenylation signals. *Nucleic Acids Res.*, **28**, 1000–1010.
8. Beaudoing,E., Freier,S., Wyatt,J.R., Claverie,J.-M. and Gautheret,D. (2000) Patterns of variant polyadenylation signal usage in human genes. *Genome Res.*, **10**, 1001–1010.
9. Prum,B., Rodolphe,F. and de Turckheim,E. (1995) Finding words with unexpected frequencies in DNA sequences. *J. R. Statist. Soc. B.*, **11**, 190–192.
10. Schbath,S. (1997) An efficient statistic to detect over- and under-represented words in DNA sequences. *J. Comp. Biol.*, **4**, 189–192.
11. Reinert,G. and Schbath,S. (1998) Compound Poisson and Poisson process approximations for occurrences of multiple words in Markov chains. *J. Comp. Biol.*, **5**, 223–254.
12. Nicodème,P. (2001) Fast approximate motif statistics. *J. Comp. Biol.*, **8**, 235–248.
13. Nicodème,P., Salvy,B. and Flajolet,P. (2002) Motif statistics. *Theor. Comp. Sci.*, **18**, 161–171.
14. Régnier,M. (2000) A unified approach to word occurrence probabilities. *Discr. Appl. Math.*, **104**, 259–280.
15. Robin,S. and Daudin,J.J. (1999) Exact distribution of word occurrences in a random sequence of letters. *J. App. Prob.*, **36**, 179–193.
16. Nuel,G. (2001) *Grandes déviations et chaine de Markov pour l'étude des occurrences de mots dans les séquences biologiques*. PhD thesis, Université d'Evry Val d'Essonne, France.
17. Van Helden,J., André,B. and Collado-Vides,J. (2000) A web site for the computational analysis of yeast regulatory sequences. *Yeast*, **16**, 177–187.

Chapitre 7

LD-SPatt : Large Deviations Statistics for Patterns on Markov chains

Article de Nuel publié en 2004 dans *Journal of Computational Biology*. Cet article présente de manière succincte les résultats de grandes déviations (niveau 1 et 2) obtenus au cours de ma thèse, ainsi que l'application *ldspatt* dans laquelle sont implémentés les calculs correspondants. Les performances des grandes déviations sont comparées à celles des approximations gaussiennes et de Poisson composées (obtenues à l'aide du logiciel R'MES).

JOURNAL OF COMPUTATIONAL BIOLOGY
Volume 11, Number 6, 2004

Pp. 1023–1033

LD-SPatt: Large Deviations Statistics for Patterns on Markov Chains

G. NUEL

ABSTRACT

Statistics on Markov chains are widely used for the study of patterns in biological sequences. Statistics on these models can be done through several approaches. Central limit theorem (CLT) producing Gaussian approximations are one of the most popular ones. Unfortunately, in order to find a pattern of interest, these methods have to deal with tail distribution events where CLT is especially bad. In this paper, we propose a new approach based on the large deviations theory to assess pattern statistics. We first recall theoretical results for empiric mean (level 1) as well as empiric distribution (level 2) large deviations on Markov chains. Then, we present the applications of these results focusing on numerical issues. LD-SPatt is the name of GPL software implementing these algorithms. We compare this approach to several existing ones in terms of complexity and reliability and show that the large deviations are more reliable than the Gaussian approximations in absolute values as well as in terms of ranking and are at least as reliable as compound Poisson approximations. We then finally discuss some further possible improvements and applications of this new method.

Key words: Cramer–Chernov, Arnoldi, Perron–Frobénius, DNA, occurrence of words, frequencies.

1. INTRODUCTION

In the framework of biological sequence analysis (DNA, proteins), differences in letter composition and the different patterns overlapping structures have as a consequence that pattern frequencies cannot be used for reliable comparisons. To avoid this issue, a classical solution consists in the using of Markov chains models for the sequences in order to detect statistically significant patterns. Such approaches have been successfully used in many biological fields allowing confirmation of existing results as well as production of new ones: Chi and restriction sites detection on bacterial genomes (El Karoui *et al.*, 1999; Gelfand and Koonin, 1997; Karlin *et al.*, 1992), transcription factor binding sites (Hampson *et al.*, 2002; Brazma *et al.*, 1998) or analysis of polyadenylation signals (van Helden *et al.*, 2000; Beaudoing *et al.*, 2000).

The problem of assessing statistical significance for patterns on Markov chains has been extensively studied in literature. A first approach consists in the computation of the exact distribution of patterns using generative series with several different methods: combinatory (Pevzner *et al.*, 1989; Régnier and Szpankowski, 1998; Régnier, 2000), finite state automaton (Atteson, 1998; Nicodème, 2001; Nicodème *et al.*, 2002), and simple recurrence (Robin and Daudin, 1999). All these methods are quite efficient for

Laboratoire Statistique et Génome, Tour Evry 2, 523 place des terasses, 91034 Evry, France.

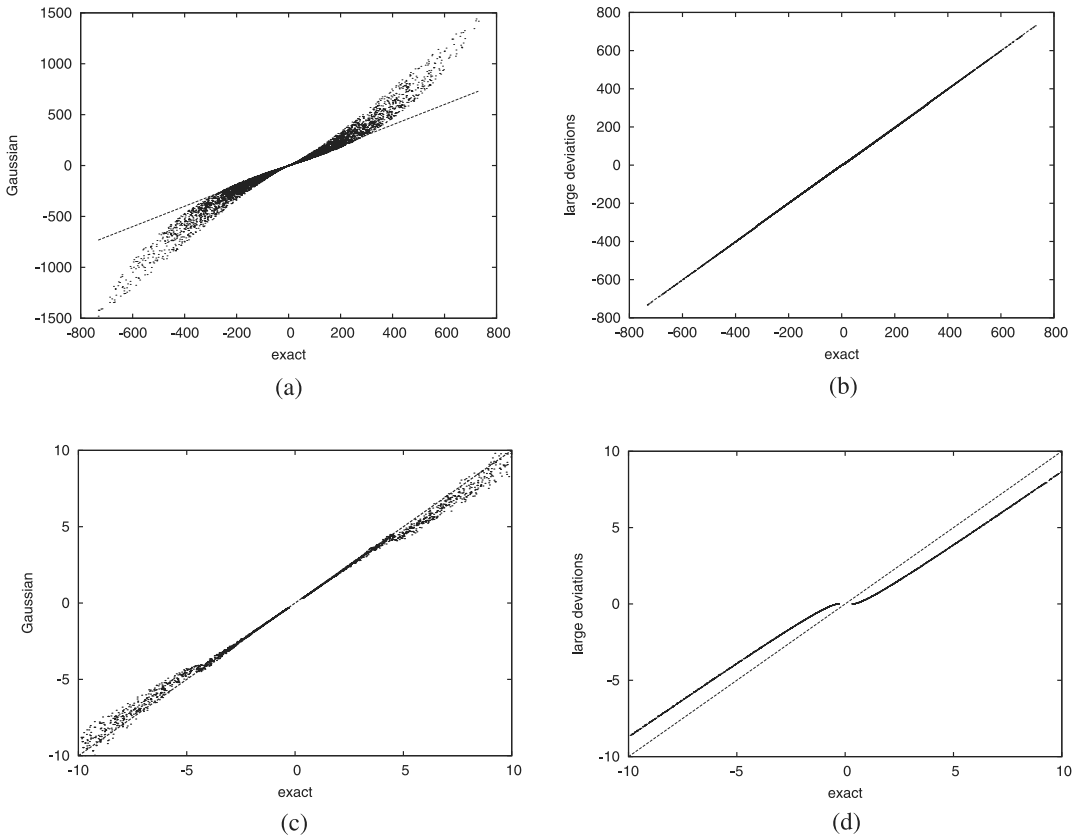


FIG. 1. Coin-tossing example. Comparison of Gaussian and large deviations for a simple coin-tossing problem. Given $n = 1{,}000$ for the number of coins tossed, p the probability to obtain "head," and a the observed proportion of "head," compute $S = -\log_{10} \mathbb{P}(a)$ if $a > p$ (we observe more heads than expected) or $S = \log_{10} \mathbb{P}(a)$ in the opposite case. Gaussian statistics are computed through the Central Limit Theorem and large deviations statistics through Cramer–Chernov. Ten thousand uniform random values for p and a are taken in order to produce statistics. Exact statistics are plotted against Gaussian statistics on (**a**) and (**c**) or against large deviations statistics on (**b**) and (**d**).

short sequences but, because of numerical complexity, they are often used to compute only expectation and variance of a given count (and thus use, in fact, Gaussian approximations). When the length of the sequences increases, an alternative is to use asymptotic compound Poisson approximations when a small number of occurrences is expected (see Godbole, 1991; Geske *et al.*, 1995; Reinert and Schbath, 1998) or Gaussian approximations (Cowan, 1991; Kleffe and Borodovsky, 1992; Prum *et al.*, 1995; Schbath, 1997) in the more general case.

Based on limit central convergence, the Gaussian approximations are known to have poor reliability for tail distribution events, and unfortunately, these events are especially interesting when tracking patterns involved in biological processes. Figure 1 provides a simple coin-tossing example illustrating this problem.

Figure 1a shows the Gaussian approximations failure for tail distribution events, whereas large deviations are very reliable (Fig. 1b). Figures 1c and 1d give a closer look on central distribution events where large deviations fail, while Gaussian approximations are much more reliable. Nevertheless, one should also note that even for such events, large deviations ranking is still reliable.

That is why the use of large deviations (Deuschel, 1957; Dembo and Zeitouni, 1998; or Den Hollender, 2000) could be a good way to improve the reliability of asymptotic approximations as soon as we focus on extreme events. In this paper, we propose to use this theory in order to compute pattern statistics with the highest reliability for the most significant patterns.

2. MATERIAL AND METHODS

2.1. Notations

We first want to introduce the notation that will be used from now on. Let's denote by $X = X_1 \dots X_n$ with $X_i \in \mathcal{A}$ a Markov chain of order m on the finite alphabet $\mathcal{A}$ of size k (for example, $\mathcal{A} = \{\mathtt{a}, \mathtt{c}, \mathtt{g}, \mathtt{t}\}$ for the DNA). A *word* W of length h ($h << n$) is a succession of h letters: $W = W_1 \dots W_h$. A *pattern* $\mathcal{W}$ is a finite family of words $\mathcal{W} = \{W^1, \dots, W^r\}$. The length of $\mathcal{W}$ is defined as $\max_{1 \leqslant i \leqslant r}\{\text{length}(W_i)\}$. The size of $\mathcal{W}$ is defined as its number of words (r in our case).

The number of occurrences of W is defined by the following formulae:

$$N(W) = \sum_{i=1}^{n-h+1} Y_i \quad \text{with} \quad Y_i = \mathbb{I}_{X_i = W_1} \times \dots \times \mathbb{I}_{X_{i+h-1} = W_h}. \tag{1}$$

Then we can define the number of occurrences of $\mathcal{W}$ by the following:

$$N(\mathcal{W}) = N(W^1) + \dots + N(W^r). \tag{2}$$

We now consider that we observe a number of occurrences $n(\mathcal{W})$ on a given sequence. We have two cases:

- $\mathcal{W}$ is *overrepresented* if $n(\mathcal{W}) \geqslant \mathbb{E}[N(\mathcal{W})]$ (we see more occurrences than expected) and its significance is given by

$$P(\mathcal{W}) = \mathbb{P}\big(N(\mathcal{W}) \geqslant n(\mathcal{W})\big) \tag{3}$$

  with the associated (positive) score defined by

$$S(\mathcal{W}) = -\log_{10}\big(P(\mathcal{W})\big). \tag{4}$$

- $\mathcal{W}$ is *underrepresented* if $n(\mathcal{W}) < \mathbb{E}[N(\mathcal{W})]$ (we see fewer occurrences than expected) and its significance is given by

$$P(\mathcal{W}) = \mathbb{P}\big(N(\mathcal{W}) < n(\mathcal{W})\big) \tag{5}$$

  with the associated (negative) score defined by

$$S(\mathcal{W}) = \log_{10}\big(P(\mathcal{W})\big). \tag{6}$$

2.2. Level 1 large deviations

In this section, we shall consider that X is an order 1 Markov chain with the irreducible transition matrix Π (this means that $\forall i, j \in \mathcal{A}$, $\exists p \in \mathbb{N}$ such as $(\Pi^p)(i, j) > 0$).

If $f : \mathcal{A}^2 \to \mathbb{R}$ is a deterministic function and we want to study the deviations of

$$S_n = \sum_{i=1}^{n-1} f(X_i, X_{i+1}), \tag{7}$$

first we need to define $\forall \theta \in \mathbb{R}$, the *twisted matrix* associated to f, as following:

$$\Pi_\theta(i, j) = \Pi(i, j) e^{\theta f(i,j)} \quad \forall i, j \in \mathcal{A}. \tag{8}$$

As this matrix is positive and irreducible (but not necessarily stochastic), the Perron–Frobénius theorem (Karlin, 1969) says that Π_θ admits its largest eigenvalue $\rho(\Pi_\theta) > 0$.

Then we can define the function Λ and its classical Legendre dual Λ^* by

$$\Lambda(\theta) = \ln\big(\rho(\Pi_\theta)\big) \quad \text{and} \quad \Lambda^*(a) = \sup_{\theta \in \mathbb{R}}\{\theta.a - \Lambda(\theta)\} \quad \forall \theta, a \in \mathbb{R}. \tag{9}$$

We can now express the Markov version of the Cramer–Chernov theorem:

Theorem (level 1). *For all* $a \geqslant \mathbb{E}\left[\frac{S_n}{n-1}\right]$ *(respectively,* $<$*) we have*

$$\lim_{n\to+\infty} \frac{1}{n-1} \times \ln \mathbb{P}\left(\frac{S_n}{n-1} \geqslant a\right) = -\Lambda^*(a) \quad (\textit{respectively, } <). \tag{10}$$

Proof. A more general result than this one can be proved by a straightforward application of the Gärtner–Ellis theorem (as done by Dembo and Zeitouni [1998, Section 3.1], for example). Another proof can be done by generalizing the i.i.d. version given, for example, by Den Hollender [2000, Section I.3]. This involves a trivial upper bound using the Markov inequality and an interesting lower bound using the key technique of the change of measure as well as a central limit theorem. ■

2.3. Level 2 large deviations

For all $h > 1$, we consider that X is a Markov chain of order $m \leqslant h-1$, and here we want to look at the deviations of the empirical distribution $L_n = \left(L_n(1), \ldots, L_n(k^h)\right)$ of the words of length h with

$$L_n(i) = \frac{N(W^i)}{n-h+1} \quad \text{for all} \quad 1 \leqslant i \leqslant k^h \tag{11}$$

where W^i is the i^{th} word.

It is clear that L_n is a random variable taking its values in the set $\mathcal{M}_1\left(\mathcal{A}^h\right)$ of the distributions on the words of length h.

We call $\nu \in \mathcal{M}_1\left(\mathcal{A}^h\right)$ a *shift-invariant* if $\forall x_1, \ldots, x_{h-1} \in \mathcal{A}$, we have

$$\overline{\nu}(x_1, \ldots, x_{h-1}) = \sum_{x\in\mathcal{A}} \nu(x_1, \ldots, x_{h-1}, x) = \sum_{x\in\mathcal{A}} \nu(x, x_1, \ldots, x_{h-1}), \tag{12}$$

and for such a ν we can define $\tilde{\nu}$ by the following: $\forall x_1, \ldots, x_h \in \mathcal{A}$,

$$\tilde{\nu}(x_1, \ldots, x_h) = \overline{\nu}(x_1, \ldots, x_{h-1}) \times \mathbb{P}\left(X_h = x_h | X_1 = x_1, \ldots, X_{h-1} = x_{h-1}\right). \tag{13}$$

Theorem (level 2). $(L_n)_{n\geqslant h}$ *satisfy a large deviations principle (LDP) with the good rate function* I *(positive, semi-continuous function with compact level sets) defined* $\forall \nu \in \mathcal{M}_1\left(\mathcal{A}^h\right)$ *by*

$$I(\nu) = \begin{cases} H(\nu|\tilde{\nu}) & \textit{if } \nu \textit{ is a shift-invariant} \\ +\infty & \textit{else} \end{cases} \tag{14}$$

where $H(.|.)$ *is the relative entropy defined by*

$$H(\mu|\nu) = \sum_{i=1}^{k^h} \mu_i \ln \frac{\mu_i}{\nu_i} \quad \forall \mu, \nu \in \mathcal{M}_1\left(\mathcal{A}^h\right) \tag{15}$$

which means that $\forall \Gamma \subset \mathcal{M}_1\left(\mathcal{A}^h\right)$*; we have*

$$-\inf_{\overset{\circ}{\Gamma}} I \leqslant \liminf_{n\to+\infty} \frac{1}{n} \ln \mathbb{P}(L_n \in \overset{\circ}{\Gamma}) \leqslant \limsup_{n\to+\infty} \frac{1}{n} \ln \mathbb{P}(L_n \in \overline{\Gamma}) \leqslant -\inf_{\overline{\Gamma}} I. \tag{16}$$

Proof. Dembo and Zeitouni (1998) show this result as a corollary of the multidimensional level 1 PGD result (as given by the Gärtner–Ellis theorem). In Den Hollender (2000, chapter 4), an application of the Varadhan's lemma proposes the elegant alternative to derive this result from the i.i.d. case. ■

2.4. Applications

2.4.1. Level 1. **The simple case.** If $\mathcal{W}$ is a pattern of length $h = 2$, we can easily set f in order to have $S_n = N(\mathcal{W})$. For example, if $\mathcal{W} = \{\mathtt{gc}, \mathtt{ac}\}$ we can choose f such that $f(x, y) = 0$ for all $x, y \in \mathcal{A}$ except for $f(\mathtt{g}, \mathtt{c}) = f(\mathtt{a}, \mathtt{c}) = 1$.

For a given observed number of occurrences $n(\mathcal{W})$, we can use the level 1 theorem to obtain the following approximations on an order 1 irreducible Markov chain:

$$P(\mathcal{W}) \simeq \exp\big(-(n-1)\Lambda^*(a)\big) \tag{17}$$

and so

$$\ln 10 \times S(\mathcal{W}) \simeq \begin{cases} +(n-1) \times \Lambda^*(a) & \text{if } \mathcal{W} \text{ is overrepresented} \\ -(n-1) \times \Lambda^*(a) & \text{if } \mathcal{W} \text{ is underrepresented} \end{cases} \tag{18}$$

with $a = \frac{n(\mathcal{W})}{n-1}$.

The general case. This result can be easily generalized to patterns of length $h \geqslant 2$ on an order $m \leqslant h-1$ irreducible Markov chain using the alphabet $\mathcal{A}^{h-1}$ instead of $\mathcal{A}$. In this new alphabet, the sequence can be seen as an order 1 Markov chain where the considered pattern is only of length 2. For example, if we want to look at the pattern $\mathcal{W} = \{\mathtt{attc}\}$ on a Markov chain of order $m = 2$, we have to switch the problem from the alphabet $\mathcal{A} = \{\mathtt{a}, \mathtt{c}, \mathtt{g}, \mathtt{t}\}$ $(k = 4)$ to $\mathcal{A}^3 = \{\mathtt{aaa}, \mathtt{aac}, \ldots, \mathtt{ttg}, \mathtt{ttt}\}$ $(k = 4^3 = 64)$ where the new transition (and very sparse) matrix $\forall x_1, x_2, x_3, y_1, y_2, y_3 \in \mathcal{A}$ is given by

$$\Pi(x_1x_2x_3, y_1y_2y_3) = \mathbb{I}\{x_2 = y_1, x_3 = y_2\} \times \mathbb{P}(X_3 = y_3 | X_1 = y_1, X_2 = y_2)\,. \tag{19}$$

One should note that the same change of alphabet is necessary for any Markov model order. The only thing that really matters is in fact the length h of the pattern. According to the simple case, we hence have to compute only $\Lambda^*(a)$.

Arnoldi. First we need to be able to compute Λ, in other words, to compute the Perron-Frobénius eigenvalue of our twisted matrix Π_θ. As explained in the general case, that matrix will be of a high order ($d = k^{h-1}$ for a pattern of length h) and very sparse (only k^h nonzero terms among the k^{2h-2} entries).

Finding the largest eigenvalue of such a matrix can be a very difficult problem with the wrong approach (such as computing all the eigenvalues and then taking the largest). Fortunately, projection-class methods such as the Arnoldi algorithm (see Stewart, 1994; Saad, 1992; Arnoldi, 1951; and Lehoucq *et al.*, 1997) are very efficient in computing the few largest eigenvalues of a sparse matrix. The following explicitly restarted version of this algorithm specifically computes the largest eigenvalue. The dimension of the projection subspace is arbitrary limited to M, which is a internal parameter of the algorithm.

1. *initialization*: $x \in \mathbb{R}^d$ is a random column vector and ρ_{new} is set to -1
2. *main loop*: $v_1 = x$ and for $j \in \{1, 2, \ldots, M\}$
   a. $w = \Pi_\theta v_j$ and $\rho_{\text{old}} = \rho_{\text{new}}$
   b. for $i \in \{1, 2, \ldots, j\}$, $h_{i,j} = v_i' w$ and $w = w - h_{i,j} v_i$
   c. $h_{j+1,j} = ||w||_2$ and $v_{j+1} = \frac{w}{h_{j+1,j}}$
   d. compute ρ_{new} the largest eigenvalue of $H_j = \big(h_{i,i'}\big)_{1 \leqslant i,i' \leqslant j}$
   e. if $|\rho_{\text{new}} - \rho_{\text{old}}| < \varepsilon$ goto 4.
3. *explicit restart*: compute y the eigenvector associated to the last ρ_{new}, set $x = \big(v_1 v_2 \ldots v_{j_{\max}}\big) \times y$ and goto 2.
4. *end*.

One should observe that the matrix Π_θ is used (step 2a) only in a matrix by vector product, an operation which can be done very efficiently for sparse matrices. With a small value for M (say, $M = 20$, for example), the eigenvalue and eigenvector computations (steps 2d and 3) are numerically tractable especially since the matrices H are always in the Hessenberg form ($h_{i,j} = 0$ if $i > j + 1$), so the classical HQR and HQR2 algorithms can be used (see Garbow *et al.*, 1977). Finally let us point out that the closer x is to an eigenvector related to the largest eigenvalue of Π_θ the faster the algorithm converges.

Brent. As each evaluation of the function Λ could be quite time consuming, it is wise to choose a maximization method that minimizes these evaluations. Mixing golden and parabolic steps, Brent's algorithm (see Brent [1973] or Press *et al.* [1997]) seems to be a good choice.

Using that algorithm, one can find

$$\theta_{\max} = \arg\max \left\{ \theta \in \mathbb{R}, \theta a - \Lambda(\theta) \right\} \tag{20}$$

and as $\Lambda'(0) = \mathbb{E}\left[\frac{N(\mathcal{W})}{n-h+1}\right]$ we know that

$$\ln 10 \times S(\mathcal{W}) \simeq \begin{cases} +(n-h+1) \times \left(\theta_{\max} \times a - \Lambda(\theta_{\max})\right) & \text{if } \theta_{\max} \geqslant 0 \\ -(n-h+1) \times \left(\theta_{\max} \times a - \Lambda(\theta_{\max})\right) & \text{if } \theta_{\max} < 0 \end{cases}. \tag{21}$$

2.4.2. Level 2. Regarding level 1, we have to maximize a function which is difficult to evaluate, whereas at level 2, function evaluations are far easier (it is only a relative entropy); a minimization in a high dimension space has to be performed (see Den Hollender, 2000). Moreover, this minimization must be done under (linear) constraints because the rate function has finite values only on the subset $\mathcal{S}$ of shift-invariants.

If a single word W of length h is observed $n(W)$ times on our sequence, we set

$$\Gamma_+(W) = \left\{ \nu \in \mathcal{M}_1(\mathcal{A}^h), \nu(W) \geqslant \frac{n(W)}{n-h+1} \right\} \tag{22}$$

and

$$\Gamma_-(W) = \left\{ \nu \in \mathcal{M}_1(\mathcal{A}^h), \nu(W) \leqslant \frac{n(W)}{n-h+1} \right\} \tag{23}$$

so that

$$P(W) = \begin{cases} \mathbb{P}\left(L_n \in \Gamma_+(W)\right) & \text{if } W \text{ is overrepresented} \\ \mathbb{P}\left(L_n \in \Gamma_-(W)\right) & \text{if } W \text{ is underrepresented} \end{cases} \tag{24}$$

and using both (14) and (16) from the level 2 theorem we obtain

$$\frac{\ln 10 \times S(W)}{n-h+1} \simeq \begin{cases} +\inf_{\nu \in \Gamma_+(W) \cap \mathcal{S}} H(\nu|\tilde{\nu}) & \text{if } W \text{ is overrepresented} \\ -\inf_{\nu \in \Gamma_-(W) \cap \mathcal{S}} H(\nu|\tilde{\nu}) & \text{if } W \text{ is underrepresented} \end{cases}. \tag{25}$$

The dimension of the minimization space is k^h and the constraints can be taken into account either with an orthogonal projection (but such a projection is hard to compute in high dimensions) or with a penalization function.

With a similar approach (only subsets Γ have to change), any complex patterns can be treated, and joint as well as conditional distributions are also easy to compute.

3. RESULTS AND DISCUSSION

3.1. LD-SPatt

We wrote a C implementation of the level 1 approach. LD-SPatt (large deviations statistics for patterns) is the name of this GPL (general public license) software. Source code and binary builds for Linux/i586 are freely available at the following address: *www.stat.genopole.cnrs.fr/ldspatt/*.

As explained in Table 1, LD-SPatt needs a set of sequences in the FASTA format, a list of patterns, and a Markov model order. By default, the model parameters are estimated on the provided sequence by maximum of likelihood, but users can alternatively specify the parameters of their choice. Then the statistics are computed for each pattern, using Brent's and Arnoldi's algorithms.

TABLE 1. INPUTS/OUTPUTS OF LD-SPATT

Inputs	*Outputs*
• A single file containing one or more sequences in FASTA format; • A Markov model order with eventually a file containing the corresponding parameters; • A pattern or a list of patterns	• The patterns statistics (log scale by default but, alternatively, z-score can be produced for easier comparison with Gaussian approaches); • Estimated parameters of the model (using maximum of likelihood); • The twisted matrices associated to the statistics

TABLE 2. TIME AND MEMORY COMPLEXITY FOR THE COMPUTATION OF A SIZE r PATTERN STATISTICS ($r = 1$ ONLY FOR COMPOUND POISSON) ON A LENGTH n SEQUENCE USING AN ORDER m ($0 \leqslant m \leqslant h-1$) MARKOV MODEL FOR SEVERAL METHODS

Method	*Time*	*Memory*
Exact	$O(k^{2m})^{\mathrm{a}} + O(r^2n^2/k^h)$	$O(k^{2m})^{\mathrm{b}} + O(n)$
Compound Poisson	$O(n^2/k^{2h})$	$O(k^{m+1})^{\mathrm{b}} + O(h + n/k^h)$
Gaussian	$O(k^{m+1})^{\mathrm{a}} + O(hr^2)$	$O(k^{m+1})^{\mathrm{b}}$
Large deviations	$O(k^h)$	$O(k^h)$

Exact refers to the simple recurrence approach (see Robin and Daudin, 1999). Compound Poisson and Gaussian methods refer to Reinert and Schbath (1998) and Schbath (1997). In the case of exact and compound Poisson methods, the number of occurrences of the pattern is critical; using a very simple approximation, we have replaced its value by n/k^h.

[a]Treatments that have to be done only once for a whole set of patterns.

[b]Corresponds to the Markov parameters memory cost.

3.2. Numerical performances

Table 2 shows time and memory complexity for several methods. As we can see, the exact method is not suitable for high-order Markov models (m), and moreover, with a time complexity proportional to the product of length of the sequence by the number of occurrences, the method is obviously not usable for long sequences. With the compound Poisson approximations, complexity is quadratic on the number of occurrences which means that for very rare patterns (few occurrences in the whole sequence), the method remains quite efficient. With Gaussian approximations, the length of the sequence is no longer a problem. Moreover, as a substantial part of the computations is performed only once for all patterns, that method is especially efficient when a large number of pattern statistics are needed (for example, statistics for all words of a given length). For the large deviations, time and memory complexity are atypically independent from the Markov model but are both proportional to k^h (k size of the alphabet, h length of the pattern). The cost could seem very high, but one should note that this complexity is exactly the same in the Gaussian case when a maximal-order Markov model is used ($m = h - 1$).

Even if a pattern of size $r > 1$ could theoretically be treated by the compound Poisson approximations, this is very difficult to do in the most general cases (with overlaps between words in the pattern), and there is still no implementation of this functionality. For exact and Gaussian methods, the complexity is quadratic with the size r. This is quite obvious as every couple of words must be considered. Once again, large deviations are atypical on this point as the complexity remains the same for all r.

3.3. Validation

In order to validate LD-SPatt (version 0.9.1), we compare its results to those from RMES (version 2.1.2) which is a C++ program implementing both Gaussian and compound Poisson approximations proposed by Schbath (1997). Statistics for all the 65,536 DNA words of length $h = 8$ have been computed on *Escherichia coli K12* using an order $m = 1$ Markov model.

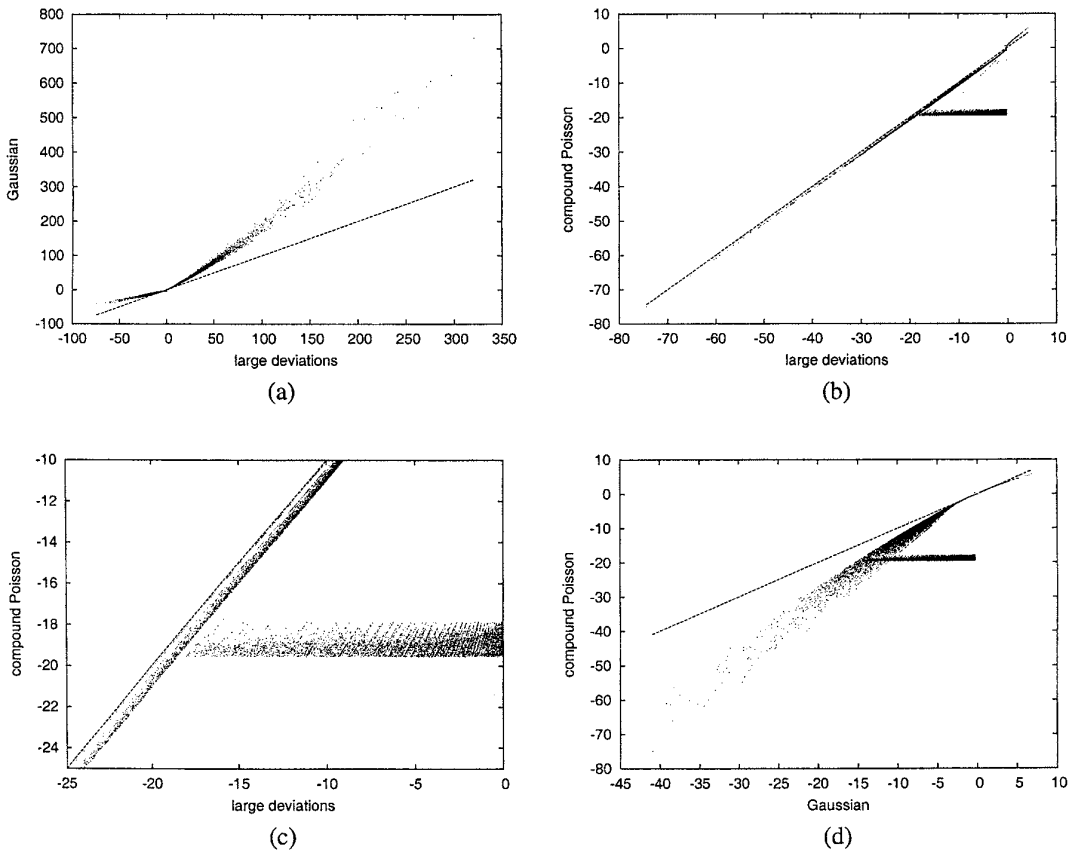


FIG. 2. Comparison of the three asymptotic methods. Gaussian, compound Poisson, and large deviations approximations. Each dot corresponds to DNA words of length $h = 8$ studied on *Escherichia coli K12* with an order $m = 1$ Markov model. Coordinates are pattern statistics with the labeled method (**a**: large deviations versus Gaussian; **b** and **c**: large deviations versus compound Poisson; **d**: Gaussian versus compound Poisson). (**c**) is a simple zoom of (**b**). All the 65,536 words are represented on (**a**). All the 28,147 words with less than 50 occurrences are represented on (**b**), (**c**), and (**d**).

3.4. Statistics values

A first point of interest is the reliability of the statistics computations. As we can see in Fig. 2a, both Gaussian and large deviations statistics are related but far from equal. Especially when we focus on the most extreme statistics equality, we can see huge differences. One should note that this behavior is similar to that in Fig. 1a and could therefore be interpreted as the failure of CLT for tail distribution events. Moreover, there are many words for which both the large deviations and compound Poisson statistics are the same while Gaussian statistics are not. This lead to the conclusion that, as expected in the case of such extreme events, the large deviations statistics are more reliable than the Gaussian ones.

Figure 2b shows the comparison between large deviations and compound Poisson approximations. Note that all words occurring more than 50 times in the sequence have been removed in order to stay in the theoretical framework of the compound Poisson method (37,389 words removed). The results of both methods are very close except for a horizontal group of roughly 9,000 words for which the results differ a lot. Figure 2c gives us a closer look on these words, where we can see small diagonal structures which seem related to the number of occurrences of their corresponding words.

Figure 2d compares Gaussian and compound Poisson statistics and shows that the issue is numerically related in the compound Poisson version of RMES. At the moment, there is no known solution for this

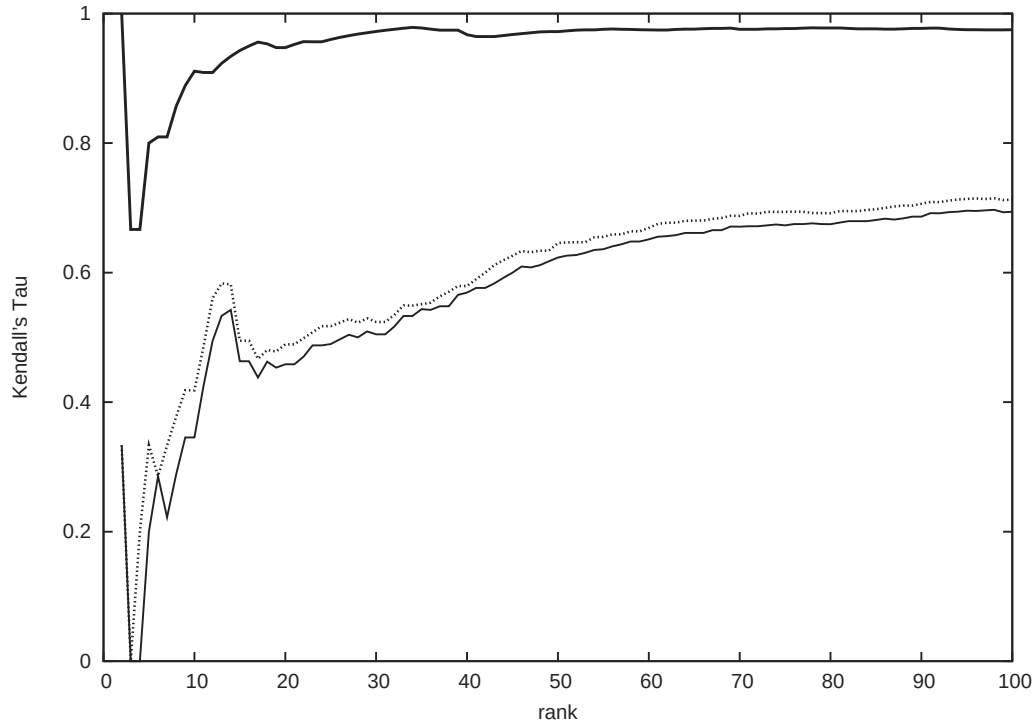


FIG. 3. Kendall's Tau for the 100 most underrepresented words of length $h = 8$ in *Escherichia coli K12* using an order 1 Markov model. The thin line represents Gaussian ranking versus large deviations ranking, the dotted line Gaussian versus compound Poisson, and the fat line large deviations versus compound Poisson.

issue, so it is important to use compound Poisson approximations produced by RMES with care (for example, filtering the results by comparing them to Gaussian approximations).

3.5. Statistics ranks

We have proved that extreme statistics are far more reliable with the large deviations, but a logical question is this: do we need so much reliability? In fact, with such statistics, p-values are very small, so it does not seem important to know whether the real p-value is 10^{-803} or 10^{-786}; differences are huge, but both p-values are obviously highly significant.

The thing that really matters here is the relative value of these statistics in respect to the others, that is, their ranks. In order to compare ranks given by the different methods, let us consider the 100 most underrepresented words. We can compute Kendall's τ statistic (see Press *et al.* [1997], for example) to compare ranking of the different methods. Here, τ is a symmetrical measurement of rank accordance and takes the value +1 for exactly the same ranking and -1 for exactly the reverse ranking.

Figure 3 shows Kendall's τ statistics for the three possible comparisons between Gaussian, compound Poisson, and large deviations statistics. As expected, with the huge differences observed on the values between Gaussian approximations and the two others, τ is very low when such approximations are involved, especially for the most extreme words. This demonstrates that Gaussian rankings have poor reliability in the tail distribution.

On the other hand, compound Poisson and large deviations rankings are very close but have also some differences. Unfortunately, without a reference ranking, it is impossible to decide which ranking is the best, and in the case of sequences of such length (several millions letters), exact p-values are very difficult to compute, which explains why such a reference is not available in our case.

4. CONCLUSION

When computing pattern statistics on Markov chains, exact methods are obviously very efficient. Unfortunately, complexity quickly increases with the length of the considered sequences or with the Markov model order (see Table 2), and this induces limitations to these methods.

To avoid the sequence length drawback, exact approaches can (and are often) used to get expectation and variance of pattern count with the aim of using Gaussian approximations. But this can be done only for low-order Markov chains. For higher orders, Gaussian approximations can be produced very quickly by tools such as RMES.

Based on the central limit theory, these approximations are very likely to fail in tail distribution. We have shown that this happens when focusing on the most over- or under-represented patterns, which are of course the patterns of interest. So in the matter of detecting statistically significant patterns, Gaussian statistics are inappropriate, and even their ranks are not reliable (30% to 60% rank discordance).

On the other hand, the compound Poisson approximations method seems far more reliable, but this method suffers several problems: 1) patterns cannot be considered, only simple words; 2) only rare words can be treated (this means in practice that the method is not reliable for words occurring more than 100 or 150 times, which means only words of length $h \geq 8$ should be treated in million-base sequences); and 3) there is an open numerical issue in the actual implementation in RMES.

With reliability similar to compound Poisson and fewer limitations in the range of possible patterns, the new large deviations approach we proposed with LD-SPatt therefore appears as a good alternative for the computations of pattern statistics. The tool is very reliable with the patterns of interest and works well with long sequences or high-order Markov models. Moreover, in a very unusual way, its complexity does not increase when the size of the pattern grows. Unfortunately, the overall complexity of the method is proportional to k^h where k is the cardinal of the considered alphabet and h the length of the pattern: the method cannot be efficiently used on long-length patterns (memory of a typical workstation would limit patterns to length $h = 12$ on a $k = 4$ DNA alphabet, for example).

One other interesting thing with LD-SPatt is that twisted matrices corresponding to specific pattern statistics can be produced. Such matrices can be seen as the parameters of the Markov chain closest (in a certain way, say, minimum entropy, for example) to the initial model where the observed pattern count is no more a tail but a central distribution event. We are looking forward to the application of these matrices in fields such as simulations or conditional statistics.

Finally, LD-SPatt is currently using only level 1 large deviations. Passing to level 2 could be a major improvement for the software, providing a way to compute joint and conditional statistics involving several patterns. But we first need to find appropriate and stable numerical solutions to deal with the corresponding problem of high multivariate minimization under constraints.

REFERENCES

Arnoldi, W.E. 1951. The principle of minimised iterations in the solution of the matrix eigenvalue problem. *Quart. Appl. Math.* 9, 17–29.

Atteson, K. 1998. Calculating the exact probability of language-like patterns in biomolecular sequences. *Proc. 6th Int. Conf. on Intelligent Systems for Molecular Biology*, 17–24.

Beaudoing, E., Freier, S., Wyatt, J.R., Claverie, J.-M., and Gautheret, D. 2000. Patterns of variant polyadenylation signal usage in human genes. *Genome Res.* 10(7), 1001–1010.

Brazma, A., Jonassen, I., Vilo, J., and Ukkonen, E. 1998. Predicting gene regulatory elements in silico on a genomic scale. *Genome Res.* 8(11), 1202–1215.

Brent, R.P. 1973. *Algorithms for Minimisation without Derivatives*, Prentice-Hall, Englewood Cliffs, NJ.

Cowan, R. 1991. Expected frequencies of DNA patterns using Whittle's formula. *J. Appl. Prob.* 28, 886–892.

Dembo, A., and Zeitouni, O. 1998. *Large Deviations Techniques and Applications*, Springer-Verlag, New York.

Den Hollender, F. 2000. *Large Deviations*, American Mathematical Society, Field Institute Monographs 15.

Deuschel, J.D. 1957. *Large Deviations*, Academic Press, Boston.

El Karoui, M., Biaudet, V., Schbath, S., and Gruss, A. 1999. Characteristics of chi distribution on different bacterial genomes. *Res. Microbiol.* 150, 579–587.

Garbow, B.S., Boyle, J.M., Dongarra, J.J., and Moler, C.B. 1977. Matrix eigensystem routines. *EISPACK Guide Extension Lecture Notes in Computer Science*, Springer-Verlag, Berlin.

Gelfand, M.S., and Koonin, E.V. 1997. Avoidance of palindromic words in bacterial and archaeal genomes: A close connection with restriction enzymes. [Published erratum appears in *Nucl. Acids Res*. 25(24), 5135–5136]. *Nucl. Acids. Res.* 25(12), 2430–2439.

Geske, M.X., Godbole, A.P., Schaffner, A.A., Skolnick, A.M., and Wallstorm, G.L. 1995. Compound Poisson approximation for word patterns under Markovian hypothesis. *J. Appl. Prob.* 32, 877–892.

Godbole, A.P. 1991. Poisson approximations for runs and patterns of rare events. *Adv. Appl. Prob.* 23, 851–865.

Hampson, S., Kibler, D., and Baldi, P. 2002. Distribution patterns of over-represented k-mers in non-coding yeast DNA. *Bioinformatics* 18(4), 513–528.

Karlin, S. 1969. *Initiation aux Processus Aléatoires*, Dunod, Paris.

Karlin, S., Burge, C., and Campbell, A.M. 1992. Statistical analyses of counts and distributions of restriction sites in DNA sequences. *Nucl. Acids. Res.* 20(6), 1363–1370.

Kleffe, J., and Borodovsky, M. 1992. First and second moment of counts of words in random text generated by Markov chains. *Comp. Applic. Biosci.* 8, 433–441.

Lehoucq, R., Sorensen, D., and Yang, C. 1997. Arpack users' guide: Solution of large scale eigenvalue problems with implicitly restarted Arnoldi methods. Technical report from *www.caam.rice.edu/software/ARPACK/*, Computational and Applied Mathematics, Rice University.

Nicodème, P. 2001. Fast approximate motif statistics. *J. Comp. Biol*. 8(3), 235–248.

Nicodème, P., Salvy, B., and Flajolet, P. 2002. Motif statistics. *Theoret. Comput. Sci*. 18, 161–171.

Pevzner, P.A., Borodovski, M.Y., and Mironov, A.A. 1989. Linguistic of nucleotide sequences: The significance of deviation from mean statistical characteristics and prediction of frequencies of occurrence of words. *J. Biomol. Struct. Dyn*. 6, 1013–1026.

Press, W.H., Teukolsky, S.A., Vettering, W.T., and Flannery, B.P. 1997. *Numerical Recipes in C*, Cambridge University Press, London.

Prum, B., Rodolphe, F., and de Turckheim, E. 1995. Finding words with unexpected frequencies in DNA sequences. *J. R. Statist. Soc. B.* 11, 190–192.

Reinert, G., and Schbath, S. 1998. Compound Poisson and Poisson process approximations for occurrences of multiple words in Markov chains. *J. Comp. Biol.* 5, 223–254.

Robin, S., and Daudin, J.J. 1999. Exact distribution of word occurrences in a random sequence of letters. *J. App. Prob.* 36, 179–193.

Régnier, M. 2000. A unified approach to word occurrence probabilities. *Disc. Appl. Math*. 104(1), 259–280.

Régnier, M., and Szpankowski, W. 1998. On pattern frequency occurrences in a Markovian sequence. *Algorithmica* 22(4), 631–649.

Saad, Y. 1992. *Numerical Methods for Large Eigenvalue Problems*, Manchester University Press, UK.

Schbath, S. 1997. An efficient statistic to detect over- and under-represented words in DNA sequences. *J. Comp. Biol.* 4, 189–192.

Stewart, W.J. 1994. *Introduction to Numerical Solution to Markov Chains*, Princeton University Press, Princeton, NJ.

van Helden, J., Olmo, M., and Perez-Ortin, J.E. 2000. Statistical analysis of yeast genomic downstream sequences reveals putative polyadenylation signals. *Nucl. Acids Res.* 28(4), 1000–1010.

Address correspondence to:
G. Nuel
Laboratoire Statistique et Génome
Tour Evry 2, 523 place des terrasses
91034 Evry, France

E-mail: nuel@genopole.cnrs.fr

Chapitre 8

S-SPatt : Simple Statistics for Patterns on Markov chains

Application note de Nuel publiée en 2005 dans *Bioinformatics*. L'approximation binomiale pour le problème des motifs y est présenté ainsi que le logiciel *sspatt* dans lequel elle est implémente. Les performances de cette approximation sont comparées à celles des approximations gaussiennes, Poisson composées et grandes déviations. L'approximation binomiale se révèle être étonnamment fiable et constitue une bonne heuristique (le plus souvent largement préférable aux approximations gaussiennes) pour le calcul des statistiques de motifs.

BIOINFORMATICS APPLICATIONS NOTE Vol. 21 no. 13 2005, pages 3051–3052
doi:10.1093/bioinformatics/bti451

Sequence analysis

S-SPatt: simple statistics for patterns on Markov chains

Grégory Nuel

Laboratoire Statistique et Génome, 523 place des terrasses de l'Agora, 91000 Evry, France

Received on February 15, 2005; revised on March 29, 2005; accepted on April 13, 2005
Advance Access publication April 19, 2005

ABSTRACT

Summary: S-SPatt allows the counting of patterns occurrences in text files and, assuming these texts are generated from a random Markovian source, the computation of the *P*-value of a given observation using a simple binomial approximation.

Availability: S-SPatt is available at: http://stat.genopole.cnrs.fr/spatt

Contact: spatt@genopole.cnrs.fr

1 PREVIOUS WORK

It is well known that patterns frequencies are highly dependent on their overlapping structures as well as on the composition bias in the considered sequences. In order to compare patterns efficiently, we need more than their simple counts. A common solution to this problem consists in using a Markov chain to modelize the sequence thereby ranking patterns according to their *P*-values.

Several software propose solutions to compute these *P*-values: QuickScore using exact computation with generative series as proposed in Régnier (2000) (under development but not yet available), RMES (Schbath, 1997) for Gaussian and Compound Poisson approximations and LD-SPatt for large deviations approximations (Nuel, 2004). Unfortunately, all these methods suffer severe drawbacks: QuickScore can not consider Markovian model of order higher than 1 or 2 and is limited to short sequences (for exact computations), Gaussian approximations are wrong for tail distribution events (which are of course the events of interest), Compound Poisson approximations are designed only for rare patterns (and have some numerical issues in the RMES implementation making its use difficult anyway) and, finally, large deviations techniques are limited to short patterns (less than 10 letters long on a DNA alphabet, for example).

The Regulatory Sequence Analysis tool (RSAT) proposes to use simple binomial approximations to compute these *P*-values. Simulations have shown (van Helden *et al*., 1998) that such approximations are wrong but very close to the optimal solution when the considered patterns are not self-overlapping too much.

As the RSAT package is only available on-line, we decided to implement the same method in a stand-alone GPL program called S-SPatt to compare its overall performances with other similar programs.

2 METHOD

2.1 Statistics

Let $X = X_1, \ldots, X_n$ be an order m stationary Markov chain on a size k finite alphabet $\mathcal{A}$, with Π ($k^m \times k^m$ dimension) sparse transition matrix and μ (k^m dimension vector) as stationary distribution.

We consider a pattern $\mathcal{W} = \{W_1, \ldots, W_r\}$ as a set of r words (of respective length $h_1, \ldots, h_r$).

We count the number of occurrences of a given word $W = w_1, \ldots, w_h$ using $N(W)$ as defined by

$$N(W) = \sum_{i=1}^{n-h+1} I_{\{W \text{ starts in } i\}} \triangleq \sum_{i=1}^{n-h+1} Y_i. \quad (1)$$

According to the model, Y_i has a Bernoulli distribution of parameter

$$P(W) = \mu(W_1^m)\Pi(W_1^m, W_2^{m+1}), \ldots, \Pi(W_{h-m}^{h-1}, W_{h-m+1}^h), \quad (2)$$

where $W_i^j = w_i, \ldots, w_j$ for all $1 \leq i \leq j \leq h$.

Couples (Y_i, Y_j) are clearly not independent nevertheless, we can use the following heuristic distribution for a single word:

$$N(W) \sim \mathcal{B}(n - h + 1, P(W)), \quad (3)$$

which hence gives the following distribution for a pattern:

$$N(\mathcal{W}) \sim \mathcal{B}\left(n - \max_{1 \leq i \leq r} h_i + 1, \sum_{i=1}^{r} P(W_i)\right). \quad (4)$$

2.2 Algorithms

2.2.1 Stationary distribution According to the theorem of Perron–Frobénius, for any irreducible Markov chain, the stationary distribution can be computed by solving an eigenvalue problem. Growing exponentially with the Markov order, the scale of this problem is usually very large. To solve this problem, we propose to use an explicitly restarted Arnoldi's algorithm (Stewart, 1994) which is known to be very efficient with large, sparse eigenproblems (for an order m Markov model we have a non zero density of k^{1-m}).

2.2.2 Number of occurrences All words of length smaller than a given L are first computed ($n \times L$ in time and memory) and deterministic finite state automata are used for larger patterns (Hopcroft and Ullman, 1979).

2.2.3 P-values *P*-value for the observations are computed with the incomplete Beta function (Press *et al*., 1992).

3 RESULTS

3.1 Numerical performances

The following table gives the computation time T (in seconds) for N computations of simple words' *P*-values (all words of length $h = 6, \ldots, 10$ in *Mycoplasma genitalium* with an order 1 Markov model; computations are performed on an Intel Pentium 4 processor at 2.8 Gz).

N	4^6	4^7	4^8	4^9	4^{10}
$T(s)$	0.10	0.17	0.42	1.41	5.33

A simple linear regression gives about 200 000 computations per second which is very fast. Let us add, as a remark, that this result

is independent of the Markov model order (thanks to the Arnoldi algorithm). In comparison, for words of length 8 on a DNA alphabet, RMES Gaussian performs about 30 000 computations per second, RMES compound Poisson ∼2000 and LD-SPatt ∼4 (computation time with LD-SPatt grows exponentially with the length of the patterns).

S-SPatt can also be used to compute very efficiently simple word frequencies. For example, counting all words of length 8 on *Escherichia coli* complete genome requires roughly 15 min with the wordcount program from the popular EMBOSS package while S-SPatt can achieve the same task in less than half a second (hence, S-SPatt is ∼2000 times faster than EMBOSS).

3.2 Reliability of the heuristic

As Gaussian approximations are expected to be good in the center of the distribution (for high P-values), they are taken as reference for such events while large deviations are used as reference for all tail distribution events (see Nuel, 2004 for more discussion on the subject).

We can see from Table 1 that our simple binomial approximation is the closest to the reference both in terms of relative error and, more important, in terms of rank agreement (see Press *et al*., 1992 for more details on Kendall's Tau).

4 CONCLUSION

S-SPatt is not only the fastest tool (about 200 000 computations per second) but also the most reliable one (after the reference ones).

Moreover, the proposed implementation have several interesting features:

- Support of any user defined alphabet (regular DNA, purin-pyrimidin, amino acids, group of amino acids, latin, case sensitive, . . .) with a simple syntax.
- Markov model parameters are estimated using maximum likelihood or are specified by the user.

Table 1. Reliability comparison

Method	Relative error		Kendall's Tau	
LD-SPatt	Ref	0.516	Ref	Ref
RMES G	0.262	Ref	0.636	0.794
RMES CP[a]	0.145	0.076	0.936	0.298
S-SPatt	0.119	0.065	0.932	1.000

Computation done with words of length 8 on *E.coli K12* with an order 1 Markov model estimated by maximum likelihood. Mean relative errors on log P-value are given first for significant words (P-value smaller than 10^{-4}) and then for the non significant ones. Mean Kendall's Tau are given first for the 50 most under-represented words and then for the 50 most over-represented ones.

[a]Indicates that the 20% worse results have been removed for RMES CP (because that software suffers severe open numerical issues in its actual implementation).

- Support for high order Markov chain including computation of the stationary distribution using efficient linear algebra methods.

In conclusion, S-SPatt seems to be a very good heuristic for the computation of pattern statistics on Markov chains.

REFERENCES

van Helden,J. *et al*. (1998) Extracting regulatory sites from the upstream region of yeast genes by computational analysis of oligonucleotide frequencies. *J. Mol. Biol.*, **281**, 827–842.

Hopcroft,J.E. and Ullman,J.D. (1979) *Introduction to Automata Theory, Languages, and Computation.* Addison-Wesley, Reading, MA.

Nuel,G. (2004) LD-SPatt: large deviations statistics for patterns on Markov chains. *J. Comp. Biol.*, **11**, 1023–1033.

Press,W.H. *et al*. (1992) *Numerical Recipes in C*. Cambridge University Press.

Régnier,M. (2000) A unified approach to word occurrence probabilities. *Discrete Appl. Math.*, **104**, 259–280.

Schbath,S. (1997) An efficient statistic to detect over- and under-represented words in DNA sequences. *J. Comp. Biol.*, **4**, 189–192.

Stewart,W.J. (1994) *Introduction to Numerical Solution to Markov Chains*. Princeton University Press.

Chapitre 9

Effective p-value computations using Finite Markov Chain Imbedding : application to local score and to pattern statistics

Article de Nuel publié en 2006 dans *Algorithms for Molecular Biology*. Un nouvel algorithme général permettant des calculs efficaces dans le cadre d'une FMCI y est présenté. Cet algorithme est appliqué à deux cas particuliers : le score local d'une séquence aléatoire et les statistiques de motifs. Dans les deux cas, on tire parti de la structure particulière de la FMCI correspondante pour obtenir un algorithme spécifique ainsi que des développements asymptotiques. Pour le score local, cet algorithme permet, pour la première fois, le calcul exact de sa distribution dans un cas concret (recherche de segments hydrophobes dans la base de données SwissProt) et met en évidence la mauvaise qualité des approximations asymptotiques.

Algorithms for Molecular Biology BioMed Central

Research

Open Access

Effective p-value computations using Finite Markov Chain Imbedding (FMCI): application to local score and to pattern statistics

Grégory Nuel*

Address: Laboratoire Statistique et Génome, UEVE, CNRS (8071), INRA (1152), Evry, France

Email: Grégory Nuel* - nuel@genopole.cnrs.fr

* Corresponding author

Published: 07 April 2006

Algorithms for Molecular Biology 2006, **1**:5 doi:10.1186/1748-7188-1-5

Received: 15 February 2006

Accepted: 07 April 2006

This article is available from: http://www.almob.org/content/1/1/5

Abstract

The technique of Finite Markov Chain Imbedding (FMCI) is a classical approach to complex combinatorial problems related to sequences. In order to get efficient algorithms, it is known that such approaches need to be first rewritten using recursive relations. We propose here to give here a general recursive algorithms allowing to compute in a numerically stable manner exact Cumulative Distribution Function (CDF) or complementary CDF (CCDF). These algorithms are then applied in two particular cases: the local score of one sequence and pattern statistics. In both cases, asymptotic developments are derived. For the local score, our new approach allows for the very first time to compute exact p-values for a practical study (finding hydrophobic segments in a protein database) where only approximations were available before. In this study, the asymptotic approximations appear to be completely unreliable for 99.5% of the considered sequences. Concerning the pattern statistics, the new FMCI algorithms dramatically outperform the previous ones as they are more reliable, easier to implement, faster and with lower memory requirements.

1 Introduction

The use of Markov chains is a classical approach to deal with complex combinatorial computations related to sequences. In the particular case of pattern count on random sequences, [5] named this method Finite Markov Chain Imbedding (FMCI, see [11] or [7] for a review). Using this technique it is possible to compute exact distributions otherwise delicate to obtain with classical combinatorial methods. More recently, [12] proposed a similar approach to consider local score on i.i.d. or Markovian ([13]) random sequences. Although these methods are very elegant, they could require a lot of time and memory if they are implemented with a naive approach. The authors of [6] first stated that recursive relation could be established for any particular case in order to provide an efficient way to perform the computations. We propose here to explore in detail this idea with the aim to provide fast algorithms able to compute with high numerical accuracy both CDF (cumulative distribution function) and CCDF (complementary CDF) of any general problem which can be written as a FMCI. We apply then these results to the particular cases of local score and pattern statistics. In each case, asymptotic developments are derived and numerical results are presented.

2 Methods

In this part, we first introduce in section 2.1 the FMCI and see the limits of naive approaches to their corresponding numerical computations. The main results are given in section 2.3 where we propose two effective algorithms

able to to compute general FMCI p-values (algorithm 1) or complementary p-value (algorithm 2). The theoretical background for these algorithms is given in the section 2.2.

2.1 Finite Markov Chain Imbedding

Let us consider $X = X_1,...,X_n$ a sequence of Bernoulli or Markov observations and E_n an event depending on the sequence X. We suppose that it is possible to build from X an order one Markov chain $Z = Z_1,...,Z_n$ on the finite state space $\mathcal{S}$ of size L. This space contains (in the order): k starting states denoted $s_1,...,s_k$, some intermediate states, and one final absorbing state f. The Markov chain is designed such as

$$(E_n|Z_1 = s_i) = (Z_n = f|Z_1 = s_i) = \prod^{n-1}(s_i, f) \quad (1)$$

where

$$\Pi = \begin{pmatrix} & R & & v \\ 0 & \dots & 0 & 1 \end{pmatrix} \quad (2)$$

is the transition matrix of Z.

If μ is the starting distribution of Z_1, we hence get

$$\mathbb{P}(E_n) = \sum_{i=1}^{k} \mu(s_i)\Pi^{n-1}(s_i, f) \quad (3)$$

Using this approach (and a binary decomposition of n - 1), it is possible to compute the p-value with $O(\log_2(n) \times L^2)$ memory complexity and $O(\log_2(n) \times L^3)$ time complexity. As L usually grows very fast when we consider more complex events E_n, these complexities are a huge drawback of the method. Moreover, numerical precision considerations prevent this approach to give accurate results when using the relation $(E_n^c) = 1 - (E_n)$ to compute the p-value of the complementary event (as the absolute error is then equal to the relative precision of the computations).

2.2 Effective computations

Proposition 1. For all $n \geq 1$ we have

$$\Pi^n = \begin{pmatrix} & R^n & & \gamma^{n-1} \\ 0 & \dots & 0 & 1 \end{pmatrix} \text{ with } \gamma^{n-1} = \sum_{i=0}^{n-1} R^i v \quad (4)$$

Proof. This trivial to establish by recurrence using matrix block multiplications. □

We hence get the

Corollary 2 (direct p-value). For all $n \geq 1$ we have

for all $1 \leq i \leq k$ $(E_n|X_1 = s_i) = \gamma_i^{n-2}$ and

$$\mathbb{P}(E_n) = \sum_{i=1}^{k} \mu_i \gamma_i^{n-2} \quad (5)$$

with γ^{n-2} computable through the following recurrence relations:

$$x^0 = \gamma^0 = v \quad \text{and, for all } j \geq 0 \quad x^{j+1} = Rx^j \quad \text{and} \quad \gamma^{j+1} = \gamma^j + x^j \quad (6)$$

Proof. Simply use proposition 1 to rewrite equations (1) and (3). Recurrence relations are then obvious to establish. □

And we also get the

Corollary 3 (complementary p-value). For all $n \geq 1$ we have

for all $1 \leq i \leq k$ $(E_n^c|X_1 = s_i) = x_i^{n-1}$ and

$$\mathbb{P}(E_n^c) = \sum_{i=1}^{k} \mu_i x_i^{n-1} \quad (7)$$

with x^0 is a size L - 1 column vector filled with ones and with $x^{n-1} = R^{n-1}x^0$ which is computable through the following recurrence relation:

$$\text{for all } j \geq 0 \quad x^{j+1} = Rx^j \quad (8)$$

Proof. $\prod$ being a stochastic matrix, $\prod^{n-1}$ is also stochastic, it is therefore clear that the sum of R^{n-1} over the columns gives 1 - γ^{n-2} and the corollary is proved. □

Using these two corollaries, it is therefore possible to accurately compute the p-value of the event or of its complementary with a complexity $O(L + \zeta)$ in memory and $O(n \times \zeta)$ in time where ζ is the number of non zero terms in the matrix R. In the worst case, $\zeta = (L - 1)^2$ but the technique of FMCI usually leads to a very sparse structure for R. One should note that these dramatic improvements from the naive approach could even get better by considering the structure of R itself, but this have to be done specifically for each considered problem. We will give detailed examples of this in both our application parts but, for the moment, we focus on the general case for which we give algorithms.

2.3 Algorithms

Using with the corollary 2 we get a simple algorithm to compute $p = (E_n)$

algorithm 1: direct p-value

x is a real column vector of size L - 1 and γ a real column vector of size k

initialization $x = (v_1,...,v_{L-1})'$ and $\gamma = (v_1,...,v_k)'$

main loop for $i = 1...n$ - 2 do

- $x = R \times x$ (sparse product)
- $\gamma = \gamma + (x_1,...,x_k)'$

end return $p = \sum_{i=1}^{k} \mu_i \gamma_i$

and using the corollary 3 we get an even simpler algorithm to compute the $q = 1 - p = (E_n^c)$

algorithm 2: complementary p-value

x is a real column vector of size L - 1

initialization $x = (1,...,1)'$

main loop for $i = 1...n$ - 1 do

- $x = R \times x$ (sparse product)

end return $q = \sum_{i=1}^{k} \mu_i x_i$

The more critical stage of both these algorithms is the sparse product of the matrix R by a column vector which can be efficiently done with ζ operations.

It is interesting to point out the fact that these algorithms do not require the stationarity of the underlying Markov chain. More surprisingly, it is also possible to relax the random sequence homogeneity assumption. Indeed, if our transition matrix $\prod$ depends on the position i in the sequence, we simply have to replace R in the algorithms with the corresponding R_i (which may use a significant amount of additional memory depending on its expression as a function of i).

For complementary p-value, we require to compute $R_1R_2...R_{n-1}R_nx$ which is easily done recursively starting from the right. In the direct p-value case however, it seems more difficult since we need to compute $x + R_1x + R_1R_2x + ... + R_1R_2...R_{n-1}R_nx$. Fortunately this sum can be rewritten as $x + R_1(x + R_2\{... [x + R_{n-1}(x + R_nx)]...\})$ which is again easy to compute recursively starting from the right.

The resulting complexities in the heterogeneous case are hence the same than in the homogeneous one (assuming that the number of non zero terms in R_i remains approximately constant). This remarkable property of the FMCI should be remembered especially in the biological field where most sequences are known to have complex heterogeneous structures which are often difficult to take into account.

3 Application 1: local score

We propose in this part to apply our results to the computation of exact p-values for local score. We first recall the definition of the local score of one sequence (section 3.1) and design a FMCI allowing to compute p-value in the particular case of an integer and i.i.d. score (section 3.2). We explain in sections 3.5 and 3.6 how to relax these two restrictive assumptions to consider rational or Markovian scores. The main result of this part is given in section 3.4 where we propose an algorithm improving the simple application of the general ones by using a specific asymptotic behaviour presented in section 3.3. As numerical application, we propose finally in section 3.7 to find significant hydrophobic segments in the Swissprot database using the Kyte-Doolittle hydrophobic scale. Our exact results are compared to the classical Gumble asymptotic approximations and discussed both in terms of numerical performance and reliability.

3.1 Definition

We consider $S = S_1,...,S_n$ a sequence of real scores and we define the local score H_n of this sequence by

$$H_n = \max\left\{0, \max_{i,j}\left(\sum_{\ell=i}^{j} S_\ell\right)\right\} \qquad (9)$$

which is exactly the highest partial sum score of a subsequence of S.

This local score can be computed in $O(n)$ using the auxiliary process

$$U_0 = 0 \quad \text{and for } 1 \le j \le n \quad U_j = \max\left\{0, \max_{i}\left(\sum_{\ell=i}^{j} S_\ell\right)\right\}$$

$$= \max\{0,\ U_{j-1} + S_j\} \qquad (10)$$

because we then have $H_n = \max_j U_j$.

Assuming the sequence S is random (Bernoulli or Markov model), we want to compute p-values relative to the event $E_n = \{H_n \ge a\}$ where $a > 0$.

3.2 Integer score

In order to simplify, we will first consider the case of integer scores (and hence $a \in$) then we will extend the result to the case of rational scores.

In the Bernoulli case, [12] introduced the FMCI Z defined by

$$Z_0 = 0 \quad \text{and} \quad Z_j = \begin{cases} U_j & \text{if there is no } a \text{ in } U_0,\ldots,U_j \\ a & \text{else} \end{cases} \tag{11}$$

(resulting with a sequence of length $n + 1$) with 0 as the only starting state and a as the final absorbing state. The transition matrix $\prod$ is given by

$$\Pi = \begin{pmatrix} f(0) & p(1) & \ldots & p(a-1) & g(a) \\ \vdots & & & & \\ f(-h) & p(1-h) & \ldots & p(a-h-1) & g(a-h) \\ \vdots & & & & \\ f(1-a) & p(2-a) & \ldots & p(0) & g(1) \\ 0 & 0 & \ldots & 0 & 1 \end{pmatrix} \tag{12}$$

where

$$p(i) = (S_1 = i) \quad f(i) = (S_1 \le i) \quad g(i) = (S_1 \ge i) \quad \forall i \in \tag{13}$$

It is possible to apply to this case the general algorithm 1 with $L = a + 1$ and $k = 1$ (please note that we have added Z_0 to the sequence and n must then be replaced by $n + 1$ in the algorithm to get correct computations) to compute the p-value we are looking for. In the worst case, R has $\zeta = a^2$ non zero terms and the resulting complexity is $O(a^2)$ in memory and $O(n \times a^2)$ in times. But in most cases, S_1 support is reduced to a small number of values and the complexities decrease accordingly.

3.3 Asymptotic development

Is it possible to compute this p-value faster ? In the case where R admits a diagonal form, simple linear algebra could help to cut off the computations and answer yes to this question.

Proposition 4. If R admits a diagonal form we have

$$\mathbb{P}(H_n \ge a) = \left[\sum_{i=0}^{\alpha-1} R^i v\right]_1 + \lambda^\alpha \frac{\left(1-\lambda^{n-\alpha}\right)}{(1-\lambda)}\left[R^\infty v\right]_1 + O\left(\nu^\alpha\right) \quad \forall n \ge \alpha \tag{14}$$

where $[]_1$ denotes the first component of a vector, with $R^\infty = \lim_{i\to\infty} R^i/\lambda^i$, where $0 < \lambda < 1$ is the largest eigenvalue of R and ν is the magnitude of the second largest eigenvalue. We also have $v = [g(a),\ldots,g(1)]'$.

Proof. By using the corollary 15 (appendix A) we know that

$$R^i - \lambda^i R^\infty = O(\nu^i) \tag{15}$$

uniformly in i so we finally get for all α

$$\sum_{i=0}^{n-1} R^i v - \sum_{i=0}^{\alpha-1} R^i v - \sum_{i=\alpha}^{n-1} \lambda^i R^\infty v = \sum_{i=\alpha}^{n-1} O\left(\nu^i\right) = O\left(\nu^\alpha \frac{\left(1-\nu^{n-\alpha}\right)}{(1-\nu)}\right) = O\left(\nu^\alpha\right) \tag{16}$$

uniformly for all $n \ge \alpha$ and the proposition is then proved by considering the first component of equation (16). □

Corollary 5. We have

$$\lim_{n\to\infty} \frac{\mathbb{P}(H_{n+1} \ge a) - \mathbb{P}(H_n \ge a)}{\lambda^n} = \left[R^\infty v\right]_1 \tag{17}$$

and

$$\lim_{n\to\infty} \frac{\mathbb{P}(H_{n+2} \ge a) - \mathbb{P}(H_{n+1} \ge a)}{\mathbb{P}(H_{n+1} \ge a) - \mathbb{P}(H_n \ge a)} = \lambda \tag{18}$$

Proof. Simply replace the terms in (17) and (18) with equation (14) to get the results. □

3.4 Algorithm

The simplest way to compute $(H_n \ge a)$ is to use the algorithm 2 in our particular case. As the number of non zero terms in R is then a^2, the resulting complexity is $O(n \times a^2)$. Using the proposition 4, it possible to get the same result a bit faster on very long sequence by computing the first two largest eigenvalues magnitudes λ and ν (complexity in $O(a^2)$ with Arnoldi algorithms) and to use them to compute a p-value.

As the absolute error is in $O(\nu^\alpha)$ we obtain a require ε error level using a α proportional to $\log(\varepsilon)/\log(\nu)$ which results in a final complexity in $O(\log(\varepsilon)/\log(\nu) \times a^2)$. Unfortunately, this last method requires to use delicate linear algebra techniques and is therefore more difficult to implement. Another better possibility is to use the corollary 5 to get the following fast and easy to implement algorithm:

algorithm 3: local score p-value

x a real column vector of size a, $(p_i)_{i\ge1}$ and $(\lambda_i)_{i\ge3}$ to sequences of real and i an integer

initialization $x = [g(a),\ldots,g(1)]'$, $p_1 = g(a)$, and $i = 0$

main loop while ($i < n$ and (λ_i) has not yet converged towards λ)

- $i = i + 1$
- $x = R \times x$ (sparse product)

Table 1: Distribution of amino-acids estimated on Swissprot (release 47.8) database and Kyte-Doolittle hydrophobic scale. Mean score is -0.244.

a. a.	F	M	I	L	V	C	W	A	T	G
in %	4.0	2.4	5.9	9.6	6.7	1.5	1.2	7.9	5.4	6.9
score	2.8	1.9	4.5	3.8	4.2	2.5	-0.9	1.8	-0.7	-0.4
a. a.	S	P	Y	H	Q	N	E	K	D	R
in %	6.9	4.8	3.1	2.3	3.9	4.2	6.6	5.9	5.3	5.4
score	-0.8	-1.6	-1.3	-3.2	-3.5	-3.5	-3.5	-3.9	-3.5	-4.5

- $p_i = p_{i-1} + x_1$
- $\lambda_i = (p_i - p_{i-1})/(p_{i-1} - p_{i-2})$ (if defined)

end • $p = p_i$

- if $(i < n)$ then $p = p + (p_i - p_{i-1}) \dfrac{\left(1-\lambda^{n-i}\right)}{(1-\lambda)}$
- return p

At any step i of the main loop we have $p_i = (H_i \geq a)$ and the final value taken by i is the α of proposition 4. One should note that only the last three terms of $(p_i)_{i\geq 1}$ and (for a simple convergence testing) the last two terms of $(\lambda_i)_{i\geq 3}$ are required by the algorithm.

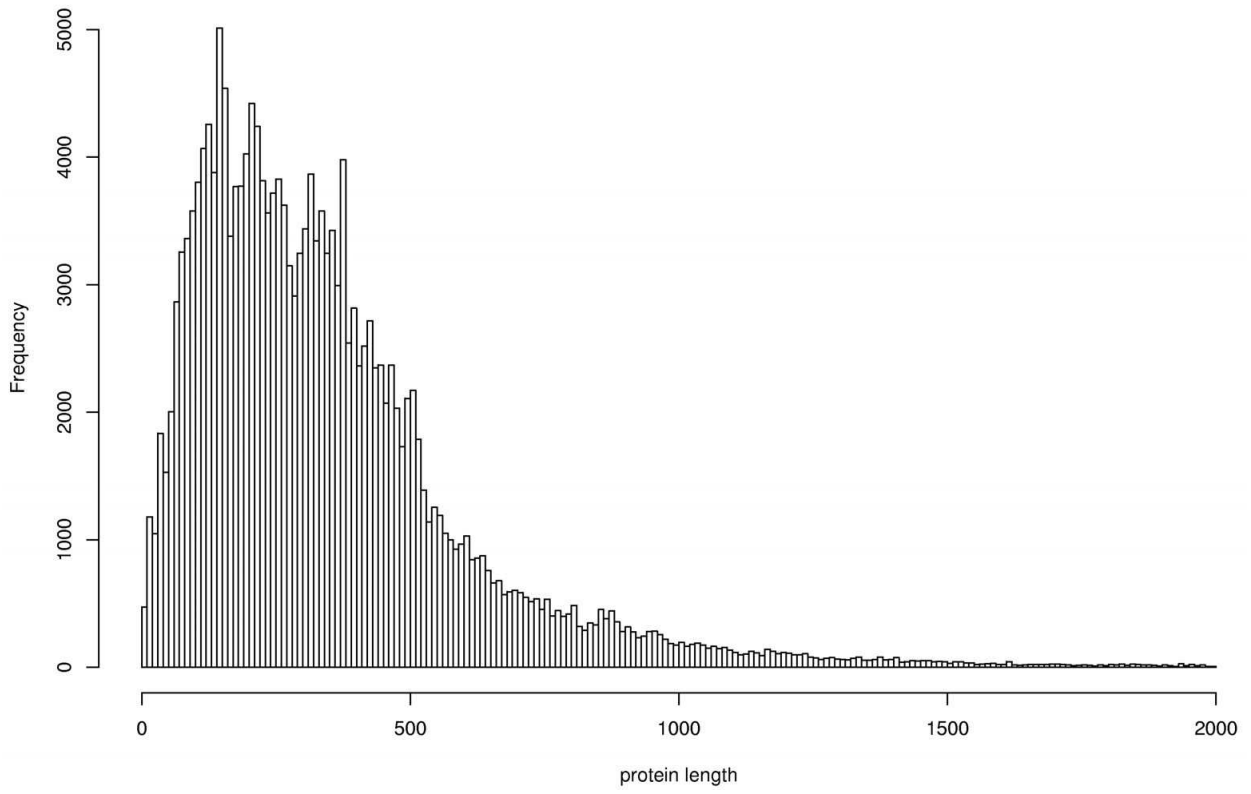


Figure 1
Empiric distribution of Swissprot (release 47.8) protein lengths. In order to improve readability, 0.5% of sequences with length $\in$ [2 000, 9 000] have been removed from this histogram.

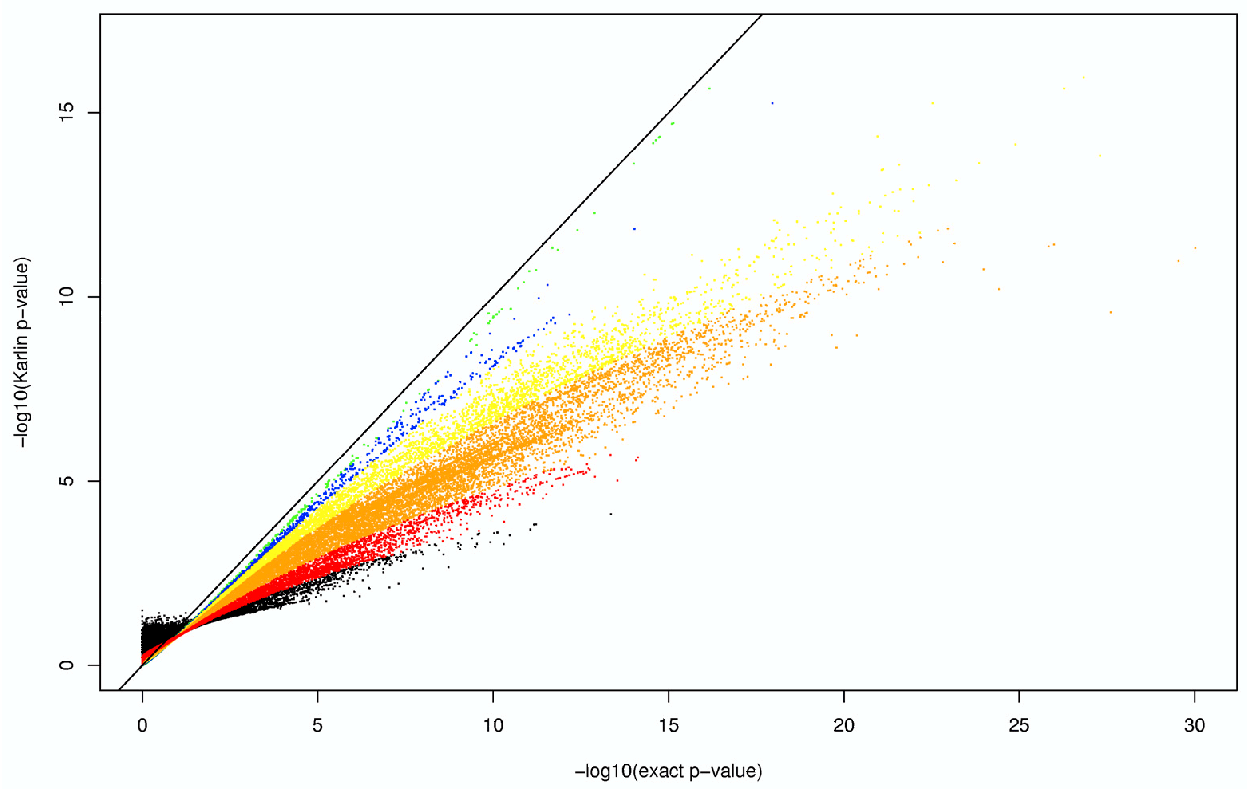


Figure 2
Exact p-value against Karlin ones (in log scale). Color refers to a range of sequence lengths: smaller than 100 in black (≃ 20 000 sequences), between 100 and 200 in red (≃ 40 000 sequences), between 200 and 500 in orange (≃ 90 000 sequences), between 500 and 1000 in yellow (≃ 30 000 sequences), between 1000 and 2 000 in blue (≃ 6 000 sequences) and greater than 2 000 in green (≃ 1 000 sequences). The solid line represents $y = x$. Range have been chosen for readability and few dots with exact p-value smaller than 10^{-30} are hence missing.

3.5 Rational scores

What if we consider now a rational score instead of an integer one ? If we denote by $\mathcal{S} \subset$ the support of S_1, let us define $M = \min_{i\in}\{i\mathcal{S} \subset\}$. Changing the scale of the problem by the factor M allows us to get back to the integer case:

$$(H_n \geq a) = (M\, H_n \geq M\, a) \quad (19)$$

This scale factor will obviously increase the complexity of the problem, but as the support cardinal (denoted η) is not changed during the process, the resulting complexities are $O(M \times a \times \eta)$ in memory and $O(M \times n \times a \times \eta)$ in time (n could vanish from the time complexity thanks to the faster algorithm presented above).

For example, if we consider the Kyte-Doolittle hydrophobicity score of the amino-acids (see [10] and table 1), it takes only $\eta = 20$ values and $M = 10$, the resulting complexity to compute $(H_n \geq a)$ is then $O(200 \times n \times a)$. If we consider now the more refined Chothia score ([4]), the scale factor increases from $M = 10$ to $M = 100$ and the resulting complexities are multiplied by 10.

3.6 Markov case

All these results can be extended to the Markov case but this require to define a new FMCI allowing us to trace the last score (in the case of an order one Markov chain for the sequence S, if a higher order m is considered, we just have to add the corresponding number of preceding scores to Z instead of one):

$$Z_j = \begin{cases} \left(S_{j-1}, U_j\right) & \text{if there is no } a \text{ in } U_0,\ldots,U_j \\ f & \text{else} \end{cases} \quad (20)$$

Doing this now we get $k = \eta$ (the cardinal of the score support) starting states instead of one so we need a starting

Table 2: Number of e-value smaller than a threshold are given for exact computations (exact) and asymptotic Karlin's approximations (Karlin). The last row gives the accuracy of asymptotic predictions (accuracy = Karlin/exact).

e-value	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}
exact	9473	7772	6271	4563	3232	2348
Karlin	3417	2047	1056	439	195	96
accuracy	34%	26%	17%	10%	6%	4%

distribution μ (which could be a Dirac) to compute the p-value.

We will not detail here the structure of the corresponding sparse transition matrix $\prod$ (see [13]) but we need to know its number ζ of non zero terms. If a is an integer value (we suppose here that the scale factor has been already included in it) then the order of R is $M \times a \times \eta$ and $\zeta = O(M \times a \times \eta^2)$ (and we get $O(M \times a \times \eta^{m+1})$ when an order m Markov model is considered).

3.7 Numerical results

In this section, we apply the results presented above to a practical local score study. We consider the complete protein database of Swissprot release 47.8 and the classical amino acid hydrophobic scale of Kyte-Doolittle given in table 1 ([10]). The database contains roughly 200 000 sequences of various lengths (empiric distribution given in figure 1).

Once the best scoring segment has been determined for each of these sequences, we need to compute the corresponding p-values. According to [9], the asymptotic distribution of H_n is given (if mean score is < 0, which is precisely the case here) by the following conservative approximation:

$$(H_n \geq a) \simeq 1 - \exp(-nKe^{-a\lambda}) \quad (21)$$

where constants λ and K depend on the scoring distribution.

With our hydrophobic scale and a distribution of amino-acids estimated on the entire database we get

$\lambda = 5.144775 \times 10^{-3}$ and $K = 1.614858 \times 10^{-2}$

(computation performed with a C function implemented by Altschul). Once the constants are computed we could get all the approximated p-values very quickly (a few seconds for the 200 000 p-values).

On the other hand, our new algorithm allows to compute (for the very first time) the exact p-values for this example. As the chosen scoring function has a one digit precision level, we need to use a scale factor of $M = 10$ to fall back to the integer case. A C++ implementation (available on request) performed all the computations in roughly three hours on a Pentium 4 CPU 2.8 GHz (this means approximately 20 p-values computed by second).

We can see on figure 2 the comparison between exact values and Karlin's approximations. The conservative design of the approximations seems to be successful except for very short unsignificant sequences. While the approximations are rather close to perfection for sequences with more than 2 000 amino-acids, the smaller the sequence is, the worse the approximations get. This is obviously consistent with the asymptotic nature of Karlin's formula but seems to indicate that these approximations are not reliable for 99.5% of the sequence in the database (protein of length < 2 000).

One should object that it exists ([1,2]) a well known finite size correction to formula (21) that might be useful, especially when considering short sequences. Unfortunately in our case, this correction does not seems to improve the quality of the approximations (data not shown) and we hence make the choice to ignore it.

In table 2 we compare the number of sequences predicted to have a significant hydrophobic segment at a certain e-value level by the two approaches. If the Karlin's approximations are used, many proteins are considered unsignificant while they are. For example, with the classical database threshold of 10^{-5}, only few sequences (6%) are correctly identified by Karlin's approximations.

We have seen that Karlin's approximations are often far too conservative to give accurate results, but what about the ranking ? Table 3 proposes the Kendall's tau rank correlation (see [16] chapter 14.6 for more details) which is equal to 1.0 for a complete rank agreement and equal to -1.0 for a complete inverse rank agreement. As we will certainly be interested in the most significant sequences produced by our study, we compute our Kendall's tau only on these sequences. When all sequence lengths are considered, Karlin's approximations show their total irrelevance to give correct ranking for the first 10 or 50 most significant p-values. Even when the 100 first p-values are taken into account, relative ranks given by Karlin's approximations are wrong in 63% of the cases, which is huge. How-

Table 3: Kendall's tau (rank correlation) comparing the most significant exact p-values (the reference) to the Karlin's approximations. The column "all" gives the result for all sequences while the R_i give the results for a certain range of sequence lengths: smaller than 100 for R1, between 100 and 200 for R2, between 200 and 500 for R3, between 500 and 1 000 for R4, between 1 000 and 2 000 for R5 and greater than 2 000 for R6.

number of p-values	all	R1	R2	R3	R4	R5	R6
10	0.30	0.64	0.24	-0.20	0.58	0.64	0.97
50	0.14	0.73	0.50	0.46	0.56	0.78	0.97
100	0.37	0.70	0.67	0.62	0.61	0.80	0.98

ever, in the case where the approximations values are close to the exact ones (sequence lengths greater than 2 000, which correspond only to 0.5% of the database), p-values obtained with both methods are highly correlated.

4 Application 2: pattern statistics

In this part, we consider the application of FMCI to pattern statistics. After a short introduction of notations (section 4.1) we explain with an example in section 4.2 how to build through the tool of DFA a particular FMCI related to a given pattern. The block structure of this FMCI (section 4.3) is then used to get in section 4.4 two efficient algorithms for under- and over-represented patterns. We derive in section 4.5 some asymptotic developments but unlike with local score application, these results are not used to improve our algorithms. In the last section 4.6 we finally compare this new method to existing ones.

4.1 Definition

Let us consider a random order m homogeneous Markov sequence $X = X_1,...,X_n$ on the finite alphabet $\mathcal{A}$ (cardinal k). If N_i is the random variable counting the number of occurrences (overlapping or renewal) of a given pattern in $X_1...X_i$. We define the pattern statistic associated to any number $N_{\text{obs}} \in$ of observations by

$$S = \begin{cases} -\log_{10} \mathbb{P}(N_n \geq N_{obs}) & \text{if } N_n \geq [N_n] \\ +\log_{10} \mathbb{P}(N_n \leq N_{obs}) & \text{if } N_n < [N_n] \end{cases} \quad (22)$$

This way, a pattern has a positive statistic if it is seen more than expected, a negative statistic if seen less than expected and, in both cases, the corresponding p-value is given (in log scale) by the magnitude of the statistic.

The problem is: how to compute this statistic ?

4.2 DFA

We first need to construct a Deterministic Finite state Automaton (DFA) able to count our pattern occurrences. It is a finite oriented graph such as all vertexes have exactly k arcs starting from them each one tagged with a different letter of $\mathcal{A}$. One or more arcs are marked as counting ones. By processing a sequence X in the DFA, we get a sequence Y (of vertexes) in which the words of length 2 corresponding to the counting transitions occur each time a pattern occurs in X.

Example: If we consider the pattern aba.a (. means "any letter") on the binary alphabet $\mathcal{A} = \{a, b\}$. We define vertex set $\mathcal{V} = \{a, b, ab, aba, abaa, abab\}$ and then the structure of the DFA counting the overlapping occurrences (set of vertexes and structure would have been slightly different in the renewal case) of the pattern is given by

(the counting arcs are denoted by a star). In the sequence

X = a a b b $\underline{\text{a b a b a}}$ a a b b $\underline{\text{a b a a a}}$ a b

of length n = 20, the pattern occurrences end in positions 9,11 and 18. Processing this sequence into the DFA gives

Y = a a ab b a ab aba $\underline{\text{abab aba}}$ $\overline{\text{abaa a}}$ ab b a ab aba $\underline{\text{abaa a}}$ a ab

which is a sequence of the same length as X, where occurrences of the pattern end exactly in the same positions.

If X is an homogeneous order one Markov chain, so is Y and its transition matrix is given by $P + Q$ where P contains the non counting transitions and Q the counting ones:

$$P = \begin{pmatrix} \mathbb{P}(a|a) & 0 & (b|a) & 0 & 0 & 0 \\ \mathbb{P}(a|b) & (b|b) & 0 & 0 & 0 & 0 \\ 0 & \mathbb{P}(b|b) & 0 & (a|b) & 0 & 0 \\ 0 & 0 & 0 & 0 & \mathbb{P}(a|a) & \mathbb{P}(b|a) \\ 0 & 0 & \mathbb{P}(b|a) & 0 & 0 & 0 \\ 0 & \mathbb{P}(b|b) & 0 & 0 & 0 & 0 \end{pmatrix}$$

and

$$Q = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \mathbb{P}(\mathrm{a}\,|\,\mathrm{a}) & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \mathbb{P}(\mathrm{a}\,|\,\mathrm{b}) & 0 & 0 \end{pmatrix}$$

It is therefore possible to work on Y rather than on X to compute the pattern statistics. In order to do that, it is very natural to use the large deviations (in this case, computations are closely related to the largest eigenvalue of the matrix $T_\theta = P + Qe^\theta$) but other methods can be used as well (binomial or compound Poisson approximations for example).

This method easily extends to cases where X is an order $m > 1$ Markov chain by modifying accordingly our vertex set. For example, if we consider an order $m = 2$ Markov model our vertex set becomes

$\mathcal{V}$ = {aa, ab, ba, bb, aba, abaa, abab}

In all cases, if we denote by L the cardinal of $\mathcal{V}$. In order to count overlapping occurrences of a non degenerate pattern of length h on a size k alphabet we get $L = k + h - 2$ when an order 1 Markov model is considered and $L = k^m + h - m - 1$ for an order $m > 1$ Markov model. For a degenerate pattern of length h, L is more difficult to know as it depends on the degeneracy of the patterns, in the worst case $L = k^{h-1}$, but L should be far smaller in most cases. One should note that L increases by the number of different words present in the pattern if we consider renewal occurrences instead of overlapping ones.

Although construction and properties of DFA are well known in the theory of language and automata ([8]), their connexions to pattern statistics have surprisingly not been extensively studied in the literature. In particular, the strong relation presented here between the FMCI technique for pattern and DFA appears to have never been highlighted before. If this interesting subject obviously need to (and will soon) be investigated more deeply, it is not really the purpose of this article which focus more on the algorithmic treatment of a built FMCI.

4.3 FMCI

Once a DFA and the corresponding matrices P and Q have been built, it is easy to get a FMCI allowing to compute the p-values we are looking for.

Let us consider

$$Z_i = \begin{cases} (Y_j, N_j) & \text{if } N_j < a \\ f & \text{if } N_j \geq a \end{cases} \qquad (23)$$

where Y_j is the sequence of vertexes, N_j is the number of pattern occurrences in the sequence $Y_1...Y_j$ (or $X = X_1...X_j$ as it is the same), where f is the final (absorbing state) and where $a \in$ is the observed number of occurrences N_{obs} if the pattern is over-represented and $N_{\text{obs}} + 1$ if it is under-represented.

The transition matrix of the Markov chain Z is then given by:

$$\Pi = \begin{pmatrix} & (R_{i,j}) & & (v_i) \\ 0 & \dots & 0 & 1 \end{pmatrix} \qquad (24)$$

where for all size L blocks i, j we have

$$R_{i,j} = \begin{cases} P & \text{if } j = i \\ Q & \text{if } j = i+1 \\ 0 & \text{else} \end{cases} \quad \text{and} \quad v_i = \begin{cases} \Sigma_Q & \text{if } i = a \\ 0 & \text{else} \end{cases} \qquad (25)$$

with Σ_Q, the column vector resulting from the sum of Q.

By plugin the structure of R and v in the corollaries 2 and 3 we get the following recurrences:

Proposition 6. For all $n \geq 1$ and $1 \leq i \leq k$ we have

$$\mathbb{P}(N_n < a \,|\, X_1 = i) = \left[u^{n-1}\right]_i \quad \text{and} \quad (N_n \geq a \,|\, X_1 = i) = \sum_{j=0}^{n-2} \left[v^j\right]_i \qquad (26)$$

where for $x = u$ or v we have $\forall j \geq 0$ the following size L block decomposition: $x^j = \left(x^j_{(a-1)}, \dots, x^j_0\right)'$ and we have the recurrence relations:

$$x_0^{j+1} = P x_0^j \quad \text{and} \quad \forall i \geq 1 \quad x_i^{j+1} = P x_i^j + Q x_{i-1}^j \qquad (27)$$

with $u^0 = (1...1)'$ and $v^0 = v$.

4.4 Algorithms

Using the proposition 6 it is possible to get an algorithm computing our pattern statistic for an under-represented pattern observed N_{obs} times:

algorithm 4under: exact statistics for under-represented pattern

$x_0,\ldots, x_{N_{\text{obs}}}$ and $y_0,\ldots, y_{N_{\text{obs}}}$ are $2 \times (N_{\text{obs}} + 1)$ real column vectors of size L

initialization for $j = 0\ldots N_{\text{obs}}$ do $x_j = (1,\ldots,1)'$

main loop for $i = 1\ldots(n - 1)$ do

- for $j = 0\ldots N_{\text{obs}}$ do $y_j = x_j$
- $x_0 = P \times y_0$
- for $j = 1\ldots N_{\text{obs}}$ do $x_j = P \times y_j + Q \times y_{j-1}$

end • $q = \sum_{i=1}^{k} \mu_i \left[x_{N_{\text{obs}}} \right]_i$

- return $\log_{10}(q)$

If we consider now an over-represented pattern we get

algorithm 4over: exact statistics for over-represented pattern

$x_1,\ldots, x_{N_{\text{obs}}}$, $y_1,\ldots, y_{N_{\text{obs}}}$ and z are $2N_{\text{obs}} + 1$ real column vectors of size L

initialization $z = (0,\ldots,0)'$, $x_1 = \Sigma_Q$ and for $j = 2\ldots N_{\text{obs}}$ do $x_j = (0,\ldots,0)'$

main loop for $i = 1\ldots(n - 2)$ do

- for $j = 1\ldots N_{\text{obs}}$ do $y_j = x_j$
- $x_1 = P \times y_1$
- for $j = 2\ldots N_{\text{obs}}$ do $x_j = P \times y_j + Q \times y_{j-1}$
- $z = z + x_{N_{\text{obs}}}$

end • $p = \sum_{i=1}^{k} \mu_i \left[z \right]_i$

- return $-\log_{10}(p)$

As we have $O(k \times L)$ non zero terms in $P + Q$, the complexity of both of these algorithms is $O(k \times L + N_{\text{obs}} \times L)$ in memory and $O(k \times L \times n \times N_{\text{obs}})$ in time.

To compute p-values out of floating point range (ex: smaller than 10^{-300} with C double), it is necessary to use log computations in the algorithms (not detailed here). The resulting complexity stays the same but the empirical running time is obviously slower. That is why we advise to use log-computation only when it is necessary (for example by considering first a rough approximation).

4.5 Asymptotic developments

In this part we propose to derive asymptotic developments for pattern p-values from their recursive expressions. For under- (resp. over-) represented patterns, the main result is given in theorem 9 (resp. 12). In both cases, theses results are also presented in a simpler form (where only main terms are taken into account) in the following corollaries.

Proposition 7. For any $x = (x_{(a-1)'},\ldots,x_0)'$ and all $\beta \geq 0$ $x^\beta = R^\beta x$ is given by $x_0^\beta = P^\beta x_0^0$ and

$$\forall 1 \leq i < a \quad x_i^\beta = P^\beta x_i^0 + \sum_{j=1}^{\beta} P^{j-1} Q x_{i-1}^{\beta-j} = P^\beta x_i^0 + \sum_{j=1}^{\beta} P^{\beta-j} Q x_{i-1}^{j-1} \qquad (28)$$

Proof. As $x_0^{j+1} = P x_0^j$ for all $j \leq 0$ it is trivial to get the expression of x_0^β. If we suppose now that the relation (28) is true for some i and β then, thanks to the relation (27) we have

$$x_i^{\beta+1} = P x_i^\beta + Q x_{i-1}^\beta \qquad (29)$$

$$= P^{\beta+1} x_i^0 + P\left(\sum_{j=1}^{\beta} P^{\beta-j} Q x_{i-1}^j \right) + Q x_{i-1}^\beta \qquad (30)$$

$$= P^{\beta+1} x_i^0 + \sum_{j=1}^{\beta+1} P^{\beta+1-j} Q x_{i-1}^j \qquad (31)$$

and so the proposition is proved through the principle of recurrence. □

Lemma 8. For all $i \geq 0$ and $a \leq b \in$ and $r > 0$ we define

$$P_a^b(r,i) = \sum_{j=a}^{b} r^j (j-1)^i \qquad (32)$$

If $r \neq 1$ we have for all $i \geq 0$ we have

$$(1-r) P_a^b(r,i) = r^a (a-1)^{i+1} - r^{b+1} b^{i+1} - r \sum_{d=0}^{i-1} C_i^d P_a^b(r,d) \qquad (33)$$

and (case $r = 1$) for all $i \geq 0$ we have

$$(i+1) P_a^b(1,i) = b^{i+1} - (a-1)^{i+1} - \sum_{d=0}^{i-1} C_{i+1}^d P_a^b(1,d) \qquad (34)$$

Proof. Easily derived from the following relation

$$P_a^b(r,i+1) = \sum_{j=a-1}^{b-1} r^{j+1} j^{i+1} = r^a (a-1)^{i+1} - r^{b+1} b^{i+1} + r \sum_{d=0}^{i+1} C_{i+1}^d P_a^b(r,d) \qquad (35)$$

Theorem 9. If P is primitive and admits a diagonal form we denote by $\lambda > \nu$ the largest two eigenvalues magnitude of P by $P^{\infty} = \lim_{i\to+\infty} P^i/\lambda^i$ (a positive matrix) and we get for all $\alpha \geq 1$ and $i \geq 0$

$$x_i^{\beta} = \lambda^{\beta} D_i^{\alpha}(\beta) + O\left(\nu^{\alpha}\beta^i\lambda^{\beta}\right) \quad \forall \beta \geq (i+1)\alpha \tag{36}$$

uniformly in β and where D_i^{α} is a polynomial of degree i which is defined by $D_0^{\alpha}(\beta) = P^{\infty}x_0^0$ and for all $i \geq 1$ by the following recurrence relation:

$$D_i^{\alpha}(\beta) = P^{\infty}x_i^0 + \sum_{j=1}^{\alpha}\lambda^{-j}P^{j-1}QD_{i-1}^{\alpha}(\beta-j) + \sum_{j=i\alpha+1}^{i\alpha}\lambda^{-j}P^{\infty}Qx_{i-1}^{j-1} + \sum_{j=i\alpha+1}^{\beta-\alpha}\lambda^{-1}P^{\infty}QD_{i-1}^{\alpha}(j-1) \tag{37}$$

Proof. See appendix B. □

Corollary 10. With the same assumptions than in the theorem 9, for all $\alpha \geq 1$ and $\beta \geq (i+1)\alpha$ we have

$$x_i^{\beta} = \beta^i\lambda^{\beta}\left(\lambda^{\beta}\frac{\left(P^{\infty}Q\right)^i P^{\infty}x_0^0}{i!\lambda^i} + O\left(\frac{1}{\beta}\right) + O\left(\nu^{\alpha}\right)\right) \tag{38}$$

Proof. Equation (37) and the lemma 8 gives

$$\left[D_i^{\alpha}\right]_i = \sum_{j=i\alpha+1}^{\beta-\alpha}\lambda^{-1}P^{\infty}Q\left[D_{i-1}^{\alpha}\right]_{i-1}(j-1)^{i-1} = \frac{P^{\infty}Q}{i\lambda}\left[D_{i-1}^{\alpha}\right]_{i-1} \tag{39}$$

and the result is then proved by a simple recurrence. □

Proposition 11. For any $x = (x_{(a-1)},\ldots,x_0)'$ and all $\beta \geq 0$, $y^{\beta} = \left(y_{(a-1)}^{\beta},\ldots,y_0^{\beta}\right)' = \sum_{b=0}^{\beta} R^{\beta}x$ is given by $y_0^{\beta} = \sum_{b=0}^{\beta} P^b x_0^0$ and

$$\forall 1 \leq i < a \quad y_i^{\beta} = \sum_{b=0}^{\beta} P^b x_i^0 + \sum_{j=1}^{\beta} P^{j-1}Qy_{i-1}^{\beta-j} = \sum_{b=0}^{\beta} P^b x_i^0 + \sum_{j=1}^{\beta} P^{\beta-j}Qy_{i-1}^{j-1} \tag{40}$$

Proof. Using equation (28) we get

$$y_i^{\beta} = \sum_{b=0}^{\beta} P^b x_i^0 + \sum_{b=0}^{\beta}\sum_{j=1}^{b} P^{b-j}Qx_{i-1}^{j-1} \tag{41}$$

$$= \sum_{b=0}^{\beta} P^b x_i^0 + \sum_{j=1}^{\beta}\sum_{b=0}^{\beta-j} P^{j-1}Qx_{i-1}^b \tag{42}$$

which gives the proposition. □

From this result (very similar to proposition 7) it is possible to get a new theorem

Theorem 12. If P is primitive and admits a diagonal form we denote by $\lambda > \nu$ the largest two eigenvalues magnitude of P by $P^{\infty} = \lim_{i\to+\infty} P^i/\lambda^i$ (a positive matrix) and we get for all $\alpha \geq 1$ and $i \geq 0$

$$y_i^{\beta} = A_i^{\alpha} + \lambda^{\beta}B_i^{\alpha}(\beta) + O\left(\nu^{\alpha}\right) + O\left(\nu^{\alpha}\beta^i\lambda^{\beta}\right) \quad \forall \beta \geq (i+1)\alpha \tag{43}$$

uniformly in β and where A_i^{α} is a constant term defined by

$$A_0^{\alpha} = \sum_{b=0}^{\alpha-1} P^b x_0^0 + \frac{\lambda^{\alpha}}{1-\lambda}P^{\infty}x_0^0 \tag{44}$$

and for all $i \geq 0$ by the following recurrence relation

$$A_i^{\alpha} = \sum_{b=0}^{\alpha-1} P^b x_i^0 + \frac{\lambda^{\alpha}}{1-\lambda}P^{\infty}x_i^0 + \sum_{j=1}^{\alpha} P^{j-1}QA_{i-1}^{\alpha} + \frac{\lambda^{\alpha}}{1-\lambda}P^{\infty}QA_{i-1}^{\alpha} \tag{45}$$

and B_i^{α} is a polynomial of degree i which is defined by

$$B_0^{\alpha}(\beta) = -\frac{\lambda}{1-\lambda}P^{\infty}x_0^0 \tag{46}$$

and for all $i \geq 1$ by the following recurrence relation:

$$B_i^{\alpha}(\beta) = -\frac{\lambda}{1-\lambda}P^{\infty}x_i^0 - \frac{\lambda^{-i\alpha}}{1-\lambda}P^{\infty}QA_{i-1}^{\alpha} + \sum_{j=1}^{\alpha}\lambda^{-j}P^{j-1}QB_{i-1}^{\alpha}(\beta-j) + \sum_{j=1}^{i\alpha}\lambda^{-j}P^{\infty}Qy_{i-1}^{j-1} + \sum_{j=i\alpha+1}^{\beta-\alpha}\lambda^{-1}P^{\infty}QB_{i-1}^{\alpha}(j-1) \tag{47}$$

Proof. Easy to derive from the proof of theorem 9. □

Corollary 13. We have the same assumptions than in the the theorem 12, for all $\alpha \geq 1$ and $\beta \geq (i+1)\alpha$ we have

$$y_i^{\beta} = A_i^{\alpha} - \frac{\lambda}{1-\lambda}\beta^i\lambda^{\beta}\left(\frac{\left(P^{\infty}Q\right)^i P^{\infty}x_0^0}{i!\lambda^i} + O\left(\frac{1}{\beta}\right) + O\left(\nu^{\alpha}\right)\right) + O\left(\nu^{\alpha}\right) \tag{48}$$

Proof. Easy to derive from the proof of corollary 10. □

4.6 Numerical results

We propose here to consider numerical applications of these new FMCI pattern statistics algorithms and to compare their results and performance to exact computations using Simple Recurrences ([17] and [18]) denoted SR from now.

All computations are performed using SPatt-1.2.0 package (see [14] or [15]) on a 2.7 Gz P4 with 512 Mo running the linux 2.6.8 system.

As we can see on table 4, the computational time to obtain the pattern statistics for all simple words of a given length are quite similar with both approaches. One exception:

Table 4: Computational time (in seconds) to get the statistics of all DNA words of length *h* in the HIV complete genome sequence (*n* = 9719) using either simple recurrences of finite Markov chain imbedding and in respect with an order *m* Markov model estimated on the genome.

Markov order word length	*m* = 0			*m* = 1			*m* = 2		
	h = 3	*h* = 4	*h* = 5	*h* = 3	*h* = 4	*h* = 5	*h* = 3	*h* = 4	*h* = 5
SR	3	4	5	61	59	62	104	102	106
FMCI	39	45	52	39	44	52	96	102	113

the Bernoulli case ($m = 0$) where SR are roughly 10 times faster than FMCI computations. This is due to the fact that the Bernoulli case uses simpler recurrences than the ones used for the true Markovian cases ($m \geq 1$). Similar simplifications in the DFA structures can reduce the computational time of FMCI approach in the independent case but they have not been implemented here (as their use is often marginal).

If we consider now degenerate patterns instead of simple words (see table 5), FMCI approach clearly outperforms the SR one. Nevertheless, as considering degenerated patterns roughly multiply their observed number of occurrences by the alphabet size for each inde-termination, the corresponding computational time grows in the same manner which usually limits the use of high degenerated patterns in practical cases.

Another interesting point is the memory requirements of the two approaches. Exact computations using SR have a $O(n + \alpha \times k^{2m})$ memory complexity where n is the sequence length, k the alphabet size, m the Markov model order and α which depends on the convergence rate of the model towards its stationary distribution. As a consequence, SR is difficult to use in practice with $m > 3$ for DNA words or $m > 1$ for protein ones. For FMCI computations, the memory requirements remain very cheap and in practice, any Markov model that fit in memory can be considered.

What about the reliability of the two methods. Once the pattern DFA has been computed, the FMCI algorithms are very simple to implement and have a high numerical stability. On the other hand, SR algorithms are quite more complicated (especially for degenerated patterns) to implement and require to approximate the iterate power of the Markov transition by the stationary distribution for large iterates. Classical convergence issues could result then to some numerical instability when high Markov orders are considered. As a consequence, FMCI results are taken as references from this point.

In table 6 we can see that for p-values larger than 10^{-300} the results given by both methods are exactly the same when we consider order 0 Markov models. As smaller p-values are not well managed by C double precision computation (the exact limit depends on the system), we get wrong results unless log computations are used. Such computations have been implemented for FMCI algorithms (they are quite simple) but not for SR ones (where it is quite more complicated) which explain the differences for patterns at and tcgatc.

When we consider order $m > 0$ Markov models, the numerical approximations done on the iterate power of the transition matrix lead to some errors. For order 1 Markov model, these errors remain quite small, but when order 2 model are considered it is more sensitive. In both cases, the larger the statistic to compute is, the greater the errors made are.

5 Conclusion

We proposed in this paper two general algorithms allowing to compute quickly and in a stable numerical way any p-value that can be imbedded in a finite Markov chain.

Table 5: Computational time (in seconds) to get the statistics of degenerate patterns (the dot means "any letter") occurring 100 times in an order *m* = 1 Markovian sequence of length *n* = 9719 which parameters are estimated on the HIV complete genome sequence using either simple recurrences of finite Markov chain imbedding.

pattern	atgca	at.ca	at..a	a...a
SR	0.60	8.20	2438.43	105 209.12
FMCI	0.51	1.15	3.01	91.80

Table 6: Reliability of pattern statistics. They are computed in respect with an order *m* Markovian sequence of length *n* = 9719 which parameters are estimated on the HIV complete genome. Relative error uses FMCI statistics as reference.

pattern	order	observed	expected	FMCI	SR	relative error
acta	0	106	48.63	+12.208	+12.208	0.0
acta	1	106	47.01	+13.090	+13.079	8.7×10^{-4}
acta	0	26	48.63	-3.567	-3.567	0.0
acta	1	26	47.01	-3.231	-3.230	4.8×10^{-4}
acta	0	6	48.63	-13.856	-13.850	3.7×10^{-4}
acta	1	6	47.01	-13.237	-13.237	3.0×10^{-5}
at	0	50	759.48	-291.610	-291.610	0.0
at	0	25	759.48	-327.214	-318.192	2.8×10^{-2}
at	0	0	759.48	-377.009	-319.607	1.5×10^{-1}
tcgatc	0	185	1.37	+294.997	+294.997	0.0
tcgatc	0	195	1.37	+314.388	+314.388	5.7×10^{-8}
tcgatc	0	205	1.37	+333.931	na	na
acacaa	2	10	6.66	+0.865	+0.855	1.1×10^{-2}
acacaa	2	20	6.66	+4.669	+4.520	3.2×10^{-2}
acacaa	2	60	6.66	+35.751	+33.532	6.2×10^{-2}
acacaa	2	100	6.66	+79.736	+73.451	7.8×10^{-2}

We used these algorithms in two applications: local score on one sequence and pattern statistics.

For local score, the resulting algorithms reduce dramatically the complexity of previously proposed naive ones allowing for the very first time to produce exact computations for practical biological studies (Kyte-Doolittle hydrophobic scale on the Swissprot database). Comparing the results to the classical and very popular Karlin's approximations, it appears that these approximations require long sequences (length greater than 2 000) which can dramatically reduce their range of applicability (only 0.5% of the data in our example). Of course, the exact computations require more time than the approximations, but are nevertheless fast enough (20 p-value per second in our example) to be used in most practicable cases. As a consequence we strongly advise to replace asymptotic approximations by these new exact ones whenever it is possible.

Concerning pattern statistics, the new FMCI algorithms appear to outperform the existing ones ([17]) in all possible ways: far easier to implement, more numerical stability, less memory requirements, as fast as SR for simple words (except in the M0 case, but this is due to a poor implementation of this particular case in FMCI approach) and dramatically faster (up to 1 000 times and more) for degenerated patterns. Even if the SR algorithms remain available in the SPatt package, FMCI ones are now used by default for exact computations.

Appendix A: power of a sub-stochastic matrix

Proposition 14. If P is an order L irreducible sub-stochastic matrix admitting a row-eigenvector basis $(e_1,...,e_L)$ where each e_j, is associated to the eigenvalue λ_j and $|\lambda_1| \geq |\lambda_2| \geq ... \geq |\lambda_L|$ then we have

$$P^i = \lambda^i P^\infty + \lambda_2^i \sum_{j=2}^{L} \left(\frac{\lambda_j}{\lambda_2} \right)^i P_j^\infty \quad \forall i \geq 0 \tag{49}$$

where $\lambda = |\lambda_1| = \lambda_1$, $P^\infty = P_1^\infty$ and $\forall j$

$$P_j^\infty = \begin{pmatrix} \left(I_1 e_j'\right) e_j \\ \vdots \ldots \\ \left(I_L e_j'\right) e_j \end{pmatrix} \text{ with } I_{\ell\ell'} = \left(\mathbb{I}_{\lambda_\ell = \lambda_{\ell'}}\right) \quad \forall 1 \leq \ell \leq L \tag{50}$$

Proof. For any vector $x = \sum_{j=1}^{L} x_j e_j$ we have

$$x \times P^i = \sum_{j=1}^{L} x_j \left(e_j \times R^i \right) = \sum_{j=1}^{L} x_j \lambda_j^i e_j = \lambda_1^i x_1 e_1 + \lambda_2^i \sum_{j=1}^{L} \left(\frac{\lambda_j}{\lambda_2} \right)^i x_j e_j \tag{51}$$

As P is an irreducible, Perron-Frobénius theorem (see [3] for example) assure that λ is real and the sub-stochastic

Table 7:

tag\vertex	a	b	ab	aba	abaa	abab
a	a	a	aba	abaa	a*	aba*
b	ab	b	b	abab	ab	b

property, that $\lambda \leq 1$ (in the particular case where P is also primitive *i.e.* $\exists m$, $P^m > 0$ then $|\lambda_2| < \lambda$) Replacing x by $I\ell$ equation (51) gives the expression of the row ℓ of P^i and the proposition is then proved. □

Corollary 15. If P is an order ≥ 2 irreducible sub-stochastic matrix admitting a diagonal form then there exists a matrix P^∞ such as

$$P^i - \lambda^i P^\infty = O(\nu^i) \quad (52)$$

uniformly in i and where $1 \geq \lambda \geq \nu$ are the two largest eigenvalues magnitudes. In the special case where P is primitive, then $\lambda > \nu$ and $P^\infty = \lim_{i \to +\infty} P^i/\lambda^i$ is a positive matrix.

Proof. Using proposition 14 we get

$$\left| P^i - \lambda^i P^\infty \right| \leq |\lambda_2|^i \sum_{j=2}^{L} \left| \frac{\lambda_j}{\lambda_2} \right|^i \left| P_j^\infty \right| \leq \nu^i \underbrace{\sum_{j=2}^{L} \left| P_j^\infty \right|}_{C} \quad (53)$$

and the corollary is proved. □

Appendix B: proof of theorem 9

Using propositions 7 and 14 we get the result for $i = 0$. Let us prove the theorem by the principle of recurrence. We assume now that the result is true until rank i - 1. Computing x_i^β for all $\beta \geq (i+1)\alpha$ we get

$$x_1^\beta = P^\beta x_i^0 + \underbrace{\sum_{j=1}^{i\alpha} P^{\beta-j} Q x_{i-1}^{j-1}}_{A} + \underbrace{\sum_{j=i\alpha+1}^{\beta-\alpha} P^{\beta-j} Q x_{i-1}^{j-1}}_{B} + \underbrace{\sum_{j=\beta-\alpha+1}^{\beta} P^{\beta-j} Q x_{i-1}^{j-1}}_{C} \quad (54)$$

For all $1 \leq j \leq i\alpha$ we have $\beta - j \geq \beta - i\alpha \geq \alpha$ so we get

$$A = \sum_{j=1}^{i\alpha} \left(\lambda^{\beta-j} P^\infty + O\left(\nu^{\beta-j}\right) \right) Q x_{i-1}^{j-1} \quad (55)$$

$$= \lambda^\beta \sum_{j=1}^{i\alpha} \lambda^{-j} P^\infty Q x_{i-1}^{j-1} + O\left(\nu^{\beta-i\alpha}\right) \quad (56)$$

For all $i\alpha + 1 \leq j \leq \beta - \alpha$ we have $j - 1 \geq i\alpha$ and $\beta - j \geq \beta - i\alpha \geq \alpha$ and so, with the help of lemma 8 we get

$$B = \sum_{j=i\alpha+1}^{\beta-\alpha} \left[\lambda^{\beta-j} P^\infty + O\left(\nu^{\beta-j}\right) \right] Q \left[\lambda^{j-1} D_{i-1}^\alpha(j-1) + O\left(\nu^\alpha \lambda^{j-1} (j-1)^{i-1}\right) \right] \quad (57)$$

$$= \sum_{j=i\alpha+1}^{\beta-\alpha} \lambda^{\beta-1} P^\infty Q D_{i-1}^\alpha(j-1) + \sum_{j=i\alpha+1}^{\beta-\alpha} \lambda^{\beta-j} O\left(\nu^\alpha \lambda^{j-1} (j-1)^{i-1}\right) \quad (58)$$

$$+ \sum_{j=i\alpha+1}^{\beta-\alpha} \lambda^{j-1} O\left(\nu^{\beta-j}\right) + \sum_{j=\alpha+1}^{\beta-\alpha} \nu^{\beta-j} O\left(\nu^\alpha \lambda^{j-1} (j-1)^{i-1}\right) \quad (59)$$

$$= \lambda^\beta \sum_{j=i\alpha+1}^{\beta-\alpha} \lambda^{-1} P^\infty Q D_{i-1}^\alpha(j-1) + O\left(\nu^\alpha \lambda^\beta \beta^i\right) \quad (60)$$

thanks to the recurrence assumption, it is easy to see than B contribute with polynomial terms of degree i (as we have a sum on $O(\beta)$ terms).

For all $\beta - \alpha + 1 \leq j \leq \beta$ we have $j - 1 \geq i\alpha$ so we get

$$C = \sum_{j=\beta-\alpha+1}^{\beta} P^{\beta-j} Q \left[\lambda^{j-1} D_{i-1}^\alpha(j-1) + O\left(\nu^\alpha \lambda^{j-1} (j-1)^{i-1}\right) \right] \quad (61)$$

$$= \sum_{j=1}^{\alpha} P^{j-1} Q \left[\lambda^{\beta-j} D_{i-1}^\alpha(\beta-j) + O\left(\nu^\alpha \lambda^{\beta-j} (\beta-j)^{i-1}\right) \right] \quad (62)$$

$$= \lambda^\beta \sum_{j=1}^{\alpha} \lambda^{-j} P^{j-1} Q D_{i-1}^\alpha(\beta-j) + O\left(\nu^\alpha \lambda^\beta \beta^{i-1}\right) \quad (63)$$

C contributing with polynomial terms of degree i - 1 (as we have a sum on α terms) Summing up all terms we get the result at rank i and the theorem is proved.

Acknowledgements

I would like to warmly thank Sabine Mercier for her kind help and her enthusiastic feedback during the writing of this article.

References

1. Altschul SF, Gish W: **Local alignment statistics.** In *Methods in Enzymology Volume 266*. Edited by: Doolittle RF. Academic Press, London; 1996:460-480.
2. Altschul SF, Bundschuh R, Olsen R, Hwa T: **The estimation of statistical parameters for local score alignment score distributions.** *Nucleic Acids Research* 2001, **29(2):**351-361.
3. Bapat RB, Raghavan TES: **Nonegative matrices and applications.** Cambridge University Press; 1997.
4. Chothia C: **The Nature of the accessible and buried surfaces in proteins.** *J Mol Biol* 1976, **105:**1-14.
5. Fu JC, Koutras MV: **Distribution theory of runs: a Markov chain approach.** *J Amer Statist Assoc* 1994, **89:**1050-1058.
6. Fu JC: **Distribution theory of runs and patterns associated with a sequence of multi-state trials.** *Statistica Sinica* 1996, **6(4):**957-974.
7. Fu JC, Lou WYW: **Distribution Theory of Runs and Patterns and Its Applications: A Finite Markov Chain Approach.** World Scientific, Singapore; 2003.
8. Hopcroft JE, Motwani R, Ullman JD: **Introduction to Automata Theory, Languages, and Computation (second edition).** Addison-Wesley; 2001.
9. Karlin S, Altschul SF: **Methods for assessing the statistical significance of molecular sequence features by using general.** *Proc Nat Acad Sci USA* 1990, **87:**2264-2268.
10. Kyte J, Doolittle RF: **A simple method for displaying the hydropathic character of a protein.** *J Mol Biol* 1982, **157(1):**105-132.
11. Lou WYW: **On runs and longest run tests: A method of nite Markov chain imbedding.** *J Am Statist Assoc* 1996, **91(436):**1595-1601.
12. Mercier S, Daudin J-J: **Exact Distribution for the Local Score of One i.i.d. Random Sequence.** *J Comp Bio* 2001, **8(4):**373-380.
13. Mercier S, Hassenforder C: **Exact distribution for the local score of a Markov chain.** *C R Acad Sci Paris* 2003, **336(10):**863-868.
14. Nuel G: **LD-SPatt: Large Deviations Statististic for Pattern on Markov chains.** *J Comp Biol* 2004, **11(6):**1023-1033.
15. Nuel G: **S-SPatt: Simple Statististic for Pattern on Markov chains.** *Bioinformatics* 2005, **21(13):**3051-3052.
16. Press WH, Teukolsky SA, Vetterling WT, Flannery BP: **Numerical recipes in C.** Cambridge University Press; 1992.
17. Robin S, Daudin JJ: **Exact distribution of word occurrences in a random sequence of letters.** *J App Prob* 1999, **36:**179-193.
18. Robin S, Daudin J-J, Richard H, Sagot M-F, Schbath S: **Occurrence probability of structured motifs in random sequences.** *J Comp Biol* 2002, **9:**761-773.

Chapitre 10

Cumulative Distribution Function of a Geometric Poisson Distribution

Article de Nuel accepté en 2006 par *Journal of Statistical Computation and Simulation*. Il y est proposé une nouvelle formule de récurrence permettant le calcul de la distribution, de la fonction de répartition et de la fonction de survie d'une loi de Poisson géométrique avec une complexité linéaire en temps et constante en espace là où le meilleur algorithme précédemment connu était pire que cubique en temps et quadratique en espace. Une application au calcul de statistiques de mots avec cette approche est présenté. Le logiciel *cpspatt* utilise ces algorithmes.

Cumulative Distribution Function of a Geometric Poisson Distribution

Abstract

The geometric Poisson distribution (also called Pólya-Aeppli) is a particular case of the compound Poisson distribution. We propose to express the general term of this distribution through a recurrence formula leading to a linear algorithm for the computation of its cumulative distribution function. Practical implementation with a special care for numerical computations is proposed and validated. The paper ends with an example of application of these results for the computation of pattern statistics in biological sequences modelized by a Markov model.

Keywords: compound Poisson, Pólya-Aeppli, confluent hypergeometric function, Kummer

1 INTRODUCTION

The geometric Poisson distribution (or Pólya-Aeppli) is a particular case of the classical compound Poisson distribution where the contribution of each term is distributed according to the geometric distribution. Johnson *et al* (1972) proposes a linear formula to compute the general term of a compound Poisson distribution which can be simplified (remaining linear) in the geometric Poisson case. This formula results in a quadratic complexity for the computation of the cumulative distribution function (cdf) which is often too slow for many problems.

We propose here to give a direct and straightforward proof for this formula and, rewriting this result with a Kummer's confluent geometric func-

tion, to derive a recurrence relation from it leading to a linear computation of the cdf.

In a first part we will recall some definitions and elementary results, then the recurrence relation will be established. The practical implementation of the resulting algorithm will be then discussed and an application to the computation of pattern statistics on random text generated by a Markov chain will be finally presented.

2 RECALLS

We first recall the

Definition 1 (Poisson distribution) *The random variable M is distributed according to $\mathcal{P}(\lambda)$, the Poisson distribution of parameter $\lambda > 0$, if*

$$\mathbb{P}(M = k) = e^{-\lambda}\frac{\lambda^k}{k!} \quad \forall k \in \mathbb{N}$$

and the

Definition 2 (compound Poisson distribution) *The random variable N is distributed according to $\mathcal{CP}\left((\lambda_k)_{k\in\mathbb{N}^*}\right)$, the compound Poisson distribution of parameter $(\lambda_k)_{k\in\mathbb{N}^*}$ such as all $\lambda_k > 0$ and $\sum_{k=1}^{\infty}\lambda_k = \lambda < \infty$ if*

$$N = \sum_{m=1}^{M} K_m$$

where $M \sim \mathcal{P}(\lambda)$ is independent from the K_m which are independent and identically distributed according to

$$\mathbb{P}(K = k) = \frac{\lambda_k}{\lambda} \quad \forall k \in \mathbb{N}^*$$

This last definition can be easily rewritten as such

Lemma 3 *If $N \sim \mathcal{CP}\left((\lambda_k)_{k\in\mathbb{N}^*}\right)$ with $\sum_{k=1}^{\infty}\lambda_k = \lambda$ then we have $\forall n \in \mathbb{N}^*$*

$$\mathbb{P}(N = n) = \sum_{m=1}^{n} e^{-\lambda}\frac{\lambda^k}{k!}\sum_{k_1,\ldots,k_m\in\mathbb{N}^*}\mathbb{I}_{\{k_1+\ldots+k_m=n\}}\frac{\lambda_{k_1}\times\ldots\times\lambda_{k_m}}{\lambda^m} \qquad (1)$$

and $\mathbb{P}(N = 0) = e^{-\lambda}$.

We are now ready to introduce the

Definition 4 (geometric Poisson distribution) *The random variable N is distributed according to $\mathcal{GP}(\lambda, \theta)$ the geometric Poisson distribution parameter $\theta \in]0, 1]$ for the geometric part and parameter $\lambda > 0$ for the Poisson part, if $N \sim \mathcal{CP}\left((\lambda_k)_{k\in\mathbb{N}^*}\right)$ with*

$$\lambda_k = \lambda(1-\theta)^{k-1}\theta \quad \forall k \in \mathbb{N}^* \tag{2}$$

(in the degenerate case where $\theta = 1$, we simply fall back to the Poisson distribution) and we recall that

$$\mathbb{E}[N] = \frac{\lambda}{\theta} \quad \textit{and} \quad \mathbb{V}[N] = \frac{\lambda(2-\theta)}{\theta^2}$$

We can see on Fig. 1 the pdf of this distribution for some choices of parameter θ with λ chosen such as the expectation $\lambda/\theta = 10$. As pointed out in the definition, we are in the Poisson case with $\theta = 1$. When the aggregative parameter θ decreases, we can see that the tail distribution becomes heavier.

As stated by Johnson *et al* (1972, pp. 378),the geometric Poisson distribution is also defined by

Proposition 5 *If $N \sim \mathcal{GP}(\lambda, \theta)$ then for all $n \geqslant 1$ we have*

$$\mathbb{P}(N = n) = \sum_{k=1}^{n} e^{-\lambda}\frac{\lambda^k}{k!}(1-\theta)^{n-k}\theta^k\binom{n-1}{k-1}$$

were $\binom{n}{k}$ is the binomial coefficient, and $\mathbb{P}(N = 0) = e^{-\lambda}$.

proof. Using relations (1) and (2) we get $\forall n \in \mathbb{N}^*$

$$\mathbb{P}(N = n) = \sum_{m=1}^{n} e^{-\lambda}\frac{\lambda^k}{k!}(1-\theta)^{n-k}\theta^k \underbrace{\sum_{k_1,\ldots,k_m\in\mathbb{N}^*} \mathbb{I}_{\{k_1+\ldots+k_m=n\}}}_{A(n,k)}$$

so we only have to compute $A(n, k)$, the number of way to make n with a sum of k non negative integers (where the order matters). Let us consider the list $1, 2, \ldots, n$. This list contains $n-1$ commas, and a selection of $k-1$ of these commas will obviously give a partition of n such as those involved in $A(n, k)$. It is therefore clear that $A(n, k) = \binom{n-1}{k-1}$ and the proposition is proved. ■

3 MAIN RESULTS

From now, $N \sim \mathcal{GP}(\lambda, \theta)$ with $\lambda > 0$ and $\theta \in]0, 1[$. It is then possible to rewrite the proposition 5 using the Kummer's M confluent hypergeometric function (see Abramovitz and Stegun (1972, chapter 13) ;all further x.x.x references will concern this book).

Proposition 6 *$\forall n \in \mathbb{N}^*$ we have*

$$\mathbb{P}(N = n) = e^{-\lambda}(1-\theta)^n z e^{-z} M(n+1, 2, z) \tag{3}$$

where M is the Kummer's M confluent hypergeometric function and

$$z = \frac{\lambda\theta}{1-\theta} \tag{4}$$

proof. We start by considering $e^{-z}M(n+1, 2, z) = M(-n+1, 2, -z)$ (thanks to 13.1.27). The definition (13.1.2) easily gives that

$$\begin{aligned} M(-n+1, 2, -z) &= 1 + \frac{(n-1)}{2}\frac{z^1}{1!} + \frac{(n-1)(n-2)}{2\times 3}\frac{z^2}{2!} + \dots \\ &= \sum_{k=1}^{n-1} \frac{(n-1)\dots(n-k+1)}{k!}\frac{z^{k-1}}{(k-1)!} \\ &= \sum_{k=1}^{n-1} \binom{n-1}{k-1}\frac{z^{k-1}}{k!} \end{aligned}$$

and so,

$$\mathbb{P}(N = n) = \sum_{k=1}^{n-1} e^{-\lambda}(1-\theta)^n \frac{z^k}{k!}\binom{n-1}{k-1}$$

which gives the result by replacing z by its value. ∎

It is now finally time to establish our recurrence relation

Proposition 7 (recurrence) *We have*

$$\mathbb{P}(N = 0) = e^{-\lambda} \quad \textit{and} \quad \mathbb{P}(N = 1) = e^{-\lambda}(1-\theta)z$$

and $\forall n \in \mathbb{N}, n \geqslant 2$

$$\mathbb{P}(N = n) = \frac{(2n-2+z)}{n}(1-\theta)\mathbb{P}(N = n-1) + \frac{(2-n)}{n}(1-\theta)^2\mathbb{P}(N = n-2)$$

where z is defined in equation (4)

proof. Using the definition 13.1.2, it is easy to show that $M(2,2,z) = e^z$ and this immediately gives $\mathbb{P}(N = 1)$. 13.4.1 also implies the following relation $\forall a, b \in \mathbb{R}$

$$M(a+1,b,z) = \frac{1}{a}\left[(2a - b + z)M(a,b,z) + (b-a)M(a-1,b,z)\right]$$

we simply apply this result to $a = n$ and $b = 2$ to obtain

$$M(n+1,2,z) = \frac{(2n-2+z)}{n}M(n,2,z) + \frac{(2-n)}{n}M(n-1,2,z)$$

and thanks to the relation (3) the proposition is proved. ■

4 ALGORITHMS

In this part, we want to compute for all $n_0 \in \mathbb{N}$ the following functions:

$$F(n_0) = \sum_{n=0}^{n_0} \mathbb{P}(N = n) \quad \text{and} \quad G(n_0) = 1 - F(n_0 - 1) = \sum_{n=n_0}^{\infty} \mathbb{P}(N = n)$$

Thanks to proposition 7, we can compute all $(\mathbb{P}(N = n))_{n \leqslant n_0}$ in $O(n_0)$ but we must face two numerical issues:

1. some terms could be out of the machine range and set to zero;
2. the relative error in the computation of a F close to one, becomes an absolute error for the corresponding G computed through the relation $G = 1 - F$.

A simple solution for the first point is to compute the $L_n = \ln \mathbb{P}(N = n)$ by adapting proposition 7 to obtain

Proposition 8 (log recurrence)

$$L_0 = -\lambda \quad \textit{and} \quad L_1 = -\lambda + \ln\left((1-\theta)z\right)$$

and $\forall n \in \mathbb{N}, n \geqslant 2$

$$L_n = L_{n-1} + \ln\left(\frac{(2n-2+z)}{n}(1-\theta) + \frac{(2-n)}{n}(1-\theta)^2 \exp(L_{n-2} - L_{n-1})\right)$$

A simple application of this proposition leads to the

Algorithm for lower tail

initialization $L_0 = -\lambda$, $L_1 = -\lambda + \ln((1-\theta)z)$, $A_1 = -\lambda$ and $S_1 = 1 + (1-\theta)z$

main loop for $n = 2 \dots n_0$ do

- compute L_n from L_{n-1} and L_{n-2} through the proposition (8)
- check range of sum $S = S_{n-1} + \exp(L_n - A_{n-1})$
  - if $S \in]0, \infty[$ then $A_n = A_{n-1}$ and $S_n = S$
  - else $A_n = A_{n-1} + \ln(S_{n-1})$ and $S_n = 1 + \exp(L_n - A_n)$

end $F(n_0) = \exp(A_{n_0}) \times S_{n_0}$

(where A_i and S_i are auxiliary variables used to compute the cumulative sum).

The second problem is illustrated by the fig (2). As one can see, using F to compute $G = 1 - F$ is acceptable so long as the value of G is not too small ($> 10^{-12}$) but when its value comes closer to the relative error of the computations (10^{-16} for double precision on 32bit machine), the results are no longer reliable.

The solution to this problem is obviously to compute G directly using the serie but, doing so, how does one know when the serie has converged ? One could think to use a rough Gaussian approximation to test the convergence. Unfortunately, the geometric Poisson distribution differs so much from Gaussian one (for example, with $\lambda = 10.0$ and $\theta = 0.1$, $\mathbb{P}(N > 1000) = 8.765 \times 10^{-24}$ while the Gaussian approximation gives 5.161×10^{-95} for the same value) that this option must be abandoned.

Without any better criterion, we will add terms until we get a numerical convergence. We first choose a small ε (typically $\varepsilon = 10^{-16}$ depending on the relative precision of the computations) and we get the

Algorithm for upper tail

initialization $L_0 = -\lambda$ and $L_1 = -\lambda + \ln\left((1-\theta)z\right)$

first loop for $n = 2 \dots n_0$ do

- compute L_n from L_{n-1} and L_{n-2} through the proposition (8)

sum initialization $A_{n0} = L_{n_0}$ and $S_{n_0} = 1$

main loop while the serie has not converged do

- $n = n + 1$
- compute L_n from L_{n-1} and L_{n-2} through the proposition (8)
- check convergence
  - if $L_n < \ln(\varepsilon) + A_{n-1} + \ln(S_{n-1})$ then the serie has converged and $n_1 = n - 1$, go to **end**
  - else the serie has not yet converged
- check range of sum $S = S_{n-1} + \exp(L_n - A_{n-1})$
  - if $S \in]0, \infty[$ then $A_n = A_{n-1}$ and $S_n = S$
  - else $A_n = A_{n-1} + \ln(S_{n-1})$ and $S_n = 1 + \exp(L_n - A_n)$

end $G(n_0) = \exp(A_{n_1}) \times S_{n_1}$

(where, like in the previous algorithm, A_i and S_i are auxiliary variables).

Of course, such a naive approach could lead to errors if the convergence rate of the serie is too low, but as we can see on Fig. 3 the algorithm works very well with, in the worst case, a relative error around 10^{-12}.

Finally, let us point out that the complexity of these two algorithms is $O(n_0)$ in time and $O(1)$ in memory (as only the last two values of L_n and the last value of A_n and S_n are needed at each step).

5 APPLICATION

We consider an homogeneous stationary Markov chain X of length ℓ on

the finite alphabet $\mathcal{A}$ (ex: $\mathcal{A} = \{\mathtt{a}, \mathtt{c}, \mathtt{g}, \mathtt{t}\}$ when DNA sequences are considered). We count N the number of occurrences of a given (non degenerate) pattern (ex: pattern `aacaa` or pattern `gctgtctg`).

Crysaphinou and Papastavridis (1988) first introduced the geometric Poisson approximation for N in the independant case. Arratia *et al* (1990) then proposed to use Chen-Stein method to get a compound Poisson approximation in the Markov case. This idea was then studied in details by Schbath (1995) and, more recently, Robin (2002) pointed out that the compound Poisson distribution was in fact a simple geometric Poisson one in the non degenerate case.

According to these references, N is approximately distributed according to a $\mathcal{GP}(\lambda, 1-a)$ distribution where

$$a = \mathbb{P}(\text{pattern self-overlaps})$$

and

$$\lambda = \ell(1-a) \times \mathbb{P}(\text{pattern appears at a a given position in } X)$$

If $a = 0$, then $\theta = 1 - a = 1.0$ and the distribution is a simple Poisson one (and its cumulative distribution function is easily computed with the incomplete gamma function, see Press *et al* (1992),chapter 6, for more details on this point). From now, we will then consider only cases where $a > 0$.

Let us consider the complete genome of the bacteria *Escherichia coli* K12 (of length $\ell = 4\,639\,221$). We estimate an order 1 Markov model on its sequence to get

$$\Pi = \begin{pmatrix} 0.2957922699 & 0.2247175468 & 0.2082510314 & 0.2712391519 \\ 0.2756564177 & 0.2303218838 & 0.2939007929 & 0.2001209057 \\ 0.2270901404 & 0.3262008455 & 0.2295111640 & 0.2171978501 \\ 0.1857763808 & 0.2342592584 & 0.2824187007 & 0.2975456600 \end{pmatrix}$$

the transition matrix between the states $\{\mathtt{a}, \mathtt{c}, \mathtt{g}, \mathtt{t}\}$ (in this order) such as $\mathbb{P}(\mathtt{g}|\mathtt{c}) = \Pi(\mathtt{c}, \mathtt{g}) = \Pi(2, 3)$. The corresponding stationary distribution μ (such as $\mu\Pi = \mu$) is therefore given by

$$\mu = \begin{pmatrix} 0.2461911663 & 0.2542308865 & 0.2536579603 & 0.2459199869 \end{pmatrix}$$

For any given pattern, we can use Π and μ to compute a and λ. For example, for the pattern gggg we get

$$a(\mathtt{gggg}) = \Pi(\mathtt{g},\mathtt{g}) = 0.2295111640$$

and

$$\lambda(\mathtt{gggg}) = \ell \times \mu(\mathtt{g}) \times \Pi(\mathtt{g},\mathtt{g})^3 \times (1 - a(\mathtt{gggg})) = 10961.53$$

So $N(\mathtt{gggg})$ is approximately distributed according to $\mathcal{GP}(10961.53, 0.7704888)$ and hence $\mathbb{P}(N(\mathtt{gggg}) \leqslant 8704) \simeq 10^{-356.942033} \simeq 1.142791496 \times 10^{-357}$.

The table 1 gives the ten smallest p-values computed with the lower tail algorithm for the self overlapping patterns of size 4 and 8 which are seen less than expected. Table 2 gives the p-values computed with the upper tail algorithm for the self overlapping patterns seen more than expected. In both cases, these p-values are compared with the one obtained through precise large deviations approximations which are, according to Nuel (2006), very close to the exact p-values (usual relative error is smaller than 0.5%).

In both tables, one should note that p-values smaller than 10^{-300} are computed, and, especially for the size 4 patterns, that large n_0 are considered (which corresponding cdf would have taken a very long time to compute with the original quadratic algorithm).

6 CONCLUSION

We have presented here a simple and efficient way to compute the cumulative distribution function of a geometric Poisson distribution. As its complexity is linear, this new algorithm allows to deal with cases where the usual quadratic algorithm was not usable. Thanks to the logarithm version of our recurrence, this algorithm is also capable to compute efficiently very small p-values such as those that usually arise when studying pattern statistics on biological sequences.

One should note that the geometric assumption is here a key stage of the proposed method since it allows to rewrite the complex sum on $k_1, \ldots, k_m \in \mathbb{N}^*$ in lemma 3 in the much more comfortable formula of proposition 5 and then to introduce our recurrence relation. For a general compound Poisson distribution however, it is much more complicated.

Some preliminary work indicates that it seems very unlikely that similar linear algorithms exist in this general case. Nevertheless, a paper should soon complete the work presented here by proposing efficient computations for more general compound Poisson distributions.

REFERENCES

Abramowitz, M., Stegun, I.A. (1972) Handbook of Mathematical Functions. *Wiley-Interscience Publication.*

Arratia R., Goldstein L. and Gordon L. (1990) Poisson approximation and the Chen-Stein method. *Stat. Sci.*, **5(4)**, 403-434.

Chrysaphinou O. and Papastavridis S. (1988) A limit theorem on the number of overlapping appearances of a pattern in a sequence of independant trials. *Proba. Theory Relat. Fields*, **79(1)**, 129-143.

Johnson N. L., Kotz S. and Kemp A. W. (1992) Univariate dis crete distributions. *Wiley, New York.*

Nuel G. (2006) Numerical solutions for Patterns Statistics on Markov chains, *Statistical Applications in Genetics and Molecular Biology.* In revision.

Press W. H., Teukolsky S. A., Vetterling W. T., Flannery B. P. (1992) Numerical Recipes in C, *Cambridge University Press.*

Robin S. (2002) A compound Poisson model for word occurrences in DNA sequences. *J. Roy. Stat. Soc. Ser. C*, **51**, 437-451.

Schbath S. (1995) Compount Poisson approximation of word count in DNA sequences. *ESAIM, Proba. Stat.*, **1**, 1-16.

pattern	n_0	$\mathbb{E}[N]$	a	geometric Poisson	large deviations
cggcgccg	7	168.42	4.86e-04	61.209046	60.881043
cgcatgcg	30	228.91	6.60e-04	60.936502	60.817153
gcgccggc	14	186.50	4.86e-04	60.081156	59.868326
gcgcatgc	47	253.49	6.60e-04	56.379357	56.307102
ggcgccgg	1	131.22	5.97e-04	54.834258	53.992639
cggcgccc	2	131.98	1.12e-04	53.366761	52.783797
ggcgcccg	2	131.69	1.12e-04	53.239547	52.657075
cgcggccg	15	168.42	4.86e-04	51.793270	51.604176
gcatgcgc	54	253.49	6.60e-04	51.490606	51.431959
gggcgccg	3	131.22	1.12e-04	51.397185	50.924344
caca	13000	20139.39	6.19e-02	565.813732	583.968289
tgtg	12830	19764.28	6.13e-02	543.598735	560.747716
gagg	8091	12772.69	1.09e-02	422.828743	430.110448
cctc	8062	12734.96	1.08e-02	422.619815	429.872555
gggg	8704	14226.72	2.30e-01	356.942033	362.994999
cccc	8853	14410.49	2.30e-01	355.737769	361.846613
acac	10964	15898.62	6.19e-02	335.990058	344.540576
gtgt	10843	15678.26	6.13e-02	327.234105	335.456416
acta	6524	9541.95	8.35e-03	231.776187	234.561629
tagt	6723	9586.76	8.40e-03	206.324250	208.824845

Table 1: The ten smallest p-values for the self overlapping patterns of size 4 and 8 seen less in *Escherichia coli* K12 complete genome than expected under an order one Markov model. Form left to right the columns give: the considered pattern, its observed number of occurrences (n_0), its expected number of occurrences (N is the random count of the pattern), its probability to overlap itself (a), its geometric Poisson log scale p-value $-\log_{10}\mathbb{P}(N \leqslant n_0)$ computed with the lower tail algorithm (using $N \sim \mathcal{GP}(\lambda, \theta)$ with $\lambda = \mathbb{E}(N) \times (1-a)$ and $\theta = 1-a$) and the same p-value using precise large deviations approximations.

pattern	n_0	$\mathbb{E}[N]$	a	geometric Poisson	large deviations
ccagcgcc	726	112.32	5.09e-04	322.902186	323.168145
cgctggcg	774	140.62	4.06e-04	299.487521	299.756283
ggcgctgg	661	109.56	4.99e-04	277.585055	277.800101
ccgccagc	663	112.32	9.52e-05	273.664407	273.879114
cgccagcg	733	143.33	4.13e-04	264.740215	264.974535
ccaccagc	510	72.58	6.15e-05	243.503618	243.630897
gctggcgg	613	109.56	9.31e-05	241.376886	241.555145
gctggtgg	499	70.10	5.96e-05	240.683818	240.814004
cttccagc	485	70.96	6.02e-05	226.647714	226.768893
gctggaag	481	69.65	5.92e-05	226.622118	226.741737
cagc	37475	22085.82	1.87e-02	1839.634208	1916.815507
gctg	36516	21695.23	1.84e-02	1744.022252	1815.665597
tggt	23904	16061.70	1.41e-02	701.021635	719.565205
acca	23981	16295.12	1.43e-02	666.399192	684.261519
cgcc	35156	26043.19	2.21e-02	595.751439	622.816682
tgat	27542	19846.44	1.74e-02	557.486921	575.279672
atca	27594	20004.81	1.75e-02	539.060422	556.415209
ggcg	34480	25893.03	2.20e-02	535.115609	559.163108
ggtg	23256	16567.08	1.41e-02	506.236812	520.965865
cacc	23467	16827.26	1.43e-02	492.183339	506.690325

Table 2: The ten smallest p-values for the self overlapping patterns of size 4 and 8 seen more in *Escherichia coli* K12 complete genome than expected under an order one Markov model. Form left to right the columns give: the considered pattern, its observed number of occurrences (n_0), its expected number of occurrences (N is the random count of the pattern), its probability to overlap itself (a), its geometric Poisson log scale p-value $-\log_{10}\mathbb{P}(N \geqslant n_0)$ computed with the upper tail algorithm (using $N \sim \mathcal{GP}(\lambda, \theta)$ with $\lambda = \mathbb{E}(N) \times (1-a)$ and $\theta = 1-a$) and the same p-value using precise large deviations approximations.

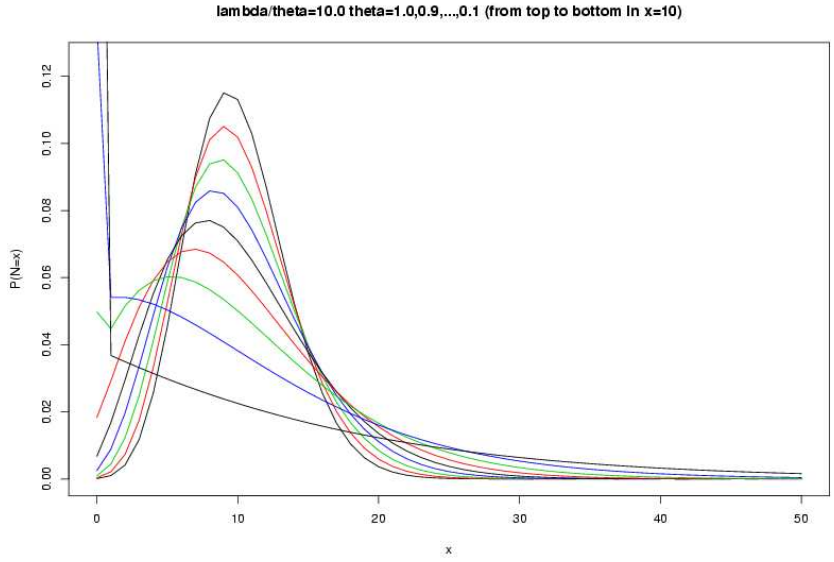


Figure 1: Probability distribution function of $N \sim \mathcal{GP}(\lambda, \theta)$ with $\lambda/\theta = 10.0$ and θ taking several values. From the top to the bottom in abscissa $x = 10.0$, θ equals 1.0 (simple Poisson case), 0.9, 0.8, ..., 0.1.

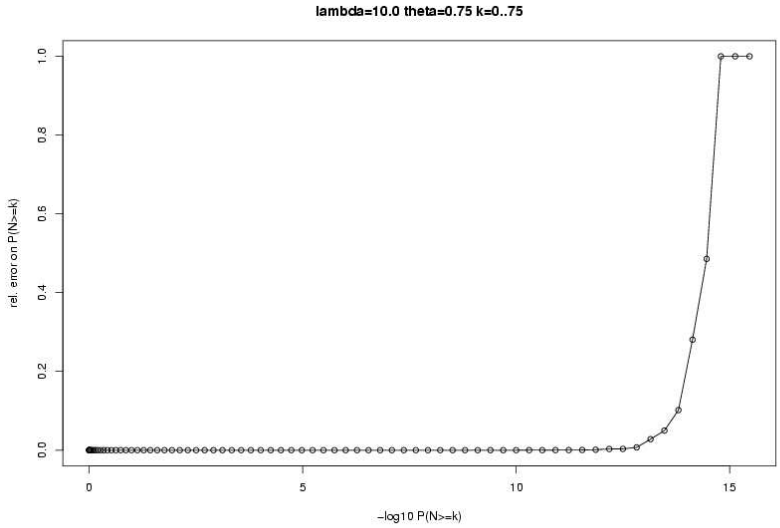


Figure 2: We consider here the distribution of $N \sim \mathcal{GP}(10.0, 0.75)$ ($0 \leqslant k \leqslant 75$). The figure plots the relative error done computing $P(N \geqslant k)$ through $1 - P(N < k)$ using double precision on a 32bit machine (10^{-16} relative precision). The value of $P(N \geqslant k)$ is computed on maple with the same method but using arbitrary precision of 1 000 digits.

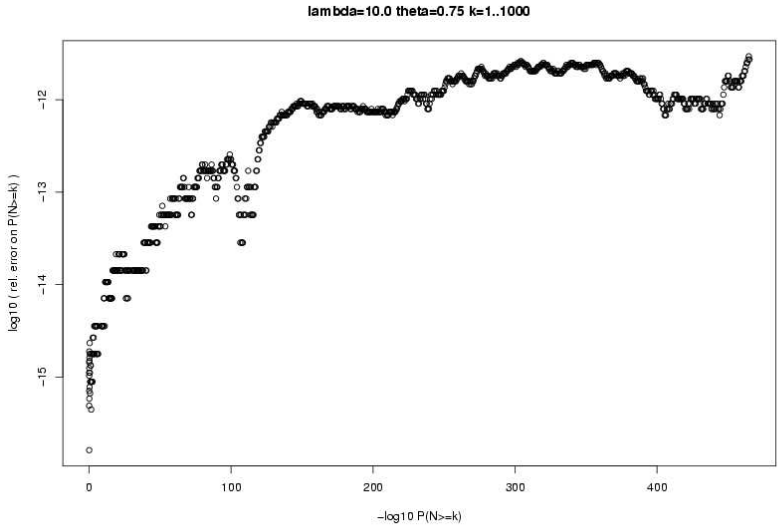


Figure 3: We consider here the distribution of $N \sim \mathcal{GP}(10.0, 0.75)$ $(0 < k \leqslant 1\,000)$. The figure plots the relative error (in log scale) done computing $P(N \geqslant k)$ through the upper tail algorithm (with $\varepsilon = 10^{-13}$) using double precision on a 32bit machine (computational time: 0.13 s). The values of $P(N \geqslant k)$ are computed on maple with the same method but using arbitrary precision of 1 000 digits (computational time: 112 s). Convergence of the serie is achieved with 67 terms (on the left of the plot) to 28 terms (on the right).

Chapitre 11

Numerical Solutions for Patterns Statistics on Markov Chains

Article de Nuel accepté en 2006 par *Statistical Applications in Genetics and Molecular Biology*. Cet article conséquent présente un large panel d'approche concurrentes au problème des motifs en discutant en détail des aspects numériques liés à leur implémentation. Le logiciel SPatt (*Statistics for Pattern*), dans lequel est implémenté l'ensemble de ces méthodes, est également présenté. Grâce à ce logiciel est ensuite conduit la plus complète comparaison numérique de méthodes permettant le calcul de statistiques de motifs. Cet article reprend bien sûr beaucoup de résultats connus mais contient également un grand nombre de nouveautés. L'étude menée conclut au manque de performance des approximations gaussiennes et à la bonne qualité des approximations binomiales (en tant qu'heuristique). Les approximations de Poisson composées et de grandes déviations (précises ou non) se révèle de performances comparables avec une préférence naturelle pour les grandes déviations lorsque l'on considère des événements extrêmes.

Statistical Applications in Genetics and Molecular Biology

Volume 5, *Issue* 1 2006 *Article* 26

Numerical Solutions for Patterns Statistics on Markov Chains

Gregory Nuel*

*Laboratoire Statistique et Genome, CNRS (8071), INRA (1152), UEVE, Evry, France, nuel@genopole.cnrs.fr

Numerical Solutions for Patterns Statistics on Markov Chains*

Gregory Nuel

Abstract

We propose here a review of the methods available to compute pattern statistics on text generated by a Markov source. Theoretical, but also numerical aspects are detailed for a wide range of techniques (exact, Gaussian, large deviations, binomial and compound Poisson). The SPatt package (Statistics for Pattern, free software available at http://stat.genopole.cnrs.fr/spatt) implementing all these methods is then used to compare all these approaches in terms of computational time and reliability in the most complete pattern statistics benchmark available at the present time.

KEYWORDS: exact, Gaussian approximations, large deviations, compound Poisson approximations, benchmark

*I would like to thank Hugues Richard for his theoretical and practical help with the exact methods presented in this paper, and to Mark Hoebeke for his participation to the implementation of SPatt.

1 Introduction

In the framework of sequence analysis (DNA, proteins), the search for patterns of interest is a difficult task. Without biological information, this search is usually done with a sequence or a set of sequences in which we assume the presence of one or more patterns with *unusual behaviour* in terms of frequencies. Unfortunately, composition bias in the sequences as well as patterns overlapping structure could both affect these frequencies which are consequently not reliable for such pattern studies. The classical solution consists in modeling the biological sequences (in most cases, by a Markov model) and then in computing the p-value corresponding to the number of occurrences of a given pattern. Using this approach, it has been possible to confirm existing results and to produce new ones in many biological fields: Chi and restriction sites detection on bacterial genomes (El Karoui et al., 1999; Gelfand and Koonin, 1997; Karlin et al., 1992), transcription factor binding sites (Hampson et al., 2002; Brazma et al., 1998) or analysis of polyadenylation signals (van Helden et al., 2000; Beaudoing et al., 2000).

The problem of assessing p-values for patterns on a Markov chain has been extensively studied in literature. A wide range of exact methods are available: moment-generating functions (Pevzner et al., 1989; Régnier and Szpankowski, 1998; Régnier, 2000), finite state automaton (Atteson, 1998; Nicodème, 2001; Nicodème et al., 2002), embedded Markov chain (Fu and Koutras, 1994; Fu, 1996) and simple recurrence (Robin and Daudin, 1999). All these methods are very efficient on short sequences but, except in few special cases, are not usable for longer sequences or if the Markov model order is greater than 2 or 3. For such cases, we have several possible approximations: binomial (van Helden et al., 1998; Nuel, 2005), Gaussian (Kleffe and Borodovsky, 1992; Cowan, 1991; Prum et al., 1995), compound Poisson (Chrysaphinou and Papastavridis, 1988; Arratia et al., 1990; Geske et al., 1995; Schbath, 1995; Reinert and Schbath, 1998) or large deviations (Nuel, 2004).

The authors of Reinert et al. (2000) proposed an overview of most of these methods, recently improving it in Reinert et al. (2005). In these references, theoretical aspects of the problem are very well covered. The quality of asymptotic approximations such as compound Poisson or Gaussian ones also are discussed on the theoretical level with the conclusion that the former are suitable for rare patterns while the latter are better for the frequent ones.

The aim of the present paper is to complete previous work by considering the pattern problem from a more practical point of view, trying to answer questions such as: What method to implement? How to perform the numerical computations efficiently? What method is best suited for a given pattern problem?

Doing so, we make here a review of most of these statistical approaches to pattern statistics giving for each of them full detail about implementation (including

error control considerations), time and memory complexities, empirical computational time, and empirical relative performances.

Let us note that Robin and Schbath (2001) already did a practical comparison of compound Poisson and Gaussian (shuffle case) approximations to the exact distribution. However, their methodology was focusing more on the global fitness of the approximations rather than on their behaviour in the tail distribution which is of course the more interesting part for testing purposes.

After reconsidering Markov models and the convergence properties of their transition matrix (which is a key-point in many numerical computations), we present a selection of the most efficient pattern methods available at the present time. For each of these methods, we first reconsider known results and then explain in detail how they should be numerically implemented. Some original algorithms or error control results are also presented. We have implemented all these methods in a software package called SPatt (Statistics for Patterns) which is presented and then used to perform the most complete benchmark (in terms of computational time, relative error and rank correlations) available for pattern statistics at the present time.

2 Material and Methods

2.1 Notations

We consider $X = X_1 \dots X_\ell$ a random sequence on $\mathcal{A}$ a size k alphabet. $w = w_1 \dots w_h$ a word (non degenerate pattern) of size h and we denote $w_i^j = w_i \dots w_j$ for all $i \leqslant j$. We define the random count of w by

$$N(w) = \sum_{i=h}^{n} \mathbb{I}_{\{w \text{ ends in position } i\}} \tag{1}$$

If a pattern w is observed n times then its associated pattern statistics is defined by

$$S(w,n) = \begin{cases} -\log_{10} \mathbb{P}(N(w) \geqslant n) & \text{if } n \geqslant \mathbb{E}[N(w)] \\ +\log_{10} \mathbb{P}(N(w) \leqslant n) & \text{if } n < \mathbb{E}[N(w)] \end{cases} \tag{2}$$

which is the log-version of the classical p-value. This way, any over-represented word (seen more than expected) has a positive statistic, while an under-represented word (seen less than expected) has a negative one. The log-scale is also useful dealing with the extreme p-values (*e.g.* 10^{-3000}) we usually get studying patterns and, as we will see with the large deviations, is moreover a natural way to take into account the length ℓ of the sequence.

We also need to introduce $f_x(w)$, the observed frequency of the word w in an observed sequence x and $P(w)$ the probability for w to occur at any given position in the sequence X.

If we consider a degenerate pattern consisting in a finite family of r different words (with none of them included in another) we can define its number of occurrences as the sum of all its words occurrences and then we can extend all the previous notations to the degenerate patterns.

2.2 Markov Model

2.2.1 M0

If the X_i are independent and identically distributed according to μ then we are in the 0-th order Markov model. This model will be denoted M0 (M00 in the special case where all letters have the same probability $1/k$ to occur).

In this model whose parameters are the $\mu(a)$ (for $a \in \mathcal{A}$), the likelihood for an observed sequence $x = x_1, \ldots, x_\ell$ is given by

$$\mu(x_1) \times \ldots \times \mu(x_\ell) = \prod_{a \in \mathcal{A}} \mu(a)^{f_x(a)} \tag{3}$$

The maximum likelihood estimator of this model is therefore given by

$$\widehat{\mu}(a) = \frac{f_x(a)}{\ell} \quad \forall a \in \mathcal{A} \tag{4}$$

In this model, the probability for w to occur somewhere in the sequence is simply given by

$$P(w) = \mu(w_1) \times \ldots \times \mu(w_h) = \prod_{a \in \mathcal{A}} \mu(a)^{f_w(a)} \tag{5}$$

2.2.2 Mm

A natural way to relax the independence assumption is to consider the case where X is a stationary and homogeneous m-th order Markov chain ($m \geqslant 1$) of parameter $\Pi \in [0,1]^{m+1}$ such as $\forall (a,b) \in \mathcal{A}^m \times \mathcal{A}$

$$\mathbb{P}(X_i = b | X_{i-m} = a_1, \ldots X_{i-1} = a_m) = \Pi(a_1 \ldots a_m, b) = \Pi(a,b) \tag{6}$$

The associated stationary distribution μ is defined by the relation $\mu\Pi = \mu$. This model is denoted Mm.

In this model, the likelihood given an observed sequence $x = x_1, \ldots, x_\ell$ is

$$\Pi(x_1^m, x_{m+1}) \times \ldots \times \Pi(x_{\ell-m-1}^{\ell-1}, x_\ell) = \prod_{(a,b)\in\mathcal{A}^m\times\mathcal{A}} \Pi(a,b)^{f_x(ab)} \tag{7}$$

We obtain the following natural maximum likelihood estimator:

$$\widehat{\Pi}(a,b) = \frac{f_x(ab)}{\sum_{c\in\mathcal{A}} f_x(ac)} \quad \forall (a,b) \in \mathcal{A}^m \times \mathcal{A} \tag{8}$$

and we have

$$\begin{aligned} P(w) &= \mu(w_1^m) \times \Pi(w_1^m, w_{m+1}) \times \ldots \times \Pi(w_{h-m-1}^{h-1}, w_h) && (9)\\ &= \mu(w_1^m) \prod_{(a,b)\in\mathcal{A}^m\times\mathcal{A}} \Pi(a,b)^{f_w(ab)} && (10) \end{aligned}$$

Let us point out the fact that the model Mm can be seen as a M1 model on the alphabet $\mathcal{A}^m$, from now, we will denote by $\tilde{\Pi}$ the associated transition matrix defined by

$$\tilde{\Pi}(a,b) = \begin{cases} \Pi(a,b_m) & \text{if } a_2^m = b_1^{m-1} \\ 0 & \text{else} \end{cases} \qquad \forall a,b \in \mathcal{A}^m \tag{11}$$

This k^m-th order matrix is in fact very sparse (only k^{m+1} non zero terms), and of course, any numerical method dealing with such matrix must use this property to speed up the computations. For example, powers of this transition matrix should never be explicitly computed when it is possible to use recursive computation using only efficient matrix-vector products to get the same results.

2.2.3 Choice of the Markov model order

Once these models have been defined, the natural question arises about how to choose the m-th order. It is obvious that higher order models will better fit to the data than low order ones. However, the former will have many more parameters to estimate than the latter so we have to find a trade-off. We thus fall back on the classical framework of model selection where several well-known strategies can be experimented. For example, one can use likelihood ratio to test the choice of a given m against another one. In order to avoid the subsequent multi-testing issue when comparing a set of possible m, another solution consists of using a likelihood criterion including a penalization for the number of estimated parameters (such as BIC or AIC). Let us add that Reinert et al. (2000) also proposed to test for the order through Peason chi-square statistics. All these methods will give similar answers

(but not always exactly the same) and hence provide for a given sequence an optimal order m^* to choose from.

For example purposes let us consider the complete genome of HIV type 1 ($\ell = 10$ kb) and *Escherichia coli* ($\ell = 4.6$ Mb) with the BIC criterion for a m-th order Markov model (3×4^m free parameters):

$$\text{BIC} = -2 \times \text{log-likelihood} + 3 \times 4^m \log(\ell)$$

we get (with the scale: $10^{-3} \times$ BIC)

m	0	1	2	3	4	5	6
HIV type 1	26.4	**25.9**	26.0	27.0	31.6	50.0	128.0
Escherichia coli	12 862	12 743	12 627	12 548	12 508	**12 497**	12 599

where minimal BIC are given in bold. As a consequence, the best order would be $m = 1$ for the HIV sequence and $m = 5$ for the bacterial one.

It is nevertheless important to point out that the method proposed here does not take into account the sensitivity of pattern statistics to parameters estimation, and it is easy to see that this sensitivity could be very important. If we consider w a simple word of length h in a m-th order Markovian model, we can see in formula (10) that computation of $P(w)$ involves the product of $h - m$ terms of Π. Small estimation variability in Π may then be magnified in $P(w)$ and possibly leads to unacceptable variability for the expected number of occurrences. This phenomenon has been investigated in detail in Nuel (2006b) who concluded that, even with long sequences such as the complete genome of *Escherichia coli*, high order (m greater than 4 or 5) should be used with care as parameter estimation will induce so much noise for pattern statistics that the risk of mistakes will be too high.

In our opinion, the best choice for m consists in sticking to orders where estimations are highly accurate (*e. g.* M0 for HIV genome, M2 or M3 for *Escherichia coli*) and to keep in mind that we are taking into account frequencies of size $m + 1$ words in the sequence with such a model. Hence, biological consideration such as: "I need to take into account the codon structure (words of size 3) and hence I want at least $m \geqslant 2$" will certainly be the most pertinent in final analysis.

From now on, Π is considered as a known parameter.

2.2.4 Stationary distribution

Let us first examine this classical theorem:

Theorem 1 (Perron-Frobénius)**.** $\mathbf{A} \in \mathcal{M}_d(\mathbb{R})$ a positive ($\forall i, j, \mathbf{A}(i,j) \geqslant 0$) and primitive matrix ($\exists D, \forall i, j, \mathbf{A}^D(i,j) > 0$) then $\mathbf{A}$ has an eigenvalue $\rho > 0$ of

multiplicity 1, which an associated eigenvector has all its components > 0 and all other eigenvalues are strictly [1] smaller in magnitude than ρ.

We consider now a Markov chain with primitive (resp. irreducible) transition matrix Π and see, using the Perron-Frobénius theorem and the fact that Π is stochastic that a stationary distribution μ exists such as $\mu\Pi = \mu$ which is the left-eigenvector associated to the eigenvalue 1 which is the one with the largest magnitude (resp. largest real part).

The implicitly restarted Arnoldi algorithm (see Lehoucq et al., 1997) allows us to compute only several eigenvalues (and optionally the corresponding eigenvectors), meeting a specific criterion (larger magnitude, larger real part, smaller imaginary part, ...) using only the matrix of interest in matrix-vector products (which is especially suitable for sparse matrices). Using this algorithm, it is therefore possible to compute efficiently the stationary distribution of any Mm Markov model in $O(k^{m+1})$ (number of non zero terms in the transition matrix).

One interesting property of this stationary distribution is given by the

Proposition 2. If Π is a primitive stochastic matrix (with stationary distribution μ) and if Π admits a diagonal form on $\mathbb{C}$ then $\forall a, b$ we have

$$\lim_{\delta\to\infty} \Pi^\delta(a,b) = \mu(b) \quad \text{and} \quad \left|\Pi^\delta(a,b) - \mu(b)\right| \underset{\delta\to+\infty}{\sim} \mathbf{C}(a,b)\lambda^\delta \tag{12}$$

where λ is the largest magnitude of the eigenvalues other than 1 and where $\mathbf{C}$ is a scalar matrix. In the particular case where there is only one eigenvalue of magnitude λ (we call v its corresponding left-eigenvector) then

$$\mathbf{C}(a,b) = \left(E_a V^{-1} E_2'\right)(vE_b) \tag{13}$$

where V is the matrix containing all the left-eigenvectors (with v in second row) and E_c is the column vector filled with zeros except with a one at the c^{th} coordinate.

Proof. We can write any $y = y_1e_1 + y_2e_2 \ldots + y_de_d \in \mathbb{R}^d$ (where $e_1, e_2 \ldots, e_d$ is the canonical basis) into the left-eigenvectors basis $y = z_1\mu + z_2v + z_3v_3 + \ldots + z_dv_d$.

$$[z_1, z_2, z_3, \ldots, z_d] = [y_1, y_2, y_3, \ldots, y_d]\, V^{-1} \tag{14}$$

Thanks to the stochasticity of Π, it is easy to see that the first column of V^{-1} is $\mathbf{1}$ (column-vector filled with ones), so, if $y = E_a$ we have

$$z_1 = \sum_{i=1}^{d} y_i = 1 \quad \text{and } \forall 2 \leqslant i \leqslant d \quad z_i = yV^{-1}E_i' \tag{15}$$

[1] If $\mathbf{A}$ is irreducible ($\forall i, j, \exists D, \mathbf{A}^D(i,j) > 0$) rather than primitive the same theorem holds but equalities in magnitude could happen.

so we get

$$E_a \Pi^\delta E'_b = \mu E'_b + (\lambda_2)^\delta z_2 v E'_b + (\lambda_3)^\delta z_3 v_3 E'_b + \ldots + (\lambda_d)^\delta z_d v_d E'_b \tag{16}$$

hence

$$\Pi^\delta(a,b) = \mu(a) + (\lambda_2)^\delta \left[\left(E_a V^{-1} E'_2\right)(v E'_b) + \left(\frac{\lambda_3}{\lambda_2}\right)^\delta z_3 v_3 E'_b + \ldots + \left(\frac{\lambda_d}{\lambda_2}\right)^\delta z_d v_d E'_b \right] \tag{17}$$

and as

$$\left|\frac{\lambda_i}{\lambda_2}\right| < 1 \quad \forall 3 \leqslant i \leqslant d \tag{18}$$

we immediately get the result. □

With the help of the Arnoldi algorithm, λ can be computed with the same complexity as for the stationary distribution ($O(k^{m+1})$ for a Mm model) but the computation of $\mathbf{C}(a,b)$ requires all the eigenvalues and eigenvectors resulting in a complexity of $O(k^{3m})$ for a Mm model.

Even if the result proposed here requires that Π admits a diagonal form in some basis, this is not restrictive in practice as it will be *always* the case for classical Markovian transition matrix estimated on a real data. Nevertheless, one should consider with caution highly structured or degenerated Markov models as their transition matrix could sometimes fail to meet this condition.

2.2.5 More complex Markov models

The basic Markov model introduced above can be generalized in several interesting ways. Most of the time, it is again possible of seeing the resulting model as a Markov one on an appropriate alphabet. Even if such an approach is possible, it is not always pertinent: a specific adaptation of the statistical methods could save a lot of computational time.

Variable Length Markov Chains (VLMC): With classical m-th order Markov chains, the probability to see a given letter somewhere in the sequence depend on the preceding m letters. For example, if we consider the binary alphabet $\mathcal{A} = \{0,1\}$ and $m = 2$ we need to now if the last two letters were '00', '01', '10' or '11' to be able to draw the next one. However, to get a better fitness with real data, Buhlmann and Wyner (1999) introduced the idea that we may require more or less letters, depending on the current context. With the same example, we would get

different distribution for the next letter if the preceding one is a '1' (order in only 1 in this case) or if the three last ones are '000', '010', '100' or '110' (which correspond to a 3-rd order Markov model. Such a model (called VLMC) can always be seen as a Markov model whose order is equal to the highest length used (denoted m and the VLMC is then said to be of this order) where some some coefficients in the transition matrix are constrained to be equal but such approach may then result in unecessary computations and memory usage. In our example, the highest length would be 3 and if we denote by Π the parameters of the model we get the constraint that for all d, all $\Pi(ab1, d)$ are equal. As proposed by Bourguignon and Robelin (2004), this idea can be extended to what they call Parsimonious Markov Models (PMM) where we do not need to take into account contiguous previous letters anymore. For example, we could say that '1', '$0 \cdot 0$' and '$1 \cdot 0$' (the dot means "any letter") are the contexts to take into account (which would have been impossible with simple VLMC). Once again, PMM can be seen as classical Markov chains once estimations have been done but with the same numerical constraints as in the VLMC case.

phased Markov models: In such models, we take into account the phase of the letter in the sequence. Different Markov models are considered for each phase. For example, such models have been used to take into account the codon structure of coding sequence in the Genmark software (Borodovsky and McIninch, 1993). By construction, the transition matrix of the resulting global Markov model is *not* irreducible and then, the stationary distribution must be computed separately for each phase.

Hidden Markov Models (HMM): These models (see Rabiner, 1989; Durbin et al., 1998) have a hidden underlying structure which is usually not observed (*e.g.* coding or not coding, exon/intron, ...). The estimation of such models is delicate (*e.g.* require EM algorithm) but once this is done, the sequence can be seen as a simple Markov model. As in the phased case, the only problem is stationary distribution. The transition matrix is still irreducible, but the magnitude λ of the second eigenvalue could be very close to 1. Even if the stationary distribution exists, the convergence rate is too slow and any method based on the stationarity of the Markov chain will fail with such models. We definitely have to adapt these methods to the HMM case.

Let us take a small example to illustrate this property. We consider two 1-st order Markov models on the binary alphabet whose transition matrices are

$$\Pi_0 = \begin{pmatrix} 0.3 & 0.7 \\ 0.6 & 0.4 \end{pmatrix} \quad \text{and} \quad \Pi_1 = \begin{pmatrix} 0.5 & 0.5 \\ 0.8 & 0.2 \end{pmatrix}$$

The magnitude $\lambda_0 = \lambda_1 = 0.3$ of the second eigenvalues of both matrices is small

enough to ensure a fast convergence toward their respective stationary distribution. However, if we consider now the Markov model whose transition matrix is given (for a fixed $0 < \varepsilon < 1$) by

$$\Pi(\varepsilon) = \begin{pmatrix} (1-\varepsilon) \times \Pi_0 & \varepsilon \times \Pi_0 \\ \varepsilon \times \Pi_1 & (1-\varepsilon) \times \Pi_1 \end{pmatrix}$$

and we called $\lambda(\varepsilon)$ the magnitude of the second eigenvalue of $\Pi(\varepsilon)$ we see the following table:

ε	0.1	0.01	0.001	0.0001	$\ldots$
$\lambda(\varepsilon)$	0.8	0.98	0.998	0.9998	$\ldots$

that $\lambda(\varepsilon)$ could be very close to 1 for small ε. As a consequence, it could take a very long time for the powers of $\Pi(\varepsilon)$ to converge toward the stationary distribution. For example we have

$$\Pi(0.0001)^{1000} = \begin{pmatrix} 0.4197044 & 0.4896528 & 0.0557694 & 0.0348734 \\ 0.4197044 & 0.4896528 & 0.0557694 & 0.0348734 \\ 0.0418459 & 0.0487969 & 0.5596034 & 0.3497538 \\ 0.0418459 & 0.0487969 & 0.5596034 & 0.3497538 \end{pmatrix}$$

which means that a thousand steps are not enough to insure stationarity.

2.3 Statistical methods

2.3.1 Exact

This class of methods has been extensively studied in literature. Two different approaches are usually considered. The first one consists of the expression of the moment-generating functions of the counting variable from which we can compute moments or p-values using Taylor expansions (overview in Régnier, 2000). With the help of computational tricks, these methods could be numerically efficient in several special cases (*e.g.* 0-th order Markov model), but most of the time they involve heavy formal computations and are not usable when a Markov model of order greater than 2 or 3 is considered or when the sequence length grows too much. Because of this, such methods are often only used to compute moments from which Gaussian or Poisson approximations are derived; this is of course completely different from an exact computation of the pattern statistic we are here looking for. The second, and more direct approach, consist of performing a combinatorial computation of the pattern p-value. Many people have done this in various special cases (0-th order Markov model, binary alphabet, $\ldots$) but the technique of embedded Markov chain (Fu and Koutras, 1994; Fu, 1996) finally gave an elegant and general

framework for these computations. As this powerful technique is usually very slow, it is often speeded up by the use of recurrence formulas. Robin and Daudin (1999) propose such an approach which appears to be, at the present time, the most efficient way of computing exact pattern statistics and we hence have chosen to present this in this paper.

In this section, propositions 3, 4 and 5 (as well as their proofs) can be found in Robin and Daudin (1999). The proposed algorithms and error control results are original work.

For all $i, k \in \mathbb{N}$ with $i > 0$ we need to compute

$$p(i,k) = \mathbb{P}(\text{the } k^{\text{th}} \text{ occurrence of } w \text{ ends in position } i) \tag{19}$$

(and $p(i) = p(i,1)$) in order to get the pattern statistic defined in (2) which is now given by

$$S(w,n) = \begin{cases} -\log_{10}\left(\sum_{i=1}^{\ell} p(i,n)\right) & \text{if } n \geqslant \mathbb{E}[N(w)] \\ +\log_{10}\left(1 - \sum_{i=1}^{\ell} p(i,n+1)\right) & \text{if } n < \mathbb{E}[N(w)] \end{cases} \tag{20}$$

We first introduce for all pairs of words $v \in \mathcal{A}^g$ and $w \in \mathcal{A}^h$ their overlapping function

$$O_{v,w}(d) = \frac{P(v_1^d w)}{P(v)} \times \mathbb{I}_{v_{d+1}^g = w_1^{g-d}} \quad \forall 1 \leqslant d \leqslant g-1 \tag{21}$$

(when $v = w$ we simple denote $O_w = O_{w,w}$)

In M0 case, $p(\ell, n)$ could be computed in $O(h\ell n)$ in time and $O(\ell)$ in memory thanks to the following result:

Proposition 3 (recurrences in the independent case)**.** We obtain the distribution of the first occurrence: $p(i) = 0$ if $i < h$, $p(h) = P(w)$ and for all $i > h$ we have

$$p(i) = p(i-1) - p(i-h)P(w) - \sum_{d=1}^{h-1} [p(i-d) - p(i-d-1)]O_w(d) \tag{22}$$

and then the distribution of the k^{th} occurrence ($k \geqslant 2$): $p(i,k) = 0$ if $i < h + k - 1$ and for all $i \geqslant h + k - 1$ we have

$$\begin{aligned} p(i,k) &= p(i-1,k) + P(w)[p(i-h,k-1) - p(i-h,k)] \\ &+ \sum_{d=1}^{h-1} \{[p(i-d,k-1) + p(i-d-1,k) - p(i-d,k) - p(i-d-1,k-1)] \\ &\qquad \times O_w(d)\} \end{aligned} \tag{23}$$

In the Markovian case the complexity to compute $p(\ell, n)$ grows to $O(\ell^2 n)$ (but memory complexity is still $O(\ell)$) with the following result:

Proposition 4 (complete recurrences in the Markovian case)**.** We obtain the distribution of the first occurrence: $p(i) = 0$ if $i < h$, $p(h) = P(w)$ and for all $i > h$ we have

$$p(i) = P(w) - \sum_{d=1}^{h-1} p(i-h+d) O_w(d) - \sum_{\delta=1}^{i-2h+1} p(i-h-\delta+1) \frac{\tilde{\Pi}^{\delta}(w_{h-m+1}^{h}, w_1^m) P(w)}{\mu(w_1^m)} \quad (24)$$

and then the distribution of the k^{th} occurrence ($k \geqslant 2$): $p(i,k) = 0$ if $i < h + k - 1$ and for all $i \geqslant h + k - 1$ we have

$$p(i,k) = \sum_{d=1}^{h-1} [p(i-h+d, k-1) - p(i-h+d, k)] O_w(d) + \sum_{\delta=1}^{i-2h+1} [p(i-h-\delta+1, k-1) - p(i-h-\delta+1, k)] \frac{\tilde{\Pi}^{\delta}(w_{h-m+1}^{h}, w_1^m) P(w)}{\mu(w_1^m)} \quad (25)$$

Fortunately, it is possible to use the convergence of $\tilde{\Pi}^{\delta}$ by replacing it by the stationary distribution μ for all $\delta > \alpha$ where α is a given integer. It is hence possible to compute $p_\alpha(\ell, n)$, an approximation of $p(\ell, n)$, in $O(\alpha \ell n)$ using the following recurrences:

Proposition 5 (short recurrences in the Markovian case)**.** Assuming that the transition matrix admits a diagonal form (which is not restrictive in practice), we get an approximation for the distribution of the first occurrence: $p_\alpha(i) = 0$ if $i < h$, $p_\alpha(h) = P(w)$, for all $i > h$ we have

$$p_\alpha(i) = p_\alpha(i-1) - \sum_{d=1}^{h-1} [p_\alpha(i-h+d) - p_\alpha(i-h+d-1)] O_w(d) - \mathbb{I}_{\{i \geqslant \alpha+2h\}} p_\alpha(i-h-\alpha) P(w) - \sum_{\delta=1}^{\min(\alpha, i-2h+1)} [p_\alpha(i-h-\delta+1) - p_\alpha(i-h-\delta)] \frac{\tilde{\Pi}^{\delta}(w_{h-m+1}^{h}, w_1^m) P(w)}{\mu(w_1^m)} \quad (26)$$

and then, an approximation for the distribution of the k^{th} occurrence ($k \geqslant 2$) is given by: $p_\alpha(i,k) = 0$ if $i < h + k - 1$ and for all $i \geqslant h + k - 1$ we have

$$\begin{aligned} p_\alpha(i,k) &= p_\alpha(i-1,k) + \mathbb{I}_{\{i \geqslant \alpha+2h\}}[p_\alpha(i-h-\alpha,k-1) - p_\alpha(i-h-\alpha,k-1)]P(w) \\ &+ \sum_{d=1}^{h-1}[p_\alpha(i-h+d,k-1) + p_\alpha(i-h+d-1,k) - p_\alpha(i-h+d,k) + p_\alpha(i-h+d-1,k-1)]O_w(d) \\ &+ \sum_{\delta=1}^{\min(\alpha,i-2h+1)} \{[p_\alpha(i-h-\delta+1,k-1) + p_\alpha(i-h-\delta,k) \\ &- p_\alpha(i-h-\delta+1,k) - p_\alpha(i-h-\delta,k-1)] \times \frac{\tilde{\Pi}^\delta(w_{h-m+1}^h, w_1^m)P(w)}{\mu(w_1^m)}\} \end{aligned} \quad (27)$$

The question of the accuracy of this approximation in arbitrary precision computations is quite difficult to answer. Nevertheless, if we work with fixed precision computations (*e.g.* $\xi = 10^{-16}$), it is sufficient to take a α large enough to have a relative error smaller than the precision ξ when computing $\tilde{\Pi}^\alpha$. The absolute error on this computation is given by $O(\lambda^\alpha)$ where λ is the magnitude of the second eigenvalue of $\tilde{\Pi}$. Using $1/k$ (k is the alphabet size) as a rough approximation of the terms of $\tilde{\Pi}$ we get

$$\alpha \geqslant \frac{\log(\xi) - \log(k)}{\log(\lambda)} \quad (28)$$

Using the recurrences we just established, it is therefore possible to compute $\mathbb{P}(N(w) \geqslant n)$ with a relative error of ξ. Unfortunately, when this probability is close to 1 (case of an under-represented word), $\mathbb{P}(N(w) \leqslant n) = 1 - \mathbb{P}(N(w) \geqslant n+1)$ is computed with an absolute (not a relative) error of the same order. As a consequence, pattern statistics for under-represented words could be wrong if the corresponding p-value is small enough.

We propose here two ways to compute our p-values with a required precision ε (*e.g.* $\varepsilon = 10^{-5}$) we assume to be larger than ξ the standard precision of the machine computations. We need to start with a rough computation s of $S(w,n)$ (using binomial or compound Poisson approximations for example). If $s \geqslant \log_{10}\xi - \log_{10}\varepsilon$, then the pattern statistic could computed directly with $\mathbb{P}(N(w) \leqslant n)$ (over-represented case) or $1 - \mathbb{P}(N(w) \geqslant n+1)$ (under-represented case). If $s < \log_{10}\xi - \log_{10}\varepsilon$ then we can use either of these two algorithms:

arbitrary precision Use arbitrary precision computations with a precision of $\varepsilon \times 10^s$ (α must also be chosen so that the approximation is not sensitive with this precision level, which, as said above, is not easy to do) and proceed in the same manner as before.

tail sum compute $\mathbb{P}(N(w) \leqslant n) = \sum_{i=\ell+1}^{\ell_0} p(i, n+1)$ with the usual precision level where ℓ_0 is chosen such as

$$\frac{\mathbb{P}(N(w) \leqslant n)}{\mathbb{P}(N_0(w) \leqslant n)} < \varepsilon \tag{29}$$

where $N_0(W)$ count occurrences of W on a random sequence of length ℓ_0 (this can be done through the same rough approximation used before computing s).

Of course, these algorithms depend on the reliability of the rough approximations, but these values only serve to control errors, and as error on error are usually small, we expect the method to be efficient.

In the case of degenerate patterns, the probabilities of seeing the k^{th} occurrence must be computed for each of the r words belonging to the pattern. These computations require the probability of seeing a couple of words separated by a gap with no occurrence of the pattern. The resulting recurrence formulas are similar to the one presented above but are much more complicated to express and implement (full details in Robin et al., 2002). The final complexity in this case increases by a factor r^2 both in time and memory in comparison with the non degenerate complexities.

The program X-SPatt (eXact Statistics for Patterns) from the package SPatt (see section 2.5 for more details) implements all these algorithms.

2.3.2 Gaussian

Kleffe and Borodovsky (1992) first propose formulas for the first and second moments of a word count on a Markov chain. These formulas are very simple to compute in the independent case but, as we will see below, are much more complicated to compute in the Markov case because we need the powers of the transition matrix. Cowan (1991) (for the expectation) and then Prum et al. (1995) (for the variance) propose a solution to this problem by computing these moments thanks to Whittle's formula (Whittle, 1955) in the case where the sequence is generated according to a shuffle model. When the parameter of a m-th order Markov model is estimated on an observed sequence (and when the sequence is long), the resulting model is very close to the m-th order shuffle model[2], and so, the moments computed with the formulas from Prum et al. (1995) are a good approximation of the solution. Unfortunately, the sequences are not always long enough to use this approach and moreover, in some cases, the Markov model parameter is given and not

[2] drawn uniformly in the set of sequence where length and counts for all size m words are exactly the same than in the original one.

estimated from a sequence (*e. g.* random parameter for simulation purpose, or parameter estimated on a set of sequences and used as background model on another one). For these reasons, we decided to focus only on the Markov case rather than on the shuffle one.

In this section, proposition 6 and corollary 7 are known results (Kleffe and Borodovsky, 1992) but the rest is original work.

We consider the first two moments of $N(w)$. The expectation is given by:

$$\mathbb{E}[N(w)] = E(w) = (\ell - h + 1)P(w) \tag{30}$$

For any m-th order model ($m \geqslant 1$), the variance can be computed in $O(\ell k^{3m})$ in time (and $O(k^{2m})$ in memory) using the following result:

Proposition 6. For any words v (of size g) and w (of size $h \leqslant g$) we have

$$\mathbb{C}ov\left[N(v), N(w)\right] = M(v,w) + O(v,w) - E(v)E(w) \tag{31}$$

where the main part ($2 \times (\ell - g - h + 1)$ terms) is given by

$$M(v,w) = \sum_{\gamma=0}^{\ell-g-h} (\ell - g - h - \gamma + 1)\left(\Pi_\gamma(v,w) + \Pi_\gamma(w,v)\right) \tag{32}$$

where the probability to see w after v and a gap of length γ is given by

$$\Pi_\gamma(v,w) = P(v)\frac{\tilde{\Pi}^\gamma(v_{g-m+1}^g, w_1^m)}{\mu(w_1^m)}P(w) \tag{33}$$

and the overlapping part ($2g - 1$ terms) by

$$O(v,w) = \sum_{i=1}^{g-1}(\ell - h - i + 1)P(v_1^i w) + \sum_{i=1}^{h-1}(\ell - g - i + 1)P(w_1^i v) + (\ell - g + 1)\sum_{i=1}^{g-h+1} P(w)\mathbb{I}_{\left\{w=v_i^{i+h-1}\right\}} \tag{34}$$

Proof. For any word w (of size h_w), we introduce the following notation for $h \leqslant i \leqslant \ell$

$$I_i(w) = \mathbb{I}_{\{w \text{ end in position } i\}} \tag{35}$$

and for any words v (of size g) and w (of size $h \leqslant g$) we denote

$$\mathbb{E}\left[I_i(v)I_{i+\delta}(w)\right] = D_\delta(v,w) \tag{36}$$

for all $\delta \in \mathbb{Z}$ and $\max(g, h-\delta) \leqslant i \leqslant \min(\ell, \ell-\delta)$ which do not depend on i. We get then that

$$\mathbb{E}[N(v)N(w)] = \sum_{i=g}^{\ell}\sum_{j=h}^{\ell} \mathbb{E}\left[I_i(v)I_j(w)\right] = \sum_{\delta=h-\ell}^{\ell-g} \eta_\delta D_\delta(v,w) \tag{37}$$

and all we have to do is to work with the different possible values of δ to get the result. □

In the special case of a 0-th order Markov model, the computation can be done in $O(h)$ with the

Corollary 7. For any words v (of size g) and w (of size $h \leqslant g$) we have

$$\begin{aligned}\mathbb{C}ov\left[N(v), N(w)\right] = O(v,w) + \\ \left(\ell - \ell g - \ell h + g^2 - 2g + h^2 - 2h + gh + 1\right) P(v)P(w)\end{aligned} \tag{38}$$

where $O(v,w)$ is exactly the same than in proposition 6.

Proof. We simply replace $\Pi_\gamma(v,w)$ and $\Pi_\gamma(w,v)$ by $P(v)P(w)$ in proposition 6. □

But thanks to the convergence of $\tilde{\Pi}^\delta$ toward the stationary distribution μ and if we assume that $\tilde{\Pi}$ admits a diagonal form (which is not restrictive in practice) we have

Lemma 8.

$$\begin{aligned}|M(v,w) - M_\alpha(v,w)| \underset{\alpha\to+\infty}{\sim} \left(\frac{\mathbf{C}(v_{g-m+1}^g, w_1^m)}{\mu(w_1^m)} + \frac{\mathbf{C}(w_{h-m+1}^g, v_1^m)}{\mu(v_1^m)}\right) P(v)P(w) \\ \times \frac{\lambda^{\alpha+1}}{(1-\lambda)}(\ell - g - h - \alpha)\left[1 - \frac{\lambda(1-\lambda^{\ell-h-g-\alpha})}{(1-\lambda)(\ell-g-h-\alpha)}\right]\end{aligned} \tag{39}$$

where $M_\alpha(v,w) = M(v,w)$ if $\alpha \geqslant \ell - g - h$ and

$$\begin{aligned}M_\alpha(v,w) = \sum_{\gamma=0}^{\alpha-1}(\ell - g - h - \gamma + 1)\left(\Pi_\gamma(v,w) + \Pi_\gamma(w,v)\right) \\ + (\ell - g - h - \alpha + 1)(\ell - g - h - \alpha)P(v)P(w)\end{aligned} \tag{40}$$

else.

Proof. This lemma comes easily from the proposition 2 and the following result:

$$\sum_{\gamma=\alpha+1}^{\ell-g-h} (\ell-g-h-\gamma+1)\lambda^\gamma = \frac{\lambda^{\alpha+1}}{(1-\lambda)}(\ell-g-h-\alpha) - \frac{\lambda^{\alpha+2}}{(1-\lambda)^2} + \frac{\lambda^{\ell-g-h+2}}{(1-\lambda)^2} \quad (41)$$

□

Using this lemma, we can compute an approximation of $\mathbb{C}ov\left[N(v), N(w)\right]$ in $O(\alpha k^{3m})$ in time (and still $O(k^{2m})$ in memory) by controlling the error done. As α is usually far smaller than ℓ, this approach improves dramatically the computations in the Markov case.

It is now possible to use these expectations and covariances to compute any word or degenerate pattern statistics. If we denote by Φ the cumulative distribution function of Gaussian random variable of mean 0 and variance 1 we get the following asymptotic approximation:

$$S(w,n) \simeq \begin{cases} -\log_{10}\left(1-\Phi\left[Z(w,n)\right]\right) & \text{if } w \text{ is over-represented} \\ \log_{10}\Phi\left[Z(w,n)\right] & \text{if } w \text{ is under-represented} \end{cases} \quad (42)$$

with

$$Z(w,n) = \frac{n-\mathbb{E}[N(w)]}{\sqrt{\mathbb{V}ar[N(w)]}} \quad (43)$$

In order to choose the value of α we can use the following result:

Proposition 9. The relative error done on the p-values using M_α instead of M is given by

$$\varepsilon \simeq \mathbf{C}(w_{h-m+1}^h, w_1^m) \times \frac{P(w)^2}{\mu(w_1^m)} \times \frac{Z^2}{V} \times \frac{\lambda^{\alpha+1}\ell}{(1-\lambda)} \quad (44)$$

where $\mathbf{C}$ is the matrix defined in proposition 2 and with $Z = Z(w,n)$ and $V = \mathbb{V}ar[N(w)]$.

Proof. This is simply done by first order Taylor expansions and the fact that $\left|\frac{z\Phi'(z)}{\Phi(z)}\right| \simeq z^2$ for all $z<0$ (replace $\Phi(z)$ by $1-\Phi(z)$ for $z>0$). □

Using the rough approximation that $V \simeq \mathbb{E}[N(w)]$ (we get an equality when we approximate the distribution of $N(w)$ by a Poisson distribution, see section 2.3.4) we hence get

Corollary 10.

$$\varepsilon = O\left(\frac{P(w)^2}{\mu(w_1^m)} \times \left(\frac{n - \mathbb{E}[N(w)]}{\mathbb{E}[N(w)]}\right)^2 \times \frac{\lambda^{\alpha+1}\ell}{(1-\lambda)}\right) \tag{45}$$

As the covariance between any pair of words is available, it is easy to compute the variance of any linear combination of words and thus the variance of any pattern. For a pattern containing r words, the resulting complexity is r^2 times more than in the case of a single word.

The program RMES (Schbath, 1997) allows the computation of the variance of any pattern in the special case of the shuffle model. The program G-SPatt (Gaussian Statistics for Patterns) from the package SPatt (see section 2.5 for more details) implements the Markov case.

2.3.3 Large Deviations

As we are focusing on tail distribution events, it could be efficient to use a statistical method dedicated to their study such as the large deviations theory. Nuel (2004) first proposed a numerical method to produce large deviations patterns statistics from well known results on Markov chains. Unfortunately, the complexity of this method grows exponentially with the length of the pattern ($O(k^h)$ in time and memory) making its use difficult in many cases. Inspired by the embedded Markov chain technique (Fu and Koutras, 1994; Fu, 1996), we propose here a dramatic improvement of the method.

Let us denote by $\mathcal{P}(w)$ (resp. $\mathcal{S}(w)$) the set of all the prefixes (resp. suffixes) of w. Assuming the random sequence X is generated by a M0 or M1 model, we consider the alphabet

$$\mathcal{L} = \mathcal{A} \cup \left(\mathcal{P}(w) \cap \left(\mathcal{A}^2 \cup \ldots \cup \mathcal{A}^{h-1}\right)\right) \tag{46}$$

(if $h = 2$ then $\mathcal{L} = \mathcal{A}$).

The following proposition is easy to establish:

Proposition 11. We define $Y_1 = X_1$ and for all $i \geqslant 1$:

$$Y_{i+1} = \text{longest word of } \mathcal{S}\left(Y_i X_{i+1}\right) \cap \mathcal{L}$$

and get the following properties:

i) $Y = Y_1 \ldots Y_\ell$ is a Markov chain whose transition matrix T is defined for all $a \in \mathcal{A}$, x such as $xa \in \mathcal{L}$ and $y \in \mathcal{L}$ by

$$T(xa, y) = \begin{cases} \Pi(a,b) & \text{if } \exists b \in \mathcal{A} \text{ such as the longest word of } \mathcal{S}(xab) \cap \mathcal{L} \text{ is } y \\ 0 & \text{else} \end{cases} \tag{47}$$

ii) The word w occurs exactly the same number of times in sequence X as the size 2 word $\alpha\beta$ in sequence Y, where α is the size $h-1$ prefix of w and β is the longest suffix of w belonging to $\mathcal{L}$.

Example 12. If $\mathcal{A}=\{0,1\}$, $w=01001$ and

$$X=0,1,0,0,0,1,0,1,0,0,1,1,0,0,1 \quad (\ell=15) \tag{48}$$

then we have $\mathcal{L}=\{0,1,01,010,0100\}$ so

$$Y=0,\beta,010,\alpha,0,\beta,010,\beta,010,\alpha,\beta,1,0,0,\beta \tag{49}$$

with $\alpha=0100$ and $\beta=01$. We see in X one occurrence of w ending in position 11 and one occurrence of $\alpha\beta$ ending in the same position in Y. The transition matrix is defined by

$$T=\left(\begin{array}{c|ccccc} & 0 & 1 & \beta=01 & 010 & \alpha=0100 \\ \hline 0 & \Pi(0,0) & 0 & \Pi(0,1) & 0 & 0 \\ 1 & \Pi(1,0) & \Pi(1,1) & 0 & 0 & 0 \\ \beta=01 & 0 & \Pi(1,1) & 0 & \Pi(1,0) & 0 \\ 010 & 0 & \Pi(0,1) & 0 & 0 & \Pi(0,0) \\ \alpha=0100 & \Pi(0,0) & 0 & \Pi(0,1) & 0 & 0 \end{array}\right) \tag{50}$$

By a simple application of the classical Cramer-Chernov for Markov chains (see Bucklew (1990) or Den Hollender (2000) for the regular form and see theorem 3.7.4 — lattice case — in Dembo and Zeitouni (1998) for the precise form we present here) we get

Theorem 13. For all $1>a\geqslant\mathbb{E}\left[\frac{N(w)}{\ell_h}\right]$ (respectively, $0<a<\mathbb{E}\left[\frac{N(w)}{\ell_h}\right]$) we have for large ℓ_h

$$\mathbb{P}\left(\frac{N(w)}{\ell_h}\geqslant a\right)=\frac{e^{-\ell_h\Lambda^*(a)}}{\sqrt{2\pi V(a)\ell_h}}\times K(a)\times\big(1+o(1)\big) \quad \text{(respectively, } <\text{)} \tag{51}$$

where $\forall\theta\in\mathbb{R}$ and $\forall a\in]0,1[$

$$V(a)=\frac{\left[(\Lambda^*)'(a)\right]^2}{(\Lambda^*)''(a)} \quad \text{and} \quad K(a)=\frac{(\Lambda^*)'(a)}{1-e^{-(\Lambda^*)'(a)}} \tag{52}$$

with

$$\Lambda^*(a)=\sup_{\theta\in\mathbb{R}}\{\theta.a-\Lambda(\theta)\} \quad \text{and} \quad \Lambda(\theta)=\ln\rho(\theta) \tag{53}$$

where $\rho(\theta)$ is the largest eigenvalue of the matrix T_θ defined for all $x,y\in\mathcal{L}$ by

$$T_\theta(x,y)=\begin{cases} T(x,y)e^\theta & \text{if } xy=\alpha\beta \\ T(x,y) & \text{else} \end{cases} \tag{54}$$

And it is therefore possible to compute our pattern statistic with

Corollary 14. If $a \times \ell_h = n \geqslant \mathbb{E}[N(w)]$ (for the under-represented case, simply change the sign of the $S(w,n)$) we get the following approximation:

$$S(w,n) \simeq \frac{\ell_h \times \Lambda^*(a)}{\ln(10)} \tag{55}$$

and if evaluations of the functions V and K are available we have the more precise approximation:

$$S(w,n) \simeq \frac{\ell_h \times \Lambda^*(a)}{\ln(10)} + \frac{1}{2}\log_{10}(2\pi V(a)\ell_h) - \log_{10} K(a) \tag{56}$$

In order to compute these approximations, we first need the transition matrix T which can be constructed in $O(L^2 \times k \times h)$ (where $L = k + h - 2$ is the cardinal of $\mathcal{L}$), then we need to evaluate the function Λ^*. Nuel (2004) proposes using the Brent algorithm for the maximisation and the Arnoldi algorithm (see Lehoucq et al. (1987)) to evaluate the function Λ. Doing so, the overall complexity of an evaluation of Λ^* is $O(L \times k)$ in time and memory, where $L \times k$ is the number of non-zero terms in T.

Computing V and K requires a bit more work. We propose to do this by getting a classical C-th order Chebyshev approximation (with $C = 2$ for example) of Λ^* and then computing its first and second derivatives thanks to this approximation. As the Chebyshev approximation requires C evaluations of Λ^*, the overall complexity of the computation of V is $O(C \times L \times k)$ (but memory is still $O(L \times k)$).

Some remarks:

a) In the special case where $a = 0$, it is not possible to use the precise approximations. Nevertheless, if we denote by

$$R = \begin{pmatrix} T_{-\infty} & 0 \\ * & 1 \end{pmatrix} \tag{57}$$

(where $*$ is set such as R is stochastic) the embedded Markov chain with an absorbing state for the pattern occurrences, we know (Fu and Koutras, 1994; Fu, 1996, according to) that $\mathbb{P}(N(w) = 0) = 1 - R^{\ell_h}(w,w)$. As the stationary distribution of R is obviously $(0,\ldots,0,1)$, the quantity we are looking for is exactly equal to the absolute error between R^{ℓ_h} and the corresponding stationary distribution. We know that this error (see 2.2.4) is well approximated by λ^{ℓ_h} where λ is the magnitude of the second eigenvalue of R. According to the structure of the transition matrix R, λ is the largest eigenvalue of $T_{-\infty}$ which is exactly $\Lambda^*(0)$ (thanks to elementary properties of the function Λ).

b) The same method easily extends to a set $\mathcal{W}$ of patterns: just replace w by $\mathcal{W}$ (for example, $y \neq w$ becomes $y \notin \mathcal{W}$). In this case, L, the size of $\mathcal{L}$ is larger (but still smaller than $k + k^2 + \ldots + k^{h-1} \simeq k^h$ in the worst case,h being the size of the longest word in $\mathcal{W}$). In this case however, occurrences of the pattern correspond to occurrences of a size 2 *pattern* (rather than word) in the sequence Y.

c) It is also possible to extend these results to m-th order Markov model with $1 < m < h$ (if $m \geqslant h$ the problem, while degenerated, still has a solution but we will not present it here). In this case, we replace the definition of $\mathcal{L}$ by

$$\mathcal{L} = \mathcal{A}^m \cup \left(\mathcal{P}(w) \cap \left(\mathcal{A}^{m+1} \cup \ldots \cup \mathcal{A}^{h-1}\right)\right) \tag{58}$$

and we just have to replace $a \in \mathcal{A}$ by $a \in \mathcal{A}^m$ in proposition 11ii). In this case, the the size of $\mathcal{L}$ simply changes to $L = k^m + h - m - 1$.

d) If we want to count renewal occurrences of pattern instead of possibly overlapping ones, we must adapt our method by adding w to the set $\mathcal{L}$ and to force transitions to $\mathcal{A}$ in w line of T. This way, the number of occurrences of the letter w in Y (changing the definition of T_θ accordingly) is exactly the renewal count of w in X.

e) The non zero terms of the matrix transition T gives the structure of a deterministic finite state automaton allowing a counting of the pattern occurrences. For, example with the pattern $w = 01001$ we get the following automaton:

state	0	1
$\emptyset$	0	1
0	0	01
1	0	1
01	010	1
010	0100	1
0100	0	(01)

how to use it? Start from state $\emptyset$ and process all letters of the sequence changing state according to the structure of the automaton. Each time the counting transition (from $\alpha = 0100$ to $\beta = 1$) is used, w occurs in the sequence. In the case of a degenerated pattern, we have several counting states, but the principle remains the same. One should note that the sequence of states is then exactly the sequence Y.

For example, processing sequence X given in (48) in this automaton uses transitions $\emptyset$ to 0, then 0 to 01 (which is also denoted β), then 01 to 010 and so on, until we get the full sequence Y given in (49).

f) It is also possible to use these approximations to get the statistic of any linear combination (with any real weights) of pattern counting functions. In this case, it is likely that the lattice correction $K(a)$ will have to be removed.

g) All these results hold good when hidden or phased Markov chains are considered (at the cost of larger set $\mathcal{L}$).

h) Structured patterns (*e.g.* two patterns separated by a given gap) can also be studied by similar methods. In fact, we can derive statistics for any pattern which can be recognized by a deterministic finite state automaton (*i. e. any regular expression)*.

i) Let us finally add that formula (55) clearly states that our log-scale pattern statistics linearly depend on the sequence length. As a consequence, dividing pattern statistics by the sequence length provides an efficient and very simple way to get normalized statistics. Using this normalization, it is therefore possible to easily perform comparisons between sequences of different lengths.

The program LD-SPatt (Large Deviations Statistics for Patterns) from the package SPatt (see section 2.5 for more details) implements all these algorithms.

2.3.4 Binomial

According to equation (1), counting pattern occurrences is simply done by summing the indicatrices of its presence all along the sequence. If we assume that these indicatrices are independent from each other, we then get a binomial approximation for the number of occurrences. Unfortunately, this assumption is obviously completely false. How ever, van Helden et al. (1998) (by simulations) and more recently Nuel (2005) (by comparison to other statistical methods) have shown that this approximation is a very good heuristic for pattern statistic computations. The only case where binomial approximations seem to be completely false concerns patterns with a high overlapping structure (*e.g.* `aaa` or `gggggg`).

If we consider a size h pattern w, $P(w)$ is its probability to occur at a given position in the random sequence which depends (as seen in section 2.2) on the stationary distribution. Hence

$$N(w) \sim \mathcal{B}\left[\ell - h + 1, P(w)\right] \tag{59}$$

and so the pattern statistic corresponding to an observation n is given by the cumulative distribution function of the binomial. As explained in Press et al. (1997), the numerical evaluation of this CDF can be done efficiently through a continued

fraction representation of the incomplete beta function with the complexity $O(\sqrt{\ell})$ in the worst case.

As we usually have ℓ large and $P(w)$ small, our binomial approximation is also very close to a Poisson distribution so we have

$$N(w) \sim \mathcal{P}\left[(\ell - h + 1)P(w)\right] \tag{60}$$

Using this distribution, it is possible to get the pattern statistic through the incomplete gamma function which can be quickly evaluated thanks to another continued fraction representation (Press et al., 1997).

Degenerate patterns can be easily treated in the same way, we just have to sum the probabilities to any of the words it contains. As a consequence, the resulting complexity for a pattern containing r words is r times more than for a simple word.

The program S-SPatt (Simple Statistics for Patterns) from the package SPatt (see section 2.5 for more details) implements this method.

2.3.5 Compound Poisson

We have seen in the last section that binomial (or Poisson) approximations for pattern counts are not reliable in the case of self-overlapping patterns (*e.g.* `accacc`). This is due to the fact that such patterns could occur by clumps (*e.g.* `accaccacc` is a clump of 2 occurrences of `accacc`). Remark: this problem vanishes if we consider renewal counting instead of classical overlapping ones.

We consider w a pattern and denote by $\mathcal{C}_\kappa$ the set of clumps containing exactly κ occurrences of w. We define then the conditional probability for w to overlap itself by

$$a = \sum_{c \in \mathcal{C}_2} \frac{P(c)}{P(w)} \tag{61}$$

and the

Proposition 15. Any clump c has a probability

i) $(1-a)P(w)$ to occur

ii) $a^{\kappa-1}(1-a)$ to belongs to $\mathcal{C}_\kappa$

Proof. In order to see a clump c we need to see one occurrence of the pattern which do not belongs to a precedent clump and so i) is proved. By the definition of a, $\mathbb{P}(c \in \mathcal{C}_1) = 1 - a$ and ii) is then proved by a simple recurrence. □

Corollary 16. $N(w)$ is approximately distributed according to a geometric Poisson distribution of parameter $(\ell - h + 1)(1-a)P(w)$ for the Poisson part and $1-a$ for the geometric part.

Proof. if we denote by M the (random) number of clumps and by K the (random) size of a given clump we have

$$N(w) = \sum_{j=1}^{M} K_j \quad \text{where } K_j \text{ (sample of } K\text{) and } M \text{ are independent} \tag{62}$$

Thanks to the last proposition, K is distributed according to a geometric distribution of parameter $1-a$. Assuming that indicatrices of clumps are independent (which is reasonable as clumps do not overlap each other) M is distributed according to a $\mathcal{B}[\ell - h + 1, (1-a)P(w)]$ and hence approximately distributed according to a Poisson distribution of parameter $(\ell - h + 1)(1-a)P(w)$. So finally we get here the exact definition of a geometric Poisson distribution (also called Pólya-Aeppli, see Johnson et al. (1992) for more details) and comes the corollary. □

Chrysaphinou and Papastavridis (1988) first introduced this approximation in the case of 0-th order Markov chains. Arratia et al. (1990) proposed the idea to use Chen-Stein Poisson approximations to get a compound Poisson distribution for pattern counts in the Markov case. This idea was then studied in detail by Schbath (1995) and, more recently, Robin (2002) pointed out that the resulting compound Poisson approximation was in fact a simple geometric Poisson distribution (in the non degenerate case).

It is possible to derive pattern statistics from this approximation by computing the CDF of a geometric Poisson distribution. This can be done in $O(n^2)$ where n is the observed number of occurrences using the definition given in Johnson et al. (1992). More recently, Nuel (2006a) proposed a recurrence formula using confluent hypergeometric function to compute the same CDF in $O(n)$ in time (and constant memory complexity).

In the case of degenerate patterns of size r, Reinert and Schbath (1999) showed that r^2 clumps must be considered (one for each couple of words belonging to the pattern) and that the resulting distribution is no longer a geometric Poisson distribution but a true compound Poisson distribution. Thanks to the formula proposed in Barbour et al. (1992), the corresponding CDF is computable in more than $O(n^3)$ in time (and a quadratic memory complexity).

One should note that in the case of non overlapping patterns ($a = 0$), our geometric (or even compound) Poisson distribution is in fact a simple Poisson distribution and we fall back to the approximation given in equation (60).

The program RMES (Schbath, 1997) allows the computation of these statistics for patterns but uses the worse than cubic algorithm rather than the linear one. Moreover, as pointed out in Nuel (2004), RMES suffers severe numeric implementation issue and its results should be considered with care. On the other hand, the

program CP-SPatt (Compound Poisson Statistics for Patterns) from the package SPatt (see section 2.5 for more details) implements the new linear algorithm in a fast and stable numeric form.

2.3.6 Algorithmic complexities

	$r=1$		$r>1$	
method	$m=0$	$m \geqslant 1$	$m=0$	$m \geqslant 1$
exact	$h\ell n$	$\alpha\ell n$	$r^2 h\ell n$	$r^2\alpha\ell n$
Gaussian	h	αk^{3m}	$r^2 h$	$r^2\alpha + k^{3m}$
large deviations	k^2+kh	k^m+kh	$(k^2+rkh)^*$	$(k^{m+1}+rkh)^*$
binomial	h	h	rh	rh
compound Poisson	hn	hn	r^3hn^3	r^3hn^3

Table 1: Order of magnitude of the time complexity: k size of the alphabet, m order of the Markov model, ℓ length of the sequence, n number of observed occurrences, h size of the pattern, r number of word in the pattern. α is the largest power iterate of the transition matrix computed; its value depends mostly on the magnitude of the largest second eigenvalue of the transition matrix (see details in the exact and Gaussian sections). $(^*)$ this complexity is always smaller than $O(k^h)$.

Table 1 and table 2 give the time and the memory complexities for the computation of the (log-scale) p-value associated to an observation for a given pattern and for each of the statistical methods available. As we can see, the binomial statistics are the fastest to compute (and with the lowest memory cost) in all cases. Large deviations seem also very efficient but one should note that the computation of their statistics requires very sophisticated linear algebra (Arnoldi algorithm) while the other methods use mostly simple algebra operations. Exact and compound Poisson methods involve either the length of the sequence or the number of observed occurrences (or both) in their complexities and thus, are expected to be slower than all others when the length of the considered sequence grows.

2.4 Statistic comparison

We want to compare the reliability of all these methods (exact, Gaussian approximations, large deviations, binomial/Poisson approximations and compound Poisson approximations). As we are usually interested by patterns with low p-values, we consider only the most under- and over-represented ones. Two criterion are taken into account.

	$r=1$		$r>1$	
method	$m=0$	$m \geqslant 1$	$m=0$	$m \geqslant 1$
exact	ℓ	ℓ	$r^2\ell$	$r^2\ell$
Gaussian	h	$h+\alpha+k^{2m}$	r^2h	$r^2h+r^2\alpha+k^{2m}$
large deviations	k^2+kh	k^m+kh	$(k^2+rkh)^*$	$(k^{m+1}+rkh)^*$
binomial	1	1	1	1
compound Poisson	1	1	r^2+n^2	r^2+n^2

Table 2: Order of magnitude of the memory complexity: k size of the alphabet, m order of the Markov model, ℓ length of the sequence, n number of observed occurrences, h size of the pattern, r number of word in the pattern. These complexities do not include the memory requirement for the Markov model parameter ($O(k^{m+1})$ for a m-th order model).

The first one is the mean of the relative error on the pattern statistics:

$$\text{rel. error} = \left| \frac{S - S_{\text{ref}}}{S_{\text{ref}}} \right| \tag{63}$$

where a given statistic is taken as a reference. One should note that the relative error on the p-values is $|S_{\text{ref}}|$ times larger than the relative error on the statistics.

The second criterion is the rank correlation given by the Kendall's tau (Press et al., 1997) which is defined by the following:

$$\text{Kendall's tau} = \frac{\text{concordant} - \text{discordant}}{\text{concordant} + \text{discordant}} \tag{64}$$

where we call a pair of patterns concordant if they are ranked the same with the two methods and discordant if they are not (in the case where we have ties, the definition of the Kendall's tau is slightly different, check Press et al. (1997) for details). This rank correlation is equal to 1.0 when we have a perfect accordance between the two statistics and is equal to -1.0 when it is perfect discordance. In our cases, Kendall's tau are always positive and can be interpreted as the probability that a given pair of patterns is ranked the same by both statistics.

2.5 Software

SPatt (Statistic for Patterns) is a C++ implementation of all the statistics methods proposed in this paper. The aim of the package is to provide state of the art high quality pattern statistic computations.

SPatt feature list:

- GNU standard command line interface (high scriptability);
- support for any custom defined alphabets (DNA, protein, latin, ...) through a simple syntax;
- Markov model parameters can be either estimated on a sequence or entered by users;
- fast computations even for very long patterns using determinist finite state automaton to count occurrences;
- complex linear algebra algorithms providing fast and reliable computation of stationary distribution even for high order Markov model
- support of very low p-values (lower than 10^{-300}) that usually arise when studying pattern statistics;

SPatt can work with a set of patterns or with all words of a given size. The package also have a web interface (called WebSPatt) so it is possible to try it even without any local installation. Let us add that SPatt is freely distributed according to the General Public Licence (GPL).

Here is the homepage of the project:
`http://stat.genopole.cnrs.fr/spatt/`

All the computations in this paper are done on an Intel P4 2.8 Gz with 512 Mo of RAM system using Linux 2.6.8 as OS and SPatt 1.1.3.

3 Results and Discussion

3.1 Second magnitude

As explained in the section 2.2.4, the second magnitude eigenvalue (λ) of the Markov model transition matrix gives the convergence speed toward the stationary distribution. The table 3 gives the exact value of this λ as well as a number of necessary steps to achieve convergence at machine precision level. The results are given for a sample of achaebacteria (the two first ones), bacteria (the four middle ones) and eucaryota (the two last ones). There is no obvious specific behaviour from a phylum to another but a simple rule seems to arise:

$$\frac{m}{10} \leqslant \lambda \leqslant \frac{m+1}{10} \tag{65}$$

where m is the Markov model order.

organism	1	2	3	4	5	6
A. fulgidus	0.16 (21)	0.27 (29)	0.51 (55)	0.53 (59)	0.65 (86)	0.72 (113)
A. pernix	0.08 (15)	0.27 (29)	0.45 (47)	0.55 (62)	0.62 (78)	0.71 (108)
M. genitallium	0.16 (21)	0.28 (29)	0.40 (41)	0.47 (49)	0.54 (60)	0.65 (86)
B. subtilis	0.10 (16)	0.28 (29)	0.43 (44)	0.52 (57)	0.52 (57)	0.49 (52)
E. coli K12	0.09 (16)	0.25 (27)	0.48 (51)	0.50 (54)	0.55 (62)	0.64 (83)
L. lactis	0.12 (18)	0.16 (21)	0.39 (40)	0.42 (43)	0.56 (64)	0.67 (92)
C. elegans Chr 1	0.15 (20)	0.30 (31)	0.38 (39)	0.42 (43)	0.49 (52)	0.57 (66)
S. cerevisiae Chr 1	0.07 (14)	0.23 (26)	0.44 (45)	0.46 (48)	0.51 (55)	0.64 (83)

Table 3: Second magnitude λ of the transition matrix for different order m (in columns) and different organisms (in rows). For each λ the corresponding α from which the absolute error between Π^α and μ is lower than 10^{-16} is given between parenthesis.

3.2 Computational time

word size	2	3	4	5	6	7	8	9
EMBOSS	3.90	4.29	5.15	7.31	16.32	57.03	230.68	891.43
SPatt	0.21	0.25	0.30	0.33	0.37	0.44	0.53	1.06

Table 4: Computational time (given in seconds) for counting all DNA words of size between 2 and 9 on *Escherichia coli K12* complete genome ($\ell = 4\,639\,221$).

The table 4 compares SPatt to the program wordcount (from the popular EMBOSS 2.10.0 package) for the simple task to count all DNA words of a given size on the complete genome of *Escherichia coli*. As we can see, SPatt is very efficient (up to 1000 times faster than EMBOSS) and takes only a few seconds to deal with huge genomes. As SPatt can also deal with any custom alphabet, we can use it to do the same task on a protein database as Swissprot. The table 5 shows that it takes only 40 seconds for SPatt to count 64 millions of words (20^6) on a protein database of several tens of millions letters.

word size	2	3	4	5	6
SPatt	1.44	1.76	2.56	7.15	41.00

Table 5: Computational time (given in seconds) for counting all protein words of size between 2 (400 words) and 6 ($\simeq 6.4$ millions words) on Swissprot database release 38 ($\ell = 29\,080\,311$).

Of course, computing pattern statistics can be far more complicated that simply

counting words. The table 6 compares the different statistic treatments in terms of empirical time complexity in the special case of a M0 model. As we can see, computing Gaussian, binomial or compound Poisson statistics is very fast while exact and large deviations take a longer time. It is important to notice that the computational time grows linearly with the sequence length in the case of compound Poisson (but remains very fast) and exact (which, as a consequence, is not usable for sequences longer than few tens thousands letters). One should also note that exact, Gaussian and large deviations complexities grow geometrically with the order of the Markov model considered.

word size	2	3	4	5	6	7	8	9
exact *,**	3.14	1.73	2.21	2.44	3.36	6.35	23.97	99.26
Gaussian ***	0.00	0.00	0.00	0.00	0.02	0.09	0.33	1.37
binomial	0.00	0.00	0.00	0.01	0.02	0.08	0.31	1.22
compound Poisson **	0.00	0.00	0.01	0.01	0.03	0.11	0.42	1.63
large deviations ***	0.01	0.05	0.26	1.31	5.79	16.79	42.58	131.71

Table 6: Computational time (given in seconds) for computing statistics for all words of size between 2 and 9 on HIV type 1 complete genome ($\ell = 9\,719$) with a 0-th order Markov model ($m = 0$). (*) takes roughly 10 times more with a 1-st order model ($m = 1$). (**) grows linearly with sequence length. (***) grows geometrically with Markov model order. The precise mode of the large deviations statistics takes roughly three more times than the regular mode (for which the computational time is given here).

3.3 Markov versus shuffle model

As said in the Gaussian section, two closely related models exist for the sequence: the classical Markov one and the shuffle one. For any order $m \geqslant 0$, the shuffle model generates sequences with fixed number of occurrences for all words of size $m + 1$. If the Markov model parameters are estimated on the same sequence, the Markov model of same order will generate a sequence also satisfying this property, but only on average. Obviously, these two models are asymptotically the same, but the question is, how different they are on sequences of finite lengths and what are the consequences on the corresponding pattern statistics.

Figure 1 compares the two models for a few orders. As the RMES (Schbath, 1997) program produces Z-score rather that the pattern statistics we have chosen in this paper, we will use this scale for this comparison. For low order models ($m = 1$ or $m = 2$), it is clear that shuffle and Markov give very similar results, but, as the order grows, the divergence starts to be more significant. The shuffle

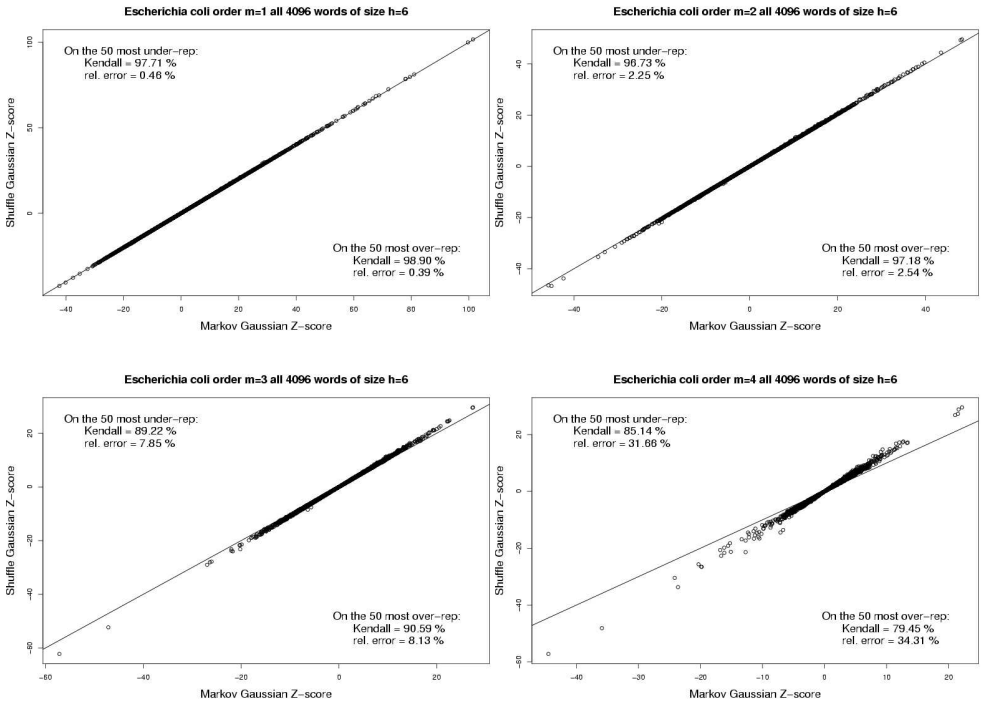


Figure 1: Comparison of the pattern Z-scores for all $4\,096$ words of length $h = 6$ on *Escherichia coli K12* complete genome ($\ell = 4\,639\,221$). The Markov Gaussian method (presented in this paper) is compared to the shuffle Gaussian one (from RMES 2.1.2p3) for several model orders ($1 \leqslant m \leqslant 4$). On all plots, relative error as well as Kendall's tau rank correlation are given both for the most under- and over-represented words.

Z-score appears to be larger than the Gaussian one (by more than 30% in mean for the 50 more significant words in the order 4 case). This trend is not a surprise as, by construction, sequences have less variability in the shuffle case than in the Markov one and hence, word counts are expected to have lower standard deviations in the former case than in the latter.

3.4 Precise versus regular large deviations

We have seen in section 2.3.3 (see corollary 14) that two kinds of large deviations approximations are available: the regular ones and, using a Chebyshev approxima-

tion, the more precise ones.

In order to limit the computational time we have chosen to use an 2-nd order Chebyshev approximation on the interval

$$\left[\frac{n_{\text{obs}}-1}{\ell-h+1};\frac{n_{\text{obs}}+1}{\ell-h+1}\right] \tag{66}$$

for a pattern of size h occurring n_{obs} times in a sequence of length ℓ.

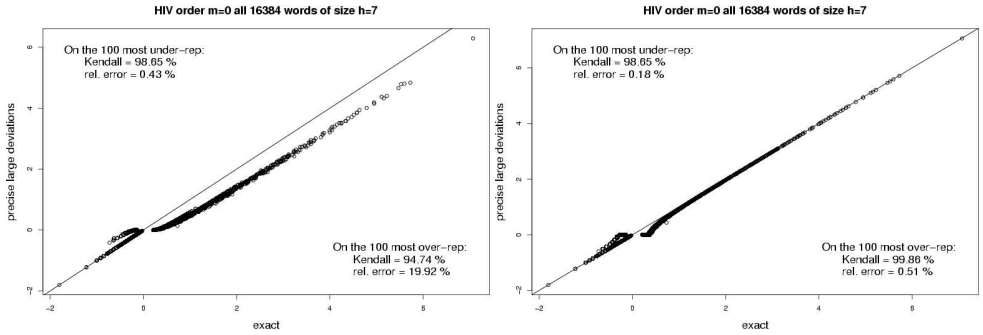


Figure 2: Comparison of the pattern statistics for all $16\,384$ words of length $h=7$ on HIV type 1 complete genome ($\ell = 9\,719$) with a 0-th order Markov model ($m=0$). The reference method (exact, 6.35s of computational time) is compared to the regular large deviations (16.79s) and to the precise large deviations (56.35s). On all plots, relative error as well as Kendall's tau rank correlation are given both for the most under- and over-represented words.

Figure 2 compares the two approaches to the exact reference. One should note first that both methods fail to give reliable results for patterns with low statistics (in absolute value). This is not a surprise as the corresponding events are in the center of the distribution where the large deviations theory is not valid.

For the regular large deviations (except in the special case of words occurring zero times, see the remark a) in section 2.3.3 for more details), we observe a good rank accordance with the reference but also a sensitive bias on the pattern statistics values. In the case of over-represented patterns, this bias is up to 20%, which is considerable. When we consider the precise large deviations statistics, we observe that this bias completely disappears. As a consequence, the relative errors improve dramatically and even the rank correlation get better in the over-represented case.

One should note that there are two different trends in the left tail of figure 2. This is due to the presence of words which are observed as zero time in the sequence.

As stated in remark a) in the large deviations section (2.3.3) this is consistent with the properties of the large deviations approximations and exact distribution in this special case.

3.5 Reliability on the extremes

In most pattern studies, we focus on the most significant results which have often very low p-values. This is the reason why it seems a good idea to examine at the reliability of these pattern statistics for extreme events. For reference purposes, we consider here a short sequence (HIV type 1 complete genome, 10 kb) so we can compute exact statistics. The problem with such small sequence is that extreme events are very uncommon. Our solution is hence to artificially consider extreme events by computing the statistics for a arbitrary fixed number n_{obs} of occurrences.

For computational considerations, only a 0-th order Markov model ($m = 0$) is considered, but the results are consistent with the one obtained with higher order models (data not shown).

3.5.1 Under-represented patterns

n_{obs}	0	5	10	15	20	25	30	35	40
Gaussian	47	42	38	35	33	30	28	26	24
binomial	4.8	5.2	5.4	5.6	5.8	6.0	6.1	6.2	6.3
compound Poisson	3.4	3.7	3.9	4.0	4.1	4.3	4.4	4.4	4.5
large deviations	9e-4	1.6	2.2	2.8	3.3	3.9	4.7	5.6	6.7
precise large dev.	9e-4	0.7	0.5	0.4	0.4	0.3	0.3	0.3	0.3
mean of statistics	-66.8	-58.1	-51.7	-46.5	-42.0	-38.0	-34.5	-31.3	-28.4

Table 7: mean relative error (in %) for the statistics (Markov model of order $m = 0$ estimated on HIV type 1 complete genome) of all DNA words of size $h = 3$ occurring n_{obs} times. The exact statistics are taken as a reference. The mean of statistics gives the mean of $S(w, n_{\text{obs}})$.

Table 7 clearly indicates that Gaussian approximations are the worst ones when we are interested in the statistic value itself. This is of no surprise for a statistical method designed to deal with "central" events. Binomial and compound Poisson approximations are similarly good with a little advantage for the latter (as it is an improvement of the former). To the difference of the other (and the exact opposite of the Gaussian ones), the large deviations results improve when n_{obs} get smaller (and so, the events are more extreme). Of course, this is classical behaviour of methods using large deviations. One should note that in the special case where $n_{\text{obs}} = 0$, the large deviations approximations are very good which is consistent with

the remark a) section 2.3.3. With precise large deviations, the results improve even further. Nevertheless, in this case, the relative error surprisingly does not improve with the significance of the events (as expected in the large deviations theory). This may be due to improper numerical evaluations of the correcting factor.

n_{obs}	0	5	10	15	20	25	30	35	40
Gaussian	95.6	96.1	96.4	97.0	97.0	97.4	97.8	97.9	97.9
binomial	96.2	96.4	96.4	96.4	96.7	96.9	96.9	97.1	97.1
compound Poisson	99.0	99.0	99.2	99.2	99.7	99.6	99.9	99.8	99.6
large deviations	100	100	100	100	100	100	100	100	100
precise large dev.	100	100	100	100	100	99.9	99.6	99.9	100
mean of statistics	-66.8	-58.1	-51.7	-46.5	-42.0	-38.0	-34.5	-31.3	-28.4

Table 8: mean Kendall's tau (in %) for the statistics (Markov model of order $m = 0$ estimated on HIV type 1 complete genome) of all DNA words of size $h = 3$ occurring n_{obs} times. The exact statistics are taken as a reference. The mean of statistics gives the mean of $S(w, n_{\text{obs}})$.

In terms of rank correlation, table 8 shows two classes of methods. In the first one, Gaussian and binomial approximations are good, if not perfect, and perform equally well. In the second, compound Poisson and large deviations approximations are closer to perfection even if, as expected, large deviations reliability increases in the extreme while the compound Poisson one slightly diminishes. On this matter, precise large deviations provide a few disappointing results (for example with $n_{\text{obs}} = 30$). Once again, the numerical evaluation of the correcting factor seems to be responsible for these imprecisions.

3.5.2 Over-represented patterns

n_{obs}	20	40	60	80	100	120	140	160	180
Gaussian	44	100	151	197	240	280	320	358	394
binomial	1.0	1.7	2.1	2.5	2.9	3.2	3.5	3.7	4.0
compound Poisson	0.6	1.2	1.6	2.0	2.3	2.5	2.8	3.0	3.3
large deviations	32	9.7	4.7	3.0	2.1	1.6	1.3	1.1	0.9
precise large dev.	7.5	0.4	0.05	0.006	0.002	0.001	0.001	0.001	0.001
mean of statistics	3.9	15.4	31.3	50.1	71.2	94.2	118.7	144.5	171.5

Table 9: mean relative error (in %) for the statistics (Markov model of order $m = 0$ estimated on HIV type 1 complete genome) of all DNA words of size $h = 5$ occurring n_{obs} times. The exact statistics are taken as a reference. The mean of statistics gives the mean of $S(w, n_{\text{obs}})$.

When considering patterns seen more than expected, table 9 and 10 show that we get similar results as the last case but with even more effect: for example, relative error can grow to 400% for Gaussian approximations.

Once again, the precise large deviations perform the best in terms of relative error (up to 1000 times better than the best one with $n_{\text{obs}} = 180$). Nevertheless, the noise due to the numeric evaluation of correction factor could lead to a few errors in the ranking (even if still close to perfection).

n_{obs}	20	40	60	80	100	120	140	160	180
Gaussian	99.8	99.1	98.2	97.6	96.9	96.3	95.7	95.1	94.1
binomial	99.1	98.3	97.3	96.6	96.0	95.4	94.7	94.1	93.1
compound Poisson	100	100	100	100	100	99.8	99.9	99.8	99.5
large deviations	100	100	100	100	100	100	100	100	99.9
precise large dev.	100	99.9	100	99.9	99.9	99.9	100	100	99.9
mean of statistics	3.9	15.4	31.3	50.1	71.2	94.2	118.7	144.5	171.5

Table 10: mean Kendall's tau (in %) for the statistics (Markov model of order $m = 0$ estimated on HIV type 1 complete genome) of all DNA words of size $h = 5$ occuring n_{obs} times. The exact statistics are taken as a reference. The mean of statistics gives the mean of $S(w, n_{\text{obs}})$.

One could object that it is well known that Gaussian approximations perform better when the expected number of occurrences increases. Even if this is true, we will see that the disastrous behaviour of Gaussian approximations in the extremes we have just pointed out, still exists (even if less intense) in more favorable cases.

3.6 Reliability on real cases

In order to compare our statistics in more realistic situations we considered two complete genomes: HIV type 1 ($\ell = 9\,719$) and *Bacillus subtilis* ($\ell = 4\,214\,814$). On both these genomes we consider all words of size $h = 4$ or $h = 7$. For HIV, exact computations are available and taken as a reference. For *Bacillus subtilis*, and according to the results of the last section, large deviations statistics are taken as a reference.

In figure 3 it is clear that all these statistical methods are consistent and close to each other. Nevertheless, Gaussian approximations seem to be less reliable for the tail distribution events while large deviations get better. Binomial and compound have only a few differences which correspond to words with high overlapping structure (*e.g.* `aaaa` or `tttt`). In terms of relative error, the best method is the compound Poisson. In terms of rank correlation, the best are compound Poisson or large deviations.

The high reliability of the compound Poisson approximations is confirmed in figure 4 where results are very close to perfection. At the same time, we see how wrong the Gaussian approximations could be. In the case of over-represented patterns, Gaussian approximations are not only completely false in terms of relative error (more than 200 %) but also very badly correlated in terms of rank (less than 50% for the Kendall's tau). Large deviations approximations reliability remains acceptable but is lower than binomial ones. This is obviously due to the fact that the considered events are not extreme (10^{-6} for the lowest p-value). One should note that large deviations statistics of words occurring zero times are very close to perfection which is once again consistent with the remark a) in section 2.3.3.

With figure 5 and 6, Gaussian approximations show once more their poor behaviour in the extremes. One notable exception concerns the ranking of over-represented words in figure 5 where Gaussian approximations perform better than binomial ones. This result is of no surprise as we consider here words with expected counts of several thousands which is a very favorable case for Gaussian approximations. However, even in this favorable case, Gaussian approximations are biased in the extremes and are still outperformed in ranking for under-represented patterns. Binomial approximations perform very well but are not good with the highly overlapping words and finally, compound Poisson and large deviations approximations give so closely related results that even if one is more reliable than the other (for example the large deviations), the second one is almost as reliable as the first.

4 Conclusion

Although the task seems simple, computing pattern statistics is a difficult problem. We proposed here a detailed review of a large sample of the methods available and we provided for each one, a complete set of solutions for its practical implementation. Using the package SPatt (freely available) it was possible to compare, for the first time, all these methods together both in terms of resource needed and reliability.

We have seen that Gaussian approximations are biased for the extreme statistics and are completely outperformed by all other methods for the matter of computing p-values. However, this bias is reduced when the expected number of occurrences increases. Nevertheless, even in the most favorable cases (several thousands of expected occurrences), binomial approximations are more reliable in terms of relative error and only slightly superceded in terms of rank accordance. Binomial approximations are always better than Gaussian approximations for p-value computations and most of the time similar to or better than Gaussian ones for ranking matters. Compound Poisson approximations give fast and accurate computations

in all cases but large deviations give better results when extreme events are considered. Finally, at the cost of a longer computational time (especially if high order Markov models are considered), the precise large deviations appear to outperform all other approximations as soon as they are used on significant patterns.

For a quick and rough evaluation of a pattern statistic, our advice would be then to use binomial approximations rather than Gaussian ones. If more reliability is needed, we strongly advise using compound Poisson approximations as a first step. Then, for more interesting patterns (*i.e.* with most extreme statistics), to get large deviations statistics: precise ones for a reliable evaluation of the statistics themselves, or the regular ones if ranking is what matters the most.

References

R. Arratia, L. Goldstein, and L. Gordon. Poisson approximation and the Chen-Stein method. *Stat. Sci.*, 5(4):403–434, 1990.

K. Atteson. Calculating the exact probability of language-like patterns in biomolecular sequences. In AAAI Press, editor, *Sixth International Conference on Intelligent Systems for Molecular Biology*, pages 17–24, 1998.

A.D. Barbour, L. H. Y. Chen, and W. L. Loh. Compound Poisson approximation for nonnegative random variables via Stein method. *Ann. Probab*, 20:1504–1527, 1992.

E. Beaudoing, S. Freier, J. R. Wyatt, J.-M. Claverie, and D. Gautheret. Patterns of Variant Polyadenylation Signal Usage in Human Genes. *Genome Res.*, 10(7): 1001–1010, 2000.

M. Borodovsky and J. D. McIninch. Genmark: parallel gene recognition for both DNA strands. *Computer & Chimistry*, 17(2):123–133, 1993.

P.-Y. Bourguignon and D. Robelin. Modèles de Markov parsimonieux: sélection de modèle et estimation. In *Proceedings of JOBIM Congress*, Montreal, 2004.

A. Brazma, I. Jonassen, J. Vilo, and E. Ukkonen. Predicting Gene Regulatory Elements in Silico on a Genomic Scale. *Genome Res.*, 8(11):1202–1215, 1998.

J. A. Bucklew. *Large deviation techniques in decision, simulation, and estimation*. Wiley, 1990.

A. Buhlmann and A. J. Wyner. Variable Length Markov chains. *Annals of Statistics*, 27(2):480–513, 1999.

O. Chrysaphinou and S. Papastavridis. A limit theorem on the number of overlapping appearances of a pattern in a sequence of independant trials. *Proba. Theory Relat. Fields*, 79(1):129–143, 1988.

R. Cowan. Expected frequencies of DNA patterns using Whittle's formula. *J. Appl. Prob.*, 28:886–892, 1991.

A. Dembo and O. Zeitouni. *Large deviations techniques and applications*. Springer-Verlag, 1998.

F. Den Hollender. *Large Deviations*. American Mathematical Society, 2000.

R. Durbin, S. R. Eddy, A. Krogh, and G. Mitchison. *Biological Sequence Analysis: Probabilistic models of proteins dans nucleic acids*. Campbridge University Press, 1998.

M. El Karoui, V. Biaudet, S. Schbath, and A. Gruss. Characteristics of Chi distribution on different bacterial genomes. *Res. Microbiol.*, 150:579–587, 1999.

J. C Fu. Distribution theory of runs and patterns associated with a sequence of multi-state trials. *Statistica Sinica*, 6(4):957–974, 1996.

J. C. Fu and M. V. Koutras. Distribution theory of runs: A Markov chain approach. *J. Am. Statist. Assoc.*, 89(427):1050–1058, 1994.

M. S. Gelfand and E. V. Koonin. Avoidance of palindromic words in bacterial and archaeal genomes: a close connection with restriction enzymes [published erratum appears in Nucleic Acids Res. 25(24), 5135-6]. *Nucl. Acids. Res.*, 25 (12):2430–2439, 1997.

M. X. Geske, A. P. Godbole, A. A. Schaffner, A. M. Skolnick, and G. L. Wallstorm. Compound Poisson approximation for word patterns under Markovian hypothesis. *J. Appl. Prob.*, 32:877–892, 1995.

S. Hampson, D. Kibler, and P. Baldi. Distribution patterns of over-represented k-mers in non-coding yeast DNA . *Bioinformatics*, 18(4):513–528, 2002.

N. L. Johnson, S. Kotz, and A. W. Kemp. *Univariate discrete distributions*. Wiley, New York, 1992.

S. Karlin, C. Burge, and A. M. Campbell. Statistical analyses of counts and distributions of restriction sites in DNA sequences. *Nucl. Acids. Res.*, 20(6):1363–1370, 1992.

J. Kleffe and M. Borodovsky. First and second moment of counts of words in random text generated by Markov chains. *Comp. Applic. Biosci.*, 8:433–441, 1992.

R. B. Lehoucq, D. C. Sorensen, and C. Yang. *ARPACK Users' Guide: Solution of Large-Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods*. SIAM, Philadelphia, 1987.

R. B. Lehoucq, D. C. Sorensen, and C. Yang. Arpack users guide: Solution of large scale eigenvalue problems with implicitly restarted Arnoldi methods. Technical report, Rice University, from `http://www.caam.rice.edu/software/ARPACK`, 1997.

P. Nicodème, B. Salvy, and P. Flajolet. Motif statistics. *Theoretical Computer Science*, 18:161–171, 2002.

P. Nicodème. Fast approximate motif statistics. *J. Comp. Biol.*, 8(3):235–248, 2001.

G. Nuel. LD-SPatt: Large Deviations Statistics for Patterns on Markov Chains. *J. Comput. Biol.*, 11(6):1023–1033, 2004.

G. Nuel. S-SPatt: simple statistics for patterns on Markov chains. *Bioinformatics*, 21(13):3051–3052, 2005.

G. Nuel. Cumulative distribution function of a geometric Poisson distribution. *J. Stat. Comp. and Sim.*, 2006a. In press, preprint available at http://stat.genopole.cnrs.fr/˜gnuel.

G. Nuel. Pattern statistics on Markov chains and sensitivity to parameter estimation. *Algo. Mol. Biol.*, 2006b. In revision, preprint available at http://stat.genopole.cnrs.fr/˜gnuel.

P. A. Pevzner, M. Y. Borodovski, and A. A. Mironov. Linguistic of nucleotide sequences: the significance of deviation from mean statistical characteristics and prediction of frequencies of occurrence of words. *J. Biomol. Struct. Dyn*, 6: 1013–1026, 1989.

W. H. Press, S. A. Teukolsky, W. T. Vettering, and B. P. Flannery. *Numerical recipes in C*. Cambridge University Press, 1997.

B. Prum, F. Rodolphe, and E. de Turckheim. Finding words with unexpected frequencies in DNA sequences. *J. R. Statist. Soc. B.*, 11:190–192, 1995.

L. A. Rabiner. Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. In *Proceedings of the IEE*, volume 77, pages 254–286, 1989.

G. Reinert and S. Schbath. Compound Poisson and Poisson process approximations for occurrences of multiple words in Markov chains. *J. Comp. Biol.*, 5:223–254, 1998.

G. Reinert and S. Schbath. Compound Poisson approximations for occurrences of multiple words. *Statistics in Molecular Biology and Genetics*, 33:257–275, 1999.

G. Reinert, S. Schbath, and S Waterman. Probabilistic and Statistical Properties of Words: An Overview. *J. Comp. Biol.*, 7(1/2):1–46, 2000.

G. Reinert, S. Schbath, and S Waterman. *Lothaire: Applied Combinatorics on Words*, chapter Probabilistic and Statistical Properties of Finite Words in Finite Sequences. Cambridge University Press, 2005.

S. Robin. A compound Poisson model for word occurrences in DNA sequences. *J. Roy. Stat. Soc. Ser. C*, 51:437–451, 2002.

S. Robin, J.-J. Daudin, H. Richard, M.-F. Sagot, and S. Schbath. Occurrence probability of structured motifs in random sequences. *J. Comp. Biol.*, 9:761–773, 2002.

S. Robin and J.J. Daudin. Exact distribution of word occurrences in a random sequence of letters. *J. App. Prob.*, 36:179–193, 1999.

S. Robin and S. Schbath. Numerical comparison of several approximations of the word count distribution in random sequences. *J. Comp. Biol.*, 8:349–359, 2001.

M. Régnier. A unified approach to word occurrence probabilities. *Discrete applied mathematics*, 104(1):259–280, 2000.

M. Régnier and W. Szpankowski. On pattern frequency occurrences in a Markovian sequence. *Algorithmica*, 22(4):631–649, 1998.

S. Schbath. Compound Poisson approximation of word counts in DNA sequences. *ESAIM, Probab. Stat.*, 1:1–16, 1995.

S. Schbath. An efficient statistic to detect over- and under- represented words in DNA sequences. *J. Comp. Biol.*, 4:189–192, 1997.

J. van Helden, B André, and J. Collado-Vides. Extracting Regulatory Sites from the Upstream Region of Yeast Genes by Computational Analysis of Oligonucleotide Frequencies. *J. Mol. Biol.*, 281:827–842, 1998.

J. van Helden, M. li del Olmo, and J. E. Perez-Ortin. Statistical analysis of yeast genomic downstream sequences reveals putative polyadenylation signals. *Nucl. Acids. Res.*, 28(4):1000–1010, 2000.

P. Whittle. Some distribution and moment fomulæ for the Markov chain. . *J. R. Statist. Soc. B.*, 17:235–242, 1955.

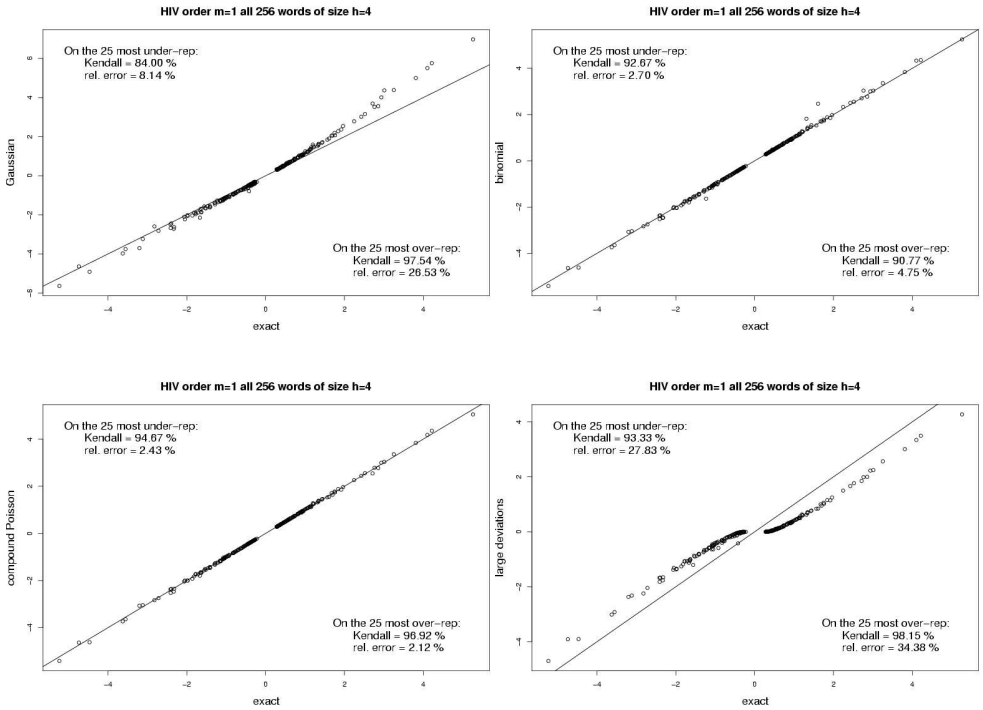


Figure 3: Comparison of the pattern statistics for all 256 words of length $h = 4$ on HIV type 1 complete genome ($\ell = 9\,719$) with a 1-st order Markov model ($m = 1$). The reference method (exact, 22.92s of computational time) is compared to the Gaussian one (0.00s), to the binomial one (0.00s), to the compound Poisson one (0.00s) and to the large deviation one (0.31s). On all plots, relative error as well as Kendall's tau rank correlation are given both for the most under- and over-represented words.

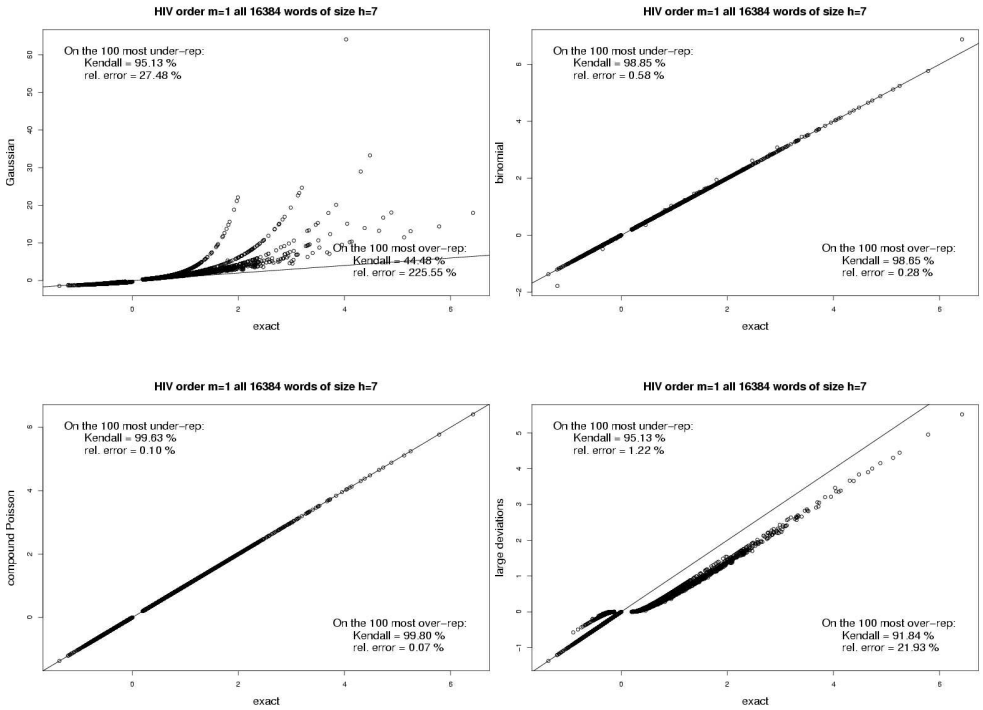


Figure 4: Comparison of the pattern statistics for all $16\,384$ words of length $h = 7$ on HIV type 1 complete genome ($\ell = 9\,719$) with a 1-st order Markov model ($m = 1$). The reference method (exact, 38.05s of computational time) is compared to the Gaussian one (0.10s), to the binomial one (0.08s), to the compound Poisson one (0.11s) and to the large deviation one (18.78s). On all plots, relative error as well as Kendall's tau rank correlation are given both for the most under- and over-represented words.

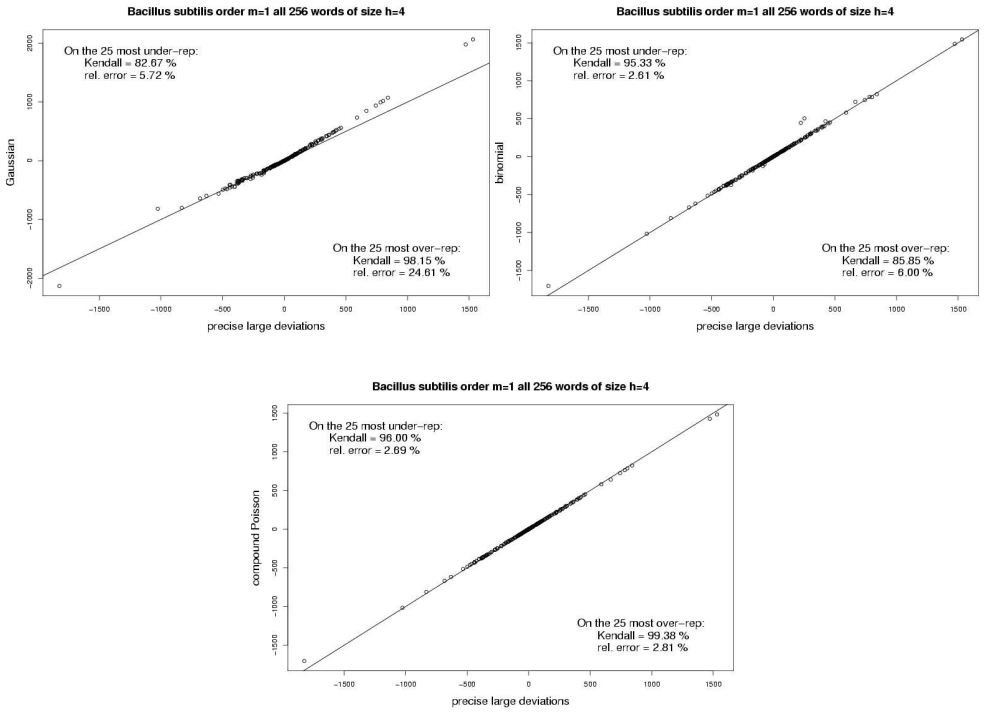


Figure 5: Comparison of the pattern statistics for all 256 words of length $h = 4$ on *Bacillus subtilis* complete genome ($\ell = 4\,214\,814$) with a 1-st order Markov model ($m = 1$). The reference method (precise large deviations, 1.35s of computational time) is compared to the Gaussian one (0.28s), to the binomial one (0.28s) and to the compound Poisson one (0.62s). On all plots, relative error as well as Kendall's tau rank correlation are given both for the most under- and over-represented words.

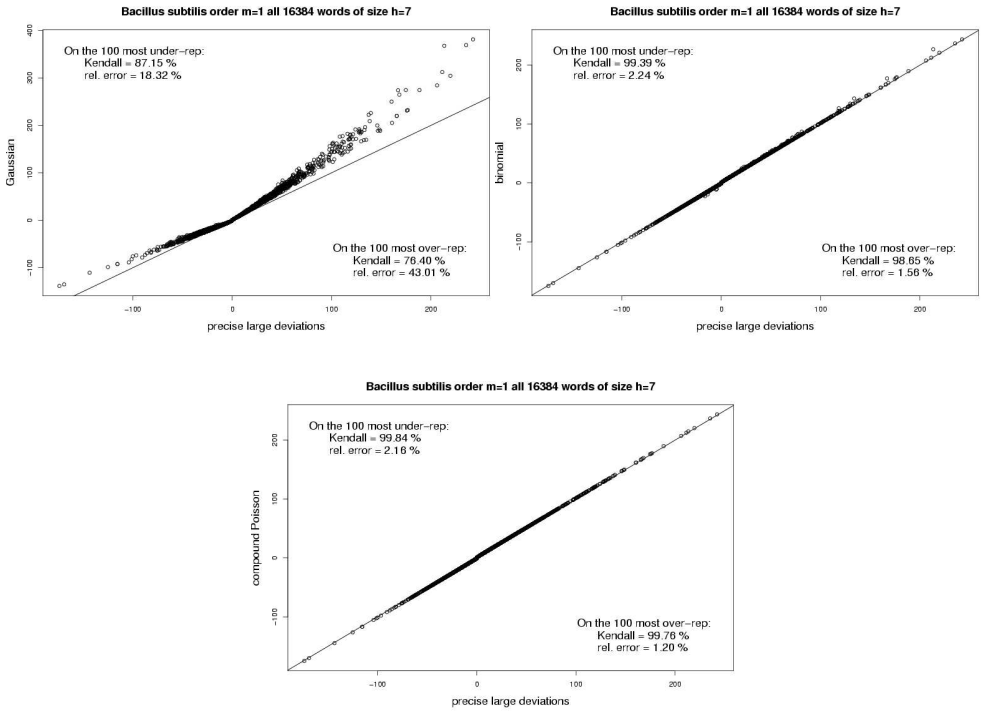


Figure 6: Comparison of the pattern statistics for all $16\,384$ words of length $h = 7$ on *Bacillus subtilis* complete genome ($\ell = 4\,214\,814$) with a 1-st order Markov model ($m = 1$). The reference method (precise large deviations, 179.01s of computational time) is compared to the Gaussian one (0.54s), to the binomial one (0.49s) and to the compound Poisson one (0.95s). On all plots, relative error as well as Kendall's tau rank correlation are given both for the most under- and over-represented words.

Chapitre 12

Pattern statistics on Markov chains and sensitivity to parameter estimation

Article de Nuel accepté par *Algorithms for Molecular Biology*. Il y est abordé le problème de la sensibilité des statistiques de motifs par rapport à l'estimation des paramètres. La technique proposée est celle de la delta-méthode. Dans le cas particulier des approximations binomiales, des formules explicites sont proposées et validées. Une étude par simulation complète ce travail en proposant d'évaluer l'impact de cette variabilité sur une étude concernant les motifs. Il est mis en évidence un manque de robustesse des statistiques de motifs qui se traduit rapidement par un grand nombre d'erreurs lorsque l'ordre des modèles considérés augmente, y compris sur des séquences de plusieurs mégabases. L'article conclut en incitant à la plus grande prudence lors du choix de l'ordre du modèle pour le calcul d'une statistique de motif.

Algorithms for Molecular Biology BioMed Central

Research

Open Access

Pattern statistics on Markov chains and sensitivity to parameter estimation

Grégory Nuel*

Address: Laboratoire Statistique et Génome, University of Evry, CNRS (8071), INRA(1152), 523, place des terrasses de l'Agora, 91034 Evry CEDEX, France

Email: Grégory Nuel* - nuel@genopole.cnrs.fr

* Corresponding author

Published: 17 October 2006

Algorithms for Molecular Biology 2006, **1**:17 doi:10.1186/1748-7188-1-17

Received: 07 April 2006
Accepted: 17 October 2006

This article is available from: http://www.almob.org/content/1/1/17

Abstract

Background: In order to compute pattern statistics in computational biology a Markov model is commonly used to take into account the sequence composition. Usually its parameter must be estimated. The aim of this paper is to determine how sensitive these statistics are to parameter estimation, and what are the consequences of this variability on pattern studies (finding the most over-represented words in a genome, the most significant common words to a set of sequences,...).

Results: In the particular case where pattern statistics (overlap counting only) computed through binomial approximations we use the delta-method to give an explicit expression of σ, the standard deviation of a pattern statistic. This result is validated using simulations and a simple pattern study is also considered.

Conclusion: We establish that the use of high order Markov model could easily lead to major mistakes due to the high sensitivity of pattern statistics to parameter estimation.

Background

In order to study pattern occurrences in biological sequences, simple frequencies are not relevant in most cases because of pattern overlapping structure as well as composition bias in the sequences. A common workaround consists to compute the significance of an observation assuming the sequence $X = X_1 \dots X\ell$ over the finite alphabet $\mathcal{A}$. (size k) is generated according to an order $m \geq 1$ homogeneous, stationary and ergodic Markov model. Let π (size k^{m+1}) defined by

$$\pi(w, a) = (X_{m+1} = a | X_1 \dots X_m = w) \quad \forall (w, a) \in \mathcal{A}^m \times \mathcal{A} \qquad (1)$$

be the parameter of this Markov model, Π its transition matrix (note that we have $\Pi = \pi$ only if $m = 1$) and μ its stationary distribution (defined by $\mu \times \Pi = \mu$).

We then introduce the pattern statistic defined by

$$S = \begin{cases} -\log_{10} \mathbb{P}(N \geq N_{\text{obs}}) & \text{if } N_{\text{obs}} \geq [N] \\ \log_{10} \mathbb{P}(N \leq N_{\text{obs}}) & \text{if } N_{\text{obs}} < [N] \end{cases} \qquad (2)$$

where N is the random number of overlapping occurrences (*i. e.* X = aababaaba contains three overlapping occurrences of aba but only two non-overlapping ones) of a given fixed pattern on the random sequence X and N_{obs} is an observation.

When π is known (and hence μ), several statistical methods are available to compute S: exact computations [1-4], Gaussian [5,6], binomial [7,8], compound Poisson [9-11] or large deviations approximations [12]. But in general, the parameter π is not available and must be estimated. Let us denote by $\mathbf{N}_0$ (resp. $\mathbf{N}_1$) the (overlap) frequencies of all words of size m (resp. $m + 1$) in the sequence $Y = Y_1 \ldots Y_n$, then the Maximum-Likelihood Estimator (MLE) of π is given by

$$\hat{\pi}(w,a) = \frac{\mathbf{N}_1(wa)}{\sum_{b\in\mathcal{A}} \mathbf{N}_1(wb)} \quad \forall (w,a) \in \mathcal{A}^m \times \mathcal{A} \tag{3}$$

and the MLE of μ (as a function of π) is therefore defined by $\hat{\mu} \times \hat{\Pi} = \hat{\mu}$ where $\hat{\Pi}$ is the transition matrix associated to $\hat{\pi}$

We introduce now the following estimators

$$\mu_{\mathbf{N}}(w) = \frac{\mathbf{N}_0(w)}{n-m+1} \quad \text{and} \quad \pi_{\mathbf{N}}(w,a) = \frac{\mathbf{N}_1(wa)}{\mathbf{N}_0(w)} \quad \forall (w,a) \in \mathcal{A}^m \times \mathcal{A} \tag{4}$$

which are known to be asymptotically equivalent with the MLE when n is large.

The quality of parameter estimation depends both on the number of parameters to estimate (k^{m+1} for an order m Markov model) and of the length (n) of the homogeneous sequence used for their estimation. When the same sequence (or set of sequences) is used both for observed frequencies and parameter estimation, m should not be greater than $h - 2$ for a pattern of length h (as else, the observed frequency of the pattern will be included in the model). As literature often suggests to use the highest possible order, it is hence common to consider $m = 6$ or more (for a DNA pattern of size $h \geq 8$). Moreover, because of the homogeneity assumption of the model, the considered genomes have often to be segmented first. As a result, the sequences length used for parameter estimations are often dramatically reduced by such segmentation (*e. g.* $n = 10^5$ to $n = 10^6$ at the very best for DNA sequences). It is hence quite common to encounter high order Markov models estimated on rather short sequences which could result in high sensitivity to parameter estimation.

Considering that Y is generated through a Markov model of parameter π, the main goal of this paper is to study the distribution of $S_{\mathbf{N}}$, the statistic S computed using the estimators $\mu_{\mathbf{N}}$ and $\pi_{\mathbf{N}}$, and the consequences of its variability in projects using pattern statistics. We first present in details how the delta-method can be used to get a Gaussian approximation for the distribution of $S_{\mathbf{N}}$ (using a binomial approximation to compute the pattern statistics). Then these approximations are validated through simulations and, at last, we consider a classical pattern study (finding the most over-represented patterns of a given size) and we evaluate the detrimental effect of parameter estimations both in terms of true positive rate and rank accordance.

Materials and methods

Distribution of N = (N₀, N₁)

As the estimators defined in (4) are expressed as functions of $\mathbf{N}_0$ and $\mathbf{N}_1$ we first study their distribution. Using a Gaussian approximation, we have

$$\mathcal{L}\left(\underbrace{\begin{bmatrix}\mathbf{N}_0\\ \mathbf{N}_1\end{bmatrix}}_{\mathbf{N}}\right) \simeq \left(\underbrace{\begin{bmatrix}\mathbf{E}_0\\ \mathbf{E}_1\end{bmatrix}}_{\mathbf{E}}, \underbrace{\begin{bmatrix}\mathbf{C}_{0,0} & \mathbf{C}_{0,1}\\ \mathbf{C}_{1,0} & \mathbf{C}_{1,1}\end{bmatrix}}_{\mathbf{C}}\right) \tag{5}$$

where, for $i, j \in \{0, 1\}$, $\mathbf{E}_i \in \mathbb{R}^{d_i}$, and $\mathbf{C}_{i,j} \in \mathbb{R}^{d_i} \times \mathbb{R}^{d_j}$ with $d_i = k^{m+i}$. One can note that $\mathbf{C}_{0,0}$ and $\mathbf{C}_{1,1}$ are symmetric, and ${}^t(\mathbf{C}_{1,0}) = \mathbf{C}_{0,1}$ (where ^{t} is the matrix transpose operator).

In the stationary case, exact expression of **E** and **C** can be computed according to [5].

Expectation is simply given $\forall w \in \mathcal{A}^m$ by

$$\mathbf{E}_0(w) = (n - m + 1)\,\mu(w) \qquad \mathbf{E}_1(wa) = (n - m)\,\mu(w)\Pi(w,a) \quad \forall (w,a) \in \mathcal{A}^m \times \mathcal{A} \tag{6}$$

In order to give more fluidity to this paper, the expression of the covariance matrix **C** have been moved in appendix A. Let us remark, before going forward that substituting **N** by **E** in (4) immediately gives

$$\mu_{\mathbf{E}} = \mu \quad \text{and} \quad \pi_{\mathbf{E}} = \left(1 - \frac{1}{n-m+1}\right)\pi \tag{7}$$

Delta method

Let us start with a simple case. We consider a single pattern which is over-represented (seen more than expected) so we have

$$S_{\mathbf{N}} = -\log_{10} F^+(\mathbf{N}) \quad \text{with} \quad F^+(\mathbf{N}) \triangleq \mathbb{P}_{\mu_{\mathbf{N}},\pi_{\mathbf{N}}}(N \geq N_{\text{obs}}) \tag{8}$$

where the function F^+ also depends on the sequence length ℓ and the considered pattern.

If F^+ is differentiate, the delta-method (a simple first order Taylor expansion around $\mathbf{N} = \mathbf{E}$, see [13]) provides the following approximation:

$$S_{\mathbf{N}} \simeq -\log_{10} F^{+}(\mathbf{E}) - \frac{{}^{t}(\mathbf{N}-\mathbf{E})\nabla F^{+}(\mathbf{E})}{\ln(10)F^{+}(\mathbf{E})} \qquad (9)$$

and hence, using (7) we have

$$S_{\mathbf{N}} \simeq S - \frac{{}^{t}(\mathbf{N}-\mathbf{E})\nabla F^{+}(\mathbf{E})}{\ln(10)F^{+}(\mathbf{E})} \qquad (10)$$

for n large enough. The distribution of $\hat{S}$ is therefore approximated by

$$\mathcal{L}(S_{\mathbf{N}}) \simeq \mathcal{N}(S, \sigma^2) \qquad (11)$$

with

$$\sigma = \frac{\sqrt{{}^{t}\nabla F^{+}(\mathbf{E}) \times \mathbf{C} \times \nabla F^{+}(\mathbf{E})}}{\ln(10)F^{+}(\mathbf{E})} \qquad (12)$$

In consequence, computing σ requires both to compute $\mathbf{C}$ (done in appendix A) and $\nabla F^{+}(\mathbf{E})$.

Single pattern

The exact expression of F^{+} is computable through many different methods [1-4] but is too much complicated to derive explicitly ∇F^{+}. To overcome this problem, we propose to consider an approximation of F^{+}. As said in introduction, many kind of approximations are available (Gaussian, binomial, compound Poisson or large deviations). In this paper, we have chosen to use a binomial approximation as it provides an expression which is analytically differentiable and is known to be a good heuristic to the problem [8].

For a single non-degenerate pattern (*i.e.* a simple word) $W = w_1 \dots w_h$ ($w_i \in \mathcal{A}$) with $h \geq m - 1$ we first denote by

$$P(\mathbf{N}) = \mu_{\mathbf{N}}(w_1 \dots w_m) \times \pi_{\mathbf{N}}(w_1 \dots w_m, w_{m+1}) \times \dots \times \pi_{\mathbf{N}}(w_{h-m} \dots w_{h-1}, w_h) \qquad (13)$$

the probability for W to occur at a given position in the sequence and then we get

$$F^{+}(\mathbf{N}) \simeq \mathbb{P}(\mathcal{B}(\ell_h, P(\mathbf{N})) \geq N_{\text{obs}}) = \frac{\beta(P(\mathbf{N}), N_{\text{obs}}, \ell_h - N_{\text{obs}} + 1)}{\beta(N_{\text{obs}}, \ell_h - N_{\text{obs}} + 1)} \qquad (14)$$

where $\mathcal{B}$ denotes the binomial distribution, with $\ell_h = \ell - h + 1$ and where the β functions (complete and incomplete) and their relation to the binomial cumulative distribution function are described in appendix B.

Note that if we consider non-overlapping occurrences instead of overlapping ones, we can still use a binomial approximation for the distribution of N, but the expression of $P(\mathbf{N})$ is more complicated as it involves the autocorrelation polynome of the pattern [14]. This point is not developed in this paper.

Replacing $\mu_{\mathbf{N}}$ and $\pi_{\mathbf{N}}$ by their expression easily gives

$$P(\mathbf{N}) = \frac{1}{n-m+1} \prod_{w \in \mathcal{A}^m} \frac{\prod_{a \in \mathcal{A}} \mathbf{N}_1(wa)^{A_1(wa)}}{\mathbf{N}_0(w)^{A_0(w)}} \qquad (15)$$

where $A_1(wa)$ counts occurrences of the word wa in $W = w_1 \dots w_h$ and $A_0(w)$ counts occurrences of the word w in $w_2 \dots w_{h-1}$. Note that in the particular case where $h = m - 1$, all $A_0(w)$ are null and we simply get $(n - m + 1) \times P(\mathbf{N}) = \mathbf{N}_1(W)$.

Using the derivative properties of the incomplete beta function (see appendix B for more details) we hence get

$$\nabla F^{+}(\mathbf{N}) \simeq \frac{P(\mathbf{N})^{N_{\text{obs}}-1}(1-P(\mathbf{N}))^{\ell_h - N_{\text{obs}}}}{\beta(N_{\text{obs}}, \ell_h - N_{\text{obs}} + 1)} \times \nabla P(\mathbf{N}) \qquad (16)$$

so all we need is to compute $\nabla P(\mathbf{N})$.

For all $(w, a) \in \mathcal{A}^m \times \mathcal{A}$ we have

$$\frac{\partial P(\mathbf{N})}{\partial \mathbf{N}_0(w)} = -\frac{A_0(w)}{\mathbf{N}_0(w)} \times P(\mathbf{N}) \qquad (17)$$

and

$$\frac{\partial P(\mathbf{N})}{\partial \mathbf{N}_1(w)} = -\frac{A_1(wa)}{\mathbf{N}_1(wa)} \times P(\mathbf{N}) \qquad (18)$$

If we denote by

$$P = \mu(w_1 \dots w_m) \times \pi(w_1 \dots w_m, w_{m+1}) \times \dots \times \pi(w_{h-m} \dots w_{h-1}, w_h) \qquad (19)$$

the *true* probability for W to occur at a given position in the sequence X then we get, using (7) in (13), that

$$P(\mathbf{E}) = p \times \left(1 - \frac{1}{n-m+1}\right)^{h-m} \simeq p \qquad (20)$$

for n large enough. We hence get

$$\nabla F^{+}(\mathbf{E}) \simeq \frac{p^{N_{\text{obs}}}(1-p)^{\ell_h - N_{\text{obs}}}}{\beta(N_{\text{obs}}, \ell_h - N_{\text{obs}} + 1)} \times \mathbf{G} \qquad (21)$$

where ${}^{t}\mathbf{G} = [{}^{t}\mathbf{G}_0\ {}^{t}\mathbf{G}_1]$ is defined by

$$\mathbf{G}_0(w) = -\frac{A_0(w)}{\mathbf{E}_0(w)} \quad \text{and} \quad \mathbf{G}_1(wa) = -\frac{A_1(wa)}{\mathbf{E}_1(wa)} \qquad (22)$$

Using equation (12) we finally get

$$\sigma \simeq Q^{+}\sqrt{{}^{t}\mathbf{G}\times\mathbf{C}\times\mathbf{G}} \qquad (23)$$

where

$$Q^{+} = \frac{p^{N_{\text{obs}}}(1-p)^{\ell_h - N_{\text{obs}}}}{\ln(10)\beta(p, N_{\text{obs}}, \ell_h - N_{\text{obs}} + 1)} \qquad (24)$$

and then, a computation of σ is possible by plug-in. Without considering the computation of **E** and **C**, the complexity of this approach is $O(h)$ (where h is the size of the pattern).

When a degenerate pattern (finite set of words) is considered instead of a single word, it is easy to adapt this method by summing the contribution p of each word belonging to the pattern. This point is left to the reader.

Under-represented pattern

In the case of an under-represented pattern we have

$$S_{\mathbf{N}} = \log_{10} F^{-}(\mathbf{N}) \quad \text{with} \quad F^{-}(\mathbf{N}) \triangleq \mathbb{P}_{\mu_{\mathbf{N}},\pi_{\mathbf{N}}}(N \le N_{\text{obs}}). \qquad (25)$$

Using a binomial approximation we get

$$F^{-}(\mathbf{N}) \simeq \mathbb{P}(\mathcal{B}(\ell_h, P(\mathbf{N})) \le N_{\text{obs}}) = \frac{\beta^{-}(P(\mathbf{N}), N_{\text{obs}}+1, \ell_h - N_{\text{obs}})}{\beta(N_{\text{obs}}+1, \ell_h - N_{\text{obs}})} \qquad (26)$$

and, by the same method than in the over-represented case we finally have

$$\sigma \simeq Q^{-}\sqrt{{}^{t}\mathbf{G}\times\mathbf{C}\times\mathbf{G}} \qquad (27)$$

where

$$Q^{-} = \frac{p^{N_{\text{obs}}+1}(1-p)^{\ell_h - N_{\text{obs}} - 1}}{\ln(10)\beta^{-}(p, N_{\text{obs}}+1, \ell_h - N_{\text{obs}})} \qquad (28)$$

Two distinct patterns

We consider now two patterns V and W instead of one and want to study the joint distribution of $S_{\mathbf{N}}(V)$ and $S_{\mathbf{N}}(W)$ their corresponding pattern statistics.

With a similar argument as in section "delta method", it is easy to show that

$$\mathcal{L}\left(\begin{bmatrix} S_{\mathbf{N}}(V) \\ S_{\mathbf{N}}(W) \end{bmatrix}\right) \simeq \left(\begin{bmatrix} S(V) \\ S(W) \end{bmatrix}, \begin{bmatrix} \sigma_V^2 & \sigma_{V,W} \\ \sigma_{V,W} & \sigma_W^2 \end{bmatrix}\right) \qquad (29)$$

where σ_V (resp. σ_W) is the standard deviation σ for the pattern V (resp. W) and where

$$\sigma_{V,W} = \frac{{}^{t}\nabla F_V^{\varepsilon}(\mathbf{E})\times\mathbf{C}\times\nabla F_W^{\eta}(\mathbf{E})}{\ln(10)F_V^{\varepsilon}(\mathbf{E})\times\ln(10)F_W^{\eta}(\mathbf{E})} \qquad (30)$$

where

$$\varepsilon \text{ (resp. } \eta) = \begin{cases} + & \text{if pattern } V \text{ (resp. } W) \text{ is over-represented} \\ - & \text{if pattern } V \text{ (resp. } W) \text{ is unter-represented} \end{cases}. \qquad (31)$$

And after using results of sections "single pattern" and "under-represented pattern" we finally get

$$\sigma_{V,W} = \left(Q_V^{\varepsilon} Q_W^{\eta}\right)\times\left({}^{t}\nabla\mathbf{G}_V \times\mathbf{C}\times\nabla\mathbf{G}_W\right) \qquad (32)$$

where Q_V^{ε} (resp. W) and $\mathbf{G}_V$ (resp. W) are the constant Q (Q^{+} and Q^{-}) and the vector **G** for the pattern V (resp. W).

Simulations

It is also possible to study the empirical distribution of a $S_{\mathbf{N}}$ (for one or more patterns) through simulations.

In order to do so, we first draw M independent sequences $Y^j = Y_1^j \ldots Y_n^j$ using an order m stationary Markov model of parameters π. Complexity of this step is $O(M \times n)$.

For each j we get the frequencies $\mathbf{N}^j = (\mathbf{N}_0^j, \mathbf{N}_1^j)$ (with complexity $O(n)$ for each sequence) of the words of size m and $m + 1$ in the sequence Y^j and use it to compute $S^j = S_{\mathbf{N}^j}$ (exact value or approximation). Complexity here depends on the statistical method used to compute S^j (*e.g.* $O(h)$ using a binomial approximation).

We now have a M – sample $S^1, \ldots, S^M$ of $S_{\mathbf{N}}$ from which we can easily estimate σ and thus, valid or invalid the approximation through the delta-method.

When used with large value of n (*e.g.* several millions or more), the complexity of this approach is slowed by the drawn of the sequences Y_j. It is therefore possible to improve the method by simulating directly the frequencies **N** through (5). As this approximation has a very small impact on the distribution of $S_{\mathbf{N}}$ (data not shown) it may dramatically speed-up the computations when considering large n or M. It is nevertheless important to point out that drawing a Gaussian vector size L requires to precompute the Choleski decomposition of its covariance matrix which could be a limiting factor when considering large L.

Results and discussion

Validation

Simple case

Let us start with a simple case: a binary alphabet $\mathcal{A}$ = {a, b} (k = 2) with an order m = 1 Markov model

$$\pi = \begin{pmatrix} 0.3 & 0.7 \\ 0.6 & 0.4 \end{pmatrix} \qquad (33)$$

which stationary distribution is μ = (6/13,7/13) and we work on a sequence of length n = 10 000.

The first thing to do is to compute **E** and **C** (see appendix A for details).

Now, we consider the pattern W = ababa occurring N_{obs} = 1221 times in a sequence of length $\ell = n$ = 10 000. We have

$$p = \mu(\text{a})\,\Pi\,(\text{a,b})^2\,\Pi\,(\text{b,a})^2 = 8.142 \times 10^{-2} \qquad (34)$$

so $\mathbb{E}[N(\text{ababa})] = (\ell - 4)p = 813.8 \simeq 0.66 \times N_{\text{obs}}$ and hence the pattern is over-represented. Its statistic (using binomial approximation) is

$$S \simeq \ell - \log_{10} \mathbb{P}(\mathcal{B}(\ell - 5 + 1, p) \geq N_{\text{obs}}) = 43.74285 \qquad (35)$$

We have

$$Q^+ = \frac{p^{N_{\text{obs}}-1}(1-p)^{\ell-4-N_{\text{obs}}}}{\ln(10)\beta(p, N_{\text{obs}}, \ell - 3 - N_{\text{obs}})} = 193.3258 \qquad (36)$$

and

$${}^t\mathbf{G}_0 = \begin{bmatrix} \frac{-1}{\mathbf{E}_0(\text{a})} & \frac{-2}{\mathbf{E}_0(\text{b})} \end{bmatrix} = \begin{bmatrix} -2.17\times10^{-5} & -3.71\times10^{-5} \end{bmatrix} \qquad (37)$$

and

$${}^t\mathbf{G}_1 = \begin{bmatrix} 0 & \frac{2}{\mathbf{E}_1(\text{ab})} & \frac{2}{\mathbf{E}_1(\text{ba})} & 0 \end{bmatrix} = \begin{bmatrix} 0 & 6.19\times10^{-5} & 6.19\times10^{-5} & 0 \end{bmatrix} \qquad (38)$$

Finally, we get

$$\sigma = Q^+ \sqrt{{}^t\mathbf{G}\times\mathbf{C}\times\mathbf{G}} = 6.1020774 \qquad (39)$$

As our pattern statistics is the decimal logarithm of the p-value, σ = 6 means that the ratio of the estimated p-value over the true one could easily range from 10^{-12} ($10^{-2\times\sigma}$) to 10^{12} ($10^{2\times\sigma}$) which is huge.

We can see on fig. 1 the empirical distribution of $S_{\mathbf{N}}$ compared to the theoretical distribution. Even if the two distributions are closely related, an adjustment test (Kolmogorov-Smirnov) shows that they are different.

In the fig. 2 we compare σ to its estimator $\hat{\sigma}$ for several values of N_{obs}. We can see that our theoretical values of σ fits very well to the empirical ones.

The equation (39) gives an explicit expression of σ as a product of two terms. Once the pattern and the true parameter π are fixed, the first term (Q) depends only on ℓ and N_{obs} while the second one only depends on the length n of the sequence used for the parameter estimation (see appendix C for an explicit expression of σ in the particular case of an order 0 Markov model).

To study the variations of $\sigma(n)$ as a function of n we therefore need to study $\mathbf{G}(n)$ and $\mathbf{C}(n)$. Using equations (6) and (22) we get that

$$\mathbf{E}(n) = O(n) \quad \text{and} \quad \mathbf{G}(n) = O\left(\frac{1}{n}\right) \qquad (40)$$

Using equations (57) and (58) in appendix A we also get that $\mathbf{C} = \mathbf{M} + \mathbf{O} + {}^t\mathbf{E}\mathbf{E}$ with

$$\mathbf{M}(n) = O(n^2) \text{ and } \mathbf{O}(n) = O(n) \qquad (41)$$

so finally

$$\sigma(n) \simeq \tilde{\sigma}(n) = Q^+ \times \sqrt{A + \frac{B}{n}} \qquad (42)$$

for large n, with

$$A = \lim_{n\to+\infty} {}^t\mathbf{G}(\mathbf{C} - \mathbf{O})\mathbf{G} \qquad (43)$$

and

$$B = \lim_{n\to+\infty} n \times {}^t\mathbf{G}\mathbf{O}\mathbf{G} \qquad (44)$$

We can see on fig. 3 that $\tilde{\sigma}$ is not a very good approximation of σ for small n, but, as the approximation is far easier to compute (and trivial to invert) than the true value, this can be useful when we need to compute a minimum length n to obtain a given σ.

We also see on the same figure that σ grows rapidly when n decreases. For example, we get $\sigma \simeq 20$ for n = 5000 (while equation (35) gives $S \simeq 264.4$).

As we consider here a binary alphabet (k = 2) and a first order Markov model (m = 1) we have only $k^m(k - 1) = 2$ parameters to estimate with a sample of size n = 5000 (so we have 2500 sample per parameter). Although this situ-

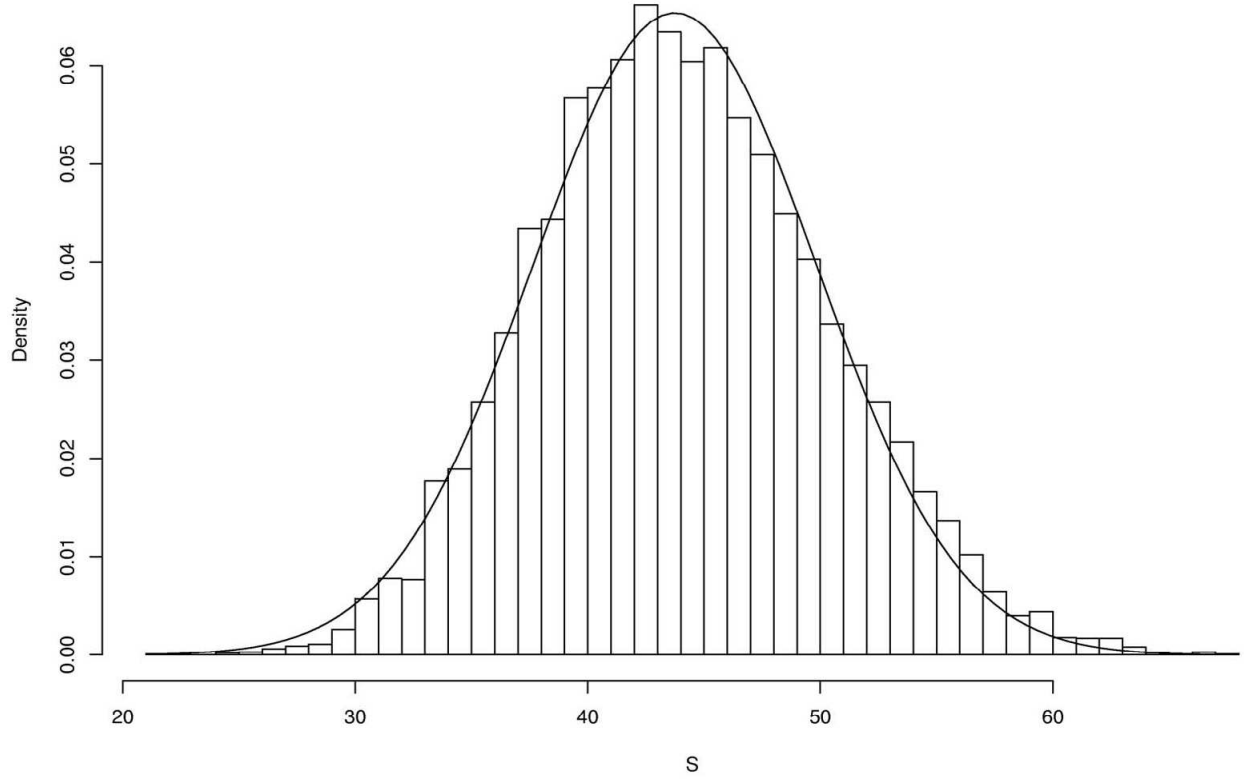


Figure 1 $\hat{S}$

Empirical and theoretical distributions of $\hat{S}$. A sample of size 10 000 have been used to get the empirical distribution. The solid line represents the density of $\mathcal{N}(S, \sigma^2)$. The adjustment test of Kolmogorov-Smirnov give $D = 0.023$ which corresponds to a p-value of $p = 5.3 \times 10^{-5}$. $N_{obs} = 1221$ and $n = \ell = 10\,000$.

ation seems quite comfortable, the sensitivity to parameter estimation appears in fact to be so large that we could have a factor 10^{40} between the true p-value and its estimate.

Practical case

We have seen with our first example that our approximation works very well in a simple case. Will this hold with more practical cases?

To answer this question, let us consider the following experimental design:

• one pattern: W = acgtacgt;

• two genomes: *Escherichia coli* K12 ($\ell = n = 4639675$) and *Mycoplasma genitalium* ($\ell = n = 580076$);

• five Markov orders: $m = 1$ to $m = 5$ (larger m are not considered since the computation of **C** becomes then intractable).

As the sequence lengths and compositions of the two considered genomes differ a lot, we have to take a different value of N_{obs} for each organism: $N_{obs} = 30$ for *M. genitalium* and $N_{obs} = 150$ for *E. coli*. Proceeding as indicated in section "simulations", we use the algorithm 1 for each experiment.

Algorithm 1 simulations for one experiment in the practical case

1: estimate the order m parameter π (and μ) from the original sequence. Although these parameters are estimated, they are considered as the true parameters;

2: compute $S = -\log_{10}(N \geq N_{obs})$;

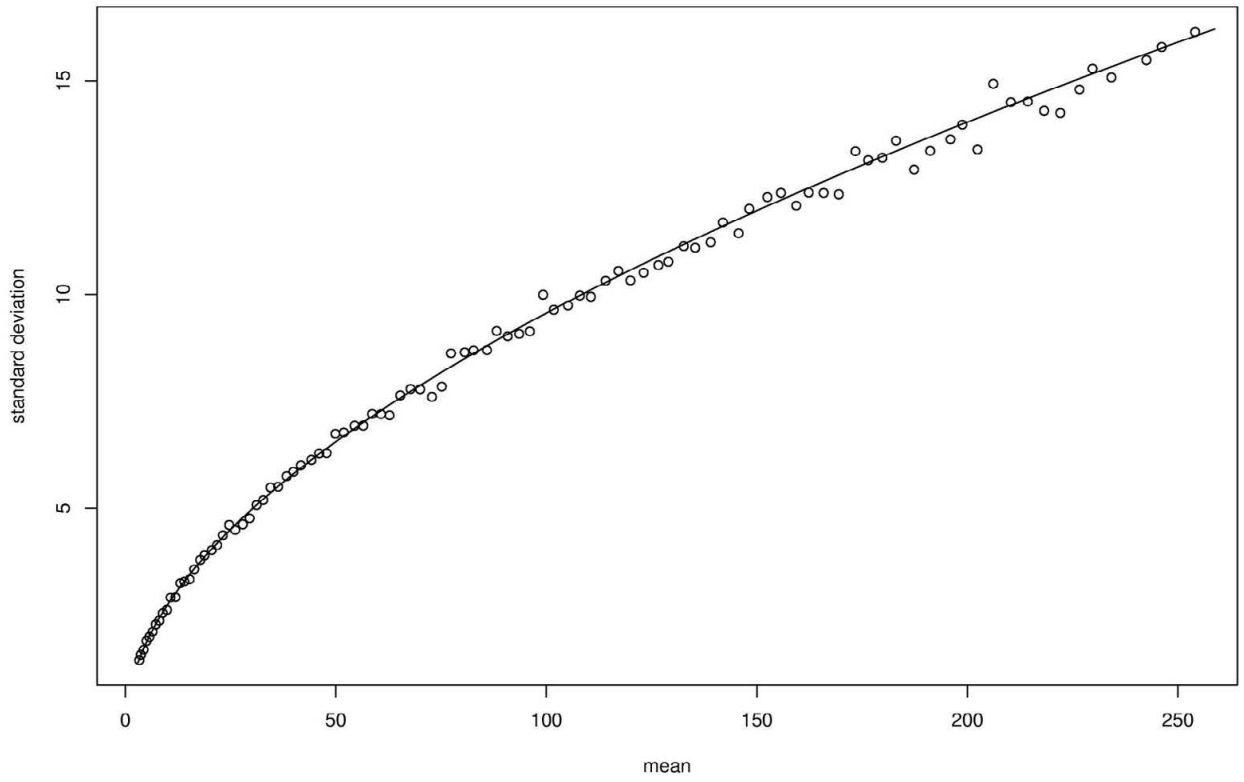


Figure 2 $\hat{\sigma}$

Comparison of σ and $\hat{\sigma}$. $\hat{\sigma}$ is estimated with a sample of size 1 000 and N_{obs} takes its values from 900 to 1 900. The solid line represents the theoretical values and the circles the empirical ones. The statistic S is used on the x-axis. $n = \ell = 10\,000$.

3: compute σ using approximation (23)

4: **for** $j = 1 \ldots 1\,000$ **do**

5: draw a random sequence $Y = Y_1 \ldots Y_n$ according to and order m stationary Markov model of parameter π;

6: compute **N** the frequency vector of all size m and size $m + 1$ words in Y;

7: compute $S^j = S_{\mathbf{N}} = -\log_{10}(N \geq N_{obs})$;

8: **end for**

9: compute $\hat{S}$ (resp. $\hat{\sigma}$) the mean (resp. standard deviation) of the sample $S^1, \ldots, S^j$.

We can see on table 1 the results for *E. coli*. For each Markov model considered, our approximation of σ is very close to the empiric ones and, as with figure 1, the Gaussian distribution fit well to the empiric one (data not shown). Table 2 shows the same behaviour with *M. genitalium* except for $m = 5$ where $\hat{\sigma}$ differs slightly more than in the other cases from its theoretical value. To understand this phenomenon, let us first recall the expression of $P(\mathbf{N})$ for $m = 5$ using equation (15):

$$P(\mathbf{N}) = \frac{\mathbf{N}_1(\text{agctac}) \times \mathbf{N}_1(\text{gctacg}) \times \mathbf{N}_1(\text{ctacgt})}{(\ell - m + 1) \times \mathbf{N}_0(\text{gctac}) \times \mathbf{N}_0(\text{ctacg})}$$

and as $(\mathbf{N}_1(\text{agctac}) = 0) \simeq 2.26 \times 10^{-6}$, $(\mathbf{N}_1(\text{gctacg}) = 0) \simeq 1.35 \times 10^{-1}$ and $(\mathbf{N}_1(\text{ctacgt}) = 0) \simeq 1.24 \times 10^{-4}$ we will have $P(\mathbf{N}) = 0$ roughly 14% of the time. This happened 123 times in our sample of size 1 000, each time preventing to compute $S_{\mathbf{N}}$. The sample is hence biased and $\hat{S}$ and $\hat{\sigma}$ are therefore not accurate.

What happen now if we use another statistical method to compute the pattern statistics. As the binomial approximation is supposed to be close to the exact solution, we expect the standard deviation obtained with other statisti-

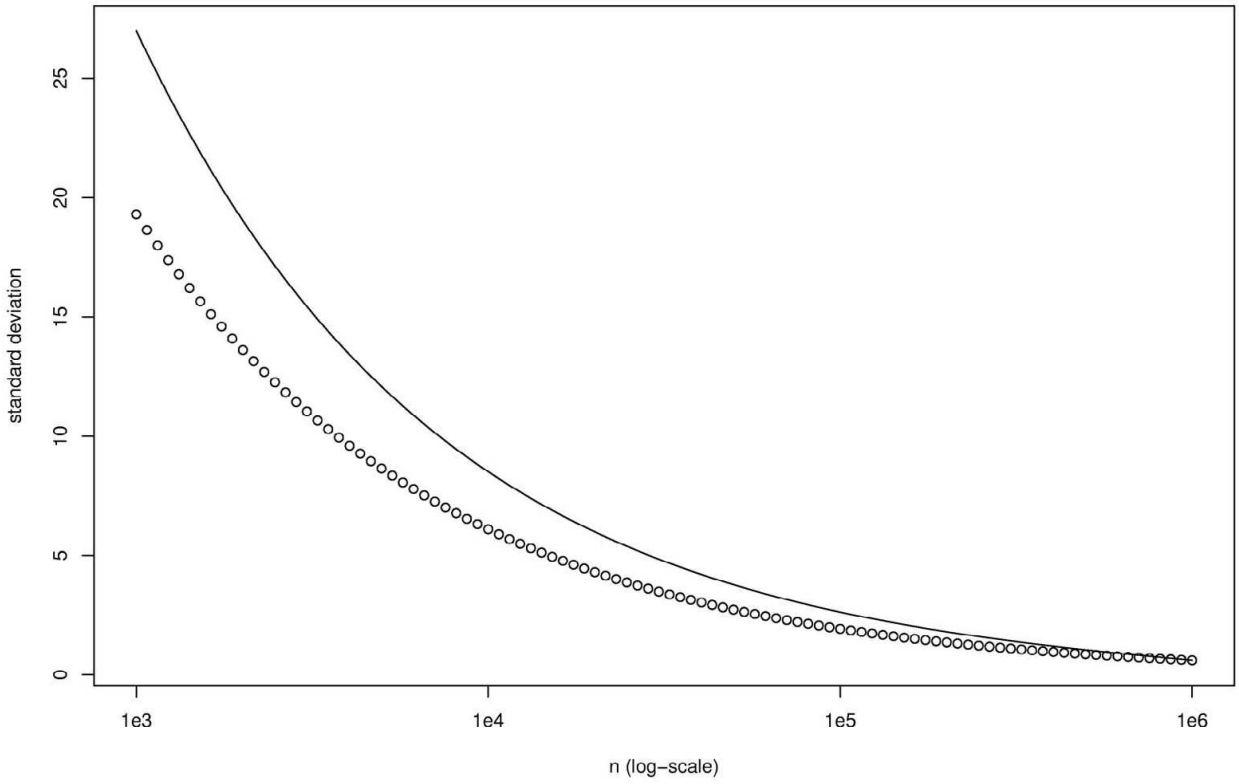


Figure 3 $\tilde{\sigma}$

Comparison of $\sigma(n)$ and $\tilde{\sigma}(n)$. The circles reprensent $\sigma(n)$ and the solid line $\tilde{\sigma}(n)$. $n_\infty = 10^6$ have been used to compute the value of A and B. $N_{obs} = 1221$ and $\ell = 10\,000$.

cal methods to remain close to σ. In table 3, we compare the empirical results using binomial approximations (like above) but also compound Poisson or large deviations approximations. Both empirical means and standard deviations are close to the theoretical ones thus validating the method.

Choice of a Markov model order

Through the computation of σ we can measure the sensitivity of pattern statistics to parameter estimations. A very natural question is then, how this variability could affect a pattern statistic study, and, as this variability grows with the Markov model order, how to choose this parameter.

Table 1: Comparison of theoretical and empirical pattern statistic mean and standard deviation on *Escherichia coli* K12.

m	S	σ	$\hat{S}$	$\hat{\sigma}$
1	35.57	0.28	35.57	0.27
2	31.61	0.49	31.60	0.50
3	46.75	1.04	46.77	1.03
4	45.33	1.74	45.32	1.81
5	62.27	3.45	62.36	3.34

We consider the pattern W = acgtacgt with $N_{obs} = 150$. The sequence length is $\ell = 4639675$, we use an order m Markov model and a sample of size $M = 1\,000$.

Table 2: Comparison of theoretical and empirical pattern statistic mean and standard deviation on *Mycoplasma genitalium*.

m	S	σ	$\hat{S}$	$\hat{\sigma}$
1	42.48	0.38	42.47	0.40
2	44.62	0.78	44.62	0.81
3	55.96	1.49	56.02	1.52
4	55.06	3.39	55.48	3.48
5	56.49	10.35	57.21*	9.09*

We consider the pattern W = acgtacgt with N_{obs} = 30. The sequence length is ℓ = 580 076, we use an order m Markov model and a sample of size M = 1 000. (*) for 123 terms in the sample we got P (**N**) = 0 and hence, $S_{\mathbf{N}}$ was not computed.

We propose here to consider the case of a very simple pattern study: we want to find the 100 most over-represented octamers (DNA words of size 8) in a given genome. Assuming the true parameter π (and hence μ) is known, we can compute REF = $\{W_1,..., W_{100}\}$, the list of these words (ordered by decreasing statistics, so that the most over-represented one is the first one).

For each estimates $\hat{\mu}$ and $\hat{\pi}$, we can compute $\widehat{\text{REF}}$ the 100 most over-represented octamers in the genome using the statistic $\hat{S}$ and compare it to the truth. In order to do so, we first compute the true positive rate (TP rate) defined by the rate of common words in $\widehat{\text{REF}}$ and REF, and the rank accordance rate (RA rate) defined by the Kendall's tau [[15], Chapter 13] between S and $\hat{S}$ ranks of $\{\widehat{\text{REF}} \cup \text{REF}\}$. Such statistic is in the range [-1,1] and has the value 1 for the complete rank accordance and the value -1 for the complete rank discordance.

As in the section "practical case", we consider two genomes: *Escherichia coli* K12 ($\ell = n$ = 4639675) and *Mycoplasma genitalium* ($\ell = n$ = 580076). For each Markov model order m from 1 to 6, we estimate π on the sequence (by maximum of likelihood), compute the REF list and then draw a sample of $\widehat{\text{REF}}$ from which we get estimates for the expectation of TP and RA rates.

Results are given in tables 4 and 5. We can see that, surprisingly, the TP rate could be very low even for long genome such as *E. coli* when high order Markov model (m = 6) are used. Of course, these rates are even worse on *M. genitalium* whose genome is ten times smaller than the first one. It is also clear that the RA rate is more affected by the variability induced by parameter estimation than the TP rate.

Based on these results, we conclude that our pattern study requires a sample size per free parameter of at least a few thousands if we want reliable results. In our examples this has for consequence that the Markov order should not be greater than 4 (or 5 at the very most) for *E. coli* and 3 (or 4 at the very most) on *M. genitalium* without resulting in important errors.

Conclusion

The delta-method allows us to approximate the distribution of $\hat{S}$ by a Gaussian distribution. This first requires to compute the expectation and covariance matrix of frequencies and then to study the derivative of a function which is specific of the method used to compute the pattern statistics. In the case of the binomial approximations, we have found an explicit expression of σ the standard deviation of $\hat{S}$.

It is clear that our approximation of σ using the delta-method relies one two major assumptions: 1) the distribution of **N** is Gaussian; 2) F^+ is regular enough (*e.g.* not

Table 3: Comparison of theoretical and empirical pattern statistics mean and deviation on *Mycoplasma genitalium*.

theoretical		binomial		compound Poisson		large deviations	
S	σ	$\hat{S}$	$\hat{\sigma}$	$\hat{S}$	$\hat{\sigma}$	$\hat{S}$	$\hat{\sigma}$
55.96	1.49	56.05	1.47	55.42	1.45	54.27	1.43

We consider the pattern W = acgtacgt with N_{obs} = 30. The sequence length is ℓ = 580076, we use an order m = 3 Markov model and a sample of size M = 1 000. The pattern statistics are computed (from left to right) through binomial, compound Poisson or large deviations approximations.

Table 4: Mean true positive rate and rank accordance rate in *Escherichia coli* K12.

Markov order	1	2	3	4	5	6
TP rate	99.0%	98.0%	97.9%	94.4%	82.1%	47.6%
RA rate	99.0%	95.5%	91.5%	83.9%	68.0%	36.5%
$\times 10^3$	383.33	95.83	23.96	5.99	1.50	0.37

Both quantities are estimated with 1 000 simulations. We consider the 1 00 most over-represented octamers, the sequence length is ℓ = 4639675. The last row gives the sample size per free parameter (length n of the sequence divided by the number $k^m(k - 1)$ of parameters).

too steep) around **E**. When m grows, **E** closes to the boundary of the definition range of F^+ hence degrading assumption 2. Moreover, it is well known that Gaussian approximations for word frequencies become weaker when the expected numbers of their occurrences become smaller, thus degrading assumption 1. It is therefore obvious that our approximation of σ will get less and less reliable as m grows.

However, the approximation of σ has been validated through simulations and appears to be very reliable (even for m = 5 or 6). As pattern statistics computed through binomial approximations are close to the exact statistics [8], the value of σ should not differ a lot when another statistical method is used. We have compared our approximations to the empiric distribution obtained using compound Poisson and large deviations approximations and, as expected, our approximations remains quite reliable even for these statistical methods.

The variability due to parameter estimation is of course related to the Markov model order m and to the size k of the alphabet (as we have k^{m+1} parameters for this model) and to the length n of the sequence used for this estimation. For example, considering an order m = 6 model with n = 4639675 (*Escherichia coli* K12 complete genome) requires to estimate $3 \times 4^6 = 4096$ free parameters which results roughly in 400 observation per free parameter. Although this situation seems quite comfortable, we have seen with our simulations that it leads an unacceptable variability for pattern statistics.

As literature often advices to use the highest possible Markov order for a given pattern problem (which means $m = h - 2$ for pattern of size h) it is easy to understand that such a practice could have very detrimental effects on the computed statistics unless huge data are available for estimation purpose. Even if we consider the more reasonable attitude to choose m using the classical framework of model selection (*e.g.* using the Akaike Information Criterion – AIC –) we get m = 5 for *Mycoplasma genitalium* and m = 6 for *Escherichia coli* K12 hence resulting in both cases in the same catastrophic results in terms of false positive and even worse ones in terms of ranking.

Moreover, we assumed here that our model was homogeneous all along the considered sequences. This is obviously completely false when complete genomes are considered. So it is more likely that the sample size n would be far smaller than a million on classical pattern studies (even of human genomes for example). As a result, the variability we pointed out in this paper will have a considerable detrimental effect on most studies unless the Markov order is carefully set.

In order to do so, we advice to compute our approximation of σ each time a pattern statistic is produced and then to evaluate, either by simulation (like in this paper) or by a theoretical work the impact of this variability on the considered study.

Competing interests

The author declares that he has no competing interests.

Appendix A

We give here the expression of the covariance matrix **C** introduced in section "distribution of $\mathbf{N} = (\mathbf{N}_0, \mathbf{N}_1)$". The sequence Y (of length n) is generated by an homogeneous, stationary and ergodic order m Markov model of parameter π and stationary distribution μ. We want to compute the covariance of the vector **N** of random frequencies of size m and m + 1 words.

Table 5: Mean true positive rate and rank accordance rate in *Mycoplasma genitalium*.

Markov order	1	2	3	4	5	6
TP rate	95.5%	93.6%	90.4%	81.8%	66.0%	25.0%
RA rate	92.6%	85.4%	79.8%	66.5%	45.1%	11.0%
$\times 10^3$	48.33	12.08	3.02	0.76	0.19	0.05

Both quantities are estimated with 1 000 simulations. We consider the 1 00 most over-represented octamers, the sequence length is ℓ = 580076. The last row gives the sample size per free parameter (length n of the sequence divided by the number $k^m(k - 1)$ of parameters).

For any word w (of size h_w), we introduce the following notation for $h_w \le i \le n$

$$I_i(w) = \mathbb{I}_{\{w \text{ end in position } i\}} = \mathbb{I}_{\{Y_{i-h_w+1}^{i} = w\}} \quad (45)$$

where $Y_i^j = Y_i \dots Y_j$ for all $i \le j$. If $h_w \ge m$, we denote by

$$p(w) = \mu(w_1^m)\Pi(w_1^m, w_{m+1})\dots\Pi(w_{h_w-m}^{h-1}, w_h) \quad (46)$$

the probability to see one occurrence of w at a given position in the sequence. At last, if we consider another word v (of size $h_v = m$) and if $h_w = m$, we denote by

$$\Pi_\delta(v,w) = \sum_{x \in \mathcal{A}^\delta} p(vxw) \quad (47)$$

the probability to see occurrences of v and w separated by a gap of length δ.

For any words v an w (to simplify, we suppose that $h_v \ge h_w$) then, for all $\delta \in \mathbb{Z}$ and

$\max(h_v, h_w - \delta) \le i \le \min(n, n - \delta)$ we have

$$\mathbb{E}\,[I_i(v)\, I_{i+\delta}(w)] = D_\delta(v,w) \quad (48)$$

which do not depend on i.

It is therefore easy to show that

$$\begin{aligned}\mathbb{E}[\mathbf{N}(v)\mathbf{N}(w)] &= \sum_{i=h_v}^{n} \sum_{\delta=h_w-i}^{n-i} D_\delta(v,w) && (49)\\ &= \sum_{\delta=h_w-n}^{n-h_v} N_\delta D_\delta(v,w) && (50)\\ &= \mathbf{M}(v,w) + \mathbf{O}(v,m) && (51)\end{aligned}$$

where the main part ($2n - h_v - h_w + 2$ terms) is given by

$$\mathbf{M}(v,w) = \sum_{\delta=h_v}^{n-h_w} N_{-\delta} D_{-\delta}(v,w) + \sum_{\delta=h_w}^{n-h_v} N_\delta D_\delta(v,w) \quad (52)$$

and the overlapping part ($h_v + h_w - 1$ terms) by

$$\mathbf{O}(v,w) = \sum_{\delta=-h_v+1}^{h_w-1} N_\delta D_\delta(v,w) \quad (53)$$

and with

$$N_\delta = \begin{cases} n - h_w + 1 + \delta & \delta \in [h_w - n, h_w - h_v[\\ n - h_v + 1 & \delta \in [h_w - h_v, 0] \\ n - h_v + 1 - \delta & \delta \in]0, n - h_v] \end{cases} \quad (54)$$

As we have

$$\mathbf{C}(v,w) = \mathbf{M}(v,w) + \mathbf{O}(v,w) - \mathbf{E}(v)\,\mathbf{E}(w) \quad (55)$$

the problem is hence to compute **M** and **O** for all pairs of size m or $m + 1$ words. In order to simplify, we will just treat here the case of a pair of size m words (other cases can be derived from this special case).

For the main part we obtain

$$\begin{aligned}\mathbf{M}(v,w) = &\sum_{\delta=m}^{n-m} N_{-\delta}\mu(w)\Pi_{\delta-m+1}(w,v) \\ &+ \sum_{\delta=m}^{n-m} N_\delta \mu(v)\Pi_{\delta-m+1}(v,w)\end{aligned} \quad (56)$$

($2n - 2m + 2$ terms). As $P_k(v,w)$ quickly converges toward $\mu(w)$ when k grows (convergence speed is given by λ^k where λ is the magnitude of the second eigenvalue of the transition matrix Π). So there exists a rank $r \ge m$ such as

$$\begin{aligned}\mathbf{M}(v,w) \simeq\ &\mu(v)\mu(w)\sum_{\delta=r}^{n-m}(N_{-\delta} + N_\delta) \\ &+ \sum_{\delta=m}^{r-1} N_{-\delta}\mu(w)\Pi_{\delta-m+1}(w,v) \\ &+ \sum_{\delta=m}^{r-1} N_\delta \mu(v)\Pi_{\delta-m+1}(v,w)\end{aligned} \quad (57)$$

which has only $2r - 2m + 1$ terms.

And for the overlapping part we get

$$\begin{aligned}\mathbf{O}(v,w) = &N_0 \times \mu(v) \times \mathbb{I}_{\{v=w\}} \\ &+ \sum_{\delta=1}^{m-1} N_{-\delta} \times p(w v_{m-\delta+1}^m) \times \mathbb{I}_{\{v_1^{m-\delta} = w_{1+\delta}^m\}} \\ &+ \sum_{\delta=1}^{m-1} N_\delta \times p(v w_{m-\delta+1}^m) \times \mathbb{I}_{\{v_{1+\delta}^m = w_1^{m-\delta}\}}\end{aligned} \quad (58)$$

which has $2m + 1$ terms.

So the overall complexity for the computation of one term of **C** is hence $O(r)$ where the value of r is directly connected to the magnitude λ of the second eigenvalue of the transition matrix.

In the particular case of an order one Markov model (m = 1), we give here the complete expressions of $\mathbf{M}$ and $\mathbf{O}$.

For all $a, b, c, d \in \mathcal{A}$, we have

$$\mathbf{M}(a,b) \simeq (n-r+1)(n-r)\mu(a)\mu(b) + \sum_{\delta=1}^{r-1}(n-\delta)\Big(\mu(b)\Pi^{\delta}(b,a) + \mu(a)\Pi^{\delta}(a,b)\Big) \quad (59)$$

$$\mathbf{O}(a,b) = n\mu(a)\,\mathbb{I}_{\{a=b\}} \quad (60)$$

$$\frac{\mathbf{M}(ab,c)}{\Pi(a,b)} \simeq (n-r)(n-r-1)\mu(a)\mu(c) + \sum_{\delta=1}^{r-1}(n-\delta-1)\Big(\mu(c)\Pi^{\delta}(c,a) + \mu(a)\Pi^{\delta}(b,c)\Big) \quad (61)$$

$$\frac{\mathbf{O}(ab,c)}{\Pi(a,b)} = (n-1)\mu(a)(\mathbb{I}_{\{a=c\}} + \mathbb{I}_{\{b=c\}}) \quad (62)$$

$$\frac{\mathbf{M}(ab,cd)}{\Pi(a,b)\Pi(c,d)} \simeq (n-r-1)(n-r-2)\mu(a)\mu(c) + \sum_{\delta=1}^{r-1}(n-\delta-2)\Big(\mu(c)\Pi^{\delta}(d,a) + \mu(a)\Pi^{\delta}(b,c)\Big) \quad (63)$$

$$\frac{\mathbf{O}(ab,cd)}{\Pi(a,b)} = (n-1)\mu(a)\mathbb{I}_{\{ab=cd\}} + (n-2)\Pi(c,d)\Big(\mu(c)\mathbb{I}_{\{a=d\}} + \mu(a)\mathbb{I}_{\{b=c\}}\Big) \quad (64)$$

With the example given in section "validation" we get for the expectation

$$\mathbf{E}_0^t = [4615.4\ 5384.6] \quad (65)$$

and

$$\mathbf{E}_1^t = [1384.5\ 3230.4\ 3230.4\ 2153.6] \quad (66)$$

The magnitude of the second eigenvalue of Π is λ = 0.3, then rank r = 19 give a relative error < 10^{-10} and we get for the covariance

$$\mathbf{C}_{0,0} = \begin{bmatrix} 1338.28 & -1338.28 \\ -1338.28 & 1338.28 \end{bmatrix} \quad (67)$$

$$\mathbf{C}_{1,0} = \begin{bmatrix} 1146.9 & 191.2 & 191.2 & -1529.2 \\ -1146.9 & -191.2 & -191.2 & 1529.2 \end{bmatrix} \quad (68)$$

and

$$\mathbf{C}_{1,1} = \begin{bmatrix} 1536.8 & -390.0 & -390.0 & -756.9 \\ -390.0 & 581.2 & 581.0 & -772.2 \\ -390.0 & 581.0 & 581.2 & -772.2 \\ -756.9 & -772.2 & -772.2 & 2301.4 \end{bmatrix} \quad (69)$$

Appendix B

The beta function is defined by

$$\beta(a,b) = \int_0^1 t^{a-1}(1-t)^{b-1}\,dt \quad (70)$$

for all $a, b > 0$. The incomplete beta function for all $x \in [0,1]$ is then defined by

$$\beta(x,a,b) = \int_0^x t^{a-1}(1-t)^{b-1}\,dt \quad (71)$$

and

$$\beta^{-}(x,a,b) = \beta(a,b) - \beta(x,a,b) \quad (72)$$

$$= \int_x^1 t^{a-1}(1-t)^{b-1}\,dt \quad (73)$$

Using a continued fraction representation, these functions can be quickly numerically evaluated in $O(\sqrt{\max(a,b)})$ in the worst case [15, Chapter 6].

A great interest of this function is that it is connected to the cumulative distribution function of a binomial distribution by the following relation:

$$\mathbb{P}(\mathcal{B}(n,p) \geq k) = \frac{\beta(p,k,n-k+1)}{\beta(k,n-k+1)} \quad (74)$$

with $(n, k) \in {}^* \times {}^*$, $0 \leq k \leq n$ and $p \in [0,1]$.

Finally, let us remark that the incomplete beta function is differentiable in x and that

$$\frac{\partial\beta(x,a,b)}{\partial x} = x^{a-1}(1-x)^{b-1} \quad (75)$$

Appendix C

We give here the complete expression of σ for a single pattern in the special case of an order m = 0 homogeneous Markov model of parameter μ.

The MLE of μ is given by

$$\mu_{\mathbf{N}} = \frac{\mathbf{N}_1}{n} \quad (76)$$

where $\mathbf{N}_1$ is the frequency of all letters.

A Gaussian approximation gives

$$\mathcal{L}(\mathbf{N}_1) \simeq \mathcal{N}(\mathbf{E}_1, \mathbf{C}_{1,1}) \quad (77)$$

with $\mathbf{E}_1 = n\mu$ and, for all $a, b \in \mathcal{A}$,

$$\mathbf{C}_{1,1}(a, b) = n\mu(a)\,\mathbb{I}_{a=b} - n\mu(a) \times n\mu(b) \quad (78)$$

We have also

$$P(\mathbf{N}) = \frac{1}{n^h} \prod_{a \in \mathcal{A}} \mathbf{N}_1(a)^{A_1(a)} \quad (79)$$

which implies for all $a \in \mathcal{A}$ that

$$\frac{\partial P(\mathbf{N})}{\partial \mathbf{N}_1(a)} = \frac{A_1(a)}{\underbrace{\mathbf{N}_1(a)}_{\mathbf{G}_1(a)}} \times P(\mathbf{N}) \quad (80)$$

So finally we get

$$\sigma \simeq Q\sqrt{{}^t\mathbf{G}_1 \times \mathbf{C}_{1,1} \times \mathbf{G}_1} \quad (81)$$

where Q is either defined by equation (24) if the pattern is over-represented or by equation (28) if under-represented.

References

1. Atteson W: **Calculating the exact probability of language-like patterns in biomolecular sequences.** *Pro 6th Int Conf on Intelligent Systems for Molecular Biology* 1998:17-24.
2. Régnier M, Szpankowski W: **On pattern frequency occurrences in a Markovian sequence.** *Algorithmica* 1998, **22(4):**631-649.
3. Robin S, Daudin JJ: **Exact distribution of word occurrences in a random sequence of letters.** *J App Prob* 1999, **36:**179-193.
4. Nuel G: **Effective p-value computations using Finite Markov Chain Imbedding (FMCI): application to local score and to pattern statistics.** *Algorithms Mol Biol* 2006, **1(1):**5.
5. Kleffe J, Borodovski M: **First and second moment of counts of words in random text generated by Markov chains.** *Comp Applic Biosci* 1992, **8:**443-441.
6. Prum B, Rodolphe F, de Turckheim E: **Finding words with unexpected frequencies in DNA sequences.** *J R Statist Soc B* 1995, **11:**190-192.
7. van Helden J, André B, Collado-Vides J: **Extracting Regulatory Sites from the Upstream Region of Yeast Genes by Computational Analysis of Oligonucleotide Frequencies.** *J Mol Biol* 1998, **281:**827-842.
8. Nuel G: **S-SPatt: Simple Statististics for Patterns on Markov chains.** *Bioinformatics* 2005, **21(13):**3051-3052.
9. Chrysaphinou O, Papastavridis S: **A limit theorem on the number of overlapping appearances of a pattern in a sequence of independent trials.** *Proba Theory Relat Fields* 1988, **79(1):**129-143.
10. Arratia R, Goldstein L, Gordon L: **Poisson approximation and the Chen-Stein method.** *Stat Sci* 1990, **5(4):**403-434.
11. Schbath S: **Compound Poisson approximation of word counts in DNA sequences.** *ESAIM Probab Stat* 1995, **1:**1-16.
12. Nuel G: **LD-SPatt: Large Deviations Statistics for Patterns on Markov Chains.** *J Comput Biol* 2004, **11(6):**1023-1033.
13. Oehlert GW: **A note on the delta method.** *American Statistician* 1992, **46:**27-29.
14. Reinert G, Schbath S, Waterman M: **Chapter 6: Statistics on words with applications to biological sequences.** In *Applied Combinatorics on Words* Cambridge Universtity Press; 2005.
15. Press WH, Teukolsky SA, Vetterling WT, Flannery BP: *Numerical Recipes in C* 2nd edition. Cambridge Universtity Press; 1988.

Chapitre 13

Distribution of patterns on Markov chains : a unified approach using deterministic finite state automata

Article de Nuel actuellement soumis (depuis juin 2006) au *Journal of Applied Probability*. Cet article présente la notion de PMC (*Pattern Markov Chain*) et ses applications aux FMCI, aux approximations gaussiennes, binomiales, Poisson composées et aux grandes déviations.

Applied Probability Trust (19 June 2006)

DISTRIBUTION OF PATTERNS ON MARKOV CHAINS: AN UNIFIED APPROACH USING DETERMINISTIC FINITE STATE AUTOMATA

GRÉGORY NUEL,* *University of Evry*

Abstract

In this paper, we use the well known theory of language and automata to unify a wide range of probabilistic methods allowing to study the distribution of pattern in random sequences. In the independent case, we introduce the notion of Pattern Markov Chain (PMC) and see how to use it to obtain exact distributions (number of occurrences and waiting time) or approximations (Large deviations, Gaussian, binomial, compound Poisson). These results are then extended to order m Markov dependent sequences (introducing the notion of non ambiguous automata) and to renewal occurrences. Finally, we see on several applications (simple patterns, gapped patterns, Prosite patterns and scan statistics) how efficient this new approach is, allowing for the first time to deal efficiently with highly degenerated patterns.

Keywords: exact distribution; large deviations; Poisson approximations; Gaussian approximations

AMS 2000 Subject Classification: Primary 65C40

Secondary

1. Introduction

The distribution of patterns on random texts (independent or Markov) has been extensively studied for its numerous applications in many fields (linguistic, reliability, insurance, computational biology, ...). A wide range of approaches have been proposed by the literature. These methods can be categorized in two classes: exact methods and asymptotic approximations. In both cases, a huge amount of references are available and we quote here only a selected sample.

Many exact methods aim to compute cumulative generating function in order to derive moments and cumulative probabilities through Taylor expansions (25; 31; 30). Other methods prefer to compute proba-

* Postal address: Laboratoire Statistique et Génome, CNRS (8071), INRA (1152), UEVE, Tour Evry 2, 523 place des terrasses, 91034 Evry cedex, France

bilities directly through simple recurrences (32) or finite Markov chain imbedding (10; 11). For asymptotic approximations, Gaussian (6; 15; 27) and Poisson distributions (4; 2; 14; 12; 35; 28) are the most popular. More recently however, (20) proposed an alternative to these classical approaches with the use of large deviations to derive more reliable approximations for tail distribution events.

All these methods have their own advantages and drawbacks but they all strongly depend on the cardinality of the considered pattern (number of words it contains). As a consequence, highly degenerated patterns are usually untractable.

On another hand, it is well known from the pattern matching theory that deterministic finite state automata can provide elegant way to overcome a pattern cardinality in order to count its occurrences in a sequence. Inspiring from finite Markov chain techniques, (37; 38) first propose such an approach using results on exponential families to get moment-generating functions. However, the authors did not seen in these papers the connexion of their work to automata, using them in an implicit manner. More recently, (19) explicitly proposed the use of automata to get generating functions through the Chomsky and Schützenberger algorithm and (8) finally connected the previous results on exponential families to the automata theory.

With the help of efficient numerical algorithms (*e. g.* fast taylor expansion), it is true that moment-generating functions can provide an effective way to compute moments or p-values for patterns. However, the computational cost of the generating function could be important and, as a consequence, more straightforward approaches (like direct moment computations) are often more efficient. In this paper we intend to push forward the connexion between patterns and automata but, unlike previous work on this subject, we will focus here on a wide range of approaches rather than on the computation of generating functions.

This task leads us to make a review of the considered methods with the necessary recalls of existing results. Even if we propose also in the process few new results, the novelty in this paper rely more on the order one Markov chain (called Pattern Markov Chain) we build for each pattern problem. We give then all details about how to use it with a wide range of classical methods, thus allowing them to deal for the first time with complex patterns that was before intractable.

In a first introductory part we recall some elements on the theory of languages and automata (section 1.1) and see how it is possible to connect it to pattern (1.2). The main results are presented in the following part (2) were exact methods (2.1), large deviations (2.2), Gaussian (2.3), binomial (2.4) and compound Poisson (2.5) approximations are successively presented. The next part (3) proposes two extensions of our approach: one considers Markov chains rather than independent sequences (3.1) and the second one, renewal occurrences rather than overlapping ones (3.2). The memory and time complexities for all these methods

are then summarized in the section 3.3. Finally, we give details in the applications part (4) the automata characteristics for several kinds of practical patterns: simple ones (4.1), gapped (4.2), Prosite (4.3) and scan statistics (4.4).

1.1. Automata and languages

In this part we introduce first some classical definitions and results of the well known theory of languages and automata (see (17) for more details).

We consider $\mathcal{A} = \{a_1, \ldots, a_k\}$ a *finite alphabet* whose elements are called *letters*. A *word* (or *sequence*) over $\mathcal{A}$ is a sequence of letters and a *language* over $\mathcal{A}$ is a set of words. We denote by ε the *empty word*. For example abbaba is a word over the binary alphabet $\mathcal{A} = \{\mathrm{a}, \mathrm{b}\}$. $\mathcal{L} = \{\mathrm{ab}, \mathrm{abbaba}, \mathrm{bbbbb}\}$ is a language over $\mathcal{A}$.

The *product* $\mathcal{L}_1 \cdot \mathcal{L}_2$ (the dot could be omitted) of two languages is the language $\{w_1 w_2, w_1 \in \mathcal{L}_1, w_2 \in \mathcal{L}_2\}$ (where $w_1 w_2$ is the concatenation (or product) of w_1 and w_2. If L is a language, $\mathcal{L}^n = \{w_1 \ldots w_n, w_1, \ldots, w_n \in L\}$ and the *star closure* of L is defined by $\mathcal{L}^* = \cup_{n \geqslant 0} \mathcal{L}^n$. The language $\mathcal{A}^*$ is hence the set of all possible words over $\mathcal{A}$. For example we have $\{\mathrm{ab}\} \cdot \{\mathrm{abbaba}, \mathrm{bbbbb}\} = \{\mathrm{ababbaba}, \mathrm{abbbbbb}\}$; $\{\mathrm{ab}\}^3 = \{\mathrm{ababab}\}$ and $\{\mathrm{ab}\}^* = \{\varepsilon, \mathrm{ab}, \mathrm{abab}, \ldots\}$

A *regular language* is either the empty word, or a single letter, or obtained by union, product and star closure of regular languages. $\mathcal{A}^*$ is regular. Any finite language is regular.

Definition 1. (*DFA.*) If $\mathcal{A}$ a finite alphabet, $\mathcal{Q}$ a finite set of states, $s \in \mathcal{Q}$ a starting state, $\mathcal{F} \subset \mathcal{Q}$ a subset of final states and $\delta : \mathcal{Q} \times \mathcal{A} \to \mathcal{Q}$ a transition function then $(\mathcal{A}, \mathcal{Q}, s, \mathcal{F}, \delta)$ is a *Deterministic Finite state Automaton* (DFA). For all $a = a_1 \ldots a_{d-1} a_d \in \mathcal{A}^d$ ($d \geqslant 2$) and $q \in \mathcal{Q}$ we recursively define $\delta(q, a_1 \ldots a_{d-1} a_d) = \delta(\delta(q, a_1 \ldots a_{d-1}), a_d)$. A word $w \in \mathcal{A}^h$ is is *accepted* (or *recognized*) by the DFA if $\delta(s, w) \in \mathcal{F}$. The set of all accepted words of the DFA is called its language.

For example we can see on fig. 1 a graphical representation of a DFA on the binary alphabet $\mathcal{A} = \{\mathrm{a}, \mathrm{b}\}$ which language is $\mathcal{L} = \mathcal{A}^* \cdot \{\mathrm{bb}\}$.

We can now give the most important result of this part which is a simple application of the classical Kleene and Rabin & Scott theorems (17):

Theorem 1. *For any rational language $\mathcal{L}$ there exists a unique (up to a unique isomorphism) smallest DFA whose language is $\mathcal{L}$.*

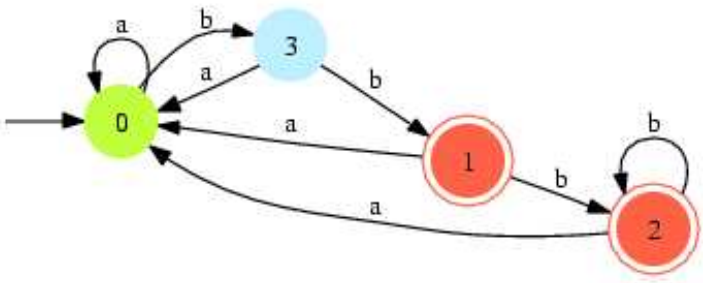


FIGURE 1: Graphical reprsentation of the DFA $(\mathcal{A}, \mathcal{Q}, s, \mathcal{F}, \delta)$ with $\mathcal{A} = \{\mathrm{a}, \mathrm{b}\}$, $\mathcal{Q} = \{0, 1, 2, 3\}$, $s = 0$, $\mathcal{F} = \{1, 2\}$ and $\delta(0, \mathrm{a}) = 0$, $\delta(0, \mathrm{b}) = 3$, $\delta(1, \mathrm{a}) = 0$, $\delta(1, \mathrm{b}) = 2$, $\delta(2, \mathrm{a}) = 0$, $\delta(2, \mathrm{b}) = 2$, $\delta(3, \mathrm{a}) = 0$ and $\delta(3, \mathrm{b}) = 1$.

1.2. Connexion with patterns

We call *pattern* over the finite alphabet $\mathcal{A}$ any finite language over the same alphabet such as no element is included into another one (this last condition is used to simplify many definitions and results by avoiding degenerated cases). For any pattern $\mathcal{W}$ any DFA that recognizes the regular language $\mathcal{A}^*\mathcal{W}$ is said to be *associated* to $\mathcal{W}$. According to theorem 1, there exists a unique (up to unique isomorphism) smallest DFA associated to a given pattern. See fig. 2 for an example.

It is well known from the pattern matching theory (5; 7) that such a DFA provides a simple way to find all occurrences of the corresponding pattern in a sequence. In the following, we will see how to exploit this remarquable property to study the distribution of patterns.

In the special case where our pattern contains only one word there is a classical way to build its smallest associated DFA:

Proposition 1. *If $\mathcal{W} = \{w = w_1 \dots w_h\}$ a single word of length h then its smallest associated DFA is of size $L = h + 1$ and defined by $\mathcal{Q} = \{\varepsilon, w_1, w_1 w_2, \dots, w\}$ the set of all prefixes of w, $s = \varepsilon$, $\mathcal{F} = \{w\}$ and for all $q \in \mathcal{Q}$ and $a \in \mathcal{A}$, $\delta(q, a)$ is simply defined as the longuest suffix of qa (concatenation of q and a) in $\mathcal{Q}$.*

In the case of a general pattern, a similar method can produce an associated DFA (consider for $\mathcal{Q}$ the union of all pattern prefixes) but it would not necessary be the smallest one. In order to be more efficient in the DFA design, one should use instead the classical and well known algorithms provided by the theory of

languages and automata (regular expression to FSA, determinization, epsilon removal).

Assuming from now that a DFA (smallest or not) associated to our pattern has been built, we can give the main result of this part:

Theorem 2. *if $X = X_1 \dots X_n$ is a i.i.d. sequence on $\mathcal{A}$, $\mathcal{W}$ a pattern and $(\mathcal{A}, \mathcal{Q}, s, \mathcal{F}, \delta)$ an associated DFA then sequence $Y = Y_0 \dots Y_n$ defined by*

$$Y_0 = s \quad \textit{and} \quad Y_i = \delta(Y_{i-1}, X_i) \quad \textit{for all} \quad 1 \leqslant i \leqslant n$$

is an order 1 *Markov chain whose transition matrix is given by*

$$\Pi(p,q) = \begin{cases} \mathbb{P}(X_1 = a) & \textit{if } \delta(p,a) = q \\ 0 & \textit{if } q \notin \delta(p, \mathcal{A}) \end{cases}$$

and such as occurrences of $\mathcal{W}$ in X correspond to occurrences of a subset of letters in Y (here $\mathcal{F}$). A Markov chain having these properties is called a Pattern Markov Chain *(PMC).*

Proof. By definition, the sequence Y is obviously an order 1 Markov chain. Moreover, if an occurrence of $\mathcal{W}$ ends at position i in X, the sequence $X_1 \dots X_i$ ends with an occurrence of the pattern and is therefore an element of $\mathcal{A}^*\mathcal{W}$ and thus is accepted by the DFA which means that $Y_i \in \mathcal{F}$ and the theorem is proved.

One should note that the built PMC has the same number of states L as the DFA used to build it (and thus, the use of the smallest DFA should be preferred). It is also clear that the transition matrix of a PMC is sparse (only $k \times L$ non zero terms among L^2, where k is the alphabet size). Finally let us note that we have a natural decomposition of this transition matrix into

$$\Pi = P + Q$$

where Q contains all transitions toward counting states and P the regular ones.

Example 1. Let us consider the pattern $\mathcal{W} = \{\text{abab}, \text{baa}\}$ over the binary alphabet. Its smallest associated DFA is represented on fig. 2. If X is the original sequence, we build the PMC Y as follows:

$X =$	$-$	a	b	b	a	a	b	a	a	b	a	b	a	b	b	a	a	a	b	b
$Y =$	0	3	4	6	7	1	4	5	1	4	5	2	5	2	6	7	1	3	4	6

In both sequences, occurrences of $\mathcal{W}$ end in positions (starting from 0 in Y): 5 (baa), 8 (baa), 11(abab, overlapping preceeding occurrence), 13 (abab, overlapping preceeding occurrence) and 16 (baa). The

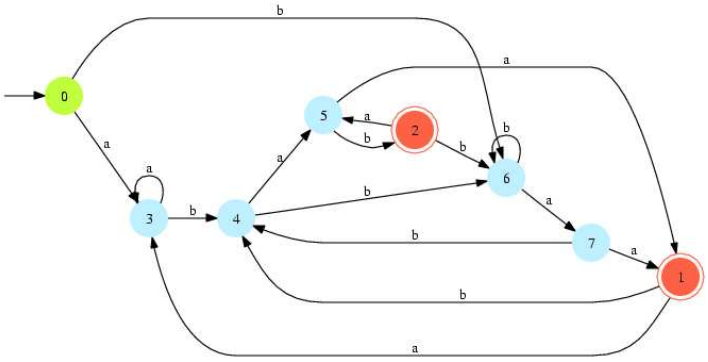


FIGURE 2: Graphical representation of the smallest DFA associated to $\mathcal{W} = \{\text{abab}, \text{baa}\}$.

transition matrix of Y is given by

$$\Pi = \begin{pmatrix} 0 & 0 & 0 & p_{\text{a}} & 0 & 0 & p_{\text{b}} & 0 \\ 0 & 0 & 0 & p_{\text{a}} & p_{\text{b}} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & p_{\text{a}} & p_{\text{b}} & 0 \\ 0 & 0 & 0 & p_{\text{a}} & p_{\text{b}} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & p_{\text{a}} & p_{\text{b}} & 0 \\ 0 & p_{\text{a}}^* & p_{\text{b}}^* & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & p_{\text{b}} & p_{\text{a}} \\ 0 & p_{\text{a}}^* & 0 & 0 & p_{\text{b}} & 0 & 0 & 0 \end{pmatrix}$$

where transitions with * belong to Q and with $p_\cdot = \mathbb{P}(X_1 = \cdot)$.

As explained in the introduction, the authors of (19) proposed to use pattern's DFA to get the pattern generating function through the Chomsky & Shltzenberger and derive from it exact results and asymptotic moments. More recently, (8) used the pattern's automaton conjointly with exponential families results in the same aim. Instead of focusing of generating function only (as done in these papers), we propose here a more straightforward and practical approach consisting to exploit our new PMC to improve a wide range of classical pattern methods.

2. Methods

In this part we consider a pattern $\mathcal{W}$, $(\mathcal{A}, \mathcal{Q}, s, \mathcal{F}, \delta)$ an associated (smallest or not) DFA and we denote by Y its corresponding PMC which transition matrix is denoted $\Pi = P + Q$.

2.1. Exact

In (10) was first introduced the elegant technique called Finite Markov Chain Imbedding (FMCI, see (18) or (11) for a review) to deal with exact computations by building a Markov chain imbedding all combinatorial aspects of the problem. In all these papers, the authors discuss in detail about the construction of this FMCI which is of course a key stage of the method, but do not see its link with the theory of language and automata. We propose here to explain the very simple connexion that exists between PMC and FMCI.

Definition 2. For any $c \in \mathbb{N}$ we define the FMCI Z by

$$Z_j = \begin{cases} (Y_j, N_j) & \text{if } N_j < c \\ f & \text{if } N_j \geqslant c \end{cases}$$

where N_j number of pattern occurrences in $X_1 \dots X_j$. We order the $cL + 1$ of states of Z as the following: $(1,0), \dots, (L,0), (1,1), \dots, (L,1), \dots, (1,c-1), \dots, (L,c-1), f$.

Proposition 2. *The corresponding transition matrix is given by*

$$\Pi = \left(\begin{array}{c|c} R & v \\ \hline 0 & 1 \end{array} \right)$$

where R (dimension $aL \times cL$) and v (dimensions $cL \times 1$) are defined by blocks of size L:

$$R_{i,j} = \begin{cases} P & \text{if } i = j \\ Q & \text{if } i + 1 = j \\ 0 & else \end{cases} \qquad \text{and} \quad v_i \equiv 0 \text{ for } 1 \leqslant i < c \text{ and } v_c = \Sigma_Q$$

where Σ_Q is the column vector resulting of the sum of Q.

Proof. Obvious since transitions in P will not increment the number of occurrences while transitions in Q will increment it by one.

Example 2. For example if $c = 3$ we get the following transition matrix:

$$\Pi = \left(\begin{array}{ccc|c} P & Q & 0 & 0 \\ 0 & P & Q & 0 \\ 0 & 0 & P & \Sigma_Q \\ \hline 0 & 0 & 0 & 1 \end{array}\right)$$

As proposed in (24) it is hence possible to get the p-values we are looking for, through efficient recurrence relations:

Theorem 3. *For all $n \geqslant 1$ and $1 \leqslant i \leqslant k$ we have*

$$\mathbb{P}(N_n < c | X_1 = i) = \left(u^{n-1}\right)_i \quad \textit{and} \quad \mathbb{P}(N_n \geqslant c | X_1 = i) = \sum_{j=0}^{n-2} \left(v^j\right)_i$$

where $(\,)_i$ denotes the i^{th} component of a vector, where for $x = u$ or v we have $\forall j \geqslant 0$ the following size L block decomposition: $x^j = \left(x^j_{(c-1)}, \ldots, x^j_0\right)'$ and we have the recurrence relations:

$$x_0^{j+1} = P x_0^j \quad \textit{and} \quad \forall i \geqslant 1 \quad x_i^{j+1} = P x_i^j + Q x_{i-1}^j$$

with $u^0 = (1 \ldots 1)'$ and $v^0 = v$.

Corollary 1. *We get recursive algorithms to compute $\mathbb{P}(N_n \geqslant N_{obs})$ or $\mathbb{P}(N_n \leqslant N_{obs})$ with $O(k \times L + N_{obs} \times L)$ memory complexity and $O(k \times L \times N_{obs} \times n)$ time complexity.*

It is also interesting to note that it is very easy to get the exact distribution of waiting time through the following proposition

Proposition 3. *We consider a subsequence of the PMC Y that we renumber starting from 0: $Y_0, Y_1, \ldots$ and for state $q \in \mathcal{Q}$ and any $f \in \mathcal{F}$ we define the waiting time $T_{q,f}$ of the first occurrence of f after q without any other occurrence of $\mathcal{F} \setminus \{f\}$. This random variable has the following (exact) distribution:*

$$\mathbb{P}\left(T_{q,f} = k\right) = \mathbb{P}(Y_1, \ldots, Y_{k-1} \notin \mathcal{F}, Y_k = f | Y_0 = q) = {}^t E_q P^{k-1} Q E_f$$

where E_i is the null column vector with a one in position i. The waiting time T_q of the first pattern occurrence after q is hence distributed according to

$$\mathbb{P}\left(T_q = k\right) = \mathbb{P}(Y_1, \ldots, Y_{k-1} \notin \mathcal{F}, Y_k \in \mathcal{F} | Y_0 = q) = {}^t E_q P^{k-1} Q E_{\mathcal{F}}$$

where $E_{\mathcal{F}} = \sum_{f\in\mathcal{F}} E_f$ *and we have*

$$\mathbb{P}(Y_k = f | T_q = k) = \frac{E_q P^{k-1} Q E_f}{E_q P^{k-1} Q E_{\mathcal{F}}}$$

Proof. It is clear that the transitions toward $\mathcal{Q} \setminus \mathcal{F}$ are, by definition, those of the matrix P while a transition ending in a final state belongs to the matrix Q. The result is then trivial to establish.

2.2. Large deviations

As explained in (20) it is possible to use classical large deviations results (9) to get the following theorem:

Theorem 4. *For all* $N_{obs} \geqslant \mathbb{E}[N_n]$ *(resp.* $<$*) we have*

$$\mathbb{P}(N_n \geqslant N_{obs}) \simeq \exp(-nI) \quad \textit{(resp. } \leqslant\textit{)}$$

with

$$I = \sup_{\theta\in\mathbb{R}} \left\{ \theta \frac{N_{obs}}{n} - \Lambda(\theta) \right\}$$

where $\Lambda(\theta)$ *is the* $\log$ *of the largest eigenvalue of* $\Pi_\theta = P + e^\theta Q$.

Using Arnoldi class methods (see (36) for a good review on the subject) it is possible to exploit the sparse structure of our transition matrix to find its few largest magnitude eigenvalues and/or associated eigenvectors in $O(k \times L)$ in time and memory. In particular, such an approach is suitable to compute a numerical approximation of the stationary distribution of our PMC. The necessary unidimensional function minimization can then be performed with classical algorithms (Golden section or Brent's algorithm for example) resulting in a final similar complexity to obtain large deviations pattern statistics.

2.3. Gaussian approximations

The result presented in this part are close to those from (15) but we focus here (like in (22)) on the effective linear computations (with the sequence length) rather than the quadratic ones. Moreover, and unlike previous work, we do not consider here the case where the Markov chain starts with the stationary distribution because this assumption is completly false for a PMC (which always starts from the starting state s in the independent case).

Theorem 5. *For all* $a, b \in \mathcal{Q}$ *we have* $\mathbb{E}[N_n(a)] = E_n(a)$ *and* $\mathbb{E}[N_n(a)N_n(b)] = E_n(a,b)$ *with*

$$E_n(a) = \sum_{0\leqslant i\leqslant n} {}^t E_s \Pi^i E_a, \quad E_n(a,b) = \mathbb{I}_{a=b} \times E_n(a) + C_n(a,b) + C_n(b,a)$$

$$\textit{and} \quad C_n(a,b) = \sum_{0 \leqslant i < j \leqslant n} \left({}^t E_s \Pi^i E_a\right) \left({}^t E_a \Pi^{j-i} E_b\right)$$

where E_p is the null column vector with a one in position p.

Proof. Simply decompose $N_n(a) = \sum_{i=1}^n \mathbb{I}_{Y_i=a}$ and $N_n(b)$ to get the result.

The computation of $E_n(a,b)$ hence require $O(n^2)$ terms which could be a serious problem for large n. Fortunately, we can use the convergence of the powers of Π toward its stationary distribution μ (which can be computed in $O(k \times L)$, see remark about Arnoldi algorithms above) to get this corollary:

Corollary 2. *We consider the case where Π admits a diagonal form (which is not restrictive in practice) and denote by ν the magnitude of its second largest eigenvalue. Then, for all $a, b \in \mathcal{Q}$, $\alpha \in \mathbb{N}$ and $n \geqslant 2\alpha$ we get*

$$E_n(a) = E_n^\alpha(a) + O(\nu^\alpha) \quad \textit{and} \quad E_n(a,b) = E_n^\alpha(a,b) + O(n\nu^\alpha)$$

where

$$E_n^\alpha(a) = \sum_{0 \leqslant i < \alpha} {}^t E_s \Pi^i E_a + (n-\alpha+1)\mu(a) \textit{ and } E_n^\alpha(a,b) = \mathbb{I}_{a=b} \times E_n^\alpha(a) + C_n^\alpha(a,b) + C_n^\alpha(b,a)$$

and with

$$\begin{aligned} C_n^\alpha(a,b) = {} & A_0^{\alpha-1}(s,a) A_1^{\alpha-1}(a,b) + \left[(n-\alpha+1)A_0^{\alpha-1}(s,a) - B_0^{\alpha-1}(s,a)\right]\mu(b) \\ & + \left[(n-2\alpha+1)A_1^{\alpha-1}(a,b) + \sum_{i=1}^{\alpha-1} A_1^i(a,b)\right]\mu(a) + \frac{(n-2\alpha+1)(n-2\alpha+2)}{2}\mu(a)\mu(b) \end{aligned}$$

where for all $i \leqslant j$ and $p, q \in \mathcal{Q}$ we have

$$A_i^j(p,q) = \sum_{d=i}^{j} {}^t E_p \Pi^d E_q \quad \textit{and} \quad B_i^j(p,q) = \sum_{d=i}^{j} d\, {}^t E_p \Pi^d E_q$$

Proof. A simple decomposition in the eigenvectors basis gives that $\exists C > 0$ such as $\forall p, q \in \mathcal{Q}$ and $\forall d \geqslant 1$ we have $|{}^t E_p \Pi^d E_q - \mu(q)| \leqslant C\nu^d$. We hence can replace all terms of the form ${}^t E_p \Pi^d E_q$ ($d \geqslant \alpha$) by $\mu(q)$ with a *uniform* error in $O(\nu^d)$. Doing so in the expression of $C_n(a,b)$ we get

$$\begin{aligned} C_n^\alpha(a,b) = {} & \sum_{(i,j) \in I_{0,0}} \left({}^t E_s \Pi^i E_a\right) \left({}^t E_a \Pi^{j-i} E_b\right) \\ & + \sum_{(i,j) \in I_{0,1}} \left({}^t E_s \Pi^i E_a\right) \mu(b) + \sum_{(i,j) \in I_{1,0}} \mu(a) \left({}^t E_a \Pi^{j-i} E_b\right) + \sum_{(i,j) \in I_{1,1}} \mu(a)\mu(b) \end{aligned}$$

with a error term in $O(n\nu^\alpha)$ and where $I_{0,0}$, $I_{0,1}$, $I_{1,0}$, $I_{1,1}$ are subsets of $\{(i,j), 0 \leqslant i,j \leqslant n\}$ such as $i < \alpha$ (resp. $\geqslant \alpha$) if first indice is a 0 (resp. 1) and $(j-i) < \alpha$ (resp. $\geqslant \alpha$) if second indice is a 0 (resp. 1). A careful study of these four subsets finally give the corollary.

Note that this result holds any first order ergodic Markov chain of length $n+1$ beginning with a s. If the Markov chain starts with a starting distribution μ_0 which is no more a Dirac, simply replace all occurrences of E_s by μ_0 (taken as a column vector). Finally, let us add that in the particular case where $\mu_0 = \mu$ (the stationary distribution), ${}^t\mu\Pi^d E_q = \mu(q)$ for all $d \in \mathbb{N}$ and $q \in \mathcal{Q}$ which leads of course to some simplifications.

In all cases, the (absolute error) in $O(n\nu^\alpha)$ insure that in order to obtain a given level of precision we have to consider a α growing in $O(-\log n/\log \nu)$ with n.

Proposition 4. *It is possible to compute approximations of the expectation and variance of a pattern whose PMC has L states and F final states with $O(F \times L)$ memory complexity and $O(\log n \times F \times L + F \times F)$ time complexity.*

Proof. For each $f \in \mathcal{F}$ we have to consider x_f the row vector of size L which is filled with zeros except in position f (this gives the memory complexity). Then use the sparse structure of Π to compute $\Pi^i \times x_f$ recursively for all $i < \alpha$. We hence get $E_n^\alpha(f, f')$ for all $f, f' \in \mathcal{F}$ and thus we obtain the variance in $O(F \times F)$.

Once expectation and variance of N_n have been computed, it is of course possible to get Gaussian approximations. Thanks to limit central theory, such asymptotic approximations are valid at finite distance if "central" events (means, not too significant p-values) are considered. In most interesting cases however, such approximations are hence poorly reliable and should be avoided if possible (see (22) for more details on this point).

2.4. Binomial approximations

Considering the results presented in the last sections, it is obvious that it is possible to obtain the probability p for an occurrence of the pattern to occur at a given position in X in $O(k \times L + F)$ (to compute of the stationary distribution and then sum F terms).

If we assume that pattern occurrences at two different positions are independent (which is obviously false) we get a simple approximation of the distribution of N_n by a binomial distribution $\mathcal{B}(n,p)$. Its tail distribution is then computable in $O(\log N_{\text{obs}})$ using incomplete beta function (26).

Several papers like (21; 22) have already pointed out the surprising good quality of such approximations which should be considered as a quick and (not so) rough heuristic to the problem (and preferred to Gaussian approximations for this matter).

2.5. Compound Poisson approximations

Definition 3. For all $p, q \in \mathcal{Q}$ and for all $m \geqslant 0$ we call $(p_0, \ldots, p_m) \in \mathcal{Q}^{m+1}$ a *path* from p to q (of length m) if and only if $p_0 = p$, $p_m = q$ and it exists a label $a = a_1 \ldots a_m \in \mathcal{A}^m$ such as $\delta(p_{i-1}, a_i) = p_i$ for all $1 \leqslant i \leqslant m$. A path $(p_0, \ldots, p_m)$ have an associated probability $P(p_0, \ldots, p_m) = \Pi(p_0, p_1) \times \ldots \times \Pi(p_{m-1}, p_m)$ and is called an *overlapping path* if $\delta(s, a_1 \ldots a_m) \neq q$.

For all $i, j \in \mathcal{F}$ we define the self-overlapping probability $A_{i,j}$ by

$$A_{i,j} = \sum_{m \geqslant 1} \sum_{\substack{(p_0 = i, \ldots, p_m = j) \\ \text{overlapping path}}} \mathbb{I}_{\{p_1, \ldots, p_{m-1}\} \cap \mathcal{F} = \emptyset} \times P(p_0, \ldots, p_m)$$

We call $\mathbf{A} = (A_{i,j})$ the pattern *self-overlapping matrix*.

Proposition 5. *For all $i, j \in \mathcal{F}$, $A_{i,j}$ is recursively computable through the relation*

$$A_{i,j} = \sum_{a \in \mathcal{A}} F_j\left(\delta(i,a), \delta(s,a)\right) \Pi\left(i, \delta(i,a)\right)$$

where for all $p, q \in \mathcal{Q}$ we have

$$F_j(p,q) = \begin{cases} 0 & \text{if } p = q \text{ or } p \in \mathcal{F} \setminus \{j\} \\ 1 & \text{if } p = j \\ \sum_{a \in \mathcal{A}} F_j\left(\delta(p,a), \delta(q,a)\right) \Pi\left(p, \delta(p,a)\right) & \text{else} \end{cases}$$

Proof. We use the recursive relation to consider all paths starting from i until either we reach j (acceptable path) or either the path is no longer an overlapping one (terminal condition $p = q$) or the path encounter another final state (terminal condition $p \in \mathcal{F} \setminus \{j\}$).

Example 3. We consider the pattern $\mathcal{W} = \{\text{abab}, \text{baa}\}$ over the alphabet $\mathcal{A} = \{\text{a}, \text{b}\}$ whose smallest DFA is given on figure 2. Using the last proposition, it is possible to get

$$\mathbf{A} = \begin{pmatrix} 0 & \Pi(1,4)\Pi(4,5)\Pi(5,2) \\ \Pi(2,5)\Pi(5,1) & \Pi(2,5)\Pi(5,2) \end{pmatrix}$$

with only 10 evaluations of functions F_1 or F_2.

Theorem 6. *For all $n \geqslant 1$ we have*

$$\mathcal{L}(N_n) \simeq \mathcal{CP}(\lambda_\kappa, \kappa \geqslant 1)$$

where

$$\lambda_\kappa = n\mathbf{M}\mathbf{A}^{\kappa-1}(\mathbf{I}-\mathbf{A})^2\mathbf{1}$$

with $\mathbf{M} = [\mu_i]_{i\in\mathcal{F}}$, $\mathbf{I}$ identity and $\mathbf{1}$ column-vector of ones.

Proof. This result has been shown through Chen-Stein approximations (4; 1; 35; 34). Although this result is only proved for asymptotically rare patterns, it has been shown (22) that the approximations remains quite reliable in practice even out of this framework.

Corollary 3. *Computing p-values through these compound Poisson approximations result in algorithms whose complexities depend on $\mathbf{A}$:*

case	$\mathbf{A} \equiv 0$	$\mathbf{A} \equiv \theta\mathbf{I}$	*else*
distribution	$\mathcal{P}$	$\mathcal{GP}$	$\mathcal{CP}$
memory	$O(1)$	$O(1)$	$O\left(N_{obs}^2\right)$
time	$O\left(\log N_{obs}\right)$	$O\left(N_{obs}\right)$	$O\left(N_{obs}^3\right)$

Proof. In the case where $\mathbf{A} \equiv 0$ all the λ_κ are nulls except the first one: $\lambda_1 = \lambda = n\mathbf{M1}$ which is precisely the expected number of pattern occurrences. The resulting distribution is a simple Poisson one (which is very close to the binomial as soon as n is large and $\mathbf{M1}$ is small).

In the case where $\mathbf{A} \equiv \theta\mathbf{I}$ we have $\lambda_\kappa = n\mathbf{M1}\theta^{\kappa-1}(1-\theta)^2$ hence $\lambda = \sum_\kappa \lambda_k = n\mathbf{M1}(1-\theta)$ so that $\lambda_\kappa/\lambda = \theta^{\kappa-1}(1-\theta)$ which define a geometric distribution of parameter θ. The resulting distribution is hence a geometric Poisson one whose cumulative distribution function can be efficiently computed through recurrences and confluent hypergeometric functions (see (23) for more details).

In the last general case, p-values can be computed through the general recurrence formula proposed by (3).

3. Extensions

The methods we have presented until now are only valid for overlapping occurrences of a pattern in a i.i.d. sequence. We propose here to extend our results to Markov sequences or to renewal occurrences.

3.1. Markov chains

In order to extend our results to Markov chain sequences, we first need to introduce the following definition

Definition 4. A DFA $(\mathcal{A}, \mathcal{Q}, \mathcal{F}, s, \delta)$ where it exists $q \in \mathcal{Q}$ and $a, b \in \mathcal{A}^m$ such as $a \neq b$ and $\delta(q, a) = \delta(q, b)$ is called m-*ambiguous*. A DFA which is not m-ambiguous is also called m-*unambiguous*.

Please note that the m-ambiguity presented here is different from the classical notion of $ambiguity$ for DFA (meaning that it exists two different path to recognize the same language element).

For any DFA $(\mathcal{A}, \mathcal{Q}, s, \mathcal{F}, \delta)$ we define for all $q \in \mathcal{Q}$ and for all $m \geqslant 1$ the following notations:

$$\delta^{-m}(q) = \{a \in \mathcal{A}^m, \exists p \in \mathcal{Q}, \delta(p, a) = q\} \quad \text{and} \quad \Delta^{-1}(q) = \{p \in \mathcal{Q}, \exists a \in \mathcal{A}, \delta(p, a) = q\}$$

Hence, such a DFA is m-unambiguous if all $\delta^{-m}(q)$ are singletons.

Theorem 7. *if $X = X_1 \dots X_n$ is an order $m \geqslant 1$ Markov sequence on $\mathcal{A}$, $\mathcal{W}$ a pattern and $(\mathcal{A}, \mathcal{Q}, s, \mathcal{F}, \delta)$ a non m-ambiguous DFA whose language is $\mathcal{A}^*\mathcal{W}$ then the sequence $Y = Y_m \dots Y_n$ defined by*

$$Y_0 = s \quad \textit{and} \quad Y_i = \delta(Y_{i-1}, X_i) \quad \textit{for all} \quad 1 \leqslant i \leqslant n$$

is an order 1 Markov chain whose transition matrix is given by

$$\Pi(p, q) = \begin{cases} \mathbb{P}(X_{m+1} = b | X_1 \dots X_m = \delta^{-m}(p)) & \textit{if } \delta(p, b) = q \\ 0 & \textit{if } q \notin \delta(p, \mathcal{A}) \end{cases}$$

and such as occurrences of $\mathcal{W}$ in X correspond to occurrences of a subset of letters in Y. Y is therefore a PMC.

Proof. The proof is very similar to the one of the i.i.d. case except that the non m-ambiguity is obviously required to insure that all $\delta^{-m}(p)$ are singletons.

Using this theorem, it is possible to apply all preceding methods to Markovian sequence. But the key question is of course: is it possible to build a non m-ambiguous pattern DFA and how ?

In (19), the authors explain (algorithm 6) that this can be done starting from a DFA associated to the pattern by duplicating states until all ambiguities have been removed. This, of course, is exactly what we need to do. However in this paper, we want to propose a more explicit algorithm:

Theorem 8. (remove m-ambiguity.) *If $A = (\mathcal{A}, \mathcal{Q}, s, \mathcal{F}, \delta)$ is a DFA associated to $\mathcal{W}$ which is $(m-1)$-unambiguous (empty condition if $m = 1$), it is possible to remove m-ambiguity while preserving the association to $\mathcal{W}$ using the following algorithm:*

- initialization*: $\mathcal{Q}_0 = \mathcal{Q}$, $\forall q \in \mathcal{Q}$, $\mathcal{D}_q = \delta^{-m}(q)$ et $\mathcal{G}_q = \Delta^{-1}(q)$*
- main loop*: for all $q \in \mathcal{Q}_0$ while $|\mathcal{D}_q| > 1$ do*
  - *take $a = a_1 \dots a_m \in \mathcal{D}_q$*
  - *add a new state q_a to $\mathcal{Q}$*
  - *if $q \in \mathcal{F}$ then add q_a to $\mathcal{F}$*
  - *define $\mathcal{D}_{q_a} = \{a\}$ and $\mathcal{G}_{q_a} = \emptyset$*
  - *for all $b \in \mathcal{A}$ do $\delta(q_a, b) = \delta(q, b)$ and add q_a to $\mathcal{G}_{\delta(q,b)}$*
  - *for all $p \in \mathcal{G}_q$, if $\delta(p, a_m) = q$ and if $\delta^{-(m-1)}(p) = a_1 \dots a_{m-1}$ (empty condition if $m = 1$) then $\delta(p, a_m) = q_a$ and add p to $\mathcal{G}_{q_a}$*
  - *for all $p \in \mathcal{G}_q$, if $q \notin \delta(p, \mathcal{A})$ then remove q from $\mathcal{G}_q$*
  - *remove a from $\mathcal{D}_q$*

Moreover, if the starting DFA is the smallest $(m-1)$-unambiguous one, then the resulting automaton is the smallest m-unambiguous DFA associated to the pattern. Let us note that we still have $\mathcal{D}_q = \delta^{-m}(q)$ and $\mathcal{G}_q = \Delta^{-1}(q)$ at the end of algorithm.

Proof. As suggested by (19), this algorithm simply duplicates state for which it exits a m-ambiguity while preserving the DFA ability to recognize its language. As only the necessary states are duplicated, this algorithm also preserves the optimality of produced DFA.

In order to achieve non m-ambiguity one could hence successively remove 1-ambiguity, then 2-ambiguity and so on till we finally remove m-ambiguity having used a total of m applications of the theorem 8

3.2. Renewal occurrences

We first recall that a renewal occurrence (also called non-overlap occurrences) of a given pattern is an occurrence which does not overlap any other pattern occurrence. For example: $X =$ abababbaba contains three overlapping occurrences of aba but only two renewal ones (as the second occurrence overlaps the first one).

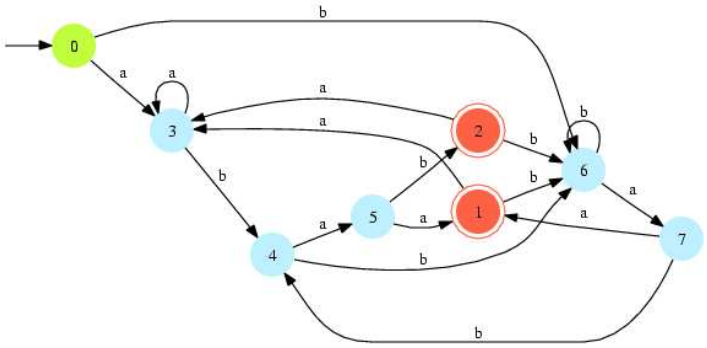


FIGURE 3: Graphical representation of the $\mathcal{W} = \{\text{abab}, \text{baa}\}$ renewal DFA built from the DFA of figure 2 through theorem 6.

Adapting pattern methods to such kind of occurrences usually requires a lot of work, but with our approach (as already pointed by (19)), we only need a small modification of our DFA:

Proposition 6. *If $(\mathcal{A}, \mathcal{Q}, s, \mathcal{F}, \delta)$ is a DFA which accepts $\mathcal{L} = \mathcal{A}^*\mathcal{W}$ then*

$$\delta(f, a) = \delta(s, a) \quad \forall f \in \mathcal{F} \quad \textit{and} \quad \forall a \in \mathcal{A}$$

will transform the DFA to accept only the texts ending with a renewal (i.e. non overlapping) occurrence of $\mathcal{W}$.

Proof. This is trivial since restarting the DFA from s after each occurrence obviously means that past is not taken into account.

Once this transformation has been done, all preceding results will hold for renewal occurrences using our modified DFA. One should note that when doing so, the pattern self-overlapping matrix is obviously null and hence makes compound Poisson approximations easier to use as they are only simple Poisson approximations.

One can also extend the notion of renewal occurrences to the one of d-*renewal* occurrences for which we have to wait d steps after a given occurrence to accept another one (thus, renewal occurrences and 0-renewal ones are exactly the same). In order to consider d-renewal occurrences of a pattern $\mathcal{W}$ we simply need to count renewal occurrences of $\mathcal{W}\mathcal{A}^d$.

method	memory complexity	time complexity
exact	$k \times L + N_{\text{obs}} \times L$	$k \times L \times N_{\text{obs}} \times n$
large deviations	$k \times L$	$k \times L$
Gaussian	$k \times L + F \times L$	$k \times L + F \times L \times \log n + F^2$
binomial	$k \times L$	$k \times L + F + \log N_{\text{obs}}$
compound Poisson ($A \equiv 0$)	$k \times L + F^2$	$k \times L + F^2 + \log N_{\text{obs}}$
compound Poisson ($A \equiv \theta I$)	$k \times L + F^2$	$k \times L + F^2 + N_{\text{obs}}$
compound Poisson (else)	$k \times L + F^2 + N_{\text{obs}}^2$	$k \times L + F^2 + N_{\text{obs}}^3$

TABLE 1: Order of magnitude of memory and time complexities for the different statistical approaches. k is the alphabet size, L is the number of states of the associated DFA, F the number of final states, n the sequence length and N_{obs} the observed number of occurrences.

3.3. Summary

We propose in table 1 to summarize the time and memory complexities of the different statistical approaches we have presented. As we can see, the number of states L of the automaton as well as the number of final states F are critical parameters. For exact and large deviations however, F does not matter at all while complexities are linear (binomial approximations) or quadratic (Gaussian and compound Poisson approximations) with this parameter for the other methods. One should also note that the complexity $O(k \times L)$ appears in all methods. This corresponds to the the computation of the stationary distribution which is achieved, as explained before, through Arnoldi's algorithm.

4. Applications

We propose in this part to build associated DFA for a wide range of common patterns in order to evaluate the efficiency of our method.

4.1. Simple patterns

Let us start by pattern reduced to a simple word of length h, $\mathcal{W} = \{w_1 \dots w_h\}$. As we have seen in proposition 1, the smallest associated DFA have then $L = h + 1$ terms.

If we consider now the order m Markov case ($m \leqslant h$), it is clear that the smallest non m-ambiguous DFA will contain $k^m + h - m$ terms (k^m for $\mathcal{A}^m$ and $h - m$ for the prefixes of length greater than m) and only one final state ($F = 1$).

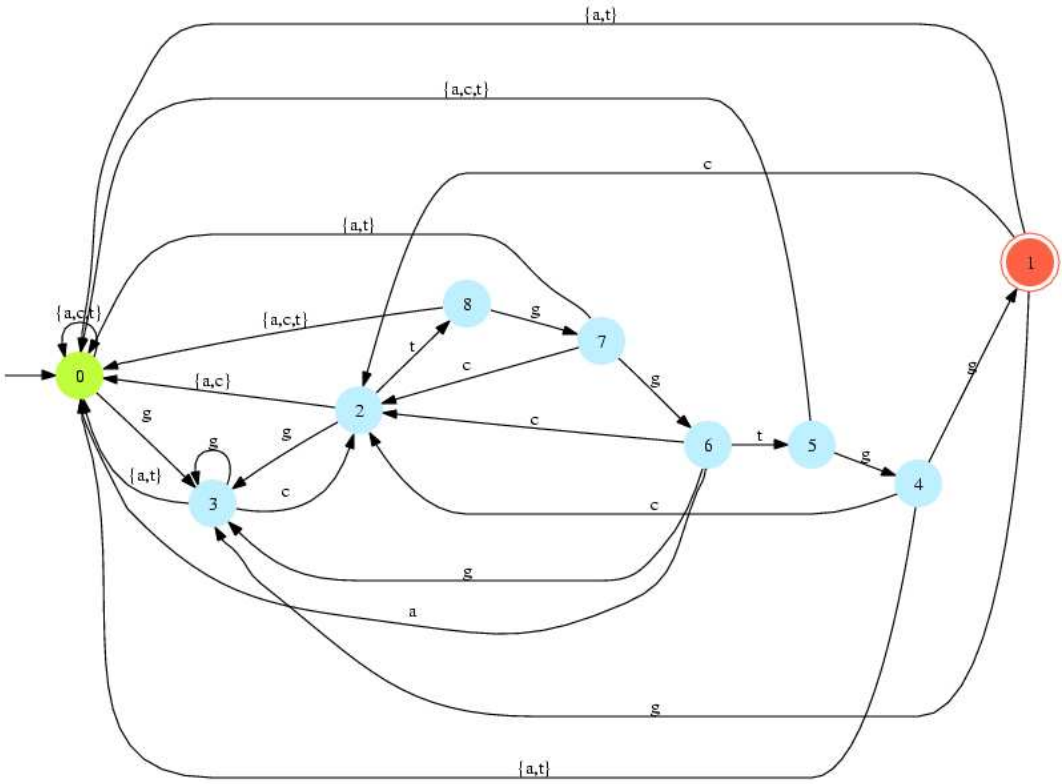


FIGURE 4: Smallest DFA associated to gctggtgg (*Escherichia coli*'s CHI pattern). $h = 8$, $k = 4$ (DNA), $F = 1$, $L = 4^0 + 8 = 9$.

In this case, it is then possible to replace L by the value proposed here in table 1 in order to get the optimal complexities for our statistical methods for a single word of length h using an order m Markov model. For example, the resulting complexity is $O(k^{m+1} + k \times (h - m))$ for an approach through large deviations.

If it is obviously possible to build this DFA directly with an approach similar to the one proposed in proposition 1, but it is also possible to build it through our remove ambiguity algorithms as a validation.

We can see on figure 4 ($m = 0$), 5 ($m = 1$) and 6 ($m = 2$) graphical representations of the smallest non m-ambiguous DFA associated to the *Escherichia coli*'s CHI pattern produced through theorem 8 for several values of m. As we can see, the observed DFA sizes are consistent with their expectation thus validating our algorithm.

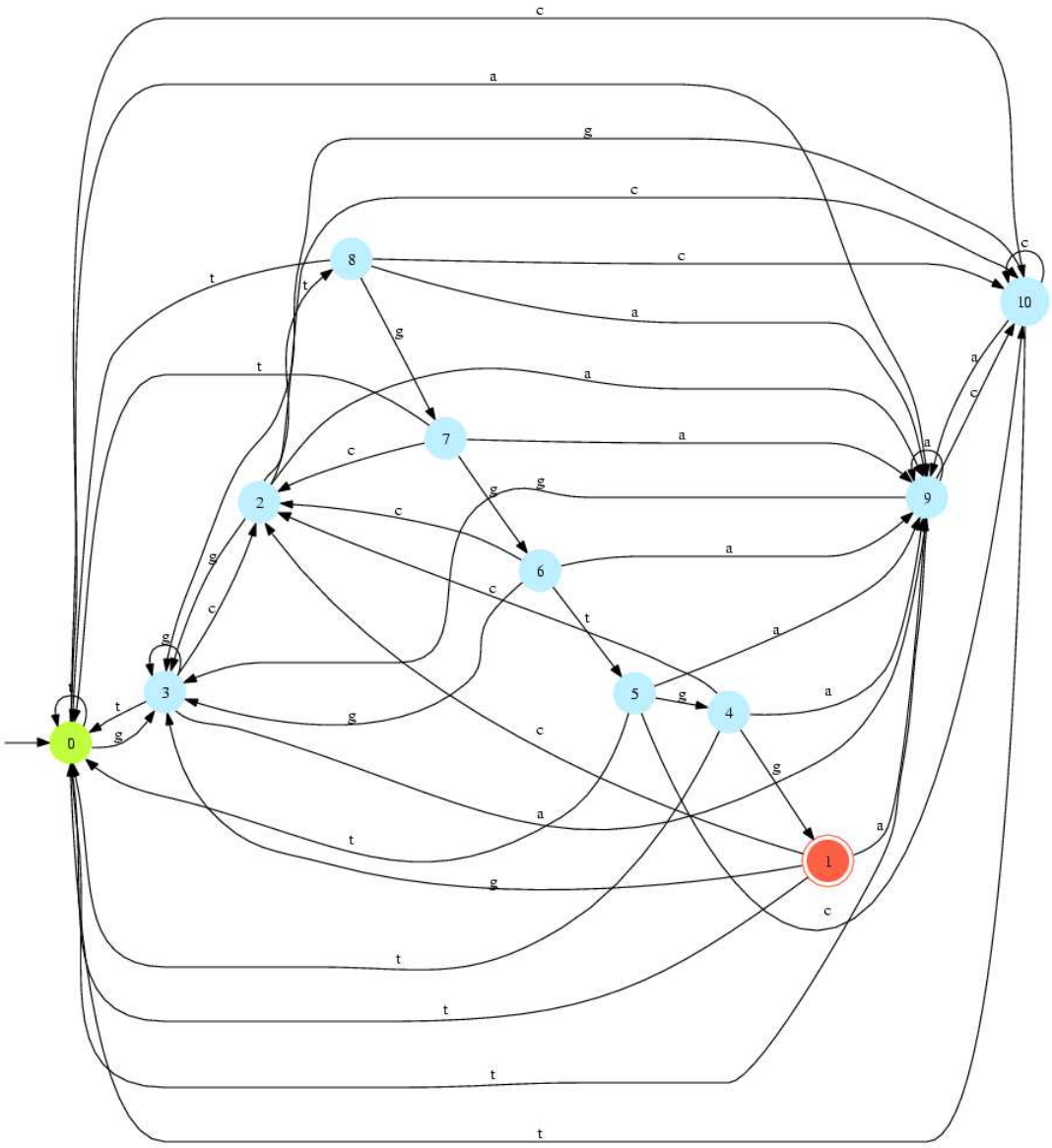


FIGURE 5: Smallest non 1-ambiguous DFA associated to gctggtgg (*Escherichia coli*'s CHI pattern). $h = 8$, $k = 4$ (DNA), $F = 1$, $L = 4^1 + 7 = 11$.

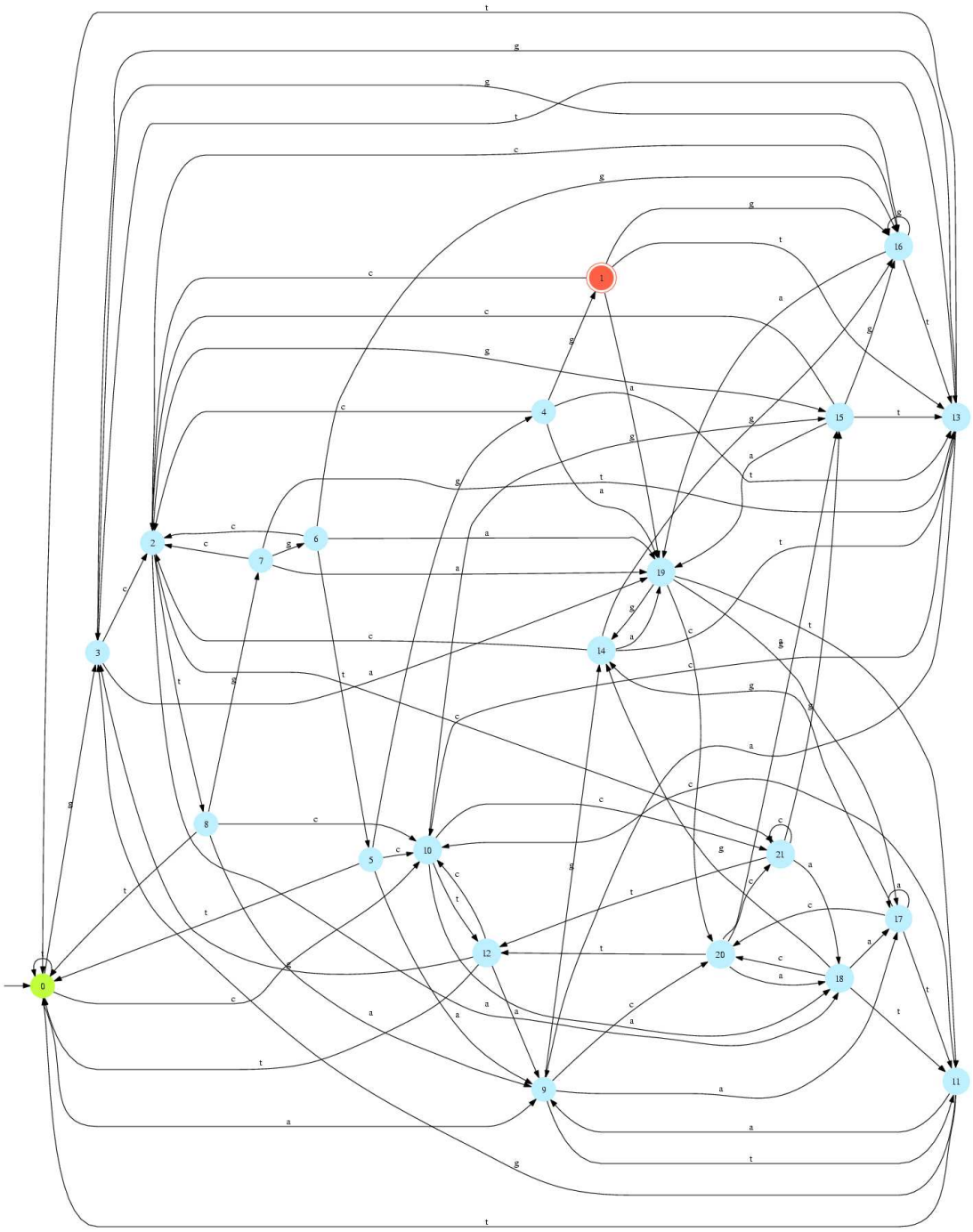


FIGURE 6: Smallest non 2-ambiguous DFA associated to gctggtgg (*Escherichia coli*'s CHI pattern). $h = 8$, $k = 4$ (DNA), $F = 1$, $L = 4^2 + 6 = 22$.

m	0	1	2	3	4
L	1 059	1 365	2 715	7 438	22 427
F	43	43	43	43	43

TABLE 2: Characteristics of the smallest non m-ambiguous DFA associated to the pattern $\mathcal{W}$ = ttgaca{16:18}tataat (cardinality $\simeq 9 \times 10^{10}$). L denotes the number of states and F the number of final states.

4.2. Gapped patterns

If the approach we proposed could obviously be useful in the case presented above of simple words (in particular if we are interested to renewal occurrences for example), it is likely that more dramatic improvement is expected for degenerated patterns.

Let us start with the gapped pattern (also called structured motifs), namely patterns separated one or several gaps of fixed or variable lengths. Such patterns are known to appear in DNA upstream regions and their distribution is difficult to study. A combinatorial technique have been proposed by (33), but this approach involves such heavy computations that the method is limited in practice to the first occurrence of patterns with only one gap.

On the other hand, our approach can take full profit of the theory of languages and automata to overcome the huge cardinality of the considered patterns.

Let us start with the pattern $\mathcal{W}$ = ttgaca{16:18}tataat (means one gap of length 16, 17 or 18) which is a consensus given by literature for promoters of *B. subtilis* (16). The cardinality of $\mathcal{W}$ is $4^{16} + 4^{17} + 4^{18} \simeq 9 \times 10^{10}$ which make this pattern untractable even by the fastest (linear with cardinal) methods based on an exhaustive approach to the pattern. Nevertheless, we can see in table 2 that the characteristics of the smallest associated DFA are obviously larger that in the case of simple patterns but remain acceptable: from a thousand states (case of independent sequence) to several thousands for the non ambiguous cases.

Of course, with such complex patterns the computational time is longer than in the simple cases, but as all our complexities grow linearly with the number of pattern states, we known that we can get a result.

Nothing prevents us to use the same approach on even more complicated patterns allowing another gap after the first one, or allowing one or several mismatch in the pattern.

For example, if we consider now the (non biological) pattern $\mathcal{W}$ = atgca{11}attat{12}gcatt, its cardinality is 7×10^{13} but we can see on table 3 that the DFA size does increase a lot.

m	0	1	2	3	4
L	2 071	2 494	4 530	10 531	26 908
F	109	109	109	109	109

TABLE 3: Characteristics of the smallest non m-ambiguous DFA associated to the pattern $\mathcal{W} =$ atgca$\{11\}$attat$\{12\}$gcatt (cardinality $\simeq 7 \times 10^{13}$). L denotes the number of states and F the number of final states.

m	0	1	2	3	4
L	329	1 393	10 688	134 746	na
F	30	78	633	3 045	na

TABLE 4: Characteristics of the smallest non m-ambiguous DFA associated to the cyclic nucleotide-binding domain signature 2 (PS00889): [LIVMF]-G-E-x-[GAS]-[LIVM]-x(5,11)-R-[STAQ]-A-x-[LIVMA]-x-[STACV] (cardinality $\simeq 10^{22}$). L denotes the number of states and F the number of final states.

4.3. Prosite patterns

It is also possible to consider proteic patterns. In this case, the alphabet is composed with the twenty amino-acids ($k = 20$). The well known Prosite database (release 19.23) contains 1 332 functional patterns which most of them are extremely degenerated.

In the paper (19), the author used a DFA approach to compute exact order one and two moments through formal computations and generating functions in the independent case. Using the extension of their method we presented here, we are able to do much more with a dramatic improvement in terms of efficiency.

For example, we consider the cyclic nucleotide-binding domain signature 2 (PS00889 entry of Prosite): [LIVMF]-G-E-x-[GAS]-[LIVM]-x(5,11)-R-[STAQ]-A-x-[LIVMA]-x-[STACV] ("x" means "any amino-acid", "[GAS]" means "any of those inside the brackets" and "x(5,11)" is a gap of length between 5 and 11). The cardinality of this pattern is 10^{22} but we can see on table 4 that its smallest associated DFA (independent case) has only 329 states. When the Markov order grows, L grows geometrically but at a slower rate that the naive 20 (size of the alphabet) rate that one could expect. Nevertheless, this rate (roughly 10.0) is obviously faster than in the DNA case (roughly 2.0) but this is not a real problem as it is very rare (and unwise) to use high order Markov model on proteic sequences (usually shorter sequences than in the DNA case and much more parameters to estimate).

4.4. Scan statistics

The scan statistics (13) have many applications in a wide range of disciplines from quality control and reliability to molecular biology through telecommunications or even sociology. Beside these numerous applications, their theoretical study is also very challenging. In this section, we will see here how our PMC approach can be applied to this particular problem.

Let us consider $X = X_1 \dots X_n$ a i.i.d. or Markov dependant sequence over the binary alphabet $\mathcal{A} = \{0, 1\}$, then for all window size $t \geqslant 1$, we define the *scan statistic* $S_i^t = \sum_{j=i}^{i+t-1} X_i$ and, for all $n, s \geqslant 1$ want to study the event "s ones out of t successive trials" :

$$A_t^s(n) = \left\{ \exists i \, (1 \leqslant i \leqslant n) \text{ such as } S_i^t \geqslant s \right\}$$

Proposition 7. *For any $t \geqslant s > 1$ (case $s = 1$ is trivial) we have $A_t^s(n) = \{N(\mathcal{W}) \geqslant 1\}$ with*

$$\mathcal{W} = \left\{ F\left(z_1, \dots, z_{s-1}\right), z_i \in \mathbb{N}, 0 \leqslant \sum_{i=1}^{s-1} z_i \leqslant t - s \right\}$$

where

$$F\left(z_1, \dots, z_{s-1}\right) = 1 \left(\prod_{i=1}^{s-1} 0^{z_i} 1 \right) \quad \textit{(concatenation)}$$

(for example, with $s = 3$, $F(0, 2) = 11001$, $F(1, 1) = 10101, \dots$). The cardinal of $\mathcal{W}$ is

$$\frac{(t - s + 1)}{(s - 1)} C_{t-1}^{s-2}$$

(C_n^k denotes the binomial coefficient).

Proof. Let us first consider all the 2^t possible words of length t over the binary alphabet. We are interested only in those with at least s ones. Fortunately, as soon as we have seen these s ones we know that the event A_t^s has occurred. It is therefore not necessary to consider words of length t with more than s ones nor those among the remaining words which start or end with zeros (only the part with ones matters). We are hence interested only in words with s ones of length s to t which give exactly the proposition as there is a bijection between this set and the way to put i balls (indistinctible) in $s - 1$ urns (C_{i+s-1}^{s-2} combinations) for $0 \leqslant i \leqslant t - s$. The cardinality of $\mathcal{W}$ is hence given by

$$\sum_{i=0}^{t-s} C_{i+s-1}^{s-2} = \frac{(t - s + 1)}{(s - 1)} C_{t-1}^{s-2}$$

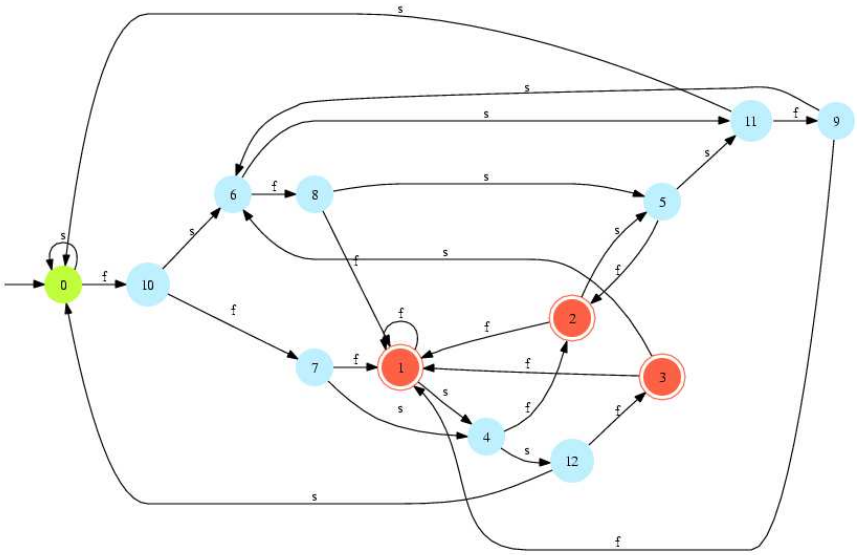


FIGURE 7: Smallest associated DFA to the pattern corresponding to the event "having 3 failure out of 5 successive trials". Pattern cardinal is 6 and the DFA has $L = 13$ states including $F = 3$ final states.

Let us consider that we track accumulated failures in a given system through scan statistics. Or binary alphabet will hence be $\mathcal{A} = \{\mathrm{f}, \mathrm{s}\}$ (failure/success) and A_t^s will be the even "having s failure out of t successive trials".

If we consider the event A_5^3, the proposition 7 tells us that it correspond to the pattern

$$\mathcal{W} = \{F(0,0), F(1,0), F(0,1), F(2,0), F(1,1), F(0,2)\} = \{\mathrm{fff}, \mathrm{fsff}, \mathrm{ffsf}, \mathrm{fssff}, \mathrm{fsfsf}, \mathrm{ffssf}\}$$

which associated smallest DFA is given in figure 7. If we consider now the event A_7^4, the cardinal of the pattern is 20 and the smallest associated DFA is given in figure 8.

On table 5 and 6 we can see the DFA characteristics for several scan statistics patterns. Unlike the degenerated pattern of previous section, the number of states is greater than the cardinality. This is due to the specific form of the scan statistic patterns which tend to explore a high combinatorial. Nevertheless, as L complexity tends to be of the same order than the cardinality (as many specific scan statistics methods), our DFA approach for scan statistics remain tractable for all cases where the cardinal is not too large providing for scan statistics the full range of statistical approaches we have presented in this paper (exact, large deviations, ...).

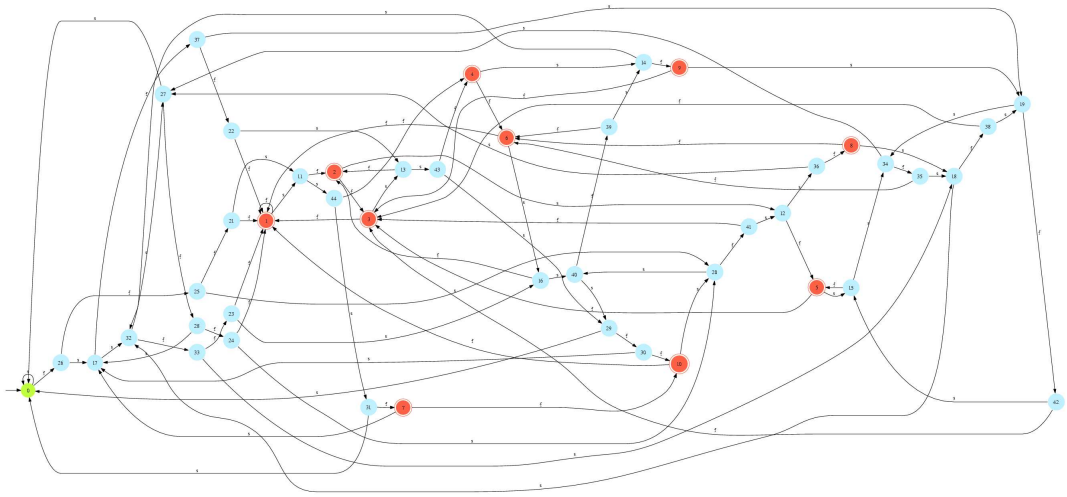


FIGURE 8: Smallest associated DFA to the pattern corresponding to the event "having 4 failure out of 7 successive trials". Pattern cardinal is 20 and the DFA has $L = 45$ states including $F = 10$ final states.

h	3	4	5	6	7	8	9
card	1	3	6	10	15	21	28
L	4	8	13	19	26	34	43
F	1	2	3	4	5	6	7

TABLE 5: characteristics of the smallest DFA associated to the pattern corresponding to the event "having 3 failure out of h successive trials". card $= (h-2)(h-1)/2$ is the pattern cardinal, L the number of states and F the number of final states.

h	4	5	6	7	8	9
card	1	4	10	20	35	56
L	5	13	26	45	71	105
F	1	3	6	10	15	21

TABLE 6: characteristics of the smallest DFA associated to the pattern corresponding to the event "having 4 failure out of h successive trials". card $= (h-3)(h-2)(h-1)/3$ is the pattern cardinal, L the number of states and F the number of final states.

5. Conclusion

In this paper, we have used the theory of finite languages and automata in order to introduce an auxiliary order 1 pattern Markov chain (PMC) for each pattern problem. With this tool, it has been possible to unify a wide range of statistical approaches (FMCI, large deviations, Gaussian and Poisson approximations). Moreover, this unification comes with some simplifications and substantial optimizations. This technique has then been extended to Markov dependent sequences introducing the notion of m-unambiguous automata and providing efficient algorithms to build them. The case of renewal or even d-renewal occurrences rather than classical overlapping ones have also been treated in a very simple way.

The interest of this new framework is obvious for simple patterns, but the most dramatic improvements come when we consider highly degenerated patterns for which the automaton approach allows to overcome pattern cardinality. Several examples of such patterns have been considered in the applications part (gapped patterns, Prosite patterns, Scan statistics). In each case, we have shown the power of our new approach by computing automata's characteristics (number of states and number of final states).

Allowing for the first time to deal with complicated patterns that was before impossible to consider, our results open new exciting perspectives for pattern statistics applications. Moreover, the simplificated framework provided by the PMC should also dramatically facilitate further theoretical developments.

Finally, let us add that all these results will be soon implemented in the Statistic for Patterns package (SPatt, freely available at http://stat.genopole.cnrs.fr/spatt).

References

[1] Arratia, R., Goldstein, L. and Gordon, L. 1990. Poisson approximation and the Chen-Stein method. *Stat. Sci. (1990)* **5 (4)**, 403-434.

[2] Arratia R., Goldstein L., and Gordon L. 1990. Poisson approximation and the chen-stein method. *Stat. Sci.* **5 (4)**, 403-434.

[3] Barbour, A. D., Brown, T. (1992) The Stein-Chen method, point process and compensators. *Ann. Proba.*, **20**, 1504-1527.

[4] Chrysaphinou, O. and Papastavridis, S. 1988. A limit theorem on the number of overlapping appearances of a pattern in a sequence of independent trials. *Proba. Theory Relat. Fields* **79 (1)**, 129-143.

[5] Cormen, T.H., Leiserson, C.E., Rivest, R.L., 1990. Introduction to Algorithms, Chapter 34, pp 853-885, MIT Press.

[6] Cowan R. 1991. Expected frequencies of DNA patterns using Whittle's formula. *J. Appl. Prob.* **28**, 886-892.

[7] Crochemore, M., Hancart, C. 1997. Automata for matching patterns in *Source Handbook of formal languages, vol. 2: linear modeling: background and application*, pp 399-462, Sptinger-Verlag New York, Inc.

[8] Crochemore, M., Stefanov, V. T. 2003. Waiting time and complexity for matching patterns with automata. *Info. Proc. Letters* **87** 119-125.

[9] Den Hollender, F. 2000. Large Deviations. *American, Mathematical Society*.

[10] Fu, J. C., Koutras, M. V. 1994. Distribution theory of runs: a Markov chain approach *J. Amer. Statist. Assoc* **89**, 1050-1058.

[11] Fu, J. C., Lou, W.Y.W. 2003. Distribution Theory of Runs and Patterns and Its Applications: A Finite Markov Chain Approach. *World Scientific*, Singapore.

[12] Geske M. X., Godbole A. P. , Schaffner A. A. , Skolnick A. M. , Wallstorm G. L. 1995. Compound poisson approximation for word patterns under markovian hypothesis. *J. Appl. Prob.* **32**, 877-892.

[13] Glaz, J., Balakrishnan, N. *Scan statistics and applications*, Statistics for industry and technology, Birkhuser, Berlin.

[14] Godbole A. P. 1991. Poisson approximations for runs and patterns of rare events. *Adv. Appl. Prob.* **23**, 851-865.

[15] Kleffe J., Borodovsky M. 1992. First and second moment of counts of words in random text generated by markov chains. *Comp. Applic. Biosci.* **8**, 433-441.

[16] Helmann, J. D. 1995. Compilation and analysis of *Bacillus subtilis* α-dependant promoter sequences: evidence for extended contact between RNA polymerase and upstream promoter DNA. *Nucleic Acids Res.*, **23**, 2351-2360

[17] Hopcroft, J. E., Motwani, R., Ullman, J. D. *Introduction to Automata Theory, Languages, and Computation (second edition)*, Addison-Wesley, 2001.

[18] Lou, W.Y.W. 1996. On runs and longest run tests: A method of nite Markov chain imbedding. *J. Am. Statist. Assoc.* **91(436)**, 1595-1601.

[19] Nicodème, P., Salvy, B. and Flajolet, P. 2002. Motifs statistics. *Theor. Comput. Sci.*, **287(2)**, 593-617.

[20] Nuel, G. 2004. LD-SPatt: Large Deviations Statistics for Patterns on Markov Chains. *J. Comput. Biol.*, **11(6)**, 1023-1033.

[21] Nuel, G. 2005. S-SPatt: Simple Statistics for Patterns on Markov Chains. *Application note in Bioinformatics*. **21(13)**, 3051-3052.

[22] Nuel, G. 2005. Numerical solutions for pattern statistics on Markov chains. *Preprint 224 university of Evry, submitted to J. C. B (available on http://stat.genopole.cnrs.fr/˜gnuel)*.

[23] Nuel, G. 2005. Cumulative distribution of a geometric Poisson distribution. *Preprint 19-05 university of Evry, submitted to J. Stat. App. Sim. (available on http://stat.genopole.cnrs.fr/˜gnuel)*

[24] Nuel, G. 2006. Effective p-values computations using finite Markov chain imbedding: application to local score and to pattern statistics. *Algo. Mol. Biol.* **1(5)**.

[25] Pevzner P. A. , Borodovski M. Y. , and Mironov A. A. 1989. Linguistic of nucleotide sequences: the significance of deviation from mean statistical characteristics and prediction of frequencies of occurrence of words. *J. Biomol. Struct. Dyn*, **6**, 1013-1026.

[26] Press, W. H., Teukolsky, S. A., Vetterling, W. T., Flannery, B. P. 1992. Numerical recipes in C. *Cambridge University Press*.

[27] Prum B., Rodolphe F., and Turckheim E. de. 1995. Finding words with unexpected frequencies in DNA sequences. *J. R. Statist. Soc. B.* **11**, 190-192.

[28] Reinert G., Schbath S. 1998. Compound poisson and poisson process approximations for occurrences of multiple words in markov chains. *J. Comp. Biol*. **5**, 223-254.

[29] Reinert G., Schbath S. 1999. Compound poisson approximations for occurrences of multiple words. *Statistics in Molecular Biology and Genetics*, **33**, 257-275.

[30] Régnier M. 2000. A unified approach to word occurrence probabilities. *Discrete applied mathematics* **104 (1)**, 259-280.

[31] Régnier M., Szpankowski W. 1998. On pattern frequency occurrences in a markovian sequence. *Algorithmica*, **22 (4)**, 631-649.

[32] Robin S., Daudin J.J. 1999. Exact distribution of word occurrences in a random sequence of letters. *J. App. Prob.* **36**, 179–193.

[33] Robin, S., Daudin, J.-J., Richard, H., Sagot, M.-F., Schbath, S. Occurrence Probability of Structured Motifs in Random Sequences, *J. Comp. Biol.*, 9 (6):761-773, 2002.

[34] Roquain, E. et Schbath, S. 2006. Efficient Compound Poisson approximation for the number of occurrences of multiple words in Markov chains. *unpublished yet*

[35] Schbath, S. 1995. Compound Poisson approximation of word counts in DNA sequences. *ESAIM, Probab. Stat.* **1**, 1-16.

[36] Sorensen, D. C. 1995. Implicitly Restarted Arnoldi/Lanczos Methods For Large Scale Eigenvalue Calculations. *technical report, Rice university, Houston, Texas*.

[37] Stefanov, V. T., Pakes, A. G. 1997. Explicit distributional results in pattern formation. *Ann. Appl. Prob.* **7**, 666-678.

[38] Stefanov, V. T., Pakes, A. G. 1999. Explicit distributional results in pattern formation II. *Austral. and New Zealand J. Statist*. **41**, 79-90.

Quatrième partie

Etudes cas-témoins

Chapitre 14

A fast, unbiased and exact allelic test for case-control association studies

Article de Guedj, Wojcik, Della-Chiesa, Nuel, et Forner publié en 2006 par *Human Heredity*. La validité du test allélique pour l'association dans les étude cas-témoins y est discuté. Les calculs asymptotiques classiques introduisent un biais qui est mis en évidence. Un algorithme efficace permettant le calcul exact est proposé comme alternative.

A fast, unbiased and exact allelic test for case-control association studies

M. Guedj,[1,2] **J. Wojcik,**[2] **E. Della-Chiesa,**[1] **G. Nuel**[1] **and K. Forner**[2]

[1]Statistique et Genome Laboratory, FRANCE

[2]Serono Pharmaceutical Research Institute, SWITZERLAND

October 9, 2006

Running title:

A fast, unbiased and exact allelic test.

Corresponding author: Jerome Wojcik,

Serono Pharmaceutical Research Institute,

14, chemin des Aulx, CH-1228 Plan-les-Ouates, SWITZERLAND

Tel: +41-22-7069161

Fax: +41-22-7069992

e-mail: jerome.wojcik@serono.com

Abstract

Association studies are traditionally performed in the case-control framework. As a first step in the analysis process, comparing allele frequencies using the Pearson's chi-square statistic is often invoked. However such an approach assumes the independence of alleles under the hypothesis of no association, which may not always be the case. Consequently this method introduces a bias that deviates the expected type-I error-rate.

In this article we first propose an unbiased and exact test as an alternative to the biased allelic test. Available data require to perform thousands of such tests so we focused on its fast execution. Since the biased allelic test is still widely used in the community, we illustrate its pitfalls in the context of genome-wide association studies and particularly in the case of low-level tests. Finally, we compare the unbiased and exact test with the Cochran-Armitage test for trend and show it perfoms similarly in terms of power. The fast, unbiased and exact allelic test code is available in R, C++ and Perl at: `http://stat.genopole.cnrs.fr/software/fueatest`.

KEY WORDS: Association studies, exact test, allelic test, power

1. Introduction

Recent great progresses in the field of genotyping technologies have led to a decrease in cost and time for data production [1]. Geneticists can now consider launching large-scale genetic association studies in order to shed light on mechanisms responsible for complex human diseases and to provide opportunities for pharmaceutical companies to discover new drug targets.

Association analyzes are generally performed via a case-control design that has been preferred to family-based frameworks [2] since it is cheaper and quicker, and data are easier to collect. This approach involves unrelated individuals sampled from the same general population and clustered into two sub-populations (cases and controls) according to their disease status. When comparing the two populations, differences in terms of genotype frequencies should be due to an association between the marker and the phenotype under consideration.

Single-point strategies which treat one marker at a time are widely used as a preliminary step in the analysis. Individuals are organized into contingency tables according to their marker and disease status. Various approaches are proposed to test for association, based on either genotypes or alleles. Among them the genotypic, allelic and Hardy-Weinberg [3] chi-square tests as well as the Cochran-Armitage test for trends [4] are mainly invoked.

In this article, we focus on the approach based on allele frequencies to test for association. It has been shown to be unadapted when alleles are not paired independently [5] but it remains widely used. As an alternative we propose an unbiased test based on the exact computation of the p-value. Since high-throughput data require to perform a lot of tests, we have paid attention to the speed of execution by optimizing the implementation to make each exact test the less time con-

suming possible. Then an application on experimental data gives an idea about the real impact of using the biased test in the context of genome-wide association studies. The Cochran-Armitage test for trends has also been suggested as an alternative to the biased alelic test. To guide the choice between the two alternatives, we propose to study their respective powers focusing on situations where genotype frequencies do not comply with Hardy-Weinberg equilibrium (HWE).

2. Methods

2.1 *Testing for association*

Let us denote X, a case-control sample. Testing for association requires first to consider a null hypothesis (H_0) used to test a particular distribution of the observations. To do so, we consider a statistic defined as a function of the observations: $\mathcal{S} = f(X)$, and carefully chosen such that $\mathcal{S}$ grows when H_0 is less likely. Using the distribution of $\mathcal{S}$ under H_0, we can find a threshold (t_α) such as $\alpha = \mathbb{P}_{H_0}(\mathcal{S} \geqslant t_\alpha)$, where level α can be set to 5% for instance. The significance of an observed value (s^{obs}) is termed p-value and corresponds to the probability to observe data as or more extreme than the observation if H_0 were true: p-value $= \mathbb{P}_{H_0}(\mathcal{S} \geqslant s^{\text{obs}})$.

2.2 *The biased allelic test*

Two strategies exist to tabulate data from genetic case-control studies: the first classifies cases and controls according to their genotypes (Table 1) and the second according to their alleles (Table 2).

[Table 1 about here.]

[Table 2 about here.]

Allelic association tests aim at detecting significant differences in allelic frequencies between cases and controls testing the null hypothesis of no association with the disease. Alleles are supposed to be sampled from the case and control populations with probabilities p_d and p_h corresponding to the proportions of the susceptibility allele in cases and controls respectively. Under H_0, alleles are sampled from the same general population and p_d and p_h are equal to p. To test this hypothesis we classically consider the statistic:

$$\mathcal{Z}_A = \frac{\hat{p}_d - \hat{p}_h}{\sqrt{N\hat{p}(1-\hat{p})}},$$

with $\hat{p}_d = \frac{2d_2+d_1}{2N_d}$, $\hat{p}_h = \frac{2h_2+h_1}{2N_h}$ and $\hat{p} = \frac{2N_2+N_1}{2N}$, or the equivalent Pearson statistic applied to the allelic table:

$$\mathcal{S}_A = (\mathcal{Z}_A)^2 = \frac{2N.\left((2d_2+d_1)(2h_0+h_1) - (2d_0+d_1)(2h_2+h_1)\right)^2}{2N_d 2N_h.(2N_2+N_1)(2N_0+N_1)}.$$

Focusing on $\mathcal{S}_A$, the p-value can be calculated using two testing procedures: **(i)** Under H_0 and when data are in accordance with Cochran's conditions (each expected cell count greater than 5) [6], the distribution of $\mathcal{S}_A$ can be asymptotically well-approximated by a one degree-of-freedom (df) chi-square distribution. This comes down to the well-known chi-square test on contingency tables applied, in our case, to the 2×2 allelic table. **(ii)** An exact procedure also exists: the probability to observe a table conditionally to the margins is given by the hypergeometric distribution. Hence enumerating the whole set of 2×2 tables in accordance with the observed margins of the allelic table, the exact p-value comes down to the sum of the probabilities to observe 2×2 tables for which $\mathcal{S}_A$ is at least as great as s_A^{obs}.

Strategies based on allelic contingency tables may appear appealing since the sample size is doubled in comparison with the genotypic approach. However the

validity of this test has been recently discussed and some authors recommend against its use [5,7]. As described below, the null hypothesis makes two assumptions:

$$H_0^{(\text{biased})} : \left\{ \begin{array}{l} 1.\ p_d = p_h \\ 2.\ \text{Alleles are sampled independently} \end{array} \right\}.$$

Although genotypes are sampled independently from cases and controls, it is however clear that alleles are not: in the allelic contingency table, each individual contributes to two observations so alleles are actually sampled by two at a time. As a result, the way alleles are paired (corresponding to a departure from HWE) deviates $\mathcal{S}_A$ from its wrongly supposed distribution under H_0 and hence dramatically biases the p-value.

2.3 *The Cochran-Armitage test for trends*

Instead, Sasieni recommended to use the Cochran-Armitage test for trends [5]. It has been shown to be asymptotically equivalent to the allelic test when the combined population is in HWE (cases and controls) but makes no assumption about alleles matching. It is based on the genotypic contingency table. Genotypes are sampled from the case and control populations with probabilities $(p_{d_0}, p_{d_1}, p_{d_2})$ for cases and $(p_{h_0}, p_{h_1}, p_{h_2})$ for controls, which is more consistent with reality. The null hypothesis for the Cochran-Armitage test is no trend, which means that the proportions p_{d_i} are the same for all genotypes:

$$H_0^{(\text{trend})} : \{p_{d_0} = p_{d_1} = p_{d_2}\}.$$

In practice, the trend statistic $\mathcal{S}_T$ measures a linear trend in proportions weighted by a dose effect score x_i associated to each column of the contingency table. When x_i quantifies the number of high-risk allele, this test is equivalent to the score test in the logistic regression model where each SNP is coded according to its count

of high-risk allele. In our case:

$$\mathcal{S}_T = \frac{N.\left[N.(d_1+2d_2) - N_d.(N_1+2N_2)\right]^2}{N_d N_h.\left[N.(N_1+4N_2) - (N_1+2N_2)^2\right]} \underset{H_0}{\sim} \chi^2(1)$$

2.4 *An unbiased and exact allelic test*

As a more natural alternative to the biased allelic test, we propose to perform an unbiased test still based on the allelic Pearson statistic ($\mathcal{S}_A$) and on the sampling of genotypes instead of alleles taken independently.

$$H_0^{(\text{unbiased})} : \left\{ \; p_d = p_h \; \right\}.$$

Following this direction, Schaid and Jacobsen found out that using the classical but biased allelic test alters the type-I error-rate in a predictable manner and proposed a correction on the chi-square approximation [8]. Our alternative relies on an exact computation of p-values and can be thus termed as an unbiased and exact test. The principle is quite similar to the biased and exact procedure described below with the difference that instead of enumerating the 2×2 tables, we enumerate all the 2×3 tables in accordance with the observed margins of the genotypic table. In such a way, alleles are not considered individually, H_0 takes the allele matching into account and makes only the assumption of no association. Probabilities are computed using a multiple hypergeometric distribution and as previously, the exact p-value comes down to the sum of the probabilities to observe 2×3 tables for which $\mathcal{S}_A$ is at least as great than s_A^{obs}.

3. Implementation

Here we propose an accurate description of the implementation for this test. Divided in three main steps, it intends to avoid most of errors due to computation.

3.1 *Enumerating the genotypic tables*

Conditionally to the margins, 2×3 contingency tables have two degrees of freedom. Consequently a table is completely determined by setting two of its elements, for example d_0 and d_1 that we refer now to as i and j to stress out that they are integer variables (Table 3). These variables can take a range of values which depends on the margins:

- i takes all integer values in:

$$[i_{\min}, ..., i_{\max}] \overset{def.}{=} [\max(0, N_1 - N_h), ..., \min(N_1, N_d)],$$

- for a given value of i, j can take all integer values in:

$$[j_{\min}(i), ..., j_{\max}(i)] \overset{def.}{=} [\max(0, N_d - N_3 - i), ..., \min(N_2, N_d - i)].$$

Determining these intervals of variation allows us to enumerate the whole set of 2×3 tables in accordance with the margins of the observed genotypic table.

[Table 3 about here.]

3.2 *Comparing Pearson statistics*

This issue can appear trivial but can generate errors and lead to inaccurate results: on a computer, real numbers are stored in floating-point format for which the arithmetic is not exact [9]. As a result, two scores (s_1, s_2) that are supposed to be mathematically equal can show different computational representations so that the result of the logical expression $s_1 \geq s_2$ is unpredictable. In our case, it is possible to circumvent this problem by coming back to integer arithmetic that is mercifully exact on computers.

To do so we first outline a simplified expression for the Pearson statistic on $2 \times$

2 tables. By definition, for a given allelic table $\begin{array}{|c|c|}\hline a & b \\ \hline c & d \\ \hline \end{array}$ this statistic is usually expressed as:

$$\mathcal{S}_A = \frac{(a-\bar{a})^2}{\bar{a}} + \frac{(b-\bar{b})^2}{\bar{b}} + \frac{(c-\bar{c})^2}{\bar{c}} + \frac{(d-\bar{d})^2}{\bar{d}}$$

where $\bar{a}, \bar{b}, \bar{c}, \bar{d}$ are the expected values of a, b, c and d under H_0, *i.e.* the products of the margins divided by the total number $n = a + b + c + d$. For instance $\bar{a} = \frac{(a+b)(a+c)}{n}$. An interesting property is that all the numerators are equal. For example $(b-\bar{b})^2 = \left(b - \frac{(b+a)(b+d)}{n}\right)^2 = \left(\frac{nb-(a+b)(n-(a+c))}{n}\right)^2 = (a-\bar{a})^2$. The expression of $\mathcal{S}_A$ comes down to:

$$\mathcal{S}_A = \left(\frac{1}{\bar{a}} + \frac{1}{\bar{b}} + \frac{1}{\bar{c}} + \frac{1}{\bar{d}}\right)(a-\bar{a})^2 = \lambda(a-\bar{a})^2 \tag{1}$$

where $\bar{a}$ and λ only depend on the margins and hence are constant. We take advantage of this property to compare statistics by the way of integers: for two tables $T_1 = (a_1, b_1, c_1, d_1)$ and $T_2 = (a_2, b_2, c_2, d_2)$ with respective statistics $\mathcal{S}_1$ and $\mathcal{S}_2$, it follows that:

$$\mathcal{S}_1 \geq \mathcal{S}_2 \Longleftrightarrow |na_1 - n\bar{a}| \geq |na_2 - n\bar{a}|\,,$$

which is computationally interesting since the statement $|na_1 - n\bar{a}| \geq |na_2 - n\bar{a}|$ only involves integers.

3.3 *Computing the probability of a* 2×3 *table*

The probability of a 2×3 table T under H_0 is given by the multiple hypergeometric distribution:

$$p(T) = \frac{N_d!\, N_h!\, N_1!\, N_2!\, N_3!}{N!\, d_0!\, d_1!\, d_2!\, h_0!\, h_1!\, h_2!}$$

Numbers under consideration are so huge that the straightforward approach is not computationally feasible. Instead we use the log-factorial [9] and exponential

functions to compute approximates.

Let us note $\text{LF}(k) = \log(k!) = \sum_{i=1}^{k} \log(i)$, so that $\exp(\text{LF}(k)) \sim k!$. Thus:

$$p(T) \simeq \exp\left(\sum \text{LF}(k) - \sum \text{LF}(k')\right) \quad (2)$$

for $k \in \{N_d, N_h, N_1, N_2, N_3\}$ and $k' \in \{N, d_0, d_1, d_2, h_0, h_1, h_2\}$. The accuracy of these approximations is greater than 1.10^{-10}.

Algorithm 1 Pseudo-code of the basic implementation

```
compute all margins N_d, N_h, N_1, N_2, N_3, N and n = 2N
compute the observed allelic count a =
2d_0 + d_1 and nā
compute the observed threshold t = |na − nā|
let p − value = 0
for i = max(0, N_1 − N_h) to min(N_1, N_d)
  for j = max(0, N_d − N_3 − i) to min(N_2, N_d − i)
    if |n(2i + j) − nā| ≥ t then
      compute p(T_{i,j})
      p-value = p-value + p(T_{i,j})
output the p − value
```

3.4 *Optimizations*

In terms of implementation, the main difference between the exact biased and unbiased tests relies on the number of tables to enumerate. In most cases, considering 2×3 contingency tables in accordance with the data is likely to increase the execution time. The dimension of available data requires to compute thousands of tests. In order to speed up analyzes, we optimized our implementation, working on **(i)** a sensitive reduction of the number of tables to enumerate and **(ii)** a recursive relation between table probabilities $p(T)$. Interested readers can refer to the Appendixes 1 and 2 for more details.

4. Application

We use an application on real data to assess the consequences of using the biased test, first in terms of p-value deviation from the unbiased p-value (qualitative results), but also in terms of effective impact on predictions (quantitative results).

4.1 *Data*

We applied the biased and unbiased exact tests on genome-wide data concerning the multiple sclerosis. The data set consists in 66,990 SNPs. DNA from 279 patients and 301 controls were genotyped with this marker set using the 100K Affymetrix chip. The algorithm used for making genotype calls has been previously described by Affymetrix [10].

4.2 *Notation*

In what follows, p_u and p_b refer to the unbiased and biased exact allelic p-values respectively. Let p be the frequency of the allele of susceptibility, p_{HW} the p-value for the HW test on combined population and f the coefficient of consanguinity quantifying a deficit ($f > 0$) or excess ($f < 0$) of heterozygotes.

4.3 *Qualitative results*

First, Figure 1 displays the absolute error made using p_b to approximate p_u with respect to the strength of deviation to HWE and the allele frequency. As expected, the amount of deviation is directly proportional to the departure from HWE (Figure 1-A). The way alleles are paired deviates p_b in a predictive direction: a deficit of heterozygotes increases p_b and at the same time the rate of false positives. By opposition an excess of heterozygotes decreases p_b and so increases the conservatism of the test. These observations are also illustrated in the examples in Table 4. It is clear that p_u and p_b are strictly identical when the combined population is in HWE and that the absolute error increases with the amount of

departure from the equilibrium.

Figure 1-B represents the link that exists between allele frequencies (p) and the deviation of p_b. In the case of a deficit of heterozygotes, there is absolutely no visible dependence but in the case of an excess, the maximum amount of possible deviation is proportional to the quantity $p(1-p)$ and hence maximal for $p = 0.5$ and minimal at the bounds $p = 0$ and $p = 1$.

[Figure 1 about here.]

[Table 4 about here.]

4.4 *Quantitative results*

Even more interesting is the percentage of erroneous predictions made with the biased allelic test (Figure 1-C). It is quite substantial and increases as the level of the test decreases up to 9, 16, 30 and 40% for respectively a 5, 1, 0.1 and 0.01% level respectively. Since most of researchers would exclude SNPs that clearly fail HWE in controls [11], we also computed these erroneous prediction rates excluding SNPs for which the HWE test p-values for controls does not exceed the 5% signifance level (when taking into account multiple-testing using the Benjamini-Hochberg procedure [12]). In this case, these rates are less important but remain substantial, up to 28% for a test level of 0.01

4.5 *Validation*

Our implementation intends to avoid errors of computation. To be convinced of the exactness of our procedure, we compared our p-values with p-values calculated with an infinite precision, which requires an important time of execution. Both appear to be identical (data not shown). Moreover, we compared our p-values with p-values calculated with Monte-Carlo simulations (Table 4). The

distribution of a statistic under H_0 can be empirically deduced via a set of simulations, permuting at each iteration the case and control labels. Considering the precision due to the number of simulations performed, the situations tested show that exact and empirical p-values are not significantly different.

4.6 *Computational time*

We ran some benchmarks on simulated data to evaluate the gain made by the proposed optimizations. The machine used is an Itanium 2 (64 bits, 1.6 GHz) running GNU/Linux. We performed the unbiased exact test with the basic and optimized implementations on 10,000 markers for different sizes of population (Figure 2). The time of execution is strongly linked to the power of the computer but we can notice the great advantage of the optimized implementation. For large populations, the gain grows with the population size up to 6 and 12 times faster for data implicating 4,000 and 8,000 individuals respectively.

[Figure 2 about here.]

5. Unbiased and Trend tests: comparative assessment

Two alternatives are proposed to the biased allelic test based on two different statistics ($\mathcal{S}_A$ and $\mathcal{S}_T$) and testing for two slightly different null hypotheses, as described previously. These two approaches have been compared in a power study based on Monte-Carlo simulations: as long as it is possible to generate a case-control sample X under an alternative hypothesis (H_1), it is very easy to get an estimation of the power $\pi(\alpha) = \mathbb{P}_{H_1}(\mathcal{S} \geqslant t_\alpha)$. We first draw B samples denoting $x^{(i)}$ the i^{th} sample. From this sample we get B statistics $s^{(1)}, \ldots, s^{(B)}$ from which we compute the estimate:

$$\widehat{\pi}(\alpha) = \frac{\sharp\left\{s^{(i)} \geqslant t_\alpha\right\}}{B}.$$

This method is widely used in the field of statistical genetics when alternative distributions are hard to derive analytically [13]. But computing $\widehat{\pi}(\alpha)$ requires first to define a genetic model. Let a bi-allelic disease susceptibility locus (DSL) with its two possible alleles (the allele of susceptibility A and a), p be the frequency of the allele A and r_0, r_1 and r_2 the frequencies of the genotypes aa, aA and AA respectively. According to the Wright's model [14], we have: $r_2 = p^2 + fp(1-p)$, $r_1 = 2p(1-p) - 2fp(1-p)$ and $r_0 = (1-p)^2 + fp(1-p)$ with f the coefficient of consanguinity defined previously. Now we introduce the prevalence of the disease (K_p), and the penetrances (f_i) associated to each genotypes (i). Considering the relative risks (RR_i) such as $RR_i = \frac{f_i}{f_0}$ for $i = 1$ or 2, we define the four main modes of inheritance (MOI) underlying the mode of action of the DSL on the disease: recessive ($RR_1 = 1$), multiplicative ($RR_1 = \sqrt{RR_2}$), additive ($RR_1 = \frac{RR_2+1}{2}$) and dominant ($RR_1 = RR_2$). From these parameters, we can easily derive $f_0 = K_p/(r_0 + RR_1.r_1 + RR_2.r_2)$, $f_1 = RR_1.f_0$ and $f_1 = RR_1.f_0$ and finally genotype frequencies in case and control populations:

$$\begin{aligned}(p_{d_0}, p_{d_1}, p_{d_2}) &= \left(\frac{f_0.r_0}{K_p}, \frac{f_1.r_1}{K_p}, \frac{f_2.r_2}{K_p}\right) \\ (p_{h_0}, p_{h_1}, p_{h_2}) &= \left(\frac{(1-f_0).r_0}{1-K_p}, \frac{(1-f_1).r_1}{1-K_p}, \frac{(1-f_2).r_2}{1-K_p}\right)\end{aligned}$$

In such a context, H_0: $\{RR_2 = 1\}$ and H_1: $\{RR2 > 1\}$.

5.1 *Simulations*

Simulations are performed considering three distinct sets. The power is estimated first with respect to the susceptibility allele frequency (p), then according to the coefficient of consanguinity (f) introduced in the general population and finally we investigate the effect due to the variation of sub-population sizes (N_d and N_h). All situations are considered for a prevalence $K_p = 0.05$, the four MOI and

two strengths of disease implication ($RR_2 = 1.5$ and $RR_2 = 2$). Each estimate is done on the basis of 10,000 simulations.

5.2 *Results*

Main results are summarized in Figures 3 and 4. Full results are compiled in supplementary data available upon request. With the relative risks considered, the additive and multiplicative models present actually close values of RR_1 and hence very similar results. In the case where p varies (Figure 3), the power of the trend and the unbiased allelic tests are not significantly different. They are also similar to the biased allelic test which is consistent with the fact the three tests are strictly identical when HWE holds in the combined population. However differences can still be observed since a departure from the equilibrium in cases can be generated by the disease.

[Figure 3 about here.]

When the equilibrium does not hold in the general population ($f \neq 0$, Figure 3), both trend and unbiased allelic tests show comparable power with each MOI, increasing along with f. We also represented the power of the biased allelic test (dotted) although there is no sense to take it in consideration. Previous conclusions about the deviation of the type-I error-rate with respect to f corresponds here to an artificial decrease ($f < 0$) and increase ($f > 0$) of power. Despite their comparative efficiency in all situations studied, the trend and unbiased allelic tests however show small but significant differences. Under a dominant model, the trend test tends to be more powerful than the unbiased allelic test which, by opposition, appears more powerful under a recessive model. These differences are clearer when the population contains a high proportion of heterozygotes ($f < 0$)

and increase with the ratio N_d/N_h (Figure 4). In addition, investigating the potential influence of the sample size, results outline that an augmentation of N clearly accentuates the bias introduced by the biased allelic test.

[Figure 4 about here.]

On our real dataset, the biased p-value distribution appears closer to the unbiased one than to the trend one (Figures 5-A,B,C). This is logical since the biased and unbiased tests are based on the same statistic ($\mathcal{S}_A$) and almost the same H_0 hypothesis. In practice, differences in term of findings are not important between the unbiased and the trend test, up to 8%, with a mean of 4% along with the level of the test (Figure 5-D).

[Figure 5 about here.]

6. Discussion

With the work of Sasieni, it is today well-known that the classical allelic test which assumes an independence of alleles is biased by the allele matching [5]. The solution we developed here constitutes an unbiased and exact test to compare allele frequencies. It has the advantage of making no assumption about the way alleles are paired and unlike asymptotic tests, its validity is not constraint to the filling of the contingency tables. Based on real genome-wide data, our results clearly outline that the strength of deviation directly depends on the importance of the departure from HWE. Moreover, the direction in which the p-value deviates is linked to the way the equilibrium is not respected. The sample size also appears to have an effect on the width of the bias as well as the allele frequency in the case of an excess of heterozygotes. Some of these observations actually confirm and illustrate on genome-wide real data results analytically deduced by Schaid

and Jacobsen [8]. Finally, we show that this bias can have a substantial impact on predictions that increases as the level of the tests decreases.

In practice, association studies are not working in a favorable situation with regard to our conclusions: **(i)** the threshold chosen to decide for the acceptance or the rejection of the H_0 hypothesis is generally low, commonly set to 5 or 1% and even lower when one takes the multiple-testing into account, **(ii)** recent studies aim to detect common susceptible alleles responsible for common diseases and **(iii)** involve hundreds of subjects. As a result, if the impact on true findings is not easy to assess in real data, it remains potential and likely to be substantial. One usual way to deal with HWE violations is to discard markers that fail HWE in controls with the argument that disequilibrium may be indicative of genotyping errors [11]. If such a strategy manages to reduce the percentage of erroneous predictions, it still represents an important amount of errors. In addition, HWE may be violated because of genotyping error but not only: chance, nonrandom mating, inbreeding, genetic drifting, selection, population stratification or combination of these reasons are mainly evoked and also susceptible to introduce a statistical bias on allele-based contrast [15]. Because the origin of HW disequilibrium can not be ensured, unbiased tests should reasonably be applied on all markers, with additional attention on interpretations when HWE does not hold in controls. And besides the impact on predictions, the exactness of p-values can be really important when one applies methods based on p-value distributions such as FDR-based approaches [16].

Compared to the trend test, the unbiased allelic test presents comparable power under various alternatives and results on real data are quite similar. Consequently the choice between the trend or the unbiased allelic tests should rather be based on

practical reasons. As underlined by Knapp, the allelic statistic ($\mathcal{S}_A$) appears more intuitive than the trend statistic ($\mathcal{S}_T$) to the genetics community [17]. Moreover, we see the unbiased test as a more natural alternative to the biased test and we have developped the unbiased test in various languages to support its diffusion. With the recent accumulation of data, genetic studies require the simultaneous treatment of a large amount of markers (up to 500,000). Besides the statistical problems that the analysis raises (multiple-testing, curse of dimensionality), the time of execution is also an important issue. Exact testing procedures are time consuming and so we found relevant to improve the basic implementation in order to accelerate the algorithm execution. Using the optimized implementation is clearly advantageous for the treatment of large populations. Such computational considerations, that can be easily extended to other contexts, along with the current power of computers make the use of exact tests reasonable.

A major issue in genetic studies is to observe replications in independent populations to ensure the reliability of the results but they are generally difficult to obtain [18,19]: with the complexity of the diseases under investigation, lack of power, multiple-testing [12], genotyping errors [20] and population stratification [21,22] are generally invoked to explain the difficulty in replicating findings. In such a context it is important to avoid errors due to the choice of the method of analysis. The procedure we propose meets this goal. Unbiased, it avoids false predictions produced by the use of the biased allelic test. Exact, it avoids additional errors due to asymptotic approximations and computations. Fast and available under different versions, its execution is easy and adapted to very large data sets: `http://stat.genopole.cnrs.fr/software/fueatest`.

ACKNOWLEDGEMENTS

We would like to thank Serono for generating data and more particularly Hiroaki TANAKA as well as Bernard PRUM and Franck PICARD for fruitful discussions.

REFERENCES

[1] Hirschhorn JN, Daly MJ: Genome-wide association studies for common diseases and complex traits. Nature Reviews Genetics 2005;6:95–108.

[2] H Zhao H: Family-based association studies. Statistical Methods in Med Research 2000;9:563–587.

[3] Nielsen DM, Ehm MG, Weir BS: Detecting marker-disease association by testing for hardy-weinberg disequilibrium at a marker locus. Am J Hum Genet 1999;63:1531–1540.

[4] Armitage P: Tests for linear trends in proportions and frequencies. Biometrics 1955;11:375–386.

[5] Sasieni PD: From genotypes to genes : doubling the sample size. Biometrics 1997;53:1253–1261.

[6] Cochran WG: The chi-square test of goodness of fit. Annals of Mathematical Statistics 1952;23:315–345.

[7] Jackson MR, Genin E, Knapp M, Escary JL: Accurate power approximation for χ^2-tests in case-control association studies of complex disease genes. Annals of Hum Genet 2002;66:307–321.

[8] Schaid DJ, Jacobsen SJ: Biased tests of association : comparisons of allele frequencies when departing from hardy-weinberg proportions. Am J of Epi-

demiololgy 1999;149:706–711.

[9] Press WH, Teukolsky SA, Vetterling WT, Flannery BP: Numerical recipies in C. Cambridge university press, 1992.

[10] Matsuzaki H, Dong S, Loi H, Di X, Liu G: Genotyping over 100,000 snps on a pair of oligonucleotide arrays. Nature Methods 2004;1:109–111.

[11] Hosking L, Lumsden S, Lewis K, Yeo, A, McCarthy L, Bansal A, Riley J, Purvis I, Xu CF: Detection of genotyping erros by Hardy-Weinberg equilibrium testing. E J Hum Genet 2004;12:395–399.

[12] Benjamini Y, Hochberg Y: Controlling the false discovery rate: a practical and powerful approach to multiple testing. Journal of the Royal Statistical Society B 1995;57:289–300.

[13] Longmate JA: Complexity and power in case-control association studies. Am J Hum Genet 2001;68:1229–1237.

[14] Wright S: Systems of mating. Genetics 1921;6:111–178.

[15] Trikalinos TA, Salanti G, Khoury MJ, Ioannidis JPA: Impact of violations and deviations in Hardy-Weinberg equilibrium on postulated gene-disease associations. Am J Epidemiol 2006;163:300–309.

[16] Storey JD, Tibshirani R: Statistical significane for genome-wide studies. PNAS 2003;100:9440–9445.

[17] Knapp M: re: "biased tests of association: comparisons of alleles frequencies when departing from Hardy-Weinberg proportions". Am J Epidemiol 2003;153:287.

[18] Page G, George V, Go R, Page P, Allison D: "Are we there yet?": deciding when one has demonstrated specific genetic causation in complex diseases and quantative traits. Am J Hum Genet 2003;73:711–719.

[19] Shen H, Liu Y, Liu P, Recker RR, Deng HW: Nonreplication in genetic studies of complex diseases – lessons from studies of osteoporosis and tentative remedies. Journal of bone and mineral research 2005;20:365–376.

[20] Gordon D, Heath SC, Liu X, Ott J: A transmission/disequilibrium test that allows for genotyping errors in the analysis of single nucleotide polymorphism data. Am J Hum Genet 2001;69:371–380.

[21] Devlin B, Roeder K: Genomic control for association studies. Biometrics 1999;55:997–1004.

[22] Pritchard JK and Rosenberg NA: Use of unlinked genetic markers to detect populations stratificiations in association studies. Am J Hum Genet 1999;65:220–228.

Appendix 1: Optimized implementation

Reducing the number of tables to enumerate

To compute p-values, we can restraint the enumeration to tables that present a better/equal or worse statistic than the observed one, termed as "better" and "worse" tables respectively:

$$p - \text{value} = \sum_{\text{better tables}} p(T) = 1 - \sum_{\text{worse tables}} p(T)$$

Considering the smallest set of tables (instead of the whole one) decreases the time of execution but also the accumulation of imprecision. Since in genome-wide data most of the variables are likely to be under H_0, it may generally exists less worse tables than better tables. Therefore we will now focus on obtaining this subset of worse tables. The analog reasoning can be proposed for better tables.

For a given value of i, let $J_{\text{w}}(i)$ be the set of j values corresponding to worse tables. Defining floor(x) as the highest integer $\leq x$ and ceil(x) as the lowest

integer $\geq x$, it can be shown that:

$$J_{\text{w}}(i) = \left[\text{floor}\left(\frac{(n\bar{a}-t)}{n}\right) - 2i + 1,\ \text{ceil}\left(\frac{(n\bar{a}+t)}{n}\right) - 2i - 1\right] \cap [j_{\min}(i), j_{\max}(i)]$$

As a result, a given value of i is obviously useless if $J_{\text{w}}(i)$ is empty for this value. So we can also describe the useful set I_u of values for i for which $J_{\text{w}}(i) = \emptyset$. See Appendix 2 for more details.

Recursive relation between table probabilities under H_0

In principle, the computation of the probability $p(T_{i,j})$ requires a call to the exponential function (see Relation 2). Noticing the recursive relation that exists between $p(T_{i,j})$ and $p(T_{i,j+1})$, it is possible to use this function only once per i value:

$$p(T_{i,j+1}) = \frac{(N_d - i - j)(N_1 - j)}{(j+1)(N_h - N_0 - N_1 + i + j + 1)} \times p(T_{i,j})$$

To save some more computational time, we can pre-calculate $\text{LF}(t)$ for $t \in [0, ..., N]$ as well as $K = \sum \text{LF}(k) - \text{LF}(N)$ for $k \in \{N_d, N_h, N_1, N_2, N_3\}$ that is constant through the algorithm execution.

Appendix 2: Determining the set I_u

From appendix 1 we have $J_w(i) = [\alpha - 2i,\ \beta - 2i] \cap [j_{\min}(i), j_{\max}(i)]$ with $\alpha = \text{floor}\left(\frac{(n\bar{a}-t)}{n}\right) - 1$ and $\beta = \text{ceil}\left(\frac{(n\bar{a}+t)}{n}\right) + 1$. In the very particular case where $\alpha > \beta$, we can see that $J_w(i) = \emptyset\ \forall i$, meaning that there is no "worse" table and

that the p-value is exactly 1. So in the following we will suppose that $\alpha \leq \beta$:

$$J_w(i) \neq \emptyset \;\Leftrightarrow\; (\alpha - 2i \leq j_{\max}(i)) \text{ AND } (j_{\min}(i) \leq \beta - 2i),$$

but

$$\begin{aligned}
\alpha - 2i \leq j_{\max}(i) &\Leftrightarrow \alpha - 2i \leq \min(N_1, N_d - i), \\
&\Leftrightarrow \left\{ \begin{array}{c} \frac{\alpha - N_1}{2} \leq i \leq N_d - N_1 \\ i > N_d - N_1 \text{ AND } i \geq \alpha - N_d \end{array} \right\}, \\
&\Leftrightarrow \left\{ \begin{array}{c} \text{ceil}(\frac{\alpha - N_1}{2}) \leq i \leq N_d - N_1 \\ \max(\alpha - N_d, N_d - N_1 + 1) \leq i \end{array} \right\},
\end{aligned}$$

and

$$\begin{aligned}
j_{\min}(i) \leq \beta - 2i &\Leftrightarrow \max(0, N_d - N_2 - i) \leq \beta - 2i, \\
&\Leftrightarrow \left\{ \begin{array}{c} i \leq N_d - N_2 \text{ AND } i \leq \beta - N_d + N_2 \\ i > N_d - N_2 \text{ AND } i \leq \text{floor}(\frac{\beta}{2}) \end{array} \right\}, \\
&\Leftrightarrow \left\{ \begin{array}{c} i \leq \min(N_d - N_2, \beta - N_d + N_2) \\ N_d - N_2 + 1 \leq i \leq \text{floor}(\frac{\beta}{2}) \end{array} \right\},
\end{aligned}$$

hence

$$i \in I_u \Leftrightarrow \left\{ \begin{array}{c} \text{ceil}(\frac{\alpha - N_1}{2}) \leq i \leq \min(N_d - N_2, \beta - N_d + N_2, N_d - N_1) \\ \max(\text{ceil}(\frac{\alpha - N_1}{2}), N_d - N_2 + 1) \leq i \leq \min(N_d - N_1, \text{floor}(\frac{\beta}{2})) \\ \max(\alpha - N_d, N_d - N_1 + 1) \leq i \leq \min(N_d - N_2, \beta - N_d + N_2) \\ \max(\alpha - N_d, N_d - N_1 + 1, N_d - N_2 + 1) \leq i \leq \text{floor}(\frac{\beta}{2}) \end{array} \right\}.$$

Finally, I_u is the union of 4 disjoint, possibly empty, intervals $I_1 \cup I_2 \cup I_3 \cup I_4$ where $I_k \cap I_\ell = \emptyset \, \forall \, k \neq \ell$ and:

$$\begin{aligned}
I_1 &= \left[\max\left(i_{\min}, \text{ceil}(\frac{\alpha - N_1}{2}) \right), \min\left(N_d - N_2, \beta - N_d + N_2, N_d - N_1, i_{\max} \right) \right], \\
I_2 &= \left[\max\left(i_{\min}, \text{ceil}(\frac{\alpha - N_1}{2}), N_d - N_2 + 1 \right), \min\left(N_d - N_1, \text{floor}(\frac{\beta}{2}), i_{\max} \right) \right], \\
I_3 &= \left[\max\left(i_{\min}, \alpha - N_d, N_d - N_1 + 1 \right), \min\left(N_d - N_2, \beta - N_d + N_2, i_{\max} \right) \right], \\
I_4 &= \left[\max\left(i_{\min}, \alpha - N_d, N_d - N_1 + 1, N_d - N_2 + 1 \right), \min\left(\text{floor}(\frac{\beta}{2}), i_{\max} \right) \right].
\end{aligned}$$

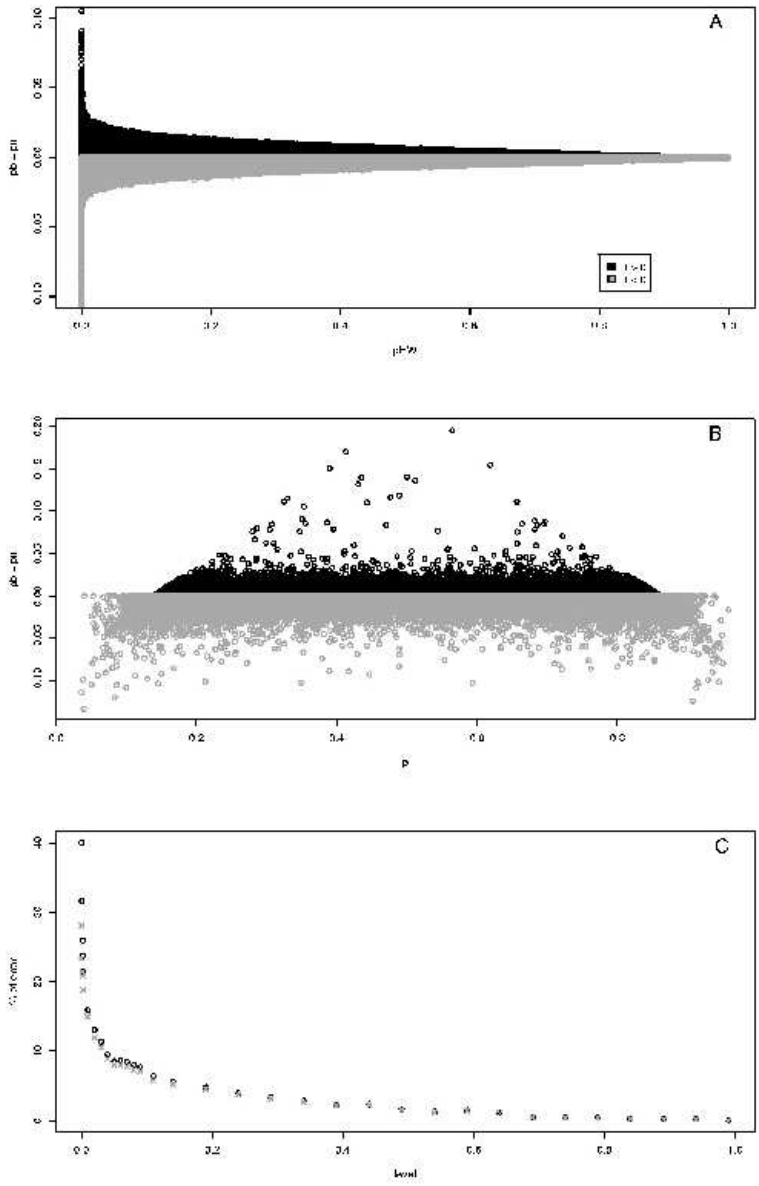


Figure 1. Deviation of p_b from p_u: A- This graphic represents the absolute error made using p_b to approximate p with respect to the strength of departure from HWE. **B-** It outlines the dependence between the absolute error and the frequency of the allele of susceptibility (p). **C-** The percentage of errors is estimated by the ratio of total false positives and negatives made by the use of the biased test on the count of positives predicted by the unbiased exact test. It is represented with respect to the level of the test for the whole set of SNPs (**o**) and this same set excluding SNPs that fail the HWE test (**x**).

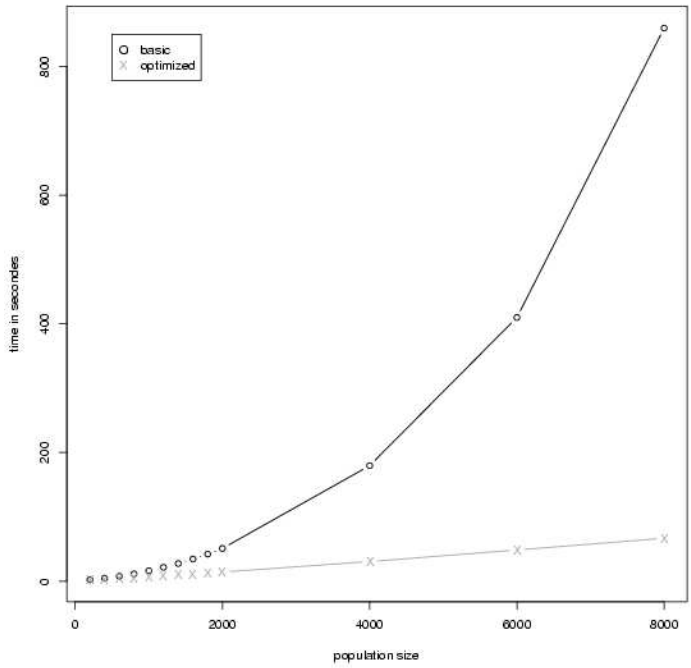


Figure 2. Computational time: the time of execution is assessed for the basic and the optimized algorithms on 10,000 simulated SNPs and increasing sizes of population.

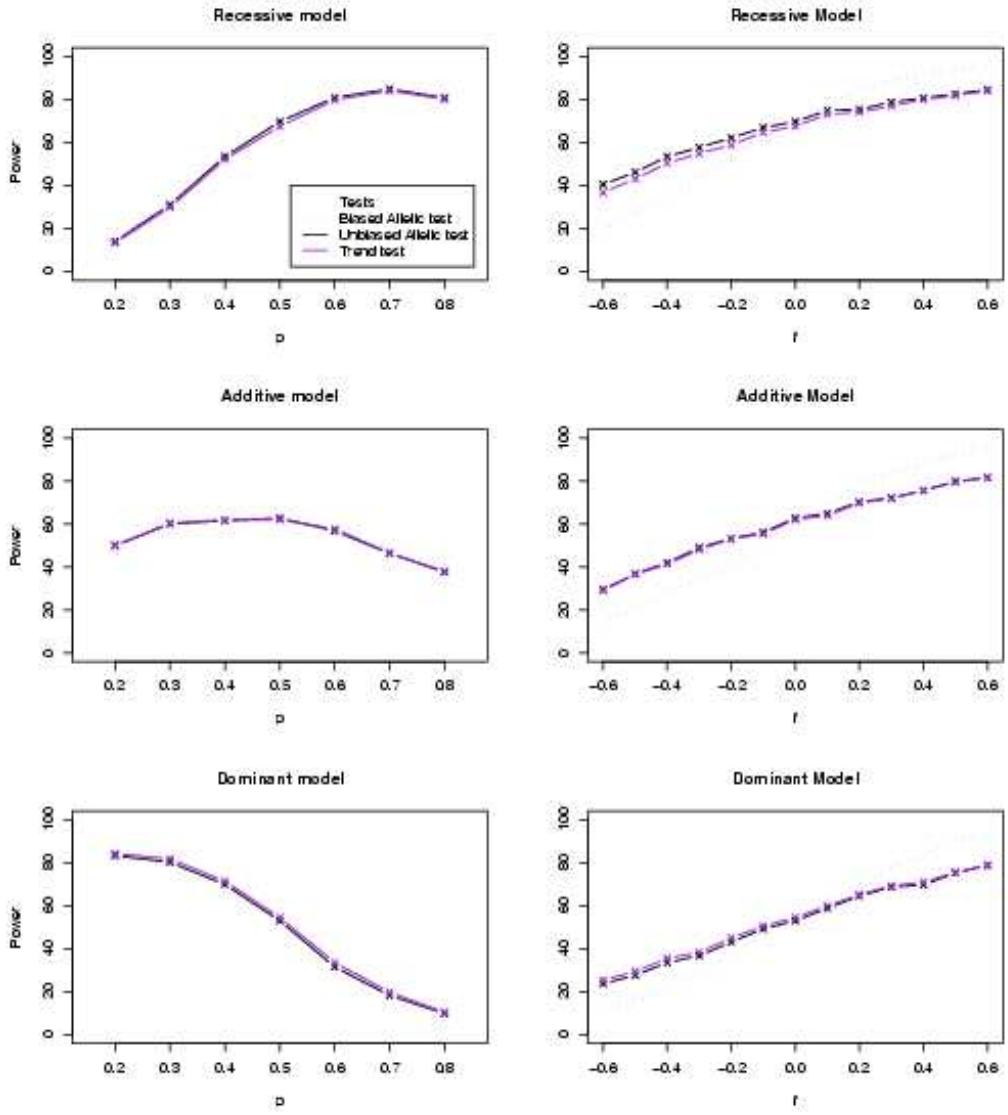


Figure 3. Power estimation with respect to p and f ($K_p = 0.05$, $RR_2 = 1.5$ and $n_D = n_C = 500$).

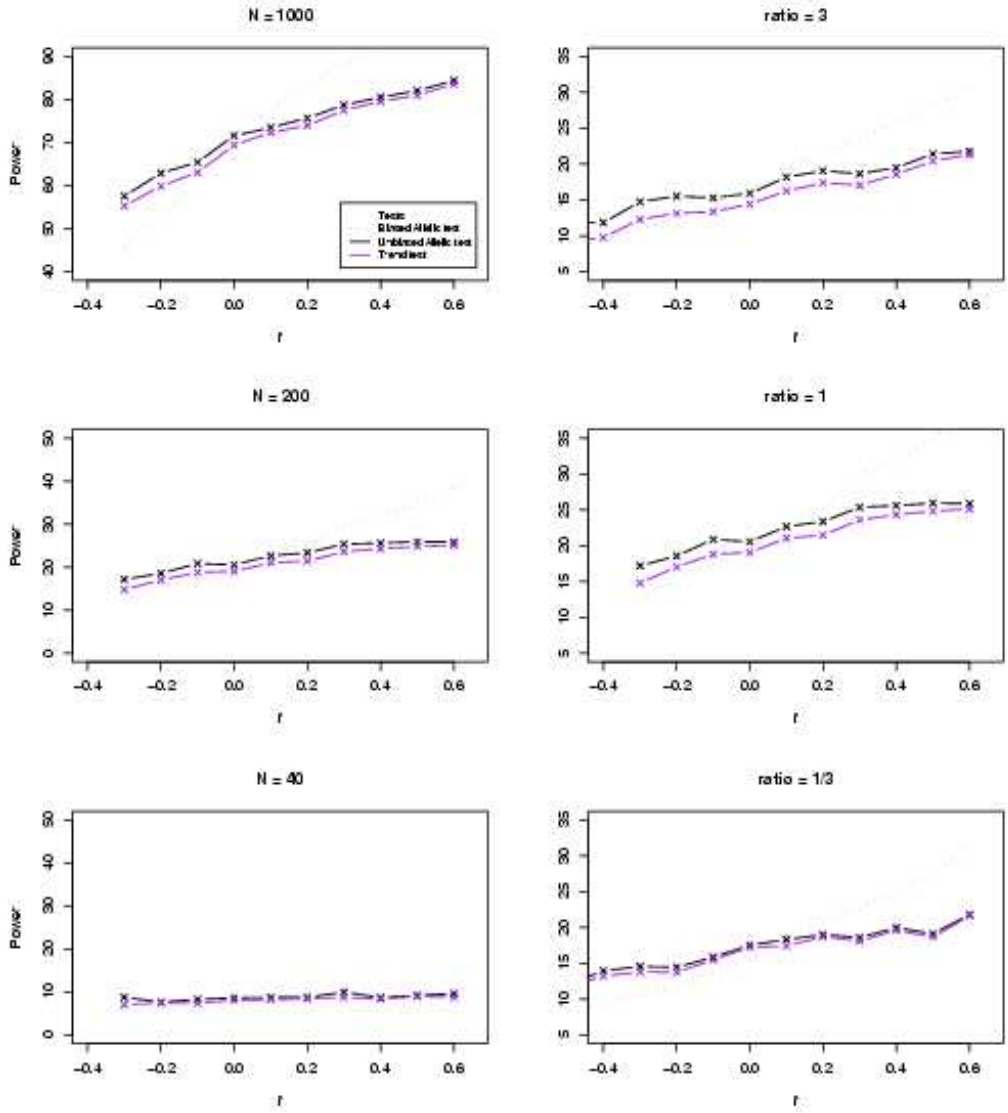


Figure 4. Power estimation with respect to f, N and the ratio N_d/N_c ($K_p = 0.05$, $p = 0.5$, $RR_2 = 1.5, RR_1 = 1$.)

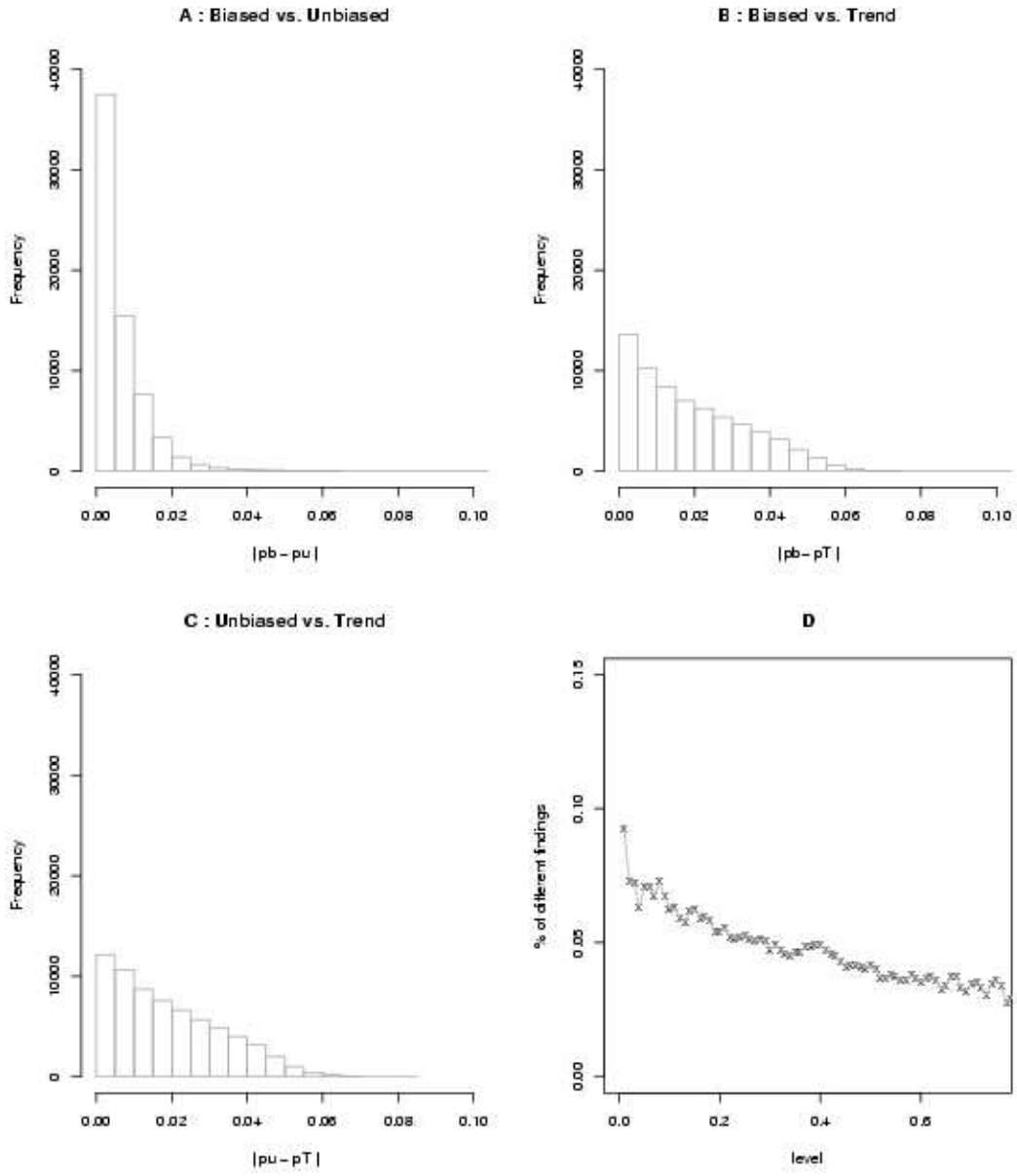


Figure 5. A,B,C- Histograms of absolute differences between the p-values of the biased, unbiased and trend tests. **D-** It represents the percentage of different findings between the biased and trend test with respect to the level of the test.

	aa	aA	AA	margins
case (diseased)	d_0	d_1	d_2	N_d
control (healthy)	h_0	h_1	h_2	N_h
margins	N_0	N_1	N_2	N

Table 1

Genotypic contingency table (d_i/h_i stands for the frequency of diseased/healthy individuals having i alleles A).

	a	A	margins
case	$2d_0 + d_1$	$2d_2 + d_1$	$2N_d$
control	$2h_0 + h_1$	$2h_2 + h_1$	$2N_h$
margins	$2N_0 + N_1$	$2N_2 + N_1$	$2N$

Table 2
Corresponding allelic contingency table.

	aa	aA	AA	margins
case	i	j	$N_d - i - j$	N_d
control	$N_0 - i$	$N_1 - j$	$N_h - N_0 - N_1 + i + j$	N_h
margins	N_0	N_1	N_2	N

Table 3
$T_{i,j}$ the 2×3 genotypic table with respect to the values of i, j and the margins.

p_{HW}	f	**Genotypic table**	p_u	p_b	p_{MC}
1	$= 0$	29 117 124 34 135 128	0.571	0.571	0.571
0.5	> 0	26 109 113 28 111 144	0.326	0.318	0.326
0.5	< 0	97 113 26 128 123 28	0.347	0.353	0.347
0.05	> 0	10 77 140 24 82 161	0.310	0.288	0.310
0.05	< 0	155 94 9 171 90 5	0.228	0.249	0.229

Table 4
Biased, unbiased and Monte-Carlo p-values for five representative examples: *each example varies in the strength of departure from the Hardy-Weinberg equilibrium and in the direction of departure. p_{MC} was computed based on 10^6 simulations.*

Chapitre 15

Detecting Local High-Scoring Segments : a First-Stage Approach for Genome-Wide Association Studies

Article de Guedj, Robelin, Hoebeke, Lamarine, Wojcik, et Nuel accepté en 2006 par *Statistical Applications in Genetics and Molecular Biology*. Cet article propose d'utiliser le score local pour détecter des accumulations significatives de signaux d'associations simple locus dans les études d'association. On recherche l'ensemble des segments optimaux et sous optimaux de SNPs, puis un certain nombre d'entre eux sont automatiquement sélectionnés d'après un critère de significativité. Une p-value globale est finalement retournée. LHiSA (*Local High-scoring Segments for Association*) est le nom logiciel qui met cette méthode à la disposition de la communauté.

Statistical Applications in Genetics and Molecular Biology

Volume 5, *Issue* 1 2006 *Article* 22

Detecting Local High-Scoring Segments: a First-Stage Approach for Genome-Wide Association Studies

Mickael Guedj[*] David Robelin[†] Mark Hoebeke[‡]
Marc Lamarine[**] Jérôme Wojcik[††] Gregory Nuel[‡‡]

[*]Laboratoire Statistique et Génome, guedj@genopole.cnrs.fr
[†]Laboratoire Statistique et Génome, robelin@genopole.cnrs.fr
[‡]Laboratoire Statistique et Génome, hoebeke@genopole.cnrs.fr
[**]Serono Pharmaceutical Research Institute, marc.lamarine@serono.com
[††]Serono Pharmaceutical Research Institute, Jerome.Wojcik@serono.com
[‡‡]Laboratoire Statistique et Genome, nuel@genopole.cnrs.fr

Detecting Local High-Scoring Segments: a First-Stage Approach for Genome-Wide Association Studies*

Mickael Guedj, David Robelin, Mark Hoebeke, Marc Lamarine, Jérôme Wojcik, and Gregory Nuel

Abstract

Genetic epidemiology aims at identifying biological mechanisms responsible for human diseases. Genome-wide association studies, made possible by recent improvements in genotyping technologies, are now promisingly investigated. In these studies, common first-stage strategies focus on marginal effects but lead to multiple-testing and are unable to capture the possibly complex interplay between genetic factors.

We have adapted the use of the local score statistic, already successfully applied to analyse long molecular sequences. Via sum statistics, this method captures local and possible distant dependences between markers. Dedicated to genome-wide association studies, it is fast to compute, able to handle large datasets, circumvents the the multiple-testing problem and outlines a set of genomic regions (segments) for further analyses. Applied to simulated and real data, our approach outperforms classical Bonferroni and FDR corrections for multiple-testing. It is implemented in a software termed LHiSA for Local High-scoring Segments for Association and available at: http://stat.genopole.cnrs.fr/software/lhisa.

KEYWORDS: association studies, local score, sum statistics, SNP

*We would like to thank Serono for providing data as well as Bernard Prum's team for fruitful discussions; more particularly Franck Picard, Karl Forner and Hiroaki Tanaka.

Introduction

Localizing trait loci responsible for human diseases is a challenge for new drug target discovery. Limited results from linkage analysis scans and the increasing availability of dense Single Nucleotid Polymorphisms (SNP) maps (Collins et al. 1998) due to rapid improvements in SNP genotyping technologies (Shendure et al. 2004, Hirschhorn et al. 2005) have recently led geneticists towards genome-wide association studies with hopes of encouraging results concerning our understanding of the genetic basis of complex diseases (Risch and Merikangas 1996, Risch 2000, Carlson et al. 2004, Hirschhorn et al. 2005). Association analyses can be carried out in a family-based framework using the Transmission Disequilibrium Test (TDT) (Spielman and Ewens 1996) for instance, or in a case-control design where individuals are unrelated, sampled from the same general population, and separated into cases and controls. The TDT method has been shown to be more robust to confounding factors such as stratification but also more susceptible to technological false-positive results (Ott 2004). Gordon et al. (2001) have proposed an approach to tackle genotyping errors in a TDT analysis. However case-control studies are still largely preferred mainly because they require less cost and time to collect data (McGinnis et al. 2002). They merely rely on the fact that genotypes associated with the disease should accumulate among cases compared to controls.

Nevertheless, association analyses appear to be insufficient to shed light on the causative genes of non-mendelian diseases. One reason pointed out by Page et al. (2003) is that complex diseases result from the interplay of multiple genetic and environmental factors. As a result each causal gene only makes a small contribution to overall heritability and "disease" alleles may be present in the healthy population. Classical marker-by-marker analyses based on contingency tables can then hardly succeed, since they rely on the detection of marginal effects. In addition, the statistical problem of multiple-testing arises, inflating the number of false positive results (type-I errors) when a large number of SNPs are tested independently. To face such complexity, statisticians and computer scientists have developed numbers of multi-marker strategies that can be divided into two classes, as underlined by Hoh and Ott (2003): "multipoint analysis" for the joint analysis of neighbouring marker loci, with the purpose of localizing one disease locus independently from the others (Morris et al. 2003, Tregouet et al. 2004, Schaid 2004) and "multi-locus" approaches that are specifically designed to find multiple disease loci, possibly on different chromosomes (Nelson et al. 2001, North et al. 2003, Yoon et al. 2003, Bastone et al. 2004, Bureau et al. 2005). Such strategies appear very promising in considering various interaction patterns between loci, but present

some weaknesses (Wille et al. 2003): First, most of them are not appropriate to treat data of large dimensionality. Moreover these methods require generally a difficult and unintuitive specification of parameters. Finally the statistical assessment of the results is not always available. See Hoh and Ott's review (2003) on mathematical multi-locus approaches for more detailed discussions.

Consequently, further efforts have to be made to propose methods that are easy to use, computationally fast, able to handle large data sets while reducing the multiple-testing problem. Sum statistics based methods can help to meet these needs. Based on simulation studies conclusions, Longmate (2001) suggested that combining single-marker genetic marginal effects should increase the power to detect susceptibility genes, possibly interacting with each other. One approach to combine individual association scores without assuming any interaction pattern among markers can be done with sums. Two examples of such strategies termed "scan statistics" (Hoh and Ott 2000) and "set association" (Hoh et al. 2001) have been proposed: they have been shown to be more powerful than conventional single-point analyses and presented promising results when applied to real data (Whille et al. 2003, Hao et al. 2004).

As part of the sum statistics family methods for association, we propose the use of the local score. This statistic has been a matter of theorical works (Altschul and Erickson 1986a, Altschul and Erickson 1986b, Karlin and Altschul 1990, Dembo and Karlin 1991, Karlin and Dembo 1992, Mercier and Daudin 2001) and frequently applied to long DNA or protein sequences to locate transmembrane or hydrophobic segments, DNA-binding domains, regions of concentrated charges (Karlin et al. 1991, Brendel et al. 1992, Karlin and Brendel 1992). It has been extended to identify similar regions among two or more sequences (Altschul et al. 1990) . See Karlin (2005) for a review on the topic.

In association studies, the local score approach intends to delimit segments (or subsequences) presenting unexpected accumulations of high scores, assuming that such features may be biologically relevant and outline etiological genomic regions. It represents a natural improvement of sliding-frame methods since it does not require to specify the size of the segments. Via sum statistics, we will see how this method considers local information to identify each segment and can be thus termed as a multi-point and multi-locus strategy. It proposes at the end a selection of candidate regions through the genome associated with a measure of statistical significance.

Methods

Definition

Let $\mathbb{X} = (X_i)_{i=1,...,n}$ a sequence of real random variables:

$$H = \max_{1 \leqslant i \leqslant j \leqslant n} \left(\sum_{i}^{j} X_k \right)$$

defines the local score assigned to $\mathbb{X}$ (Figure 1). In practice, the local score statistic is defined as the value of the segment with the maximal sum of scores X_i; this segment is also termed as "maximal scoring subsequence", "locally optimal subsequence" or "maximal sum interval" in the literature. Note that the segment is maximal in the sense that it can not be extended or shortened without reducing the local score. Consequently the variables X_i must have a negative expectation otherwise the maximal segment would easily span the entire sequence, which is not consistent with the aim of this approach.

This definition of the local score restrains our search to the highest scoring sequence but data may not contain only one region of interest: next highest scoring subsequences are as potentially interesting. However, segments with the successive high scores could overlap the highest-scoring one and lead to the identification of more or less the same genomic region, yielding to no additional useful information. Consequently, it appears better to look for disjoint segments, with the k^{th} best segment defined as the local score of the initial sequence disjoint from the preceding $k-1$ best segments. We consider in this case $H^{(1)} \geqslant ... \geqslant H^{(k)}$ as being the scores of the k first and distinct highest-scoring segments.

Test for association

Given the local score statistic, advantages to simple-marker strategies will arise from the definition of a more appropriate alternative hypothesis (H_1): there is at least one contiguous segment of the genome associated with the disease. This alternative is justified by haplotypes that have remained intact for many generations around a disease susceptibility locus (DSL). As a result, instead of looking for one marker at a time, it has been shown relevant to look for a stretch of DNA inherited in all mutation carriers (Clayton et al. 2004). From this point of view, our approach can be categorized as a haplotype similarity based method (Tzeng et al. 2003). Note that when more than one mutation is suspected to alter the gene function, we expect an accumulation of high scores of association in a quite small genomic region, situation also well detected by the local score statistic.

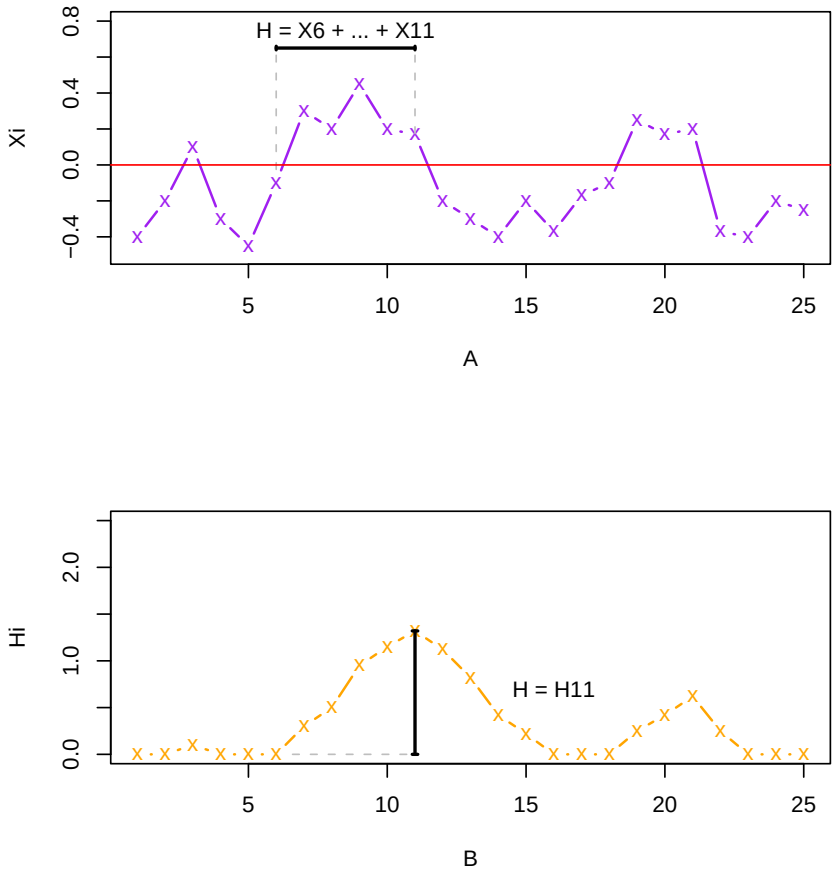


Figure 1. Local score statistic : A - A set of scores X_i represents the sequence $\mathbb{X}$. The highest-scoring segment is delimited. A naive approach to get the local H score from $\mathbb{X}$ consists in comparing the value of $\sum_i^j X_k$ for all possible pairs $(i,j)_{i \leqslant j}$; such an approach is quadratic with the length of the sequence. **B -** Considering the processus $\mathbb{H} = (H_i)_{1,\dots,n}$ with $H_i = \max(0, H_{i-1} + X_i)$ and $H_0 = \max(0, X_0)$, the local score statistic is merely defined as $H = \max(H_i)$. Finding the maximal scoring subsequence comes down to find the maximal value $\mathbb{H}$ what is linear instead of quadratic.

Statistical results

Based on the extreme values theory and the work of Iglehart (1972) , Karlin and Dembo (1992) underlined that when X_i are independent, identically distributed and $\mathbb{E}(X_i) < 0$, the distribution of H can be well approximated for large n by the Gumbel distribution:

$$\lim_{n\to+\infty} \mathbb{P}\left(H \geqslant \frac{\log n}{\lambda} + x\right) = 1 - \exp\left(-Ke^{-\lambda x}\right) \tag{1}$$

The random distribution of the local score is thus expressed in terms of three parameters that are the length of the sequence n and two parameters of normalization, K and λ depending on the distribution of scores X_i. Given all parameters, the statistical theory can be more simply expressed by the use of the normalized score $H' = \lambda H - \log(nK)$:

$$\lim_{n\to+\infty} \mathbb{P}\left(H' \geqslant x\right) = 1 - \exp\left(-e^{-x}\right) \tag{2}$$

When searching for multiple genomic regions, $H^{(1)}, ..., H^{(k)}$ can also be written using normalized expressions $H'^{(1)}, ..., H'^{(k)}$ with $H'^{(i)} = \lambda H^{(i)} - \log(nK)$ for which Karlin and Altschul (1993) and Altschul (1997) provided some interesting results.

Considering values $h'_1 \geqslant h'_2 \geqslant ... \geqslant h'_k$, the asymptotic joint probability density function for $H'^{(1)} \geqslant ... \geqslant H'^{(k)}$ is well approximated by:

$$\lim_{n\to+\infty} f(h_1, ..., h_k) = \exp\left(-e^{-h_k} - \sum_{i=1}^{k} h_i\right) \tag{3}$$

From this, Karlin and Altschul (1993) deduced a generalization of Formula 1 for the k^{th} score:

$$\lim_{n\to+\infty} \mathbb{P}\left(H'^{(k)} \geqslant h'_k\right) = 1 - \exp\left(e^{-x}\right) \times \sum_{i=0}^{k-1} \frac{e^{-ix}}{i!} \tag{4}$$

The authors also provided the asymptotic probability density function for the sum of the k highest normalized scores $T'^{(k)} = H'^{(1)} + ... + H'^{(k)}$:

$$\lim_{n\to+\infty} f_k(t') = \frac{e^{-t'}}{k!(k-2)!} \int_0^{+\infty} y^{k-2} \exp\left(-e^{(y-t')/k}\right) dy \tag{5}$$

Algorithm

Knowledge about the local score is compiled in the following four-step algorithm (Figure 2) in order to propose a new approach for SNP-data analysis:

1 - First we assign an "individual score" X_i to each marker, a high score meaning a high chance of association to the disease. It can be based on common statistics like allelic, genotypic, Hardy-Weinberg or Armitage statistics; p-values p_i can as well be used but need a transformation such as $X_i = -\log_{10}(p_i)$. As previously underlined, a constraint of the strategy is to have negative expected individual scores; that does not happen with scores based on chi-square statistics or p-values for instance. In such cases, a constant δ must be subtracted from the whole signal $\mathbb{X}$ to get a corrected sequence $\mathbb{X}' = (X'_i)_{i=1,...,n}$, with $X'_i = X_i - \delta$ and $\mathbb{E}(X'_i) < 0$. SNPs with a score higher than δ will improve the cumulative score of a given segment whereas SNPs with a score below the threshold will penalize it.

2 - Then the aim is to identify the best high-scoring segments and to compute their respective scores $H^{(1)}, ..., H^{(k)}$. A simple strategy is to use an iterative algorithm: **(i)** find the highest-scoring segment (refer to Figure 1), **(ii)** remove it, **(iii)** apply the algorithm again. The process is stopped when the next best score is non-positive. In the worst case, all maximal scoring subsequences can be found in $O(n^2)$ time. Considering the length of the sequences we want to analyze, we will prefer instead of this naive approach, a more recent and faster one developed by Ruzzo and Tompa (1999) that finds all maximal scoring subsequences in a linear running time, and which has been proved to be 15 to 20 times faster on megabase sequences. From this step we get cumulative scores $H^{(1)}, ..., H^{(k)}$. Testing n markers has been reduced to the test of k sums.

3 - Given $H^{(1)}$, ...,$H^{(k)}$, the third step proposes a way to select a set of interesting segments. The successive local scores are combined into new sum statistics $T^{(1)}$, ..., $T^{(k)}$ with $T^{(i)} = H^{(1)} + ... + H^{(i)}$. Corresponding p-values $p_{T1}, ..., p_{Tk}$ can be computed using theoretical results and normalized scores (we obtain the tail probability that $T'^{(k)} \geqslant x$ integrating formula 5 for t' from x to infinity) or by Monte-Carlo simulations (permuting case and control labels). As proposed by Karlin and Altschul (1993), we select interesting segments as being the r first ones with $r = \text{argmin}_i(p_{Ti+1} > p_{Ti})$. In practice, it makes the hypothesis that segments are biologically interesting as long as they improve the overall result (p_{Ti} decreases) and segments that are actually noise do not improve it anymore. This gives us the number r of segments to select and an associated statistic $p^{(0)}_{\min} = p_{Tr}$ for this selection.

4 - The last step consists of assessing the global significance (p_G) of the

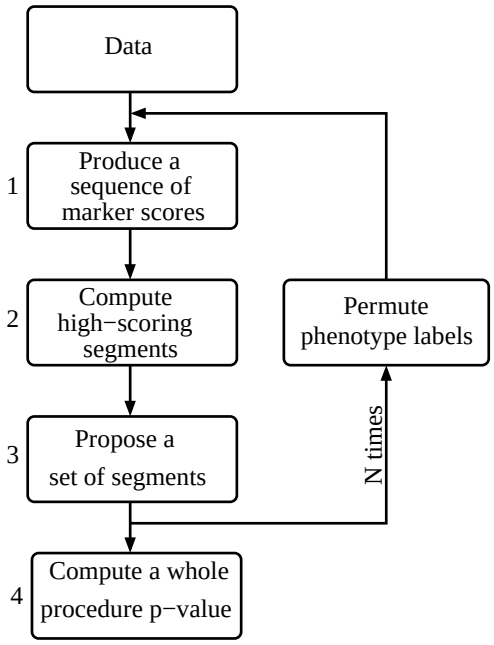


Figure 2. Algorithm

process via Monte-Carlo simulations. We iterate N times steps 1 to 3, permuting each time case and control labels, and computing $p_{\min}^{(i)}$ attached to the i^{th} iteration. This operation leads to the distribution of $p_{\min}$ under the assumption H_0 of total independence between genotypes and the disease. Finally, the p-value of the whole procedure is:

$$p_G = \frac{\text{card}\left\{i, p_{\min}^{(i)} \leqslant p_{\min}^{(0)}\right\}}{N}$$

Our method circumvents the multiple-testing problem by reducing the number of tests performed from n marker scores (X_i) to k local scores ($H^{(i)}$) and then from k to one statistic ($p_{\min}^{(0)}$).

Power study

This power study aims at showing the advantages of such a strategy compared to classical simple-marker with Bonferroni and FDR (Benjamini and Hochberg 1995) corrections. In addition we investigated the effect of the parameter δ

on results, as well as the efficiency of our selection when more than one DSL contributes to the phenotype.

Computing power

The power calculations were performed using Monte-Carlo simulations based on real data by applying a genetic model to the functional site(s). We used diplotype frequencies of a given population as an empirical distribution of the range of possible diplotypes. This way of simulating preserves the LD patterns expected in real data (Chapman et al. 2003, Nielsen et al. 2004). We then use the following genetic model to generate samples of affected and unaffected individuals, along with their genotypes at each marker of the dataset. Let a bi-allelic disease susceptibility locus with its two possible alleles (the allele of susceptibility A and a), p be the frequency of the allele A and r_0, r_1 and r_2 the frequencies of the genotypes aa, aA and AA respectively. According to Wright's model (1921), we have: $r_2 = p^2 + fp(1-p)$, $r_1 = 2p(1-p) - 2fp(1-p)$ and $r_0 = (1-p)^2 + fp(1-p)$ with f the coefficient of consanguinity. Now we introduce the prevalence of the disease (K_p), and the penetrances (f_i) associated to each genotype (i). Considering the relative risks (RR_i) such as $RR_i = \frac{f_i}{f_0}$ for $i = 1$ or 2, we define the four main modes of inheritance (MOI) underlying the mode of action of the DSL on the disease: recessive ($RR_1 = 1$), multiplicative ($RR_1 = \sqrt{RR_2}$), additive ($RR_1 = \frac{RR_2+1}{2}$) and dominant ($RR_1 = RR_2$). From these parameters, we can easily derive $f_0 = K_p/(r_0 + RR_1.r_1 + RR_2.r_2)$, $f_1 = RR_1.f_0$ and $f_1 = RR_1.f_0$ and finally genotype frequencies for the DSL in case and control populations:

$$\begin{aligned}(p_{d_0}, p_{d_1}, p_{d_2}) &= \left(\frac{f_0.r_0}{K_p}, \frac{f_1.r_1}{K_p}, \frac{f_2.r_2}{K_p}\right)\\ (p_{c_0}, p_{c_1}, p_{c_2}) &= \left(\frac{(1-f_0).r_0}{1-K_p}, \frac{(1-f_1).r_1}{1-K_p}, \frac{(1-f_2).r_2}{1-K_p}\right)\end{aligned}$$

Simulations

Simulations were based on 674 SNPs covering the chromosome 19 and DNA from 301 controls were genotyped using the 100K Affymetrix chip. DSL were chosen randomly as long as they satisfy polymorphim constraints (minor genotype frequency more than 5%) and random union of gametes was assumed ($f = 0$). We computed the genotypic Pearson statistic for each SNP and compared the local score strategy with common Bonferroni and FDR corrections. The sampling and testing procedure was repeated 200 times for an additive MOI, different values of RR_2 and δ, first with one (hidden or not) DSL and then with three.

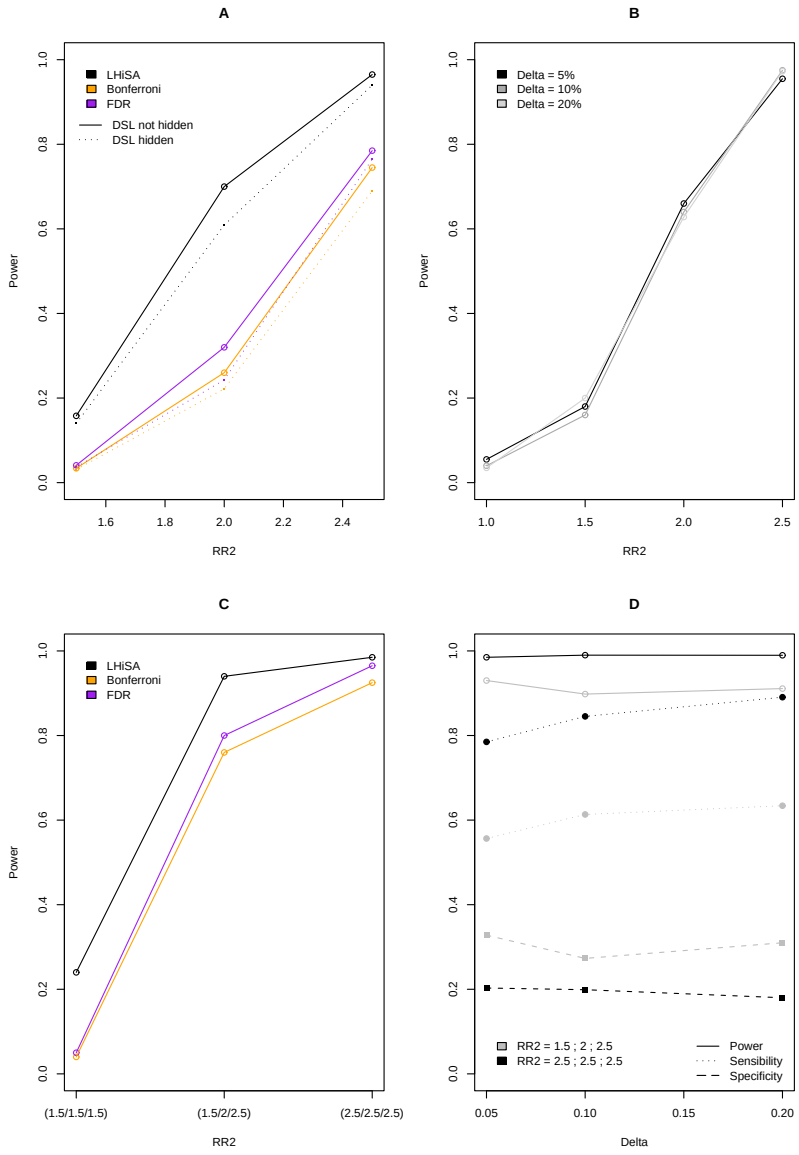


Figure 3. Power study: A - LHiSA, Bonferroni and FDR power estimates for three H_1 ($RR_2 = 1.5$, 2 and 2.5), DSL hidden or not. **B -** LHiSA power estimates for H_0 ($RR_2 = 1$) and three H_1 ($RR_2 = 1.5$, 2 and 2.5) according to δ. **C -** Comparison of power for three DSL H_1 (RR_2 =(1.5/1.5/1.5), (1.5/2/2.5) and (2.5/2.5/2.5)). **D -** LHiSA power, specificity and sensibility estimates for two H_1 ($RR_2 = (1.5/2/2.5)$ and $(2.5/2.5/2.5)$) according to δ.

Results

Results are summarized in Figure 3. In the two situations tested (DSL hidden or not, Figure 3-A), our strategy shows better performances than the classical Bonferroni and FDR ones with an average increase of 0.25. Simulating a hidden DSL logically leads to a loss of power. Note also that FDR is more powerful than Bonferroni which is consistent with the literature. On situations considered, the parameter δ does not seem to have any effect on the power (Figure 3-B). Moreover, under H_0 ($RR_2 = 1$), the type-I error-rate remains around 5% for any value of δ which is reassuring concerning the statistical assessment of results. When dealing with three DSL, the local score strategy also presents greater power (Figure 3-C). In this case, δ does not seem to have more effect on the probability to reject H_0 (Figure 3-D). However, the ability of the method to detect the true positives (sensitivity) increases and the count of true positives among the positives (specificity) tends to decrease along with δ.

Application

Data

We applied our strategy to genetic data implicating G72 and DAAO (D-amino acid oxydase) genes in schizophrenia (Chumakov et al. 2002). The dataset consists of 172 SNPs: 8 on chromosome 12 (surrounding DAAO) and 164 on chromosome 13 (including G72). DNA from 213 schizophrenic patients and 241 normal individuals were genotyped with this marker set. We computed the allelic chi-square statistics for each SNP (i) and associated p-values (p_i). Let $X_i = -\log_{10}(p_i)$ and $X'_i = X_i - \delta$ with $\delta = -\log_{10}(0.1)$ so that we consider SNPs with a p-value lower than 0.1 as potentially interesting and contributing positively to the score of a segment (Figure 4). The implication of G72 in schizophrenia has been reported many times in the literature (Detera-Wadleigh and McMahon, 2006)

Results

The local score strategy identifies three segments ($r = 3$), and $p^{(0)}_{\min} = 0.1660$ corresponding to the sum of the three first segments (Table 1). The global significance is $p_G = 0.22$. The two first segments are part of the G72 gene and the third one is part of the DAAO gene. Our method success in detecting the two genes expected. However, the significance is not convaincing enough to conclude to an association and confirms recent doubts about the implication of G72 and DAAO in schizophrenia (Riley and Kendler 2006).

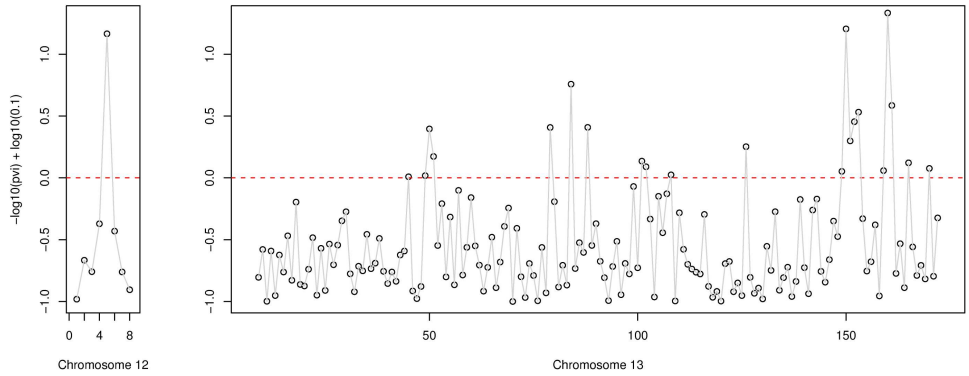


Figure 4. Schizophrenia - data: for each SNP i, an individual score $X_i = -log_{10}(p_i) + log_{10}(0.1)$ is assigned. SNPs labelled 1 to 8 are located on chromosome 12 and SNPs labelled 9 to 172 on chromosome 13.

rank	chr	segment	H	T	p_T
1	13	149-153	2.542	2.542	0.2459
2	13	159-161	1.978	4.520	0.1737
3	12	5	1.165	5.686	0.1660
4	13	84	0.758	6.444	0.1702
5	13	49-51	0.587	7.031	0.1747

Table 1
Schizophrenia - results of the local score strategy: $N = 10000$, $\delta = -\log_{10}(0.1)$, $p_{min}^{(0)} = 0.1660$ *and* $p_G = 0.22$.

Discussion

Based on $H^{(i)}$ and $T^{(i)}$, the sums of contiguous marker scores and the combination of distant segments, our strategy proposes to test the null hypothesis of no association against the existence of at least one region involved in the disease. The size of the regions is not predefined. This is an important advantage over sliding-frame strategies (Hoh and Ott 2000, Marques-Bonet et al. 2005) for which the choice of the frame size is arbitrary, often difficult, and may depend on multiple biological *a priori* (pattern of LD and density of markers for instance). Additionally, there is absolutely no biological reason to constrain the successive high-scoring segments to the same size. Our method handles the multiple-testing problem by reducing the number of tests from n markers to one statistic $p^{(0)}_{\min}$. On our power study it outperforms classical Bonferroni and FDR strategies. On schizophrenia data, it detects expected genes but with a significance that does not allow us to really conclude to an association with the disease.

The method requires the specification of one control parameter δ. This parameter represents the level upon which one marker is considered to be putatively interesting and hence can be intuitively set. One can choose classical 1% or 5% levels for the considerate marker score (X_i). However, since effects we want to detect are generally weak, we decide to increase it to 10%. Results predictively change along with this parameter: lowering δ will logically increase the size of the segments and the number of segments detected whereas augmenting it will lead to the contrary. Moreover, to correctly determinate the significance of the process using the Gumbel approximation, it is a necessary assumption for marker scores X_i to be weakly dependent. Because of the LD that may exist between successive loci (particulary for dense maps), this assumption is not ensured, so we recommend for this step to use Monte-Carlo simulations instead of the extreme value theory. It also has the advantage of avoiding asymptotic approximations and hence works on small sequences. The software has been implemented so that the use of Monte-Carlo simulations does not dramatically extend the algorithm computation time. The theoretical time complexity is in $O\left(N\log(N) + Nn\log(n)\right)$ and the memory complexity is in $O\left(n + N\log(n)\right)$. On a quite basic machine (Intel Pentium 4 CPU 2.80GHz, 512Mo RAM) it takes approximatively 10, 180 and 800 seconds to process respectively 200, 2000 and 10000 SNPs with $N = 2000$ simulations. The use of a more powerful machine will obviously improve this computation time, and considering the large number of independent Monte-Carlo simulations performed, our algorithm allows for parallelisation if applied on a cluster of machines. Finally, our method handles data from SNP genotyping but can

easily be adapted to other frameworks and types of markers.

Developed to manage large data sets in a quite reasonable time and to propose a selection of segments surrounding putative etiological sites, we believe that our approach is well adapted to genome-wide association studies. Having found significant associations with given regions, this selection can then be subject to further attention (haplotype reconstruction, fine-mapping, annotation, interaction patterns...) in order to formulate assumptions on the way they may contribute to the disease.

References

Altschul, S. (1997). *Theorical and compuational methods in genome research*, chapter Evaluating the statistical significance of multiple distinct local alignments, pages 1–14. New York: Plenum Press.

Altschul, S. and Erickson, B. (1986a). Locally optimal subalignments similarity and its significance levels. *Bulletin of Mathematical Biology* **48**, 633–660.

Altschul, S. and Erickson, B. (1986b). A nonlinear measure of subalignments similarity and its significance levels. *Bulletin of Mathematical Biology* **48**, 617–632.

Altschul, S., Gish, W., Miller, W., Myers, E. and Lipman, D. (1990). Basic local alignement search tool. *Journal of Molecular Biology* **215**, 403–410.

Bastone, L., Reilly, M., Rader, D. and Foulkes, A. (2004). Mdr and prp: a comparison of methods for high-order genotype-phenotype associations. *Human Heredity* **58**, 82–92.

Benjamini, Y. and Hochberg, Y. (1995). Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society B* **57**, 289–300.

Brendel, V., Bucher, P., Nourbakhsh, I., Blaisdell, B. and Karlin, S. (1992). Methods and algorithms for statistical analysis of protein sequences. *Proceedings of the National Academy of Science USA* **89**, 2002–2006.

Bureau, A., Dupuis, J., Falls, K., Lunetta, K., Hayward, B., Keith, T. and Van Eerdewegh, P. (2005). Identifying snps predictive of phenotype using random forests. *Genetic Epidemiology* **28**, 171–182.

Carlson, C., Eberle, M., Kruglyak, L. and Nickerson, D. (2004). Mapping complex disease loci in whole-genome association studies. *Nature* **429**, 446–452.

Chapman, J. M., Cooper, J. D., Todd, J. A. and Clayton, D. G. (2003). Detecting disease associations due to linkage disequilibrium using haplotype tags: a class of tests and the determinants of statistical power. *Human Heredity* **56**, 18–31.

Chumakov, I., Macciardi, F., Sham, P., Straub, R., Weinberger, D., Cohen, N. and Cohen, D. (2002). Genetic and physiological data implicating the new human gene G72 and the gene for D-amino acid oxidase in schizophrenia. *Proceedings of the National Academy of Science USA* **99**, 13675–13680.

Clayton, D., Champan, J. and Cooper, J. (2004). Use of unphased multilocus genotype data in indirect association studies. *Genetic Epidemiology* **27**, 415–428.

Collins, F., Brooks, L. and Chakravarti, A. (1998). A dna polymorphism discovery resource for research on human genetic variation. *Genome Research* **8**, 1229–1231.

Dembo, A. and Karlin, S. (1991). Strong limit theorems of empirical functionals for large exceedances of partial sums of iid variables. *Annals of Probability* **19**, 1737–1755.

Detera-Wadleigh, S. D. and McMahon, F. J. (2006). G72/g30 in schizophrenia and bipolar disorder: review and meta-analysis. *Biological Psychiatry [in press]* .

Gordon, D., Heath, S., Liu, X. and Ott, J. (2001). A transmission/disequilibrium test that allows for genotyping errors in the analysis of single nucleotide polymorphism data. *American Journal Human Genetics* **69**, 371–380.

Hao, K., Xu, X., Laird, N., Wang, X. and Xu, X. (2004). Power estimation of multiple snp association test of case-control study and application. *Genetic Epidemiology* **26**, 22–30.

Hirschhorn, J. N. and Daly, M. J. (2005). Genome-wide association studies for common diseases and complex traits. *Nature Reviews Genetics* **6**, 95–108.

Hoh, J. and Ott, J. (2000). Scan statistics to scan markers for susceptibility genes. *Proceedings of the National Academy of Science USA* **97**, 9615–9617.

Hoh, J. and Ott, J. (2003). Mathematical multi-locus approaches to localizing complex human trait genes. *Nature Reviews Genetics* **4**, 701–709.

Hoh, J., Wille, A. and Ott, J. (2001). Trimming, weighting, and grouping snps in human case-control association studies. *Genome Research* **11**, 2115–2119.

Iglehart, D. (1972). Extremes values in the gi/g/1 queues. *Annals of Mathematical Statistics* **43**, 627–635.

Karlin, S. (2005). Statistical signals in bioinformatics. *Proceedings of the National Academy of Science USA* **102**, 13355–13362.

Karlin, S. and Altschul, S. (1990). Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes. *Proceedings of the National Academy of Science USA* **87**, 2264–2268.

Karlin, S. and Altschul, S. (1993). Applications and statistics for multiple high-scoring segments in molecular sequences. *Proceedings of the National Academy of Science USA* **90**, 5873–5877.

Karlin, S. and Brendel, V. (1992). Chance and significance in protein and dna sequence analysis. *Science* **257**, 39–49.

Karlin, S., Bucher, P., Brendel, V. and Altschul, S. (1991). Statistical-methods and insights for protein and dna-sequences. *Annual Review of Biophysics and Biophysical Chemistry* **20**, 175–203.

Karlin, S. and Dembo, A. (1992). Limit distributions of maximal segmental score among markov-dependant partial sums. *Advances in Applied Probability* **24**, 113–140.

Longmate, J. (2001). Complexity and power in case-control association studies. *American Journal Human Genetics* **68**, 1229–1237.

Marques-Bonet, T., Lao, O., Goertsches, Robert ad Comabella, M., Montalban, X. and Navarro, A. (2005). Association cluster detector: a tool for heuristic detection of significance cluters in whole-genome scans. *Bioinformatics* **21**, ii180–ii181.

McGinnis, R., Shifman, S. and Darvasi, A. (2002). Power and efficiency of the tdt and case-control design for association scans. *Behavior Genetics* **32**, 135–144.

Mercier, S. and Daudin, J. (2001). Exact distribution for the local score of one i.i.d random sequence. *Journal of Computational Biology* **8**, 373–380.

Morris, A., Whittaker, J. and Balding, D. (2003). Multipoint linkage-disequilibrium mapping narrows location interval and identifies mutation heterogeneity. *Proceedings of the National Academy of Science USA* **11**, 13442–13446.

Nelson, M., Kardia, S., Ferrell, R. and Sing, C. (2001). A combinatorial partitioning method to identify multilocus genotypic partitions that predict quantitative trait variation. *Genome Research* **11**, 458–470.

Nielsen, D. M., Ehm, M. G., Zaykin, D. V. and Weir, B. S. (2004). Effect of two- and three-locus linkage disequilibrium on the power to detect marker/phenotype associations. *Genetics* **168**, 1029–1040.

North, B., Curtis, D., Cassell, P., Hitman, G. and Sham, P. (2003). Assessing optimal neural network architecture for identifying disease-associated multi-marker genotypes using a permutation test, and application to calpain 10 polymorphisms associated with diabetes. *Annals Human Genetics* **67**, 348–356.

Ott, J. (2004). Issues in association analysis: error control in case-control association studies for disease gene discovery. *Human Heredity* **58**, 171–

174.

Page, G., George, V., Go, R., Page, P. and Allison, D. (2003). "are we there yet?": deciding when one has demonstrated specific genetic causation in complex diseases and quantitative traits. *American Journal Human Genetics* **73**, 711–719.

Riley, B. and Kendler, K. (2006). Molecular genetic studies of schizophrenia. *European Journal of Human Genetics* **14**, 669–680.

Risch, N. (2000). Searching for genes in complex diseases: lessons from systemic lupus erythematosus. *American Society for Clinical Investigation* **105**, 1503–1506.

Risch, N. and Merikangas, K. (1996). The future of genetic studies of complex human diseases. *Science* **273**, 1516–1517.

Ruzzo, W. and Tompa, M. (1999). A linear time algorithm for finding all maximal scoring subsequences. In *7th International Conference of Intelligent Systems for Molecular Biology*, pages 234–241.

Schaid, D. (2004). Evaluating association of haplotypes with traits. *Genetic Epidemiology* **27**, 348–364.

Shendure, J., Mitra, R., Varma, C. and Church, G. (2004). Advanced sequencing technologies: methods and goals. *Nature Review Genetics* **5**, 335–344.

Spielman, R. and Ewens, W. (1996). The tdt and other family-based tests for linkage disequilibrium and association. *American Journal of Human Genetics* **59**, 983–989.

Tregouet, D., Escolano, S., Tiret, L., Mallet, A. and Golmard, J. (2004). A new algorithm for haplotype-based association analysis: the stochastic-em algorithm. *Annals of Human Genetics* **68**, 165–177.

Tzeng, J.-Y., Devlin, B., Wasserman, L. and Roeder, K. (2003). On the identification of disease mutations by the analysis of haplotype similarity and goodness of fit. *American Journal of Human Genetics* **72**, 891–902.

Wille, A., Hoh, J. and Ott, J. (2003). Sum statistics for the joint detection of multiple disease loci in case-control association studies with snp markers. *Genetic Epidemiology* **25**, 350–359.

Wright, S. (1921). Systems of mating. *Genetics* **6**, 111–178.

Yoon, Y., Song, J., Hong, S. and Kim, J. (2003). Analysis of multiple single nucleotide polymorphisms of candidate genes related to coronary heart disease susceptibility by using support vector machines. *Clinical Chemestry and Laboratory Medecine* **41**, 529–534.

Chapitre 16

Computing power in case-control association studies and application to meta-statistics

Article de Guedj, Della-Chiesa, Picard, et Nuel accepté en 2006 par *Annals of Human Genetic*. Cet article traite du calcul de la puissance de tests dans les études cas-témoins. Plusieurs méthodes classiques sont passées en revue et une approche nouvelle (delta-méthode d'ordre 2) est proposée. Le cas particulier des tests utilisant des méta-statistiques (ex : sommes ou produits de statistiques classiques) est discuté en détails. L'efficacité des différentes méthodes est évaluée par simulations.

Computing power in case-control association studies through use of quadratic approximations : application to meta-statistics.

M. Guedj,[1,2] **E. Della-Chiesa,**[1] **F. Picard**[1] **and G. Nuel**[1]

[1]Laboratoire Statistique et Genome, FRANCE

[2]Serono, FRANCE

9 octobre 2006

Corresponding author : Mickaël GUEDJ

Laboratoire Statistique et Genome,

523 place des terrasses de l'Agora, 91000 EVRY

Tel : +33-1-6087-3800

Fax : +33-1-6087-3809

e-mail : guedj@genopole.cnrs.fr

SUMMARY

In the framework of case-control studies, many different test statistics are available to measure the association of a marker with a given disease. Nevertheless, choosing one particular statistic can lead to very different conclusions. In the absence of a consensus for this choice, a tempting option is to evaluate the power of these different statistics *prior* to make any decision. We propose to make a review of available methods dedicated to power computation and to assess their respective reliability to treat a wide range of tests on a wide range of alternative models.

Considering Monte-Carlo, non-central chi-square and Delta-Method estimates, we compete empirical, asymptotic and numerical approaches. Additionally, we introduce the use of the Delta-Method extended to the order 2 intending to provide better results than the traditional order-1 Delta-Method. Supplementary data can be found at :

`http://stat.genopole.cnrs.fr/software/dm2`.

KEY WORDS: Power, association tests, Delta-Method

1. Introduction

Case-control association studies are considered as the simplest framework to help to elucidate the genetic basis of complex diseases (Risch 2000). Even if they present some weaknesses to potential confounding factors such as population stratification, they remain an important tool in genetical epidemiology, often preferred to family-based studies (Zhao 2000) due to the availability of data. Such an approach involves unrelated individuals split into cases who are diagnosed with the disease of interest and unaffected controls. It merely relies on the assumption that disease related genetic determinants should accumulate among cases.

Tests of association are used as a first step in the analysis process. Various tests are proposed based on either genotypes (Table 1) such as the genotypic, Hardy-Weinberg equilibrium or Cochran-Armitage tests, or alleles such as the allelic test.

[TAB. 1 about here.]

Since different single-statistics can be used to test for association, another strategy is to combine them via meta-statistics with the hope of gaining power.

If they aim to establish an association between markers and disease, each test has a slightly different null hypothesis (H_0) and hence a different efficiency with

respect to the underlying hypothesis. One way to compete them is to assess their power (π) defined as the ability of a test to reject the null hypothesis when the alternative hypothesis (H_1) is true. Power studies require the distribution of statistics under H_1.

This article reviewes and discusses the most used mathematical frameworks to approximate the H_1 distribution (and hence to compute π) in the context of genetic association studies. Our study includes two popular approaches. The first is empirical and based on Monte-Carlo simulations under the alternative hypothesis. The other is based on the asymptotic non-central chi-square distribution of the statistics under H_1. We compare these two approaches with the Delta-method with emphasis on its extension to the order 2. As expected, non-central chi-square approximations appear to be very reliable (whenever available) while the order-1 Delta-Method is not. To treat non-explicit cases (combination of statistics for instance) for which computationally expensive Monte-Carlo simulations are usually considered, we show that the order-2 Delta-Method approximations is sufficiently efficient to represent a valid and cheaper alternative.

2. Method

2.1 *Testing for association*

Let us denote by x a case-control sample that is a realisation of the random variable X, and which can be represented by a genotypic contingency table (Table 1). To establish an association we consider a null hypothesis (H_0) used to test a particular distribution of the observations. To do so, we consider a statistic defined as a function of these observations : $\mathcal{S} = f(X)$, and carefully chosen such that $\mathcal{S}$ grows when H_0 is less likely. Using the distribution of $\mathcal{S}$ under H_0, we can find a threshold (t_α) such as $\alpha = \mathbb{P}_{H_0}(\mathcal{S} \geqslant t_\alpha)$, where level α can be set to 5% for

example.

2.2 *Computing power*

Using the distribution of $\mathcal{S}$ under an alternative association hypothesis H_1, we can compute the power $\pi(\alpha)$ of the test such that : $\pi(\alpha) = \mathbb{P}_{H_1}(\mathcal{S} \geqslant t_\alpha)$. To calculate $\pi(\alpha)$, the first step is to define a genetic model as well as the null and alternative hypothesis.

2.2.1 Genetic Model

Consider a bi-allelic disease susceptibility locus (DSL) with A being the allele of susceptibility and a the other, p the frequency of the allele A and r_0, r_1 and r_2 the genotype frequencies in the general population. Assuming that the Hardy-Weinberg equilibrium (HWE) holds in the population, genotypic frequencies reduce to $r_2 = p^2$, $r_1 = 2p(1-p)$ and $r_0 = (1-p)^2$. Now we introduce the prevalence of the disease (K_p), and penetrances (f_i) associated to each genotype (i). Considering the relative risks (RR_i) such that $RR_i = \frac{f_i}{f_0}$ for $i = 1$ or 2, we define the four main mode of inheritances (MOI) corresponding to the modes of action of the DSL on the disease : recessive ($RR_1 = 1$), multiplicative ($RR_1 = \sqrt{RR_2}$), additive ($RR_1 = \frac{RR_2+1}{2}$) and dominant($RR_1 = RR_2$).

Considering these parameters, we can easily derive $f_0 = K_p/(r_0 + RR_1.r_1 + RR_2.r_2)$ and $f_i = RR_i.f_0$ for $i = 1$ or 2. With the further assumption of infinite population, the genotype distributions (D_0, D_1, D_2) in cases and (C_0, C_1, C_2) in controls are multinomial with parameters :

$$\begin{aligned}(D_0, D_1, D_2) &\sim \mathcal{M}\left(n_D; \frac{f_0 r_0}{K_p}, \frac{f_1 r_1}{K_p}, \frac{f_2 r_2}{K_p}\right), \\ (C_0, C_1, C_2) &\sim \mathcal{M}\left(n_C; \frac{(1-f_0) r_0}{1-K_p}, \frac{(1-f_1) r_1}{1-K_p}, \frac{(1-f_2) r_2}{1-K_p}\right).\end{aligned}$$

In such a context, $H_0 : \{RR_2 = 1\}$ and $H_1 : \{RR2 \neq 1\}$. Once the alter-

native hypothesis is explicit, power can be calculated using one of the following frameworks to approximate the distribution of $\mathcal{S}$ under H_1.

2.2.2 Monte-Carlo estimation As long as it is possible to generate a case-control sample $X = \begin{cases} D_0 & D_1 & D_2 \\ C_0 & C_1 & C_2 \end{cases}$ under H_1, it is very easy to get an estimation of the power. We first draw N samples denoting $x^{(i)}$ the i^{th} sample. From this sample we get N statistics $s^{(1)}, \ldots, s^{(N)}$ from which we get the estimation of the power :

$$\widehat{\pi}(\alpha) = \frac{\sharp\left\{s^{(i)} \geqslant t_\alpha\right\}}{N}.$$

This well-known method is often very easy to perform and is consequently widely used, particularly in the field of statistical genetics when alternative distributions are hard to calculate analytically (Longmate 2001). But such an approach may generally require a lot of time to reach a given level of precision. As $\widehat{\pi}$ is distributed according to a binomial distribution, using the central limit theorem we get that $\widehat{\pi} \sim \mathcal{N}(\pi, \pi(1-\pi)/N)$ which gives the following 95% confidence interval :

$$\left[\widehat{\pi} - 1.96\frac{\sqrt{\widehat{\pi}(1-\widehat{\pi})}}{\sqrt{N}}; \widehat{\pi} + 1.96\frac{\sqrt{\widehat{\pi}(1-\widehat{\pi})}}{\sqrt{N}}\right].$$

Consequently the precision of the power estimate increase with speed $1/\sqrt{N}$. One could remark that the same method can be used to estimate the threshold of tests involving statistics for which the distribution under H_0 is not easily available (*e.g.* meta-statistics).

2.2.3 Asymptotic non-centrality parameter Mitra (1958) demonstrated that under H_1, the asymptotic distribution of a chi-square frequency test applied to a $2 \times c$ contingency table follows a non-central chi-square distribution $\chi'^2(k, \lambda)$

where k is the degree of freedom and λ the non-centrality parameter such that

$$\lambda = N_1 N_2 \times \sum_{j=1}^{c} \frac{(p_{1j} - p_{2j})^2}{N_1 p_{1j} + N_2 p_{2j}},$$

with p_{ij} the frequency of case ij and N_1, N_2 the total counts of the first and second row. The author derived the asymptotic power for the test :

$$\pi(\alpha) \underset{\infty}{\rightarrow} 1 - \chi'^2_{1-t_\alpha}(k, \lambda).$$

Given the expression of the non-centrality parameter, this approach can be adapted to any statistic following a chi-square distribution under H_0 (see below for the particluar case of trend test) and is appropriate when sample sizes are large enough. It has been recently presented as an appealing and fast way to approximate power in association studies (Sham et al 2000, Gordon et al 2002, Kang et al 2004).

2.2.4 Delta-Method The Delta-Method is used to approximate the distribution of $\mathcal{S}$ with X. The multinomial distribution of X (derived from the genetic model) is asymptotically distributed according to a Gaussian distribution $\mathcal{N}(M, \Sigma)$. Using a order-1 Taylor development of $\mathcal{S} = f(X)$ around M we hence approximate $\mathcal{S}$ by :

$$\mathcal{S} \simeq f(M) + {}^t(X - M) \times \nabla f(M),$$

where ^{t} is the transpose operator and ∇f is the gradient of f. This 1-order development allows us to approximate the distribution of $\mathcal{S}$ by a Gaussian distribution $\mathcal{N}(m, \sigma^2)$ with $m = f(M)$ and $\sigma^2 = {}^t\nabla f(M) \times \Sigma \times \nabla f(M)$. Then we have :

$$\pi(\alpha) \underset{\infty}{\rightarrow} 1 - \Phi\left(\frac{t_\alpha - m}{\sigma}\right)$$

where Φ is the cumulative distribution function (CDF) of a Gaussian variable with zero mean and variance of one. Of course, the closer the distribution under H_1 will be to a Gaussian distribution, the better will be this 1-order approximation.

For cases where the Gaussian distribution of the statistic under H_1 is not realistic, we propose to use a order-2 Taylor expansion around M. We hence get a more precise approximation based on the distribution of a quadratic form in normal variables (QFNV) :

$$\mathcal{S} \simeq f(M) + {}^t(X-M) \times \nabla f(M) + \frac{1}{2}{}^t(X-M) \times \nabla^2 f(M) \times (X-M),$$

where $\nabla^2 f$ is the Hessian of f.

In the case of the first order development, the computation of power only requires to evaluate the CDF of a normal distribution. With the second order development however, the distribution of $\mathcal{S}$ is approximated by a combination of chi-squares and the CDF is not straighforward to derive. Technical details can be found in Appendix 1 and derivations of the distribution for the statistics considered are available at : `http://stat.genopole.cnrs.fr/software/dm2`.

3. Application

3.1 *Statistics considered*

Here we consider four statistics. **(i)** The *genotypic test* compares genotypic frequencies between affected and unaffected subjects by using the Pearson's chi-square statistic :

$$\mathcal{S}_{\mathrm{G}} = \sum_{i=0}^{2} \frac{\left(D_i - \frac{n_D \times n_i}{n}\right)^2}{\frac{n_D \times n_i}{n}} + \frac{\left(C_i - \frac{n_C \times n_i}{n}\right)^2}{\frac{n_C \times n_i}{n}} \underset{H_0}{\sim} \chi^2(2),$$

From the formula given in (2.2.3) and the parameters of the genetic model (2.2.1), we can derive the non-centrality parameter for this statistic :

$$\lambda_G = n_D n_C \times \sum_{i=0}^{2} \frac{\left(\frac{f_i r_i}{Kp} - \frac{(1-f_i)r_i}{1-Kp}\right)^2}{n_D \frac{f_i r_i}{Kp} + n_C \frac{(1-f_i)r_i}{1-Kp}} \text{ and } \mathcal{S}_G \underset{H_1}{\sim} \chi^2(2, \lambda_G)$$

(ii) Another test based on genotypes is the *Cochran-Armitage test for trends* (Armitage 1955). It measures a linear trend in proportions weighted by a dose effect score x_i associated to each column with x_i corresponding to the number of susceptibility allele :

$$\mathcal{S}_{\mathrm{T}} = \frac{n.\left[n.(D_1 + 2D_2) - n_D.(n_1 + 2n_2)\right]^2}{n_D n_C.\left[n.(n_1 + 4n_2) - (n_1 + 2n_2)^2\right]} \underset{H_0}{\sim} \chi^2(1).$$

For this particular case (trend test with three categories) Gordon et al (2005) derived the expression of the non-centrality parameter, based on previous work of Chapman and Nam (1968). With our notation, it comes down to :

$$\lambda_T = n_D n_C \times \frac{\left[\sum x_i \left(\frac{(1-f_i)r_i}{1-Kp} - \frac{f_i r_i}{Kp}\right)\right]^2}{\sum x_i^2 \left(n_D \frac{f_i r_i}{Kp} + n_C \frac{(1-f_i)ri}{Kp}\right) - \frac{\left[\sum x_i \left(n_D \frac{f_i r_i}{Kp} + n_C \frac{(1-f_i)ri}{Kp}\right)\right]^2}{n}} \text{ and } \mathcal{S}_T \underset{H_1}{\sim} \chi^2(1, \lambda_T)$$

Note that chi-square approximations are appropriate when sample sizes are in accordance with the Cochran's condition (each expected cell count > 5 - Cochran 1952).

(iii) Another strategy is to combine simple statistics via meta-statistics with the hope of gaining power. Nevertheless, the actual null hypothesis tested is not explicit and distributions of such statistics (under H_0 or H_1) are not easy to assess out of Monte-Carlo simulations. Our aim considering $\mathcal{S}_\Sigma = \mathcal{S}_G + \mathcal{S}_T$ and $\mathcal{S}_\Pi = \mathcal{S}_G \times \mathcal{S}_T$ is to assess the efficiency of competing approaches to handle their power computation.

3.2 *Simulations*

Simulations are performed using the susceptibility allele frequency (p) as a factor of variation. All simulations are considered for a prevalence $K_p = 0.05$, $n_D = n_C = 500$ and the four MOIs ($RR_2 = 1.5$). Each Monte-Carlo estimate of power is done on the basis of $N = 10,000$ simulations and is considered as a reference to compare with the other approaches. Using this approach to compute a power $\widehat{\pi}$, we get a 95% confidence interval of radius $0.0196\sqrt{\widehat{\pi}(1-\widehat{\pi})}$ centered on $\widehat{\pi}$. For example, this radius gives 0.588% for $\widehat{\pi} = 10\%$ (or 90%), 0.784% for $\widehat{\pi} = 20\%$ (or 80%) and is always smaller than 0.98% (case $\widehat{\pi} = 50\%$).

3.3 *Results*

Results concerning $\mathcal{S}_G$ and $\mathcal{S}_T$ are compiled Figure 1. For the set of parameters considered, the additive and multiplicative models give very close results so we display them only for the additive, recessive and dominant models.

[FIG. 1 about here.]

The non-central chi-square approach (NC) is fully adapted to chi-square distributed statistics and hence gives accurate estimates of power. As non-central chi-square distributions are particular cases of QFNV, the order-2 Delta-Method (DM2) unsurprisingly also presents good results. By comparison, the order-1 Delta-Method (DM1) underestimates the power in the two cases. This underscores that the Gaussian approximation under H_1 made by this approach is not realistic. However it provides better estimates for the trend test than for the genotypic test. This variation is due to the fact that the distribution of $\mathcal{S}_T$ under H_0 and H_1 is closer to a Gaussian distribution - that requires this approach - than $\mathcal{S}_G$. For instance, the expected value for the Wilk-Shapiro statistic test for normality is 0.69

for one degree-of-freedom chi-square distributed samples and 0.81 for two degree-of-freedom chi-square distributed ones.

In the literature, it has been suggested that factors such as the ratio of cases to controls, minor allele frequency anf total sample size affect the accuracy of the analytic power calculations (Ji et al 2005). We investigated such effects considering the genotypic and trend tests for the four MOIs and values of (0.04, 0.2, 1), (0.2,0.3,0.4,0.5) and (40,200,1000) for the ratio, the minor allele frequency and the total sample size respectively. Normalized absolute differences have been computed for the NC and DM2 calculations (data not shown). However, we do not observe any clear effect of these factors on the accuracy of the analytic calculations.

Figure 2 presents the results for the two meta-statistics ($\mathcal{S}_\Sigma$ and $\mathcal{S}_\Pi$).

[FIG. 2 about here.]

In these cases, the non-central chi-square approach is not applicable. Even if $\mathcal{S}_\Sigma$ is the sum of two chi-square distributed statistics, $\mathcal{S}_G$ and $\mathcal{S}_T$ are not independent and hence $\mathcal{S}_\Sigma$ is not merely distributed according to a three degree-of-freedom chi-square distribution under H_0. DM1 still badly estimates power. DM2 is really efficient to treat $\mathcal{S}_\Sigma$. As previously underlined, a linear combination of (dependent or not) chi-square distributed statistics is a QFNV which explains that DM2 works well on $\mathcal{S}_\Sigma$. Nevertheless DM2 does not manage to assess the power of $\mathcal{S}_\Pi$. We can imagine that such a product of chi-square distributed statistics would have required the use of the Delta-Method to a higher order, for which the determination of the CDF would have been numerically very expensive and unrealistic in practice.

In term of comparison between the four strategies considered $\mathcal{S}_T$, $\mathcal{S}_G$, $\mathcal{S}_\Sigma$ and $\mathcal{S}_\Pi$ (Figure 3), meta-statistics power estimates mainly lie between trend and genotypic ones and hence do not clearly represent a better alternative to single-statistics. However, they do more than merely averaging power estimates of single statistics and hence can appear as a clever alternative to combine efficiency according to the model.

[FIG. 3 about here.]

4. Discussion

Studying power is an important tool in statistics to compare the efficiency of different tests or to help to design a study. With the accumulation of new analysis methods that have recently arisen from the accumulation of large scale data, statistical genetics does not escape this rule. In this article, we focus on the computation of power in the context of simple-marker analyzes *via* the genotypic and trend statistics as well as simple combinations of them.

Easy to implement, Monte-Carlo simulations are often the preferred approach to compute power estimations. Nevertheless it is computationally expensive since the precision of the estimates is directly dependent to the number of simulations performed. In particular the length of the confidence interval decrease with $1/\sqrt{N}$ and hence evolves quite slowly with N. Computing power through the non-centrality parameter is logically well adapted for statistics distributed according to a chi-square distribution under H_0. The order-1 Delta-Method is based on a Gaussian distribution of the statistic. As a result, it is not efficient in the situation considered here. In the literature, approaches based on the order-1 Delta-Method have been successfully developed (Slager and Schaid 2001, Jackson et al 2002) to compute accurate power approximations for allelic and trend tests (Slager and

Schaid 2001, Jackson et al 2002). Shortly, the required Gaussian distributions of the statistics was obtained by the authors considering $\mathcal{Z} \sim \mathcal{N}(0,1)$ such that $(\mathcal{Z})^2 = \mathcal{S}$ instead of $\mathcal{S}$ directly (as we have done here). This approach provides very good power approximations. However, its application is restrained to z-scores or by extension to 1 degree-of-fredom chi-square distributed statistics. It is hence less general than the non-central chi-square approach. To go further, we introduce the use of the order-2 Delta-Method. This approach provides good estimates and can be used to treat simple statistics and linear combinations of them which is an advantage over other approaches. Besides a less straitghforward CDF evaluation, it represents an alternative much less computionally expensive than Monte-Carlo simulations, more general than the non-central chi-square framework and more accurate than the order-1 Delta-Method.

This work has been restricted to the study of the trend and genotypic tests under alternatives that differ in the susceptibility allele frequency and the MOI only. However, our conclusions can easily be extended to other simple-marker tests (Hardy-Weinberg and allelic tests for instance) based on more complicated meta-statistics and applied on more elaborate alternative models taking, for instance, the coefficient of consanguinity, the linkage disequilibrium and genotyping errors into account.

Even if the fail to provide greater power estimates than single statistics, meta-statistics do not suffer from substantial power loss when compared with the best single statistic for each of the situations considered in this work. We thus suggest that meta-statistics may provide a useful means for combining such tests.

If they failed to provide better results than single-statistics, meta-statistics do not present a sensible loss of power comparing to the best simple statistic in each

situation considered and hence appear as a possible clever way to combine such tests.

ACKNOWLEDGEMENTS

The authors would like to thank Bernard Prum's team as well as members of the Serono Pharmaceutical Institute, in particular Jérôme Wojcik and Hiroaki Tanaka for encouraging this work.

REFERENCES

Armitage, P. (1955). Tests for linear trends in proportions and frequencies. *Biometrics* **11**, 375–386.

Chapman, D. and Nam, J. (1968). Asymptotic power of chi-square tests for linear trends in proportions. *Biometrics* **24**, 315–327.

Cochran, W. (1952). The chi-square test of goodness of fit. *Annals of Human Genetics* **23**, 315–345.

Davies, R. B. (1973). Numerical inversion of characteristic function. *Biometrika* **60**, 481–482.

Davies, R. B. (1980). The distribution of a linear combination of chi-square random variables. *Appl. Stat.* **29**, 323–333.

Gordon, D., Finch, S. J., Nothnagel, M. and Ott, J. (2002). Power and sample size calculations for case-control genetic association tests when errors are present : application to single nucleotid polymorphisms. *Human Heredity* **54**, 22–33.

Gordon, D., Haynes, C., Blumenfeld, J. and Finch, S. (2005). Pawe-3d : visualizing power for association with error in case-control genetic studies of complex traits. *Bioinformatics* **21**, 3935–3937.

Jackson, M., Genin, E., Knapp, M. and J.L., E. (2002). Accurate power approximation for χ-tests in case-control association studies of complex disease genes. *Annals of Human Genetics* **66**, 307–321.

Ji, F., Yang, Y., Haynes, C., Finch, S. and Gordon, D. (2005). Computing asymptotic power and sample size for case-control genetic association studies in the presence of phenotype and/or genotype misclassification errors. *Statistics Applied to Genetic and Molecular Biology* **4**.

Kang, S. J., Gordon, D. and Finch, S. J. (2004). What snp genotyping errors are most costly for genetic association studies. *Genetic epidemiology* **26**, 132–141.

Longmate, J. A. (2001). Complexity and power in case-control association studies. *American Journal Human Genetics* **68**, 1229–1237.

Lu, Z.-H. and King, M. L. (2002). Improving the numerical techniqie for computing the accumulated distribution of a quadratic form in normal variables. *Econometric reviews* **21**, 149–165.

Mitra, S. (1958). On the limiting power function of the frequency chi-square test. *Annals of Mathematical Statitics* **29**, 1221–1233.

Risch, N. (2000). Searching for genes in complex diseases : lessons from systemic lupus erythematosus. *American society for clinical investigation* **105**, 1503–1506.

Sham, P., Cherny, S., Purcell, S. and Hewitt, J. (2000). Power of linkage versus association analysis of quantitative traits, by use of variance components models, for sibship data. *American Journal Human Genetics* **66**, 1616–1630.

Slager, S. and Schaid, D. (2001). Case-control studies of genetic markers : power and sample size approximations for armitage's test for trend. *Human Heredity* **52**, 149–153.

Zhao, H. (2000). Family-based association studies. *Statistical Methods in Medical Research* **9**, 563–587.

Appendix 1 : Quadratic Form in Normal Variables

In this appendix we propose to recall the definition of this distribution and to explain how it is possible to compute its cumulative distribution function (CDF).

Definition 1 *If $X \sim \mathcal{N}(\mu, \Sigma)$ is a dimension $d \geqslant 1$ (column) vector of normal variables we call*

$$Q = A + BX + X'CX$$

a quadratic form in normal variables *(QFNV) of dimension d with parameters $A \in \mathbb{R}$, $B \in \mathbb{R}^{(1,d)}$ and $C \in \mathbb{R}^{(d,d)}$ and with $\mu \in \mathbb{R}^{(d,1)}$ and $\Sigma \in \mathbb{R}^{(d,d)}$ are mean and covariance matrix of the normal variables.*

In particular, a linear combination of (central or not, independent or not) chi-square is a QFNV.

For non degenerate parameters, it is possible to express a QFNV as a linear combination of independent non-central chi-square distribution (Lu and King, 2002). Namely :

Proposition 2 *For any QFNV Q with non-singular covariance matrix Σ and matrix C, it exists $1 \leqslant n \leqslant d$, $\lambda_j \in \mathbb{R}$, $d_j \in \mathbb{N}^*$ and $\nu_j > 0$ (for all $1 \leqslant j \leqslant n$) such as*

$$Q = K + \sum_{j=1}^{n} \lambda_j \chi_j$$

where $K \in \mathbb{R}$ and $\chi_j \sim \chi^2(d_j, \nu_j)$ are independent chi-square variables with d_j degrees of freedom and ν_j non centrality parameters.

Proof. We first factorize Q in

$$Q = \underbrace{\left(A - \frac{BC^{-1}B'}{4}\right)}_{K} + \underbrace{\left(X + \frac{C^{-1}B'}{2}\right)' C \left(X + \frac{C^{-1}B'}{2}\right)}_{Y'CY}$$

where $Y \sim \mathcal{N}(\tilde{\mu}, \Sigma)$ with $\tilde{\mu} = \mu + C^{-1}B'/2$. We consider then the linear transformation $Z = \Sigma^{-1/2}Y$ so $Z \sim \mathcal{N}(\Sigma^{-1/2}\tilde{\mu}, I)$ and

$$Q = K + Z' \underbrace{\Sigma^{-1/2} C \left(\Sigma^{1/2}\right)'}_{\tilde{C}} Z$$

We consider then the orthogonal matrix P of the eigenvector of $\tilde{C}$ and denote by $D = P'\tilde{C}P$ the diagonal matrix of the corresponding eigenvalues. With $W = P^{-1}Y$ we get

$$Q = K + W'DW$$

with $W \sim \mathcal{N}(P^{-1}\Sigma^{-1/2}\tilde{\mu}, I)$. For $1 \leqslant j \leqslant n$, we denote by λ^j the (distinct) eigenvalue and by d_j their orders of multiplicity (one should note that they are also those of $C\Sigma$ or ΣC). Finally, we consider $\nu_j = \sum_{q=1}^{d_j} \gamma_{j,q}^2$ where $\gamma_{j,q}$ are the elements of $P^{-1}\Sigma^{-1/2}\tilde{\mu}$ corresponding to the same eigenvalue λ_j and the result is established.

From now, we hence focus of the numerical CDF evaluation of

$$Q = \sum_{j=1}^{n} \lambda_j \chi^2(d_j, \nu_j)$$

a linear combination of independent and non central chi-square distributions.

A numerical inversion of the characteristic function is then possible resulting through truncation and trapezoidal integration (Davies 1973, Davies 1980) to the following formula

$$\mathbb{P}(Q < c) = \frac{1}{2} - \sum_{m=0}^{M} \left(\frac{\sin\{\theta_c[(m+0.5)\Delta]\}}{\pi(m+0.5)\gamma[(m+0.5)\Delta]} \right)$$

where Δ is the (small) step interval, M the (large) number of step intervals, and $U = (M + 0.5)\delta$ the truncation value. The functions θ_c and γ are give by

$$\theta_c(u) = \sum_{j=1}^{n} \left[\frac{d_j}{2} \tan^{-1}(2u\lambda_j) + \nu_j u \lambda_j (1 + 4u^2\lambda_j^2)^{-1} \right] - cu$$

and

$$\gamma(u) = \prod_{j=1}^{n} (1 + 4u^2\lambda_j^2)^{d_j/4} \exp\left(2u^2 \sum_{j=1}^{n} \frac{\nu_j \lambda_j^2}{1 + 4u^2\lambda_j^2} \right)$$

The numerical evaluation of the CDF using this formula leads to an error of truncation ε_T depending on the truncation bound U and to an error of integration ε_I depending on the step interval Δ. It exists many concurrent ways to choose both these values and Lu and King (2002) provides a complete review of them.

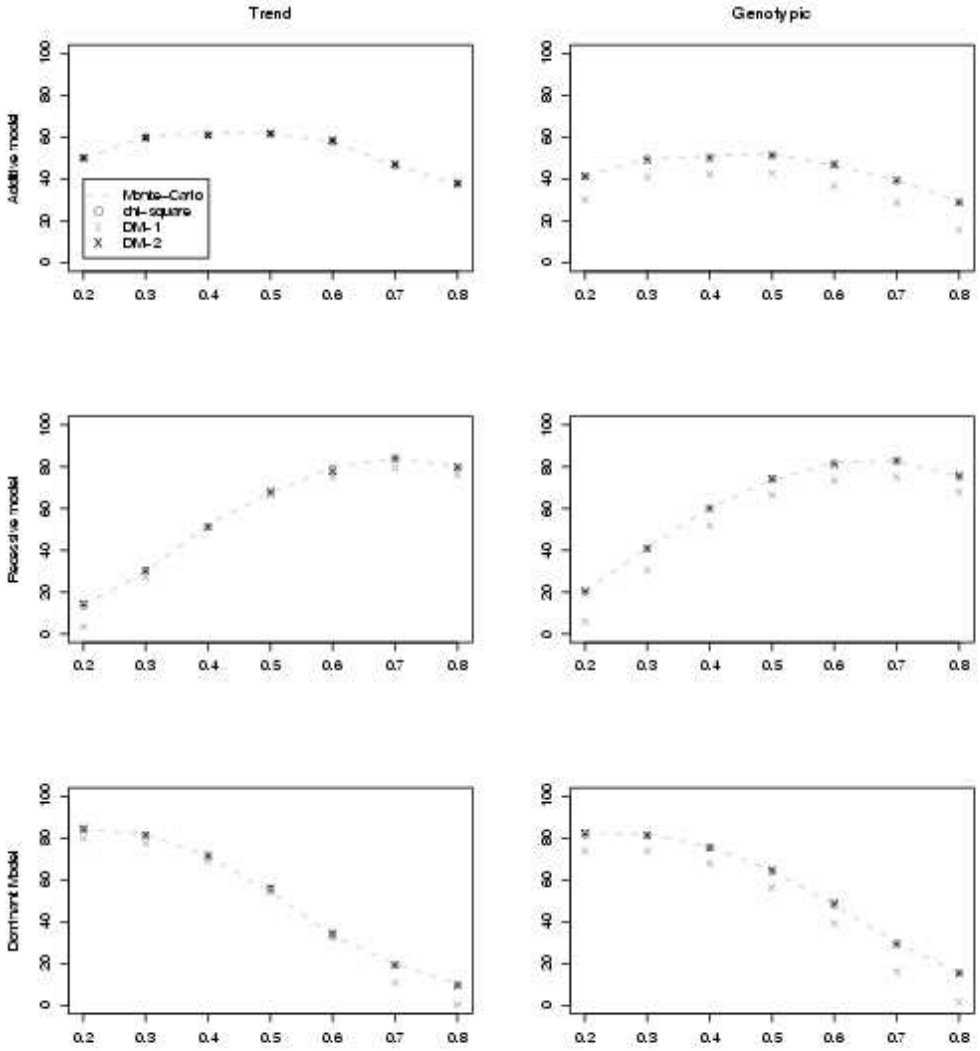


FIG. 1. Power estimation (at the 5% significance level) for the trend and genotypic tests according to the allele frequency (p).

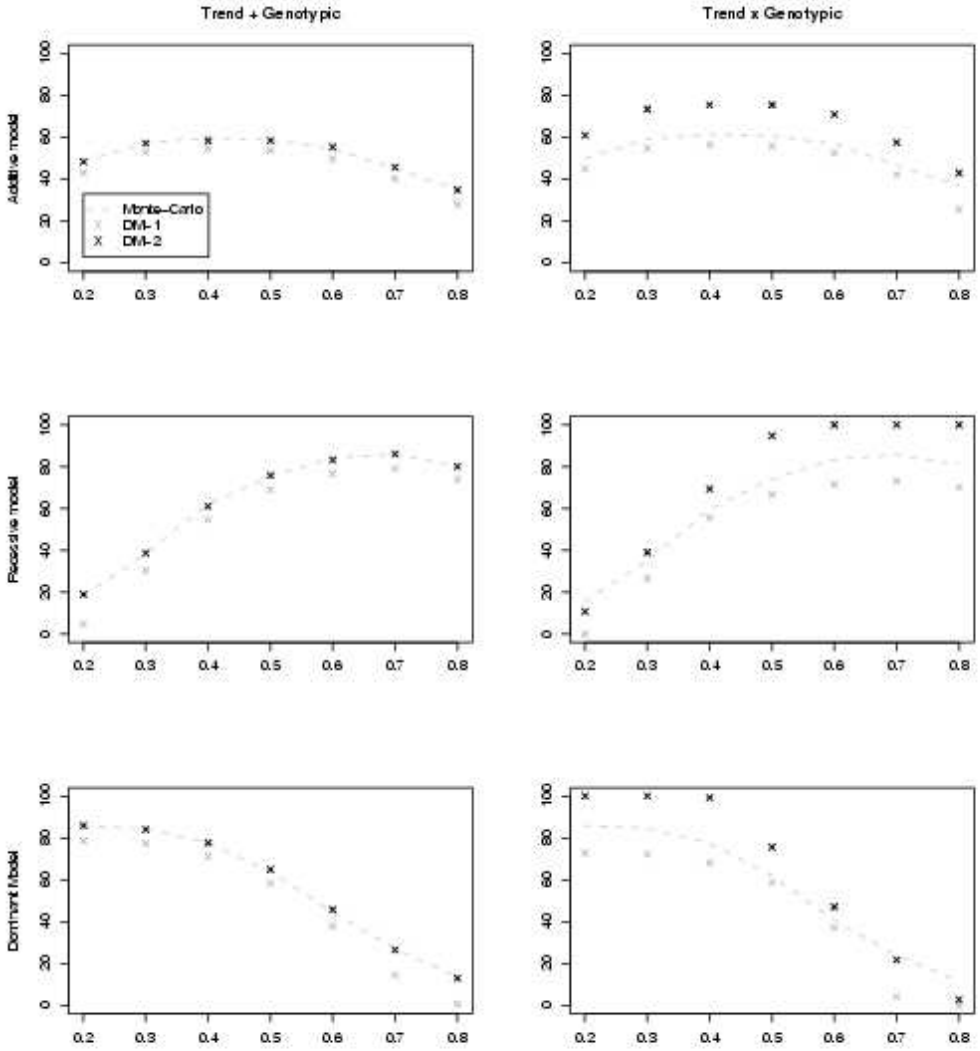


FIG. 2. Power estimation (at the 5% significance level) for the meta-statistics according to the allele frequency (p).

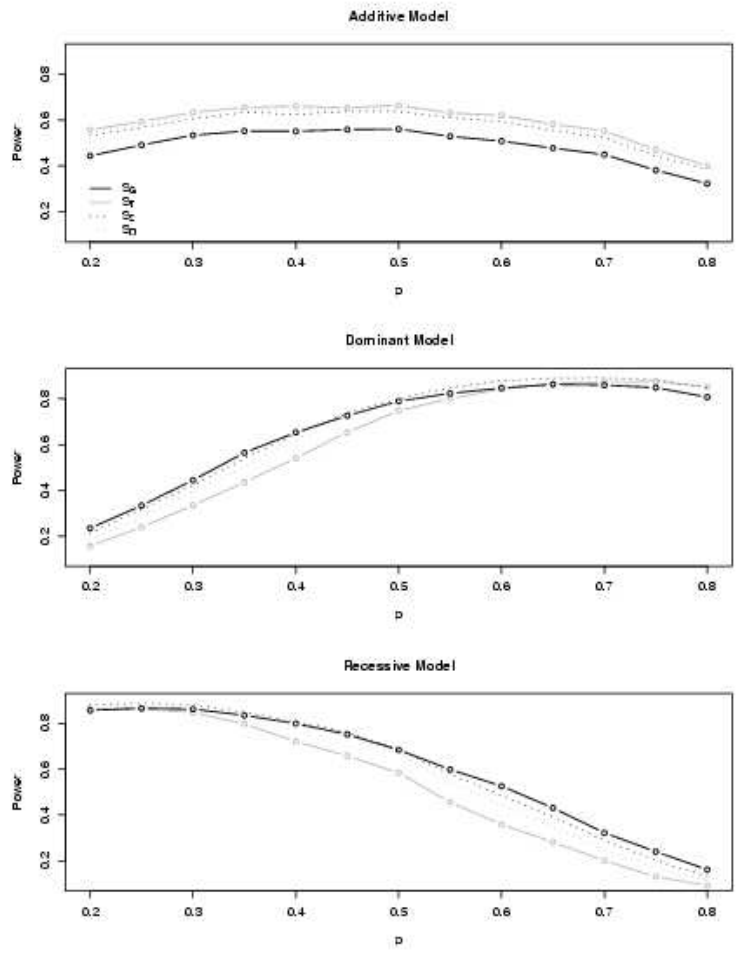


FIG. 3. Power comparison : this figure compares the power of the four statistics $\mathcal{S}_G$, $\mathcal{S}_T$, $\mathcal{S}_\Sigma$ and $\mathcal{S}_\Pi$ according to the allele frequency (p). It is done for the additive, recessive and dominant models. Power is computed by Monte-Carlo at the 5% significance level.

	aa	aA	AA	total
diseased	D_0	D_1	D_2	n_D
control	C_0	C_1	C_2	n_C
total	n_0	n_1	n_2	n

TAB. 1
The genotypic contingency table

Cinquième partie

Autres publications

Chapitre 17

Varietal Distinctness Assisted by Molecular Markers : a Methodological Approach

Article de Nuel, Baril, et Robin publié en 1999 dans *Acta Horticulturae*. On y présente une méthode pour faciliter la distinction variétale en utilisant des données génotypiques à partir desquelles on effectue des prédictions de distances phénotypiques. Un ensemble de règles de décisions permet ensuite de savoir quelles espèce du catalogue il faudra expérimenter aux champs. Une étude par simulation valide l'approche et montre qu'elle pourrait permettre de réaliser des économies substantielles avec un taux d'erreurs acceptable.

VARIETAL DISTINCTNESS ASSISTED BY MOLECULAR MARKERS: A METHODOLOGICAL APPROACH

G. Nuel
Université d'Evry-Val d'Essonne, Analyse et Probabilités, F-91000 EVRY, France

C. Baril
GEVES Domaine Expérimental de La Minière, F-78285 GUYANCOURT CEDEX, FRANCE

S. Robin
Institut National de la Recherche Agronomique, Unité de Biométrie, F-78352 JOUY EN JOSAS, FRANCE

Keywords: Linear Model, Phenotypic Distance, Genetic Distance, Mahalanobis Distance, Maize

Abstract

The varietal distinctness procedure involves expensive, space and time consuming measurements of phenotypical traits. Moreover, for most species and traits, interactions between genotype and environment complicate the cultivar evaluation. Molecular markers could provide a distinctness procedure that actually help increasing reliability of decisions and saving field experiments and time. In the statistical framework of the linear model, the authors propose to define a new distinctness procedure using molecular data to predict phenotypical distance between cultivars. As an example, this method is applied to maize data and leads to 29% of savings for less than 5% of errors. Improvement of these methods are finally discussed.

1. Introduction

While phenotypic observations depend on genetic sensitivities to environmental conditions, molecular data can give a direct access to most part of the genome. It is therefore of prime interest to be able to predict phenotypic distances through molecular data. Many authors have already studied different genetic distances (see Nei and Roychoudhury (1973), Nei and Li (1979), Sneath and Sokal (1973), Rogers (1972) or, for a general review, Gower (1985) or Piepho and Laidig (1996)). These distances are essentially used to predict specific combining abilities (see Baril *et al.* (1997) for a general review) and also to estimate coefficient of coancestry (see Bernardo (1993)). This paper deals with a different problem: the aim is to propose a statistical approach to manage large collections of genotypes combining phenotypic and molecular information. Most of the statistical developments are given in Nuel *et al.* (1999). An application to distinctness of plant varieties is developed in order to save some field trials and focus the field measurements on a subset of questionable varieties.

2. Material

144 maize inbred lines, representative of material released in France, were described using both Restriction Fragment Length Polymorphism (RFLP) markers and morphological quantitative traits. This genetic material consist in a selection of 80 monolocus Enzyme Primer Combinations (EPCs) corresponding to different loci (see Dillmann *et al.* (1997) for more details) and the phenotypic material consists in the observation of ten morphological quantitative traits during four years (from 1989 to 1992) in three sites with two replications per site following a block design (see Bar–Hen *et al*. (1995) for more details).

3. Methods

3.1. Model

We shall assume here that each quantitative trait Y^j is normally distributed with a mean linearly depending on a subset B_j of the whole set M of the observed markers. Because of the small number of lines, no interaction between markers can be considered, although the theoretical results could easily take them into account. In the following, the markers of B_j shall be called relevant markers for Y^j.

In this linear model, the different phenotypic traits do not depend on the same markers; Burstin and Charcosset (1997) have shown that this model gives a good description of the relationship between phenotypic and molecular distances.

The set of the relevant markers B_j has to be determined for each trait Y^j. Here, these markers are selected using a 'stepwise' procedure optimizing a fitting criterion, the F statistic. At each step, the F statistic is also used to check significancy of the markers of B_j and is based on type III sums of squares with a α test level. The type III sums of squares associated to the marker i is defined by the difference between the sums square of model's residuals including all markers except the i marker, and the the one of complete model. This type of sum square is relevant in the case of non orthogonal design because of linkage desequilibrium between markers, and so is relevant in our case.

The different models obtained for each phenotypic trait can be rearranged in a single model using the common matrix $\mathbf{X}$, with the convention that the parameters of non relevant markers are set to zero. We get then

$$\underset{[n\times p]}{\mathbf{Y}} = \underset{[n\times(A+1)]}{\mathbf{X}} . \underset{[(A+1)\times p]}{\mathbf{\Theta}} + \underset{[n\times p]}{\mathbf{E}}$$

where n is the number of lines, p the number of traits, A the total number of alleles of all the markers. The i–th row of the matrix $\mathbf{E}$ has a multivariate distribution with mean $\mathbf{0}$ and variance $\mathbf{\Sigma}$. For further details, see Nuel *et al.* (1999).

3.2. Distance Prediction

If the morphological distance between the lines a and b is defined, using the metric $\mathbf{M}$, as

$$D^2_{\mathbf{M}}(a,b) = \left\|\mathbf{Y}_a - \mathbf{Y}_b\right\|^2_{\mathbf{M}} = (\mathbf{Y}_a - \mathbf{Y}_b).\mathbf{M}.(\mathbf{Y}_a - \mathbf{Y}_b)',$$

we propose a natural prediction of this distance using the predicted values $\hat{\mathbf{Y}}_a = \mathbf{x}_a.\mathbf{\Theta}$ and $\hat{\mathbf{Y}}_b = \mathbf{x}_b.\mathbf{\Theta}$, so we get

$$\delta^2_{\mathbf{M}}(a,b) = \left\|\hat{\mathbf{Y}}_a - \hat{\mathbf{Y}}_b\right\|^2_{\mathbf{M}} = (\mathbf{x}_a - \mathbf{x}_b).\mathbf{\Theta}.\mathbf{M}.\mathbf{\Theta}.(\mathbf{x}_a - \mathbf{x}_b)'.$$

In a sense, this distance can be considered as a molecular distance calculated on $(\mathbf{x}_a - \mathbf{x}_b)$ with the metric $\mathbf{\Theta}.\mathrm{M}.\mathbf{\Theta}'$, taking into account only the observed markers explaining the phenotypic variability.

If the metric M is the inverse of the covariance matrix of $(\mathbf{Y}_a - \mathbf{Y}_b)$ (*i.e.* $\mathbf{M} = \frac{1}{2}\mathbf{\Sigma}^{-1}$), the morphological distance $D^2(a,b)$ is the Mahalanobis distance. The interest of this distance is that it takes into account the variances and covariances of the phenotypic variables (see Mahalanobis (1936)). As shown in Nuel *et al.* (1999), it has a non–central chi–square distribution with p degrees of freedom and with non–centrality parameter $\delta^2(a,b)$:

$$D^2(a,b) \sim \chi^2_p[\delta^2(a,b)]. \quad \textbf{(1)}$$

If we use another metric **M**, the distribution of $D^2_{\mathbf{M}}(a,b)$ turns out to be a linear combination of non−central chi−squares (see Nuel *et al.* (1999)).
In the following, D^2 and δ^2 shall stand for Mahalanobis distances.

3.3. Decision Rule

The distinctness between two lines is generally determined according to a threshold t on the phenotypic distance:

$$\{\text{lines } a \text{ and } b \text{ are distinct}\} \Leftrightarrow \{D^2(a, b) > t\}.$$

Applying formula **(1)** and given the genotypes $\mathbf{x}_a$ and $\mathbf{x}_b$, we can give a prediction interval with level $1 - \gamma$ for D^2:

$$\Pr\left\{ D^2(a,b) \in \left[\chi^2_{p;\gamma/2}\left(\delta^2(a,b)\right); \chi^2_{p;1-\gamma/2}\left(\delta^2(a,b)\right) \right] \right\} = 1-\gamma \quad \textbf{(2)}$$

denoting $\chi^2_{p;\beta}(\delta^2)$ the β−quantile of the non central distribution $\chi^2_p(\delta^2)$.
This result can be applied to predict whether lines a and b are distinct or not according to the following decision rule:

- if $\chi^2_{p;\gamma/2}\left[\delta^2(a,b)\right] > t$, the whole prediction interval on D^2 is above the threshold t: lines a and b are predicted distinct considering their genotypes;
- if $\chi^2_{p;1-\gamma/2}\left[\delta^2(a,b)\right] < t$, the whole prediction interval on D^2 is beyond the threshold t: lines a and b are predicted similar considering there genotypes;
- if $\chi^2_{p;\gamma/2}\left[\delta^2(a,b)\right] \le t \le \chi^2_{p;1-\gamma/2}\left[\delta^2(a,b)\right]$, the threshold t is included in the prediction interval: no decision can be made on the basis of the simple genotypes, an experiment has to be made to measure phenotypic variables.

3.4. Errors

Using the molecular thresholds b_{inf} and b_{sup} given by

$$t = \chi^2_{p;1-\gamma/2}(b_{\text{inf}}),\ t = \chi^2_{p;\gamma/2}(b_{\text{sup}})$$

it is possible to define the two kind of errors that decisions made according to molecular data can lead to.

- if $\delta^2(a,b) < b_{\text{inf}}$ (a and b predicted similar) and $D^2(a,b) > t$ (a and b are actually distinct), then the decision is wrong.
- if $\delta^2(a,b) > b_{\text{sup}}$ (a and b predicted distinct) and $D^2(a,b) < t$ (a and b are actually similar), then the decision is wrong too.

3.5. Proposition of a New Distinctness Procedure

3.5.1. Distinctness

The aim of a distinctness procedure is to decide whether candidates lines (set C) are distinct from a reference collection (set R). A candidate line c of C is distinct from the collection R if it is distinct from all the elements r of R:

$$\{c \text{ distinct from } R\} \Leftrightarrow \{\forall r \in R : D^2(a,b) > t\}. \quad \textbf{(3)}$$

On the opposite c is called similar to R as soon as it resembles to one of the element of R:

$$\{c \text{ similar to } R\} \Leftrightarrow \{\exists r \in R : D^2(a,b) < t\}.$$

3.5.2. Savings

Obviously, all candidate lines have to be set on trial to observe their phenotypes, but we hope to avoid a part of the experiments for the reference lines by using the predicted distance (through molecular data) rather than the observed one.
For a reference line r, we can avoid the experimentation if a decision can be made for all the molecular comparisons between r and all the candidate lines. Let us denote S (for savings) the set of the reference lines which do not need to be set on trial ($S \subset R$):

$$S = \{r \in R : \forall c \in C, \delta^2(a,b) \notin [b_{inf} ; b_{sup}]\}. \quad \textbf{(4)}$$

3.5.3. False distinctness

A candidate line c can be predicted distinct from r while it is not. This happens when all the following conditions are fulfilled:
D1 c is similar to r: $\exists\, r \in R, D^2(r,c) < t$;
D2 c is not predicted similar considering molecular data: $\forall\, r \in R, \delta^2(r,c) > b_{inf}$;
D3 c is distinct from all the experimented reference lines: $\forall\, r \in R–S, D^2(r,c) > t$;
D4 none of the errors is corrected by any experiment: $\forall\, r \in R, \{\delta^2(r,c) > b_{sup}$ and $D^2(r,c) < t\} \Rightarrow \{r \in S\}$.

3.5.4. False similarity

A candidate line c can be predicted to be similar to r while it is actually distinct. This happens when all the following conditions are fulfilled:
S1 c is distinct from R: $\forall\, r \in R, D^2(r,c) > t$;
S2 at least one decision concerning c is wrong and the concerned reference line is not experimented: $\exists\, r \in R, \delta^2(r,c) < b_{inf}$ and $r \in S$.
In the context of DUS testing, false similarities will never occur because no similarity decision will be taken based on just the molecular data. One reason for this is that plant breeders need a phenotypic evaluation of their lines when proposing a new variety for registration; another reason is that the reference collection has to contain the widest possible biodiversity so it seems preferable to accept some irregular elements (that is to make a conservative choice) rather than to reject original lines.
This paper will therefore focus on the upper bound b_{sup}. In the following, we shall assume that b_{inf} is equal to zero – the D2 is always fulfilled. This is equivalent to apply the decision rule with a unilateral prediction interval $[\chi^2_{p;\gamma}(\delta^2);\infty)$.
In both cases (false distinctness or false similarity), the molecular prediction leads to some errors that cannot be corrected because of the savings.

4. Results

Among the 144 lines, a subset of 100 lines has been sampled to stand for the reference collection R; the set C of candidate lines shall be sampled among the 44 remaining. Two types of couples of lines can be considered according to their membership of the reference or candidate set:
Type 1 : a mixed couple of a reference and a candidate line: $(r,c) \in R \times C$;
Type 2: a couple of reference lines: $(r,r') \in R^2$.

4.1. Observed confidence

Formula **(2)** gives a prediction interval for each distance D^2 with theoretical confidence $1 - \gamma$. We compare it to the observed confidence defined as the proportion of observed D^2 that lie in the prediction interval. As the parameter of the models have been estimated with the markers and traits of the lines of *R* only, the observed confidence is expected to be smaller for type 1 than for type 2. We can measure the distance prediction adjustment through the type 2 observed confidence. To check the robustness of the model to new data, we have to consider the observed confidences of type 1. We found 72.8% for type 1 and 90.2 % for type 2 instead of a theoretical 95% = 1 - γ. The differences between theoretical and observed confidence can be explained in the case of the type 2 by the fact that the parameter **Θ** is estimated (and not given). That last point is not taken into account in formula **(1)** and this induces a bias (see Nuel *et al.* (1999) for a complete discussion).

To analyze the performances of this new distinctness procedure, we simulate several varietal distinctness sessions with a varying candidate set *C* and a constant reference collection *R*. At the present time, in the maize case, French number of candidate lines (350) represents 20% of the size of the reference collection (1700 lines) . In order to respect this proportion, we sample 20 lines among the 44, for each simulation.
Because the binomial coefficient C_{44}^{20} exceeds 10^{12}, we never consider the total number of possible sets *C*, but only a random sample.

4.2. Phenotypic threshold *t*

To determine the phenotypic threshold, we sample 10 000 sets *C*; for each sample, we calculate *t* so that the acceptance rate (according to formula **(3)**) is 30% (the usual rate for maize). The final threshold is the mean of these 10 000 values.

4.3. False distinctness and savings

The choice of the level γ is a crucial step in the procedure: for a given phenotypic threshold *t*, the molecular threshold b_{sup}, defined by $t = \chi^2_{p;\gamma}(b_{sup})$, is a decreasing function of γ. Thus, a small value of γ will provide a high value of b_{sup} and, therefore, a large field trial zone. In this case, we expect very few errors, and savings. On the opposite, a high level γ will lead to numerous savings and errors.
To optimize the procedure, we consider all the values of γ from 0.5% to 10% by 0.5%. For computational time reasons, we only use 1 000 samples to estimate the number of false distinctnesses and savings for each value of γ. These quantities are estimated and not directly calculated because the combinatory problem seems really intricated.

Table 1 – level γ, false distinctness and percentage of savings.

level γ (%)	0.5	1	**1.5**	2	**2.5**	3	3.5	4	4.5	5
false distinctness	0.1	0.2	**0.6**	1.2	**1.2**	2.1	2.7	2.6	5.2	5.2
Savings (%)	10	18	**29**	39	**40**	54	54	67	67	84

5.5	6	6.5	7	7.5	8	8.5	9	9.5	10
5.2	5.2	5.2	8.9	8.9	8.9	8.9	12.2	12.2	12.2
84	84	84	95	95	95	95	99	99	99

4.4. Results

Table 1 shows the evolution of the number of false distinctnesses and of the percentage of savings as the level γ grows. We observe that both these quantities increase step by step; this is due to the limited size of the reference collection (100). Since the lower bound b_{inf} is set to zero, the definition **(4)** can be rewritten as

$$S = \{ r \in R : \inf_{c \in C} [\delta^2(r,c)] > b_{sup} \}.$$

The set S changes only when the bound b_{sup} becomes lower than one of 100 different values of $\inf_{c \in C}[\delta^2(r,c)]$. The same kind of mechanism explains the steps in the number of false distinctnesses.

To choose the level γ, we set a maximum acceptable number false distinctnesses. If we are only ready to accept one error among the 20 candidate lines, we see on table **1** that the corresponding level is γ =1.5% which provides 29% of savings. If we accept a little bit more errors, say 1.2, we get γ = 2.5% and 40% of savings.

5. Discussion and conclusion

One could note that the choice of the linear model which residuals provide an estimation of phenotypic values is fundamental. It gives the metric matrix used for the calculation of phenotypic distances. When the interaction terms are significant, it could be adapted to use as many traits as there are phenotypic traits × environments combinations. This would multiply the number of variables but would probably affect positively the quality of the phenotypic distance estimation.

The distinctness of varieties exclusively based on phenotypic data being both area and time consuming, necessarily leads to compromises between the number of observed varieties and the quality of experimental design, because of the difficulty of mastering all the environmental variation factors. The present modelization provides a valuable tool to concentrate the field work on the most difficult evaluations. With a conservative approach, one could avoid 29% of the trials making at most 1 false distinctness.

The error occurrence is calculated in comparaison to the phenotypic results which could have been themselves different using a more accurate experimental design. Nevertheless to be more efficient, this approach has to be validated on more data and first of all using more relevant molecular markers. For this last point, substantial improvements can be expected soon thanks to important progress made in the localization of genes coding for agronomic traits, using Expressed Sequences Tags (EST) of cDNA for example.

As said in the section 'Model selection' the fact that parameters are estimated rather than given induces some bias of the predicted distance. Nevertheless this bias seems to remain small, and the statistical development given in see Nuel *et al.* 1999 could further improve the method. Finally, the generalization of this approach to qualitative traits would be of great interest considering the important proportion of discrete variables observed in distinctness trials.

Acknowledgments

We are grateful to all the persons who obtained the data used in this study and especially to Z. Karaman and D. Guérin. We are also grateful to Y Dattée who initiated the project. This

research was funded by the SEPROMA association, the French Ministry of Agriculture, INRA and GEVES.

References

Baril CP, Verhaegen D, Vigneron Ph, Bouvet J-M, Kremer A (1997) Structure of the specific combining ability between two species of *Eucalyptus*. I. RAPD data. Theor Appl Genet **94:** 796–803.

Bar-Hen A, Charcosset A (1995) Relationship between molecular and morphological distances in a maize inbred lines collection. Application for breeder's right protection. pp 57-66 *in* Van Ooijen JW, Jansen J (eds). Biometrics in plant breeding: Applications of molecular markers (Proc 9th Meet Eucarpia Sect Biometrics Plant Breed). CPRO-DLO, Wageningen.

Bar–Hen A, Charcosset A, Bourgoin M, Guiard J (1995) Relationship between genetic markers and morphological traits in a maize lines collection. Euphytica **84:** 145–154.

Bernardo R (1993) Estimation of coefficient of coancestry using molecular markers in maize. Theor. Appl. Genet. **85:** 1055-1062.

Burstin J, Charcosset A (1997) Relationship between phenotypic and marker distances: theoretical and experimental investigations. Heredity **79** (5): 477-483.

Dillmann C, Bar–Hen A, Guerin D, Charcosset A, Murigneux A (1997) Comparison of RFLP and morphological distances between maize *Zea Mays* L. inbred lines. Consequences for germplasm protection purposes. Theor Appl Genet **95:** 92–102.

Gower JC (1985) Measures of similarity and distance, pp397-405 *in* Kotz, S., Johnson, N.L., (eds) Encyclopedia of statistical sciences, vol 5. Wiley: New-York.

Mahalanobis PC (1936) On the generalized distances in statistics. Proc Nat Inst Sci India. **2:** 49–55.

Nei M, Li WH (1979) Mathematical models for studying genetic variation in terms of restriction endonuclease. Proc Nat Acad Sci USA **76:** 3269-3273.

Nei M, Roychoudhury AK (1973) Sampling variances of heterozigosity and genetic distance. Genetics **76:** 379-390.

Nuel G, Robin S, Baril C (1999)) Predicting distances using a linear model: the case of varietal distinctness. (Submitted for publication).

Piepho HP, Laiding F (1996) A review of methods for cluster analysis of marker data. UPOV TWC/14/8.

Rogers JS (1972) Measures of similarity and genetic distance. Studies in Genetics VII. Univ Texas Publ **7213:** 145–153.

Sneath PHA, Sokal RR (1973) Numerical taxonomy. W. H. Freeman and co.: San Francisco.

Chapitre 18

Predicting distances using a linear model : the case of varietal distinctness

Article de Nuel, Robin, et Baril publié en 2001 dans le *Journal of Applied Statistics*. L'article détaille le problème de la prédiction de distances dans un modèle linéaire. L'application de cette technique à la distinction variétale est ensuite proposée.

Predicting distances using a linear model: the case of varietal distinctness

Nuel G.[1], Robin S.[2], Baril C.P.[3]

J. Appl. Statist. **28 (5)** 607–621.

([1]) Laboratoire d'Analyse et Probabilités, Université d'Evry - Val d'Essone, 91000 Evry, FRANCE
([2]) Unité Mathématique, Informatique et Génome, Institut National de la Recherche Agronomique, 78026 Versailles, FRANCE (*corresponding author*)
([3]) Groupe d'Étude des Variétés et des Semences, Domaine de la Minière, 78285 Guyancourt, FRANCE

Abstract

Differences between plant varieties are based on phenotypic observations, which are both space and time consuming. Moreover the phenotypic data result from the combined effects of genotype and environment. On the contrary, molecular data are easier to obtain and give a direct access to the genotype. In order to save experimental trials and to concentrate efforts on the relevant comparisons between varieties, the relationship between phenotypic and genetic distances is studied. It appears that the classical genetic distances based on molecular data are not appropriate for predicting phenotypic distances. In the linear model framework, we define a new pseudo genetic distance which is a prediction of the phenotypic one. The distribution of this distance given the pseudo genetic distance is established. Statistical properties of the predicted distance are derived when the parameters of the model are either given or estimated. We finally apply these results to distinguishing between 144 maize lines. This case study is very satisfactory because the use of anonymous molecular markers (RFLP) leads to saving 29% of the trials with an acceptable error risk. These results need to be confirmed on other varieties and species and would certainly be improved by using genes coding for phenotypic traits.

keywords: genetic distances, linear model, molecular markers, phenotypic distances, varietal distinctness

1 Introduction

Every year, hundreds of new plant varieties emerge from laboratory and field trials. To supply farms with the latest products of genetic progress in an organized and coherent manner, the entry of a new variety into the official catalogue is a required condition for commercialization. To be registered, a variety must be distinct, uniform and stable (DUS) and show agronomic or technological advantages. GEVES (Groupe d'étude et

de contrôle des variétés et des semences) performs technical analyses of new varieties on behalf of the CTPS (Comité Technique Permanent de la Sélection) under the authority of the french Ministry of Agriculture. DUS tests involve observations over one to three growth cycles in two sites. In all cases, morphological and physiological characteristics are examined at several stages of plant growth, and, in some cases, biochemical characteristics as well. All countries that are members of UPOV (Union pour la Protection des Obtentions Végétales) follow the same recommendations and thus assessment methods are harmonized internationally. In this article, we only focus on the distinctness procedure.

At the present time, a variety is accepted as being new if it can be unambiguously distinguished from all other varieties on the basis of at least one characteristic. This differentiation may also be based on a number of small differences. The requirement for a more accurate and objective method to assess the distinctness of two varieties has led the CTPS to encourage feasibility studies on the use of multivariate criteria. Maize is genetically well characterized, and has thus been used as a model system. An investigation of its DNA polymorphism has been undertaken by GEVES in collaboration with maize breeders and INRA (Institut National de la Recherche Agronomique). Every year 350 lines (corresponding to 500 varieties) are tested for DUS and the acceptance rate is about 30%. The distinctness of a new variety is assessed by comparing with standard varieties, called the reference collection, which included 1700 lines (corresponding to 1000 varieties) in 1998 .

Obtaining phenotypic data is both space and time consuming; as a consequence, the number of varieties to be tested every year prevents from numerous repetitions. On the other hand molecular characteristics are comparatively easier to obtain. The aim of this paper is to predict phenotypic distances through molecular data. In the context of varietal distinctness, using these predictions could save many field trials. Efforts could be concentrated on difficult cases using better experimental designs.

Numerous genetic distances and similarity indices already exist; Gower (85) gives a general review. The simplest measure of genetic distance without any extra information but molecular is the Rogers' distance calculated as the proportion of markers having different modalities. The relationship between phenotypic (Mahalanobis) and genetic (Rogers) distances is not linear but may be described as triangular (see figure **1**). This shape, which is observed for all genetic distances, has some biological explanations: (*i*) compensation of elementary traits in complex traits expression, (*ii*) low part of genome involved in phenotypical traits expression, (*iii*) lack of environmental conditions necessary for the expression of all the observed traits, (*iv*) variation of linkage disequilibrium according to the origin of genotypes. As demonstrated in Burstin & Charcosset (97), this triangular shape would still be observed if one used only molecular markers linked with the phenotypic traits, such as QTL (quantitative traits loci).

Obviously, this kind of relationship prevents one from predicting the phenotypical distance from the molecular one. A new genetic distance has therefore to be defined. We propose to use a linear model for predicting the phenotypic traits from the molecular markers. We are then able to calculate a prediction of the phenotypical distance based only on the molecular data. This predicted distance can therefore be considered as a pseudo

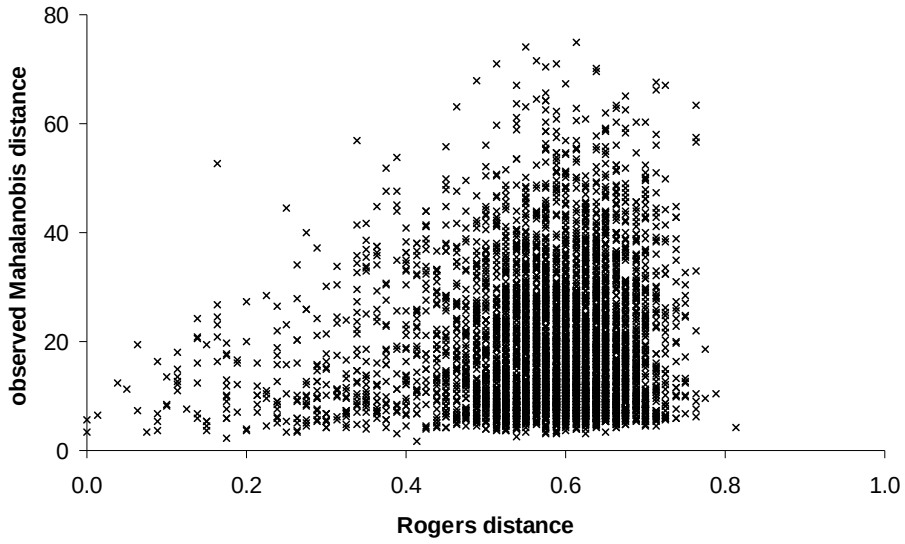


Figure 1: Plot of the Mahalanobis distance D versus the Rogers' (genetic) distance.

genetic distance. Probabilistic and statistical properties of this distance are derived; in particular, its distribution is established. In section **4**, we use this model to distinguish between maize lines.

2 Linear model and Distance prediction

Let us consider n individuals each described by p quantitative morphological variables Y^j $(j = 1 \dots p)$ and m molecular markers $(k = 1 \dots m)$ each having s_k modalities corresponding to the alleles observed in the collection. Let $\mathcal{M}$ denote the set of the observed markers $(|\mathcal{M}| = m)$.

2.1 The linear model

Model for one variable. Each morphological variable is governed by a set $\mathcal{A}_j$ of markers. We assume here that the relation between Y^j and the markers of $\mathcal{A}_j$ can be modeled by a classical linear model $\mathbf{Y}^j = \mathbf{X}_{\mathcal{A}_j}\theta_{\mathcal{A}_j} + \mathbf{F}^j$ where $\mathbf{X}_{\mathcal{A}_j}$ is the matrix including the genotypes (coded in terms of presence/absence of each modality of each marker) of all individuals for the markers of $\mathcal{A}_j$ and $\theta_{\mathcal{A}_j}$ is the vector of the effects of the different modalities.

Unfortunately we do not observe all the markers of the relevant set $\mathcal{A}_j$ but only the

subset $\mathcal{B}_j = \mathcal{M} \cap \mathcal{A}_j$. So we have to consider the restricted model

$$\underset{(n,1)}{\mathbf{Y}^j} = \underset{(n,1+M_j)}{\mathbf{X}_{\mathcal{B}_j}} \underset{(1+M_j,1)}{\theta_{\mathcal{B}_j}} + \underset{(n,1)}{\mathbf{E}^j} \tag{1}$$

where $\mathbf{X}_{\mathcal{B}_j}$ contains the genotypes for the markers of $\mathcal{B}_j$ ($M_j = \sum_{k \in \mathcal{B}_j} s_k$, the first element of $\theta_{\mathcal{B}_j}$ is the intercept). The residuals $\{E_a^j\}_{a=1\ldots n}$ are assumed to be independent and normally distributed with the same mean 0 and variance σ_j^2. In this model, the residual term $\mathbf{E}^j$ includes the residual term $\mathbf{F}^j$ of the complete model and the effects of the unobserved relevant markers.

This model appears to give a good representation of the relation between genetic and phenotypic distances. In particular, Burstin & Charcosset (97) have shown, using simulations, that it reflects the frequently observed triangular relationship (see figure **1**).

From now on, to make the notation more convenient, we shall use an alternative but equivalent form of model (1):

$$\underset{(n,1+M)}{\mathbf{Y}^j} = \underset{(n,1+M)}{\mathbf{X}} \underset{(1+M,1)}{\theta^j} + \underset{(n,1)}{\mathbf{E}^j} \tag{2}$$

where $\mathbf{X}$ contains the genotypes of all individuals and for all the markers of $\mathcal{M}$ ($M = \sum_{k \in \mathcal{M}} s_k$) and where the elements of θ^j corresponding to the markers of $\mathcal{M} - \mathcal{B}_j$ are set to zero.

In this model, the expected value Y_a^j of j-th variable for the individual a is

$$\mu_a^j = \mathbb{E}\left(Y_a^j\right) = \mathbf{x}_a \theta^j$$

where $\mathbf{x}_a$ denotes the a-th row of the matrix $\mathbf{X}$.

Model for p variables. To consider p variables, we just have to gather the p models (2) into a single model for $\mathbf{Y} = [\mathbf{Y}^1 \ldots \mathbf{Y}^p]$:

$$\underset{(n,p)}{\mathbf{Y}} = \underset{(n,M+1)}{\mathbf{X}} \underset{(M+1,p)}{\mathbf{\Theta}} + \underset{(n,p)}{\mathbf{E}} \tag{3}$$

where $\mathbf{\Theta} = [\theta^1 \ldots \theta^p]$ and $\mathbf{E} = [\mathbf{E}^1 \ldots \mathbf{E}^p]$.

To take into account the possible dependence of the phenotypic variables, we assume that their common covariance matrix $\mathbf{\Sigma}$ is not necessarily diagonal. The vectors $\mathbf{E}_a = [E_a^1 \ldots E_a^p]$ and $\mathbf{Y}_a = [Y_a^1 \ldots Y_a^p]$ are normally distributed:

$$\mathbf{E}_a \sim \mathcal{N}\left(\mathbf{0}_p; \mathbf{\Sigma}\right), \qquad \mathbf{Y}_a \sim \mathcal{N}\left(\mu_a; \mathbf{\Sigma}\right)$$

where $\mu_a = [\mu_a^1 \ldots \mu_a^p] = \mathbf{x}_a \mathbf{\Theta}$.

In this model, the individuals are assumed to be independent but the variables measured on a same individual are correlated.

2.2 Distance Prediction

The phenotypic distance between two individuals is defined as

$$D^2_{\mathbf{M}}(a,b) = \|\mathbf{Y}_a - \mathbf{Y}_b\|^2_{\mathbf{M}} = (\mathbf{Y}_a - \mathbf{Y}_b)\,\mathbf{M}\,(\mathbf{Y}_a - \mathbf{Y}_b)'$$

where $\mathbf{M}$ is a given metric.

The case of Mahalanobis distance. A particularly interesting distance is the Mahalanobis distance D^2_{Maha} for which $\mathbf{M}$ is the inverse of the covariance matrix of $(\mathbf{Y}_a - \mathbf{Y}_b)$, i.e. $\mathbf{M} = [\mathbb{V}(\mathbf{Y}_a - \mathbf{Y}_b)]^{-1} = \frac{1}{2}\mathbf{\Sigma}^{-1}$ so

$$D^2_{\text{Maha}}(a,b) = \frac{1}{2}(\mathbf{Y}_a - \mathbf{Y}_b)\,\mathbf{\Sigma}^{-1}(\mathbf{Y}_a - \mathbf{Y}_b)'.$$

We can define the expected distance $\delta_{\text{Maha}}(a,b)$ as the distance between the expected values of the $\mathbf{Y}_a$'s:

$$\delta^2_{\text{Maha}}(a,b) = \|\mu_a - \mu_b\|^2_{\frac{1}{2}\mathbf{\Sigma}^{-1}},$$

which can be expressed as

$$\begin{aligned}\delta^2_{\text{Maha}}(a,b) &= \frac{1}{2}(\mathbf{x}_a\mathbf{\Theta} - \mathbf{x}_b\mathbf{\Theta})\,\mathbf{\Sigma}^{-1}(\mathbf{x}_a\mathbf{\Theta} - \mathbf{x}_b\mathbf{\Theta})' \\ &= \frac{1}{2}(\mathbf{x}_a - \mathbf{x}_b)\,\mathbf{\Theta}\mathbf{\Sigma}^{-1}\mathbf{\Theta}'(\mathbf{x}_a - \mathbf{x}_b)' = \|\mathbf{x}_a - \mathbf{x}_b\|^2_{\frac{1}{2}\mathbf{\Theta}\mathbf{\Sigma}^{-1}\mathbf{\Theta}'}.\end{aligned}$$

This distance can be considered as a pseudo genetic distance, since it is only calculated through the genotypes $\mathbf{x}_a$ and $\mathbf{x}_b$, with the metric $\frac{1}{2}\mathbf{\Theta}\mathbf{\Sigma}^{-1}\mathbf{\Theta}'$.

Given the genotypes, $D^2_{Maha}(a,b)$ is random while $\delta^2_{Maha}(a,b)$ is not. It is straightforward that this phenotypic square distance has a non-central chi-square distribution with a non-centrality parameter equal to the square predicted distance:

$$D^2_{\text{Maha}}(a,b) \sim \chi^2_p\left[\delta^2_{\text{Maha}}(a,b)\right]. \tag{4}$$

This is clear since all the vectors $\mathbf{E}_a$ are independent with mean zero and common covariance matrix $\mathbf{\Sigma}$, so $\mathbb{V}(\mathbf{E}_a - \mathbf{E}_b) = 2\mathbf{\Sigma}$. Hence, for any given level $1-\gamma$ we can give a two-sided prediction interval for $D^2_{\text{Maha}}(a,b)$:

$$\left[\chi^2_{p;\gamma/2}\left\{\delta^2_{\text{Maha}}(a,b)\right\}\ ;\ \chi^2_{p;1-\gamma/2}\left\{\delta^2_{\text{Maha}}(a,b)\right\}\right] \tag{5}$$

(where $\chi^2_{p;\beta}(\delta^2)$ is the β-quantile of the non central chi-square distribution) and one-sided interval:

$$\left[\chi^2_{p;\gamma}\left\{\delta^2_{\text{Maha}}(a,b)\right\}\ ;\ \infty\right[.$$

General case. The Mahalanobis distance is not the only distance that can be used, although it is the most convenient. For other distances the following proposition may be established.

Proposition 1 *For a general metric* $\mathbf{M}$*, the distribution of the square distance* D^2 *is a linear combination of non-central chi-squares: using the two diagonal decompositions* $\mathbf{M} = \mathbf{PDP}'$ *and* $\mathbf{D}^{\frac{1}{2}}\mathbf{P}'\mathbf{\Sigma}\mathbf{P}\mathbf{D}^{\frac{1}{2}} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}'$ *where* $\mathbf{P}$ *and* $\mathbf{Q}$ *are orthogonal,* $\mathbf{D}$ *and* $\mathbf{\Lambda}$ *diagonal with* $\mathbf{\Lambda} = \mathbf{diag}\left[\lambda_1, \ldots, \lambda_p\right]$*, one has*

$$D^2_{\mathbf{M}}(a,b) \overset{\mathcal{D}}{=} 2\sum_{j=1}^{p} \lambda_j \xi_j$$

where the ξ_j *are independent,* $\xi_j \sim \chi^2_1\left[(\nu^j)^2/(2\lambda_j)\right]$ *and*

$$\nu = \left[\nu^1 \ \ldots \ \nu^p\right] = (\mu_a - \mu_b)\ \mathbf{P}\mathbf{D}^{\frac{1}{2}}\mathbf{Q}.$$

The proof of this proposition and of the following are given in the appendix. According to proposition **1**, a predicted distance can be defined as the linear combination of the non centrality parameters $(\nu^j)^2$:

$$\delta^2_{\mathbf{M}}(a,b) = 2\sum_{j=1}^{p} \lambda_j \frac{(\nu^j)^2}{2\lambda_j} = \nu\nu' = (\mu_a - \mu_b)\mathbf{M}(\mu_a - \mu_b)'.$$

Prediction intervals can thus be constructed even for a general metric $\mathbf{M}$.

2.3 The case of an unknown parameter

That the parameter Θ is estimated is not taken into account in the previous results and this may induce some bias and uncertainty in the distance prediction. Furthermore, when managing reference collections, this estimation is made on the elements of the collections, but the results may be applied to other individuals. In this section we give some results about the bias and distribution of the predicted distance. The originality of these results lays in the two following points: i) the difference between the training and validation subsets is taken into account, especially in the expression of the predicted vectors $\widehat{\mathbf{Y}}^j$; ii) the linear model is different for each phenotypic variable, which makes a major difference with the classical multivariate linear model.

Parameter estimates. In the following, for each column vector $(n \times 1)$, we shall consider that the first n_1 coordinates correspond to the training set and the remaining n_0 to the validation set $(n_0 + n_1 = n)$. Decomposing $\mathbb{R}^n$ as $\mathbb{R}^n = \mathbb{R}^{n_1} \oplus \mathbb{R}^{n_0}$, we denote by $\mathbf{\Pi}_1$ the $(n \times n)$ projector matrix that sets the last n_0 coordinates to zero and by $\mathbf{\Pi}_0$ the $(n \times n)$ projector setting the first n_1 coordinates to zero.

Since the rank of $\mathbf{\Pi}_1\mathbf{X}_{\mathcal{B}_j}$ is less than $M_j + 1$, we use side conditions to define the generalized inverse $(\mathbf{X}'_{\mathcal{B}_j}\mathbf{\Pi}_1\mathbf{X}_{\mathcal{B}_j})^-$ (see Rao and Mitra (71)). $\widehat{\theta}_{\mathcal{B}_j} = (\mathbf{X}'_{\mathcal{B}_j}\mathbf{\Pi}_1\mathbf{X}_{\mathcal{B}_j})^-\mathbf{X}'_{\mathcal{B}_j}\mathbf{\Pi}_1\mathbf{Y}^j$

is an unbiased estimate of the particular value of $\theta_{\mathcal{B}_j}$ satisfying this conditions, i.e. such as $\theta_{\mathcal{B}_j} = (\mathbf{X}'_{\mathcal{B}_j}\mathbf{\Pi}_1\mathbf{X}_{\mathcal{B}_j})^{-}\mathbf{X}'_{\mathcal{B}_j}\mathbf{\Pi}_1\mathbf{X}_{\mathcal{B}_j}\theta_{\mathcal{B}_j}$ (see Scheffé (59)).

The predicted vector $\widehat{\mathbf{Y}}^j$ is

$$\widehat{\mathbf{Y}}^j = \mathbf{X}_{\mathcal{B}_j}\widehat{\theta}_{\mathcal{B}_j} = \mathbf{A}_{\mathcal{B}_j}\mathbf{\Pi}_1\mathbf{Y}^j \tag{6}$$

where $\mathbf{A}_{\mathcal{B}_j} = \mathbf{X}_{\mathcal{B}_j}(\mathbf{X}'_{\mathcal{B}_j}\mathbf{\Pi}_1\mathbf{X}_{\mathcal{B}_j})^{-}\mathbf{X}'_{\mathcal{B}_j}$. $\widehat{\mathbf{Y}}^j$ is an unbiased estimate of μ^j: $\mathbb{E}(\mathbf{Y}^j) = \mu^j$.

Decomposing $\mathbf{I}_n = \mathbf{\Pi}_1 + \Pi_0$, we get $\widehat{\mathbf{Y}}^j = \mathbf{\Pi}_1\mathbf{A}_{\mathcal{B}_j}\mathbf{\Pi}_1\mathbf{Y}^j + \mathbf{\Pi}_0\mathbf{A}_{\mathcal{B}_j}\mathbf{\Pi}_1\mathbf{Y}^j$. The first term concerns the first n_1 coordinates: $\mathbf{\Pi}_1\mathbf{A}_{\mathcal{B}_j}\mathbf{\Pi}_1$ is the orthogonal projector on the intersection of $\mathcal{B}_j$ and the column space of $\mathbf{\Pi}_1$ and this projection does not depend on the side conditions used to calculate the generalized inverse (see Scheffé (59)). The second element concerns the last n_0 coordinates but is not exactly equivalent since $\Pi_0\mathbf{A}_{\mathcal{B}_j}\mathbf{\Pi}_1$ is not a projector: it gives the predicted values for the validation set, these values depend on the side conditions.

The predicted matrix $\widehat{\mathbf{Y}}$ is

$$\widehat{\mathbf{Y}} = \left[\begin{array}{ccc} \mathbf{A}_{\mathcal{B}_1}\mathbf{\Pi}_1\mathbf{Y}^1 & \dots & \mathbf{A}_{\mathcal{B}_p}\mathbf{\Pi}_1\mathbf{Y}^p \end{array}\right].$$

General case. When $\mathbf{\Theta}$ is unknown, for any pair (a, b), we are able only to calculate an estimate of δ^2 :

$$\widehat{\delta^2}(a,b) = \left\|\widehat{\mathbf{Y}}_a - \widehat{\mathbf{Y}}_b\right\|^2_{\mathbf{M}} = \left(\widehat{\mathbf{Y}}_a - \widehat{\mathbf{Y}}_b\right)\mathbf{M}\left(\widehat{\mathbf{Y}}_a - \widehat{\mathbf{Y}}_b\right)'.$$

Defining the vector $\mathbf{u}_a = [0 \cdots 0\,1\,0 \cdots 0]$ having n coordinates with a one in the a-th column and zero elsewhere, the difference $\widehat{\mathbf{Y}}_a - \widehat{\mathbf{Y}}_b$ is equal to the contrast $(\mathbf{u}_a - \mathbf{u}_b)\,\widehat{\mathbf{Y}}$, so

$$\widehat{\delta^2}(a,b) = (\mathbf{u}_a - \mathbf{u}_b)\,\widehat{\mathbf{Y}}\mathbf{M}\widehat{\mathbf{Y}}'\,(\mathbf{u}_a - \mathbf{u}_b)'.$$

Proposition 2 *The bias of $\widehat{\delta^2}(a,b)$ is positive and equal to*

$$(\mathbf{u}_a - \mathbf{u}_b)\,\mathbf{H}\,(\mathbf{u}_a - \mathbf{u}_b)' = h^a_a + h^b_b - 2h^b_a$$

where the general term h^b_a of the matrix $\mathbf{H}$ is

$$h^b_a = \sum_{j=1}^{p}\sum_{j'=1}^{p}\sigma^{j'}_j m^{j'}_j\left[\mathbf{u}_a\mathbf{A}_{\mathcal{B}_j}\mathbf{\Pi}_1\mathbf{A}_{\mathcal{B}_{j'}}(\mathbf{u}_b)'\right],$$

where σ^j_a and m^j_a are the general terms of $\mathbf{\Sigma}$ and $\mathbf{M}$.

Using Mahalanobis distance with one single model. We shall concentrate now on the case of the Mahalanobis distance ($\mathbf{M} = \frac{1}{2}\mathbf{\Sigma}^{-1}$), in the case where $\mathbf{\Sigma}$ is known. Assuming additionally that the same model is valid for all the phenotypic variables, we give a simpler expression of the bias in corollary **3**.

Corollary 3 *In the particular case of the Mahalanobis distance and if we use the same model for all the* $(Y^j)_{j=1\ldots p}$ *(i.e.* $\mathbf{X}_{\mathcal{B}_j} \equiv \mathbf{X}$*), the bias is*

$$\frac{p}{2}\left(\mathbf{u}_a - \mathbf{u}_b\right)\mathbf{A}\mathbf{\Pi}_1\mathbf{A}\left(\mathbf{u}_a - \mathbf{u}_b\right)'.$$

This corollary can be simplified if the generalized inverse $(\mathbf{X}'\mathbf{\Pi}_1\mathbf{X})^-$ is reflexive (see Rao and Mitra (71)): in this case $\mathbf{A}\mathbf{\Pi}_1\mathbf{A} = \mathbf{A}$.

Although in practice, both $\mathbf{\Theta}$ and $\mathbf{\Sigma}$ are estimated, the fact that $\mathbf{\Sigma}$ is unknown is not taken into account here. At the present time, we propose to 'plug' in $\widehat{\mathbf{\Sigma}}$ in place of $\mathbf{\Sigma}$ in all the formulae given here.

3 Distinctness procedure

3.1 Decision rule

The distinctness between two lines is generally determined according to a threshold t on the phenotypic distance:

$$\{\text{lines } a \text{ and } b \text{ are distinct}\} \Leftrightarrow \left\{D^2(a,b) > t\right\}.$$

Formula (5) gives a prediction interval for d^2 with level $1-\gamma$. This result can be applied to predict whether lines a and b are distinct or not according to the following decision rule:

- if $\chi^2_{p;\gamma/2}\left[\delta^2(a,b)\right] > t$, the whole prediction interval on D^2 is above the threshold t: lines a and b are predicted to be distinct considering their genotypes;
- if $\chi^2_{p;1-\gamma/2}\left[\delta^2(a,b)\right] < t$, the whole prediction interval on D^2 is beyond the threshold t: lines a and b are predicted to be similar considering their genotypes;
- if $\chi^2_{p;\gamma/2}\left[\delta^2(a,b)\right] < t < \chi^2_{p;1-\gamma/2}\left[\delta^2(a,b)\right]$, the threshold t is included in the prediction interval: no decision can be made on the basis of the simple genotypes so an experiment has to be made to measure the phenotypic traits.

This rule is summarized in figure **2**. The molecular thresholds b_{inf} and b_{sup} are given by

$$t = \chi^2_{p;1-\gamma/2}(b_{\text{inf}}) = \chi^2_{p;\gamma/2}(b_{\text{sup}}).$$

Errors. The decisions made according to molecular data can obviously lead to some errors. Let us consider the cases corresponding to the three vertical zones of figure **2**.

(1) $\delta^2(a,b) < b_{\text{inf}}$ (a and b predicted similar):
if $D^2(a,b) < t$ then a and b are effectively similar, the decision is correct;
if $D^2(a,b) > t$ then a and b are actually distinct, the decision is wrong.

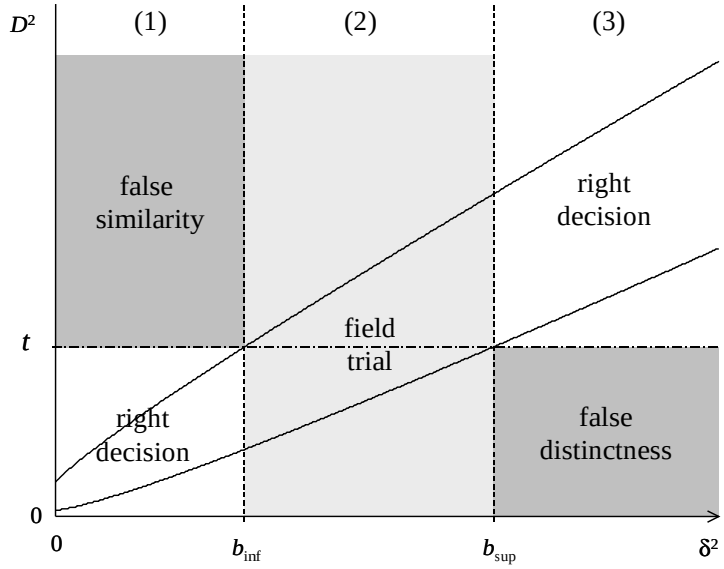


Figure 2: Decision rule (solid lines = $\chi^2_{p;\gamma/2}(\delta^2)$ and $\chi^2_{p;1-\gamma/2}(\delta^2)$)

(2) $b_{\text{inf}} < \delta^2(a,b) < b_{\text{sup}}$: no error can be made since there is no decision; the experiment will show whether a and b are distinct or not.

(3) $\delta^2(a,b) > b_{\text{sup}}$ is the symmetrical case of **(1)**: if $D^2(a,b) > t$ then the decision is correct, else we are misled.

3.2 Proposition of new distinctness procedure

Distinctness. The aim of a distinctness procedure is to decide whether candidates lines (set $\mathcal{C}$) are distinct from those of a reference collection (set $\mathcal{R}$). A candidate line c of $\mathcal{C}$ is distinct from the collection $\mathcal{R}$ if it is distinct from all the elements r of $\mathcal{R}$:

$$\{c \text{ distinct from } \mathcal{R}\} \Leftrightarrow \left\{\forall r \in \mathcal{R} :\ D^2(r,c) > t\right\}. \tag{7}$$

On the contrary c is said to be similar to $\mathcal{R}$ as soon as it resembles one of the elements of $\mathcal{R}$:

$$\{c \text{ similar to } \mathcal{R}\} \Leftrightarrow \left\{\exists r \in \mathcal{R} :\ D^2(r,c) < t\right\}.$$

Savings. As plant breeders need a phenotypic evaluation of their lines when proposing a new variety for registration, all candidate lines have to be set on trial. Our wish is to avoid a part of the experiments for the reference lines by using the predicted distance (through molecular data) rather than the observed one.

For a reference line r, we can avoid experimentation if a decision can be made for all the molecular comparisons between r and all the candidate lines. Let us denote by $\mathcal{S}$ (for

savings) the set of the reference lines which do not need to be set on trial ($\mathcal{S} \subset \mathcal{R}$):

$$\mathcal{S} = \left\{ r \in \mathcal{R} : \ \forall c \in \mathcal{C},\, \delta^2(r,c) \notin \left]b_{\text{inf}}, b_{\text{sup}}\right[\right\}. \tag{8}$$

False distinctness. A candidate line c can be predicted to be distinct from $\mathcal{R}$ while it is not. This happens when all the following conditions are fulfilled:

D1 c is similar to $\mathcal{R}$: $\exists r \in \mathcal{R},\ D^2(r,c) < t$;

D2 c is not predicted to be similar considering molecular data: $\forall r \in \mathcal{R},\ \delta^2(r,c) > b_{\text{inf}}$;

D3 c is distinct from all the experimented reference lines: $\forall r \in \mathcal{R} - \mathcal{S},\ D^2(r,c) > t$;

D4 none of the errors is corrected by any experiment: $\forall r \in \mathcal{R},\ \delta^2(r,c) > b_{\text{sup}}$ and $D^2(r,c) < t : \ r \in \mathcal{S}$.

The point (**D4**) has some major (and favourable) consequences. It recalls that a candidate line is compared not only to the reference lines from which it is close according to the molecular data, but also to all the reference lines that shall be trialed. If the candidate line c is near the reference line r but these two lines are not predicted as similar, the final error may be avoided if r is predicted to be similar to another candidate c', and therefore, trialed. In this case, multiple candidate lines can reduce savings but avoid some false distinctnesses.

False similarity. A candidate line c can be predicted to be similar to $\mathcal{R}$ while it is actually distinct. This happens when both the following conditions are fulfilled:

S1 c is distinct from $\mathcal{R} : \forall r \in \mathcal{R},\ D^2(r,c) > t$;

S2 at least one decision concerning c is wrong and the concerned reference line is put aside: $\exists r \in \mathcal{R} : \ \delta^2(r,c) < b_{\text{inf}},\ r \in \mathcal{S}$.

In the context of DUS testing, false similarities will never arise because no similarity decision can be made on the basis of single molecular data. One reason is that such an error is much more likely to occur than false distinctness; another reason is that the reference collection has to contain the widest possible biodiversity so it seems preferable to accept some irregular elements (that is to make a conservative choice) rather than to reject original lines.

This paper will therefore focus on the upper bound b_{sup}. In the following, we shall assume that b_{inf} is equal to zero. This is equivalent to applying the decision rule of section 3.1 with an unilateral prediction interval $\left[\chi^2_{p;\gamma}\left(\delta^2\right); \infty\right]$.

4 Application to varietal distinctness

4.1 Material

These results have been applied to $n = 144$ maize inbred lines, representative of material released in France, described using both Restriction Fragment Length Polymorphism (RFLP) markers and $p = 10$ discriminant morphological quantitative traits (Bar-Hen & Charcosset (95)). This genetical material, derived from the maize germplasms corresponding to the different groups cultivated in Europe, is coded to ensure the confidentiality of the results. 100 probes were chosen according to the quality of the hybridization signal and the genome coverage. DNA extracted from 15 young plants was digested with three restriction enzymes, giving rise to 3×100 Enzyme × Probe Combinations (EPC). Among these 300 EPCs, only 222 provided polymorphic and interpretable profiles corresponding to 95 probes, giving 1098 bands. In order to avoid redundancy, only $m = 80$ monolocus EPCs corresponding to different loci, were selected (see Dillmann et al. (97) for more details).

The ten morphological quantitative traits were observed over four years (from 1989 to 1992) in three sites, with two replications per site following a block design. The experimental design is unbalanced because sites were chosen for each genotype according to its earliness. Most of the inbred lines were evaluated in at least two locations for at least two years, but some of them were evaluated for only one year in two locations. An estimate of each phenotypic effect was given by the residuals of the analysis of variance performed on each trait with main effects: year, location, block and all two-way interaction terms: year×location, year×block. The quantitative traits concerned ear (length, diameter of ear, diameter of cob), number of rows of seeds, plant (height of ear, total plant length), leaf (width of blades) and tassel (length of main axis above lowest side branch, length of main axis above highest side branch, date of male flowering). See Bar-Hen et al. (95) for more details.

For each phenotypic variable, the set $\mathcal{B}_j$ of the relevant markers has been selected using a stepwise procedure using a subset of $n_1 = 100$ training lines of maize (the $n_0 = 44$ remaining lines were used to validate the model). This procedure uses a Fisher test with level α to enter or remove markers from the model. In the following, we shall discuss the choice of the selection level α.

Throughout this application we use Mahalanobis distance in order to take into account the correlations between the phenotypic traits. Furthermore, the estimate $\widehat{\mathbf{\Theta}}$ is calculated using the Moore-Penrose generalized inverse (see Rao and Mitra (71)) performed with the MatLab software: this choice has no influence on the predicted value for the 100 training lines but, as said in section **2.3**, changes the predictions for the validation set.

4.2 Distance prediction

Using the usual $\alpha = 5\%$ level for selecting the relevant markers for each phenotypic variable, we obtain very well-fitted predictions of the phenotypic distances, as shown in

figure **3**. It is clear that the triangular shape disappears: the distribution of the observed distance D is concentrated in the prediction interval. Figure **1** plots the Mahalanobis distance versus the Rogers' (molecular) distance for the $n_1 (n_1 - 1)/2 = 4950$ pairs of training lines. The comparison of figures **1** and **3** shows the improvement of the prediction using δ rather than the Rogers' distance: we obtain an almost linear relation instead of a triangular one. Note that all the points have the same y-value on both graphs; only the x-values differ.

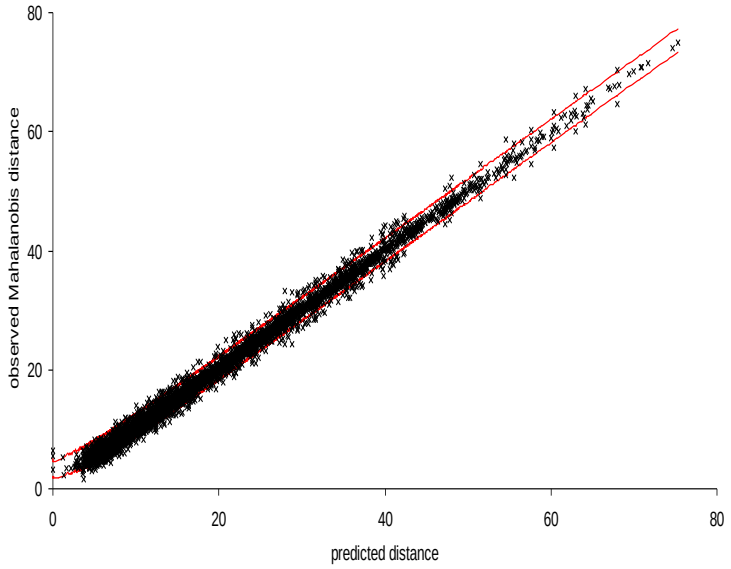


Figure 3: Plot of the Mahalanobis distance D versus the (uncorrected) predicted distance with a selection level $\alpha = 5\%$, for couples of training lines. The solid lines give the 95% prediction interval.

Robustness. In practice, this method should be used to compare lines of a reference collection with new lines. In this framework, the reference collection would be the training set and the new lines could be considered as a validation set since they would not be used to estimate $\boldsymbol{\Theta}$. Hence it is necessary to study the quality of the predictions when considering pairs involving both training and validation lines.

Unfortunately, as shown in table **1**, the quality of the predictions with a selection of the markers at level $\alpha = 5\%$ is very disappointing for such pairs: the observed confidence is 11.9% for a 95% prediction interval. As discussed in Nuel (98), this selection level is not severe enough and seems to select poorly relevant markers and induces an over-fitting of the model; these selected markers strongly depend on the training set and give very inaccurate predictions for the validation set.

However, a great effort is made by many genetic research teams to detect QTL for traits having an agronomical interest; some substantial progresses can also be expected

selection	predicted distance		corrected predicted distance	
level	training couples	mixed couples	training couples	mixed couples
5%	93.0%	11.9%	93.2%	12.3%
1%	93.9%	76.0%	94.6%	77.7%

Table 1: Observed confidence for two selection levels. The expected confidence is 95%. For each level and distance, the first column concerns couples of training lines and the second one concerns mixed couples (one training line and one validation line).

using expressed sequences tags (EST). Therefore, we can soon expect to get more relevant markers than those used in this example. To try to see what kind of performance could be obtained with such markers, we apply a more severe selection level ($\alpha = 1\%$). In this case, the observed confidence (set table **1**) raises to 76.0% for such pairs (and to 93.9% for pairs of training lines). This results allow us to be optimistic about the performances of the method when used with more accurate predictors such as QTL.

Correction of the bias. As shown in section **2.3**, the fact that $\boldsymbol{\Theta}$ has to be estimated after the data, induces that $\widehat{\delta}^2$ is biased. Figure **4** presents the plots of the observed distance versus the uncorrected and the corrected predicted distance. The bias is calculated according to proposition **2**; when the correction happens to give a negative distance, it is set arbitrarily to zero. This correction shifts the set of points slightly to the left and induces a small improvement of the observed confidence which increases of 0.7% (see table **1**), corresponding to 35 pairs and to 1.7% for the pairs containing both validation and training lines.

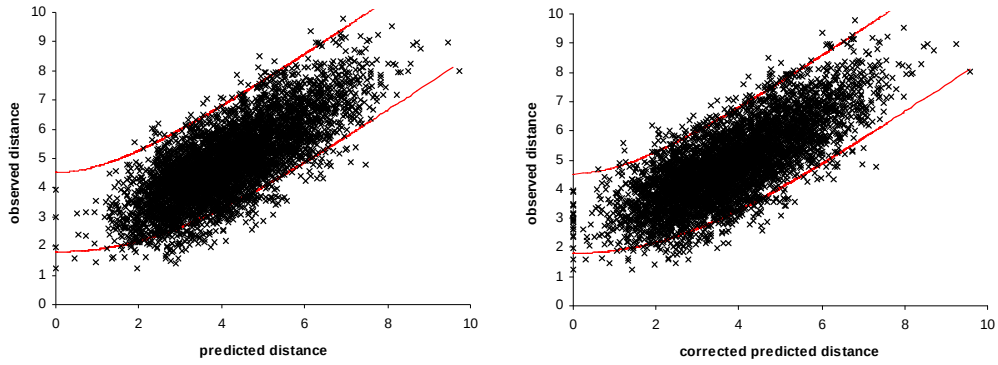


Figure 4: Plot of the Mahalanobis distance D versus the uncorrected predicted distance (left hand side) and corrected predicted distance (right hand side) with a selection level $\alpha = 1\%$, for couples of training lines. The solid lines give the 95% prediction interval.

It has to be pointed out that the calculation of the bias is not so easy with ordinary statistical softwares. Although the computation of the predicted distance involves no particular difficulties, the bias requires some special programming. In our case, the correction of the bias does not induce a great improvement so this effort may seem to be rather futile. Nevertheless it is not clear that this is a general result.

4.3 DUS

To analyze the performances of the distinctness procedure we are proposing in section **3.2**, we simulate several DUS sessions with a varying candidate set $\mathcal{C}$ and a constant reference collection $\mathcal{R}$. As said in section **1**, at the present time, the number of candidate lines represent 20% of the size of the reference collection. In order to respect this proportion, we sample 20 lines among the 44, for each simulation (the total number of possible subsets would be $C_{44}^{20} \simeq 2 \times 10^{12}$).

Phenotypic threshold t. To determine the phenotypic threshold above which two lines are deemed to be phenotypically distinct, we sample 10 000 sets $\mathcal{C}$; for each sample, we calculate t so that the acceptance rate (according to formula (7)) is 70%, as indicated in section **1**. The final threshold is the mean of these 10 000 values.

False distinctness and savings. The choice of the confidence level γ is a crucial step in the procedure: for a given phenotypic threshold t, the molecular threshold b_{sup}, defined by $t = \chi^2_{p;\gamma}(b_{\text{sup}})$, is a decreasing function of γ. Thus, a small value of γ will provide a high value of b_{sup} and, therefore, a large field trial zone. In this case, we expect very few errors, and savings. Contrariwise, a high confidence level γ will lead to numerous savings and errors.

To optimize the procedure, we consider all the values of γ from 0.5% to 10% by 0.5%. For computational time reasons, we only use 1000 samples to estimate the number of false distinctnesses and savings for each value of γ. These quantities are estimated and not directly calculated because the combinatorial problem seems really intricate.

Results. Figure **5** shows the evolution of the number of false distinctnesses and of the percentage of savings as the confidence level γ grows. We observe that both of these quantities increase step by step; this is due to the limited size of the reference collection ($n_1 = 100$). Since the lower bound b_{inf} is set to zero, the definition 8 can be rewritten as

$$\mathcal{S} = \left\{ r \in \mathcal{R} : \inf_{c \in \mathcal{C}} \left[\delta^2 (r, c) \right] > b_{\text{sup}} \right\}.$$

The set $\mathcal{S}$ changes only when the bound b_{sup} becomes lower than one of 100 different values of $\inf_{c \in \mathcal{C}} [\delta^2 (r, c)]$. The same kind of mechanism governs the number of false distinctnesses.

To choose the level γ, we set a maximum acceptable number false distinctnesses. If we are ready only to accept one error among the 20 candidate lines, we see on the left

side of figure **5** that the corresponding level is $\gamma = 1.5\%$ which provides 29% of savings (right side of figure **5**). If we accept a slighlty larger error, say 1.2, we get $\gamma = 2.5\%$ and 40%

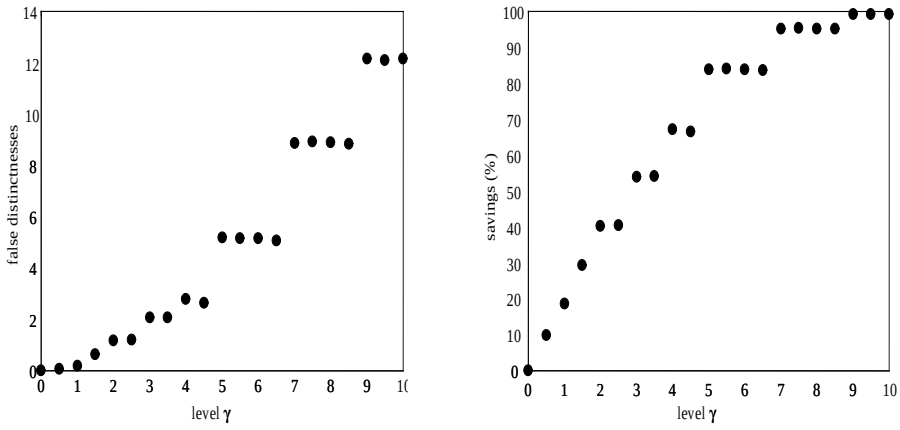


Figure 5: Evolution of the number of false distinctnesses (left) and of the percentage of savings (right) as functions of the confidence level γ (%).

5 Discussion and Conclusion

One could note that the choice of a linear model, the residuals of which provide an estimation of phenotypic values, is fundamental. It gives the weighting matrix used for the calculation of phenotypic distances. When the interaction terms are significant, it could be further adapted to use as many traits as there are phenotypic traits $\times$ environments combinations. This would increase the number of variables but would probably positively affect the quality of the phenotypic distance estimation.

The distinctness of varieties exclusively based on phenotypical data being both area and time consuming, necessarily leads to compromises between the number of observed varieties and the quality of experimental design, because of the difficulty of mastering all the environmental variation factors. The present model provides a valuable tool for concentrating the field work on the most difficult evaluations. Conservatively, one could avoid 29% of the trials making at most one false distinctness. Furthermore, the use of pseudo genetic distance can be of great help in optimising the experimental design: pairs of lines having a small predicted distance should be trialed in neighbouring plots.

The perspective for this method are as follows:

- Substantial improvements can be expected soon, thanks to important progress made in the localization of genes coding for agronomical traits, providing more relevant markers.

- This approach has to be validated on more data: currently it is being applied to 83 rapeseed varieties described with 324 amplified fragment length polymorphism (AFLP) markers and 10 phenotypic traits (see Lombard et al. (00)).
- Finally, considering the importance of discrete variables noted in the trials, the generalization of this approach to qualitative traits would be of great interest.

Appendix: Proofs

Proposition 1. This can be proved using the same kind of argument as Cochran's theorem (see for example Scheffé (59)). Since $\mathbf{M}$ is a metric, it is defined and positive, we hence decompose it as $\mathbf{M} = \mathbf{PDP}'$. We obtain

$$D^2_{\mathbf{M}}(a,b) = (\mathbf{Z}_a - \mathbf{Z}_b)(\mathbf{Z}_a - \mathbf{Z}_b)',$$

where $\mathbf{Z}_a = \mathbf{Y}_a\mathbf{P}\mathbf{D}^{\frac{1}{2}}$. Since $\mathbf{Y}_a$ is normal with mean μ_a and variance $\mathbf{\Sigma}$, the distribution of $\mathbf{Z}_a$ is

$$\mathbf{Z}_a \sim \mathcal{N}_p\left(\mu_a\mathbf{P}\mathbf{D}^{\frac{1}{2}};\ \mathbf{D}^{\frac{1}{2}}\mathbf{P}'\mathbf{\Sigma}\mathbf{P}\mathbf{D}^{\frac{1}{2}}\right).$$

Furthermore, since $\mathbf{\Sigma}$ is positive, one has $\lambda_j \geq 0$ for all $1 \leq j \leq p$. The variance of the transformed variable $\mathbf{T}_a = \mathbf{Z}_a\mathbf{Q} = \mathbf{Y}_a\mathbf{P}\mathbf{D}^{\frac{1}{2}}\mathbf{Q}$ is diagonal: $\mathbb{V}(\mathbf{T}_a) = \mathbf{Q}'\mathbf{D}^{\frac{1}{2}}\mathbf{P}'\mathbf{\Sigma}\mathbf{P}\mathbf{D}^{\frac{1}{2}}\mathbf{Q} = \mathbf{Q}'\mathbf{Q}\mathbf{\Lambda}\mathbf{Q}'\mathbf{Q} = \mathbf{\Lambda}$, so

$$\mathbf{T}_a \sim \mathcal{N}_p\left(\mu_a\mathbf{P}\mathbf{D}^{\frac{1}{2}}\mathbf{Q};\ \mathbf{\Lambda}\right)$$

and $D^2_{\mathbf{M}}(a,b) = (\mathbf{T}_a - \mathbf{T}_b)(\mathbf{T}_a - \mathbf{T}_b)'$.

Hence we have shown that $D^2_{\mathbf{M}}(a,b)$ is the squared norm of a normal vector with mean $\nu = (\mu_a - \mu_b)\mathbf{P}\mathbf{D}^{\frac{1}{2}}\mathbf{Q} = [\nu^1\ \ldots\ \nu^p]$ and independent coordinates with respective variances $2\lambda_j$, so the proposition is proved. ■

Proposition 2. One has $\mathbb{E}[\widehat{\delta}^2(a,b)] = (\mathbf{u}_a - \mathbf{u}_b)\,\mathbb{E}(\widehat{\mathbf{Y}}\mathbf{M}\widehat{\mathbf{Y}}')(\mathbf{u}_a - \mathbf{u}_b)'$ and, according to equation (6) and model (1),

$$\widehat{\mathbf{Y}}^j = \mathbf{A}_{\mathcal{B}_j}\mathbf{\Pi}_1\mathbf{Y}^j = \mathbf{A}_{\mathcal{B}_j}\mathbf{\Pi}_1\left(\mathbf{X}_{\mathcal{B}_j}\theta_{\mathcal{B}_j} + \mathbf{E}^j\right) = \mu^j + \mathbf{A}_{\mathcal{B}_j}\mathbf{\Pi}_1\mathbf{E}^j.$$

So $\widehat{\mathbf{Y}}\mathbf{M}\widehat{\mathbf{Y}}'$ can be decomposed as

$$\widehat{\mathbf{Y}}\mathbf{M}\widehat{\mathbf{Y}}' = \mu\mathbf{M}\mu' + \mathbf{2}\mu\mathbf{M}\mathbf{G} + \mathbf{G}\mathbf{M}\mathbf{G}'$$

where $\mu = [\mu^1\ \ldots\ \mu^p]$ and $\mathbf{G} = \left[\mathbf{A}_{\mathcal{B}_1}\mathbf{\Pi}_1\mathbf{E}^1\ \ldots\ \mathbf{A}_{\mathcal{B}_p}\mathbf{\Pi}_1\mathbf{E}^p\right]$; its mean is $\mathbb{E}(\widehat{\mathbf{Y}}\mathbf{M}\widehat{\mathbf{Y}}') = \mu\mathbf{M}\mu' + 2\mu\mathbf{M}\mathbb{E}(\mathbf{G}) + \mathbb{E}(\mathbf{G}\mathbf{M}\mathbf{G}')$. Since each row $\mathbf{E}$ has a normal distribution with mean $\mathbf{0}$, $\mathbb{E}(\mathbf{G})$ is null, so $\mathbb{E}[\widehat{\delta}^2(a,b)]$ is equal to

$$(\mathbf{u}_a - \mathbf{u}_b)\,\mu\mathbf{M}\mu'\,(\mathbf{u}_a - \mathbf{u}_b)' + \mathbb{E}\left[(\mathbf{u}_a - \mathbf{u}_b)\,\mathbf{G}\mathbf{M}\mathbf{G}'\,(\mathbf{u}_a - \mathbf{u}_b)'\right]$$

where the first term equals $\delta^2(a,b)$ and where the second term is the expectation of a square distance and is hence positive.

The term (a,b) of $\mathbf{GMG}'$ equals $\sum_{j=1}^{p}\sum_{j'=1}^{p} m_j^{j'} \times G_a^j G_b^{j'}$. Denoting the general term of $\mathbf{G}$ by G_a^j, one has $G_a^j = \mathbf{u}_a \mathbf{A}_{\mathcal{B}_j} \mathbf{\Pi}_1 \mathbf{E}^j$ so

$$\begin{aligned}\mathbb{E}\left(G_a^j G_b^{j'}\right) &= \mathbf{u}_a \mathbf{A}_{\mathcal{B}_j} \mathbf{\Pi}_1 \mathbb{E}\left[\mathbf{E}^j \left(\mathbf{E}^{j'}\right)'\right] \mathbf{\Pi}_1 \mathbf{A}_{\mathcal{B}_{j'}} (\mathbf{u}_b)' \\ &= \sigma_j^{j'} \left[\mathbf{u}_a \mathbf{A}_{\mathcal{B}_j} \mathbf{\Pi}_1 \mathbf{A}_{\mathcal{B}_{j'}} (\mathbf{u}_b)'\right].\end{aligned}$$

The expectation of $\mathbf{GMG}'$ gives the result. ■

Corollary 3. Since all the projectors are equal, one has $\mathbf{A}_{\mathcal{B}_j} \mathbf{\Pi}_1 \mathbf{A}_{\mathcal{B}_{j'}} = \mathbf{A}\mathbf{\Pi}_1\mathbf{A}$. Applying proposition **2**, we get

$$h_a^b = \left[\mathbf{u}_a \mathbf{A}\mathbf{\Pi}_1\mathbf{A} (\mathbf{u}_b)'\right] \sum_{j=1}^{p}\sum_{j'=1}^{p} \sigma_j^{j'} m_j^{j'}.$$

Furthermore, in the case of the Mahalanobis distance, the metric $\mathbf{M}$ is $\frac{1}{2}\Sigma^{-1}$, so $\mathbf{\Sigma M} = \frac{1}{2}\mathbf{I}_p$, hence $\sum_{j=1}^{p}\sum_{j'=1}^{p} \sigma_j^{j'} m_j^{j'} = \frac{p}{2}$ and the corollary is proved. ■

REFERENCES

Bar-Hen, A. & Charcosset, A. (1995) Relationship between molecular and morphological distances in a maize inbred lines collection. Application for breeder's right protection. *In*: Van Ooijen J.W., Jansen J. (Ed.), *Biometrics in Plant Breedings: Application of Molecular Markers* (Proc. of the 9th Meet. Eucarpia, Sect. Biometrics Plant Breed.), pp. 57–66 (CPRO-DLO, Wageningen).

Bar-Hen A., Charcosset A., Bourgoin M., Guiard J. (1995) Relationship between genetic markers and morphological traits in a maize lines collection, *Euphytica*, 84, pp. 145–154.

Burstin J. & Charcosset A. (1997) Relationship phenotypic and markers distances: theoretical and experimental investigations, *Heredity*, 79 (5), pp. 477–483.

Dillmann C., Bar-Hen A., Guerin D., Charcosset A., Murigneux A. (1997) Comparison of RFLP and morhological distances between maize *Zea Mays L.* inbred lines. Consequences for germplasm protection purposes. *Theoretical and Applied Genetics*, 95, pp. 92–102.

Gower, J.C. (1985) Measures of similarity and distance. *In*: Kotz, S., Johnson, N.L., (Ed.) *Encyclopedia of statistical sciences*, vol. 5, pp. 397–405 (Wiley: New-York).

Lombard, V., Baril, C.P., Dubreuil, P., Zhang, D. (2000) Genetic relationships and fingerprinting of rapeseed cultivars using AFLP: consequences for varietal registration. *Crop Science* (*in press*).

Mahalanobis P.C. (1936) On the generalized distances in statistics. *Proceedings of the National Institute for Science, India*, 2, pp. 49–55.

MatLab, Version 5.2, *The Mathworks inc.*

Nuel, G. (1998). Prédiction de Distances Phénotypiques à partir de Données Moléculaires pour la Distinction Variétale. Master thesis, INA-PG − Paris XI university (Paris).

Rao C. R. & Mitra S. K. (1971) Generalized inverse of matrices and its applications (New York: Wiley).

Rogers J.S. (1972) Measures of similarity and genetic distance, *Studies in Genetics*, VII: 7213, pp. 145–153 (Univ Texas Publ).

Scheffe, H. (1959) The Analysis of Variance (New York: Wiley).

Chapitre 19

AMIGene : Annotation of MIcrobial Genes

Article de Bocs, Cruveiller, Vellenet, Nuel, et Medigue publié en 2003 dans *Nucleic Acids Research*. AMIGene est un système expert pour l'annotation des génomes microbiens. Il utilise un grand nombre de sources hétérogènes (prédiction de RBS, longueurs des ORF dans les différentes phases, données d'alignement, segmentations par HMM, etc ...) dont il fait la synthèse à travers une suite de règles de décisions mimant l'expertise humaine. Beaucoup de ces règles de décisions dépendent de paramètres qu'il faut calibrer. C'est sur ce dernier point que porte ma contribution personnelle à ce travail.

Nucleic Acids Research, 2003, Vol. 31, No. 13 ***3723–3726***
DOI: 10.1093/nar/gkg590

AMIGene: Annotation of MIcrobial Genes

Stéphanie Bocs, Stéphane Cruveiller, David Vallenet, Grégory Nuel[1] and Claudine Médigue*

Génoscope/UMR-CNRS 8030, Atelier de Génomique Comparative, 2 rue Gaston Crémieux, F-91006 Evry and [1]Laboratoire Statistique et Génomes, UMR-CNRS 8071, Tour Evry2, 523 place des terrasses de l'Agora, F-91034 Evry, France

Received February 17, 2003; Revised and Accepted April 4, 2003

ABSTRACT

AMIGene (Annotation of MIcrobial Genes) is an application for automatically identifying the most likely coding sequences (CDSs) in a large contig or a complete bacterial genome sequence. The first step in AMIGene is dedicated to the construction of Markov models that fit the input genomic data (i.e. the gene model), followed by the combination of well-known gene-finding methods and an heuristic approach for the selection of the most likely CDSs. The web interface allows the user to select one or several gene models applied to the analysis of the input sequence by the AMIGene program and to visualize the list of predicted CDSs graphically and in a downloadable text format. The AMIGene web site is accessible at the following address: http://www.genoscope.cns.fr/agc/tools/amigene/index.html (Contact: sbocs@genoscope.cns.fr).

INTRODUCTION

Intrinsic methods for predicting coding regions extract information on gene locations using statistical patterns inside and outside gene regions as well as patterns typical of the gene boundaries. Several highly accurate prokaryotic gene-finding methods are based on Markov model algorithms [i.e. GeneMark (1) and Glimmer (2)]. The accuracy of these systems depends on models of protein coding regions (and non-coding regions in the case of GeneMark) derived either from experimentally validated training sets or from large amounts of anonymous DNA sequences. Identification of putative genes is followed by an examination of overlaps between selected ORFs (open reading frames) in order to eliminate doubtful candidates. In the context of bacterial genome annotation, we have intensively used these methods and compared their results using the graphical interface provided by Imagene (3). This interface allows one to superimpose results obtained by different strategies and/or gene-finding models and is very useful for pinpointing interesting features such as the coding sequences (CDSs) located at positions in which the coding prediction is good. We have noticed that although most of the predicted genes are identical, the Glimmer method tends to select additional 'suspect' CDSs (false positives). We also found examples in which Glimmer proposed a sequence on the opposite strand of the GeneMark prediction (4). Conversely, many short genes seem not to be identified by the GeneMark method and genes which are 'atypical' in their pattern of codon usage (compared to the average codon bias of the genome) could be missed (false negatives).

Using appropriate gene models with a coding prediction program such as GeneMark (1), the CDS selection is manually performed by keeping the longest CDSs which have a good coding prediction, a minimum overlap with adjacent CDSs (except in the case of frameshift detection) and maximum coverage of the nucleic sequence. These observations led us to mimic the behaviour of the expert in the AMIGene method to automatically identify the most likely CDSs in a large contig or a complete bacterial genome. Although AMIGene remains relatively similar to most existing gene finding systems, it is able to give more accurate predictions in some cases (4–6). The web interface described in this paper allows the user to run AMIGene on a raw DNA sequence, either with suitable gene models we have previously defined on several bacterial genomes, or with a new gene model computed from the user's input genomic data (http://www.genoscope.cns.fr/agc/tools/amigene/index.html).

METHODS

Generating the models for gene-finding

Running the AMIGene method requires the construction of Markov models that fit well with the input genomic data. A preliminary and essential step for a new genome annotation (anonymous DNA sequence) or a re-annotation process of an available prokaryotic genome (5) consists in the construction of appropriate gene models. To achieve this goal, two programs similar to the ones of M. Borodovsky [MakeMat (unpublished) and GeneMark (1)] have been developed (A. Viari, personal communication): (i) *prokov-learn* which uses inhomogeneous three-periodic Markov chain models of protein-coding regions along with ordinary Markov models of non-coding DNA

*To whom correspondence should be addressed. Tel: +33 1 60 87 84 59; Fax: +33 1 60 87 25 14; Email: cmedigue@genoscope.cns.fr

sequences to build gene models; (ii) *prokov-curve* that incorporates these models into a Bayesian algorithm and analyzes DNA sequences locally within a sliding window (1).

The first procedure thus requires as input a set of predicted CDSs or a set of previously annotated genes. In the case of available complete bacterial genomes, the set of annotated genes has been extracted from the International Nucleotide Sequence Database (INSD: DDBJ/EMBL-EBI/GenBank). In the case of an anonymous bacterial sequence, we first search for the longest ORFs and the *prokov-learn* program is used to determine parameters of the Markov model of the protein-coding region (pre-matrix). Depending on the length of the input DNA sequence, the order of the Markov model is equal to two, three or four. For a Markov model of a non-coding sequence, we used the zero order model with four probability parameters estimated by genome specific frequencies of mono-nucleotides. These pre-matrices are then used in the AMIGene program for predicting coding regions in the original genomic sequence (set of predicted CDSs).

However, when the models are constructed by training on the bulk set of protein-coding genes, programs such as GeneMark (1) and *Prokov-curve* become insensitive to genes of minor inhomogeneity classes. For the *Escherichia coli* genome, whose genes have been divided into three classes that differ in codon usage pattern (7), class-specific models of protein-coding regions have improved the performance of the GeneMark method (8). Thus, our group systematically investigates codon usage differences using the multivariate statistical technique of factorial correspondence analysis (FCA) to identify major trends within the data set, i.e. annotated genes or predicted CDSs (9). The *k*-means clustering algorithm is also employed on relative synonymous codon usage (RSCU) values of the coding sequences (10), k being equal to 2, 3 or 4, depending on the major trend of the codon usage bias. For example, $k=2$ for bias due to genes lying on the leading versus lagging strands in the bacterial chromosome (11,12) and $k=3$ in the case of bacterial genomes for which recent horizontal gene transfer has occurred (7,13). The gene classes define the training sets for protein-coding regions and the rest of the sequence is included in the non-coding training set. Corresponding gene models are generated by the *prokov-learn* program (two, three or four in total) and subsequently used in the core of the AMIGene method. Several illustrations of this step are given on the AMIGene web site (http://www.genoscope.cns.fr/agc/tools/amigene/html/Method.html#1).

AMIGene: an heuristic to select the most likely CDSs

Given the sequence of a complete genome, we first look for the maximal CDSs, i.e. maximal segments in-frame between start and stop codons in the six reading frames. The putative CDSs >60 bp are retained. Then, the *Prokov-curve* method (A.Viari and J.Romanet, personal communication) uses the specific models built for the gene classes of the genome in parallel (see above), in order to compute the average coding probability of each identified CDS. AMIGene indicates the model (matrix 1, 2 or 3) that fits best in terms of coding probability. In order to accelerate the following steps, CDSs are first selected according to their coding probability. Two probability thresholds, *prob-PC* and *sure-PC*, are defined (Table 1). If the value of the coding probability is: (i) below the *prob-PC* threshold, the corresponding CDS is eliminated; (ii) between the *prob-PC* and *sure-PC* thresholds, the corresponding CDS is stored in a list containing the probable CDSs if its length is longer than the *Prob-LMin* threshold (Table 1); and (iii) above the *sure-PC* threshold, the corresponding CDS is stored in a list containing the sure CDSs. Therefore we defined an heuristic to select CDSs in order to take into account ambiguous choices between two overlapping CDSs and/or the presence of frameshifts in the DNA sequence. In order to avoid overlaps generated by the choice of the left-most start codon, AMIGene searches, when necessary, for an alternative start codon fitting well with the beginning of the coding prediction curve. Then the selection of the most likely CDSs consists of the elimination of false positives according to overlapping criteria between adjacent CDSs (AMIGene parameters are listed in Table 1); these overlaps may be either total (inclusion) or partial. The AMIGene method is divided into three main steps which are precisely described on our web site (http://www.genoscope.cns.fr/agc/tools/amigene/html/Method.html#2).

Setting the threshold of the parameter values

The threshold values used in the AMIGene method were first determined empirically, based on the examination of results obtained with several AMIGene runs on various bacterial genomes. Then a statistical validation of the chosen threshold values was performed using curated annotations from three bacterial genomes: *E.coli* K12 (50.8% GC-content), *Bacillus subtilis* 168 (43.5% GC-content) and *Mycobacterium tuberculosis* H37Rv (65.6% GC-content). *E.coli* annotated genes were extracted from the last update of the EcoGene data base (14) and we have used curated data from the SubtiList database (15) and from the TubercuList database (16). For each genome, we first defined three gene classes according to their codon usage (see below), which were subsequently used to build three gene models. Predictions made by the AMIGene program were compared to the corresponding annotation reference set. An AMIGene prediction was assumed to be correct if the predicted stop codon of the CDS matched the annotated stop. The sensitivity, Sn, is thus defined as the ratio of the number of correctly predicted genes to the number of genes annotated in the corresponding curated database. The specificity, Sp, is the ratio of the number of correctly predicted genes to the total number of genes predicted by AMIGene. Optimization of the AMIGene parameter values was performed by searching for the set of values that minimize the following risk Rk function:

$$\mathrm{R}k = \left(\frac{k}{k+1}\right) \times (1 - \mathrm{Sn}) + \left(\frac{1}{k+1}\right)(1 - \mathrm{Sp})$$

where k is the penalization factor between false-negative and false-positive predictions. Successive minimizations in each direction (seven parameter values, Table 1) were performed on several iterations until the Rk variation between two successive iterations was $<0.1\%$ [i.e. $\mathrm{R}i - (\mathrm{R}i+1)/\mathrm{R}i < 0.1\%$]. This optimization process was performed for $k=10$ (false-negative predictions are penalized 10 times more than false-positive

Table 1. Definition and value of the AMIGene parameters for three reference genomes (optimization process)

Abbreviation	Definition	BACSU	ECOLI	MYCTU
Sure-Pc	Coding probability above which a CDS is interpreted as a sure CDS	0.67	0.62	0.47
Prob-Pc	Between the Sure-Pc and Prob-Pc thresholds a CDS is interpreted as a probable CDS	0.35	0.40	0.21
Prob-LMin	Minimum length (bp) of a probable CDS selected	114	141	219
Sure-ss-I	Minimum inclusion percentage between two sure CDSs transcribed on the same strand	5	5	20
Sure-os-I	Minimum inclusion percentage between two sure CDSs transcribed on the opposite strands	30	56	70
Sure-prob-O	Maximum overlapping percentage between a sure and a probable CDSs transcribed on the opposite strands	5	5	37
Prob-glob-IO	Maximum global score (%), including both inclusion and overlapping situations, between a probable CDS and all the other probable CDSs which partially or completely overlap this CDS	86	75	99

ECOLI = *Escherichia coli* K12; BACSU = *Bacillus subtilis* 168; MYCTU = *Mycobacterium tuberculosis* H37Rv.

predictions) and led to three sets of parameter values for the reference genomes (Table 1). The AMIGene average accuracy of gene finding is then characterized by 98.3% sensitivity and 92.4% specificity. Based on the learning set, such values are probably an upper limit of the accuracy of AMIGene.

The accuracy of our method was then tested on bacterial genomes closely related to our models: *Bacillus halodurans*, *E.coli* O157:H7 and *M.tuberculosis* CDC1551. Annotation data were extracted from the INSD and compared to the set of AMIGene predicted CDSs, using the three gene models built for these genomes and the optimized parameter values of each related organism (Table 1). AMIGene predictions were very good for the *B.halodurans* genome (Sn = 98.6% and Sp = 89%) and the *E.coli* O157:H7 genome (Sn = 96.8% and Sp = 93.2%). Concerning the *M.tuberculosis* CDC1551 genome, the sensitivity was 94.7% with 85% specificity. This result is somewhat surprising since the two *M.tuberculosis* strains (H37Rv and CDC1551) share >90% identity at the DNA level. Additional statistical tests have been used to demonstrate that this lower prediction emerged from heterogeneity between the different sets of *M.tuberculosis* annotations (H37Rv and CDC1551; not shown).

The AMIGene web site

AMIGene is implemented in the C language and is available upon request as a stand-alone application or via a web server at the following URL: http://www.genoscope.cns.fr/agc/tools/amigene/index.html.

The home page of our software allows users to choose the AMIGene input parameters and is divided into four main sections. A precise description of this page can be found at the following URL: http://www.genoscope.cns.fr/agc/tools/amigene/html/helpForm.html; the 'Gene Model' section allows the user to either select existing matrices that have been computed on several bacterial gene classes or to build a new gene model (see below). In the latter case, the minimum recommended length of the input sequence is 10 kb. The 'AMIGene parameters' section allows the user to either choose the parameter values that have been optimized for three reference genomes with different GC contents (i.e. *B.subtilis* for a low GC%, *E.coli* for a medium GC% and *M.tuberculosis* for a high GC%; see below), or to define his/her own parameter values. It is however recommended to carefully read the detailed description of the heuristic we have implemented in AMIGene (http://www.genoscope.cns.fr/agc/tools/amigene/html/Method.html#2). The proposed default values are close to those obtained after the optimization procedure on the *E.coli* genome (see below). The third section ('Sequence'), allows the user to enter a DNA sequence (either a large contig or a complete bacterial genome) and to choose the adapted genetic code. In the last section ('Options'), several additional functions are proposed such as the translation of the predicted genes (leading to a Fasta file format which can be downloaded from the results home page) or the search for putative frameshifts using our ProFED method (16).

The home page of the AMIGene results includes the list of predicted CDSs in text format and a graph representing the protein coding potentials (both CDS positions and coding prediction curves in the six reading frames). The map is fully dynamic and allows the user to navigate along the genome (or contig) while the corresponding list of predicted CDSs is updated accordingly. The predicted CDSs are drawn using only the left-most start position on the sequence. The positions of putative frameshifts are also clearly indicated on this map and the nucleic or peptidic sequence of each predicted CDS can be retrieved independently. Finally this home page includes several files which can also be downloaded: two files containing predicted nucleic and protein sequences and one file containing the positions of the putative frameshifts (ProFED results; 16). More details on the AMIGene results page are available at the following URL: http://www.genoscope.cns.fr/agc/tools/amigene/html/helpViewer.html.

CONCLUSIONS

The AMIGene method, together with the web software presented here, has already been used to analyze >30 complete prokaryotic genomes and its gene-finding accuracy was assessed by comparison with existing annotations (5). Several interesting discrepancies were in favour of a better selection of CDSs using the biological heuristic developed in AMIGene. Although an alternative start codon is sometimes used in the CDS selection step, our method is not yet suitable for identifying true translation initiation sites. If an alternative start codon is proposed in the AMIGene output file, this only indicates that probably the left-most start codon is not correct.

The next version of AMIGene will include correct gene start predictions, taking into account overlaps between adjacent CDSs, coding prediction curves, translation signals such as the Ribosome Binding Site, together with results of similarities in the protein databanks.

The use of two, three or four gene models instead of only one is clearly an improvement in the final selection. Whatever the reasons for intragenomic variations (e.g. codon bias, base content), the construction of several gene classes based on codon usage leads to Markov models that can uncover small genes which are difficult to spot using the typical model. Genes with atypical composition are candidates for being horizontally transferred genes, although additional evidence would be necessary to confirm this hypothesis (17,18). Indeed, identifying the number of gene classes based on their codon usage (with FCA and clustering statistical methods; see below) is more pertinent when performed by human experts. This work is currently under development using the GenoStar platform (19) (GenoAnnot and GenoBool modules; http://www.genostar.org) and deals with pitfalls which arise from the use of correspondence analysis in codon usage studies; the transformation performed on the original data (i.e. the absolute codon frequencies) decreases the amount of information and may introduce new biases (20). In addition to gene models currently available for 12 genomes, we plan to regularly add new gene models in our AMIGene web site, computed based on the codon usage analysis of other bacterial genomes.

ACKNOWLEDGEMENTS

This work was supported by the French Centre National de la Recherche Scientifique (CNRS-URA8030) and the GENOPOLE of Evry. We thank Antoine Danchin (Genetics of Bacterial Genomes, Pasteur Institute) and Hugues Roest Crollius (Genoscope/CNRS-UMR8030, Evry) for their critical comments and suggestions.

REFERENCES

1. Borodovsky,M. and McIninch,J.D. (1993) GeneMark: parallel gene recognition for both DNA strands. *Comp. Chem.*, **17**, 123–133.
2. Salzberg,S.L., Delcher,A.L., Kasif,S. and White,O. (1998) Microbial gene identification using interpolated Markov models. *Nucleic Acids Res.*, **26**, 544–548.
3. Médigue,C., Rechenmann,F., Danchin,A. and Viari,A. (1999) Imagene: an integrated computer environment for sequence annotation and analysis. *Bioinformatics*, **15**, 2–15.
4. Médigue,C., Wong,B.C.Y., Lin,M.C.M., Gu,Q., Bocs,S. and Danchin,A. (2002) The *secE* gene of *Helicobacter pylori. J. Bacteriol.*, **184**, 2837–2840.
5. Bocs,S., Danchin,A. and Médigue,C. (2002) Re-annotation of genomes microbial CoDing Sequences: finding new genes and inaccurately annotated genes. *BMC Bioinformatics*, **3**, 5.
6. Camus,J.C., Pryor,M.J., Médigue,C. and Cole,S. (2002) Re-annotation of the genome sequence of *Mycobacterium tuberculosis* H37Rv. *Microbiology*, **148**, 2967–2973.
7. Médigue,C., Rouxel,T., Vigier,P., Hénaut,A. and Danchin,A. (1991) Evidence for horizontal gene transfer in *Escherichia coli* speciation. *J. Mol. Biol.*, **222**, 851–856.
8. Borodovsky,M., McIninch,J., Koonin,E., Rudd,K., Médigue,C. and Danchin,A. (1995) Detection of new genes in the bacterial genome using Markov models for three gene classes. *Nucleic Acids Res.*, **23**, 3554–3562.
9. Hill,M.O. (1974) Correspondence analysis: a neglected multivariate method. *Appl. Stat.*, **23**, 340–354.
10. Cover,T.M. and Thomas,J.A. (1991) *Elements of Information Theory.* John Wiley, New York.
11. McInerney,J.O. (1998) Replicational and transcriptional selection on codon usage in *Borrelia burgdorferi. Proc. Natl Acad. Sci. USA*, **95**, 10698–10703.
12. Rocha,E.P., Danchin,A. and Viari,A. (1999) Replication biases in bacteria. *Mol. Microbiol.*, **32**, 11–16.
13. Moszer,I., Rocha,E.P.C. and Danchin,A. (1999) Codon usage and lateral gene transfer in *Bacillus subtilis. Curr. Opin. Microbiol.*, **2**, 524–528.
14. Rudd,K.E. (2000) EcoGene: a genome sequence database for *Escherichia coli* K-12. *Nucleic Acids Res.*, **28**, 60–64.
15. Moszer,I., Jones,L.M., Sandrine, Moreira,C. and Danchin,A. (2002) SubtiList: the reference database for the *Bacillus subtilis* genome. *Nucleic Acids Res.*, **30**, 62–65.
16. Médigue,C., Rose,M., Viari,A. and Danchin,A. (1999) Detecting and analysing sequencing errors: toward a high quality of the *Bacillus subtilis* genome sequence. *Genome Res.*, **9**, 1116–1127.
17. Wang,B. (2001) Limitations of compositional approach to identifying horizontally transferred genes. *J. Mol. Evol.*, **53**, 244–250.
18. Koski,L.B., Morton,R.A. and Golding,B. (2001) Codon bias and base composition are poor indicators of horizontally transferred genes. *Mol. Biol. Evol.*, **18**, 404–412.
19. Durand,P., Médigue,C., Morgat,A., Vandenbrouck,Y., Viari,A. and Rechemmann,F. (2003) Integration of data and methods for genome analysis. *Curr. Opin. Drug Disc. Devel.*, **6**, 346–352.
20. Perrière,G. and Thioulouse,J. (2002) Use and misuse of correspondence analysis in codon usage studies. *Nucleic Acids Res.*, **30**, 4548–4555.

Chapitre 20

Short inverse complementary amino acid sequences generate protein complexity

Article de Goldstein, Fondrat, Muri, Nuel, Saragueta, Tocquet, et Prum publié en 2003 dans les Comptes-Rendus de Biologie. L'article met en évidence la présence significative de court inverses-complémentaires dans les séquences d'ADN et étudie les conséquences de ce phénomène pour les séquences protéiques.

SHORT INVERSE COMPLEMENTARY AMINO ACID SEQUENCES (*PRINCOMS*) ARE A UNIVERSAL PROTEOMIC FEATURE INVOLVED IN THE GENERATION OF PROTEIN COMPLEXITY.

Daniel J. Goldstein[1,6,7], Christian Fondrat[2], Florence Muri[3,4], Gregory Nuel[3] , Patricia Saragueta[5], and Bernard Prum[3,6,7]

[1]*Departmento de Ciencias Biológicas*, *Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires, Argentina;* [2]*Université de Paris V*, *Paris, France;* [3]*Laboratoire de Statistique, Upresa CNRS 8071et Département de Mathématique, La Génopole, Université d'Evry, Val d'Essone, France;* [4]*Departement STID, Institut Universitaire de Technologie, Paris, France;* [5]*Departamento de Química Biológica, Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires, Argentina.*

[6]These authors contributed equally to this work.
[7]Corresponding authors.
e-mail djgol@biolo.bg.fcen.uba.ar
prum@genopole.cnrs.fr

ABSTRACT

Inversions of short genomic sequences play a central role in the generation of protein complexity. More than half of the 1300 motifs registered in ProSite have protein inverse complementary sequences (princoms) among proteins registered in the SwissProt data bank. Out of 594 Prosite motifs, the observed number of princoms occurrences exceeded by far the expected number ($p < 10^{-10}$). Princoms are a universal feature of the proteome and underline the importance of lateral genomic transfers (Jain et al. 1999). Princoms often endow their host proteins with a whole new range of biochemical and physiological capabilities, including the possibility of intramolecular and intermolecular disulfide bond formation. These results isupport the idea that the duplication and inversions of small genomic fragments has been a fundamental mechanism for shaping genomes and increasing protein complexity.

INTRODUCTION

Double helical DNA is made up by two complementary polynucleotide chains, Watson (W) and Crick (C) with opposing polarities. Messenger RNAs (mRNA, represented by arrows) are transcribed from both strands with a 5' —> 3' polarity.

$$3' < — 5'$$
$$5' W_s \ldots W_s 3'$$
$$3' C_s \ldots\ldots C_s 5'$$
$$5' —> 3'$$

Duplications with or without inversions of the duplicated DNA segments are common genetic phenomena. Inversions generate transposed segments $[C /W]^T$. Given a complementary double sequence $[W /C]_j$ in a supporting stand , $[W /C]_S$,

$$W_S \ldots W_j \ldots W_S$$
$$C_S \ldots C_j \ldots C_S$$

its inversion and reinsertion will generate a structure of the type

$$W_S \ldots W_j \ldots C^T_j \ldots W_S$$
$$C_S \ldots C_j \ldots W^T_j \ldots C_S$$

W^T_j is the **d**na **in**verse **com**plementary **s**equence (dincom) of C_j and C^T_j is the dincom of W_j. Since every inversion generates a dincom pair, the issue is to determine which of the two DNA strands, C^T or W^T encodes a polypeptide.

Contemporary accounts of gene structure assume that when a protein-encoding double helical DNA segment is inverted, the transposed C^T in W is transcribed with the polarity of C if it carries its own promoter or is driven by a W_S promoter. If the insertion of C^T_j is out of frame, the encoded polypeptide will be a different molecular species, with a length determined by the emergence of a non-sense triplet.

When C^T_j is inserted outside a transcription unit, it will not be transcribed. But its complementary W^T_j may be transcribed as a part of a C_S transcription unit – obviously in the direction of its supporting strand. The nucleotide sequence of W^T_j will be the inverse complementary (i.c.) of C_j,and its amino acid sequence will be therefore different from that encoded by C_j. In a previous paper, we announced that the proteome contains princoms (**pr**otein **in**verse **com**plementary **s**equences), the translation products of dincoms pairs. We found cases where a princoms pair – the direct and the i.c. amino acid sequences – is contained in the same protein, and others in which each member of the princoms pair is found in a different protein species [Goldstein et al. 2000].

In this paper, we provide evidence indicating that the princoms pairs reported earlier are a subset of a much larger set present in protein data banks, and that each member of the pair may belong to either phylogenetically related or unrelated

genomes

RESULTS

ProSite

ProSite is a catalogue of patterns identified by sequence or profile (weight matrix). This paper is based on the analysis of the May 2000 release (release 38), that contains more than 1300 patterns. Since profiles allow for the detection of signatures in sequences with a high degree of divergence, we only worked with patterns. ProSite patterns are sequences of brackets, each containing the list of the possible amino acids that can be found in a given position, (e.g. [NF]), separated by a number d of gaps each characterized by lower and upper boundary values (e.g. a ≤ d ≤ b).

The princoms of a Prosite pattern was obtained by replacing the amino acids within each bracket by the amino acids encoded by the i.c. of all their possible codons (e.g. N [IV], F [KE], hence [NF] [KEIV]), and the order of the brackets and the gaps, inverted.

We asked how many of the ProSite patterns have princoms in the proteins registered in SwissProt. The probability of a bracket equals the sum of the frequencies of its letters (e.g. $p_{[KEIV]} = p_K + p_E + p_I + p_V$.). The probability ***p*** of the occurrence of a sequence of brackets separated by gaps is the product of the probabilities of the brackets in the sequence, multiplied by the factor b – a + 1 for each gap of length d with a ≤ d ≤ b. Taking into account the number ***N*** of amino acids in Swissprot (80,000 entries with ***N*** = 29,085,265 amino acids), the expected number ***E*** of occurrences for this sequence is equal to the product ***N***.***p***. The standard deviation s.d. of ***E*** was calculated through a Poisson approximation. The statistical signification of the observed number of occurrences, ***O***, was based on the statistic (***O***-***E)***/s.d.

To avoid very large files we selected the patterns for which the expected number ***E*** of princoms occurrences does not exceed 100, the cut-off number 100 being arbitrary. This selection generated a set of 594 sequences. Out of these 594 motifs, the mathematical expectation of the number of occurrences greater than 1.96 s.d. is less than 30, and we observed 273 (Group A, $p < 10^{-10}$). The mathematical expectation of the number of occurrences greater than 5 s.d. is less than 2×10^{-4}, and we observed 93 (Group B, $p < 10^{-10}$). From these results we conclude that the princoms we reported earlier are a subset of a much larger set. This strongly suggests that princoms are a common feature in the proteome.

The amino acid composition of the Group B patterns is significantly different from that of all ProSite entries [Table 1]. Cysteine is almost 2.5 times more frequent in Group B than in the whole ProSite ; arginine, glycine, lysine, alanine, and tryptophan almost 1.25 times. The over-representation of these amino acids could be explained by the fact that all of them can be generated by a single site mutation of the cysteine codon, while lysine is an habitual replacement of arginine. A similar behavior could be expected from serine, which codons can also be derived from those of cysteine

by a one letter change. The codons for cysteine, tryptophan, arginine and glycine, as well as one of the stop codons, all have a central guanine, and the frequency of central guanine in the codon usage of group B (40.60 %) is significantly higher than in the whole ProSite ($p < 10^{-18}$) [4] [TABLE 2].

Protein data banks have certain inherent characteristics that can lead to sampling biases – e.g. the arbitrary selection of proteins studied and reported, the inconsistencies of the annotation systems and the existence of numerous entries corresponding to the same protein in different species (redundancy). Furthermore, the criteria for identification of the patterns registered in ProSite are not exhaustive.

We addressed the problem of redundancy by comparing the statistical signification of princoms data obtained from the whole SwissProt and that derived from four specialized data banks – Human proteins in SwissProt (5,913 proteins), Yeast Protein Database (4531 proteins), an Enzyme sub-bank (1519 proteins), and a PDB sub-bank (2139). The statistical signification was comparable except in the case of the PDbank. (Pasek 2001) [TABLE 3].

Biological signification of princoms

Do princoms play a functional role in proteins? We arbitrarily selected the first pattern in Group A, PS 01113. This pattern corresponds to the domain signature of C1q, a subunit of the C1 enzyme complex that activates the serum complement system. We found thirty eight princoms of PS 01113 in Swissprot, a number that significantly exceeds the expected number 16.40 ($p < 3.10^{-8}$) [TABLE 4]. These princoms are found in eukaryotes (animals and plants), prokaryotes and viruses.

The thirty eight host proteins containing these princoms form an heterogeneous group, both structurally and functionally, which includes several types of intracellular and extracellular proteins. The intracellular proteins are nucleotide-binding proteins, DNA and RNA binding proteins, ribosomal proteins, and Ca^{2+}-binding proteins. The extracellular proteins are the constant region of an immunoglobulins heavy chain (IgG-1c), a von Willebrand Factor (vWF), and the heme-transporter hemopexin (Hp). While we have not been able to assign a biochemical role to the princoms in the intracellular proteins, several of them play key structural functions in three types of extracellular proteins.

IgGc. The princoms provides the hinge region of the IgG-1c [P01868 and P01869], including the cysteine involved in the formation of the heavy chain-light chain disulfide bond. This particular princoms has remarkable homology with several plant, insect, and vertebrate metallothioneins. [TABLE 5].

vWF. vWF belongs to a protein family endowed with a C-terminal cysteine knot (CTCK). Approximately one third of the vWF-CTCK is contributed by a princoms. This particular princoms is also homologous to several plant, insect, and vertebrate metallothioneins. (Data not shown)

---

[4] In the case of serine, we estimated its frequency in 1/3.

Hp. Hp consists of a single polypeptide chain divided in two homologous domains, the result of a duplication of an ancestral gene. The two Hp domains, in positions 32 to 235 and 239 to 460 are separated by a four residue hinge region; the two Hp domains share about 25 per cent sequence homology and have the same 3D structure. We found a princoms in the amino terminal domain of rabbit Hp (position 125-155), where it provides the metal binding histidine in position 152. The C-terminal domain of rabbit Hp also has in position 334-365 a trace of the same princoms (denoted princoms*). The princoms and princoms* in Hp have partial homology with the pattern Hp, [LIVMFY]-[DENQS]-[STA]-[AV]-[LIVMFY], in the segment of polypeptide chain between the pattern Hp and princoms. [FIGURE 1]

These data suggest that each Hp domain itself is the result of the duplication of a smaller ancestor gene. If this were so, they should be other vestiges of the princoms of the pattern Hp. We found these traces in the amino terminal Hp-domain (positions 88-103) and in the C-terminal Hp-domain (positions 295-310). As expected, homologous princoms* can be detected in rat, pig, and human hemopexins, and in several matrix metalloproteinases which contain an Hp domain. However, this sequence is also found in the extracellular domain of the -a minobutyric acid (GABA) receptors GAB1 (human, bovin), GAB2 (human, mouse), GAB3 (human, mouse, chick), and a hypothetical protein [P40882] of *Pseudomonas aeruginosa*.

Apoptotic proteins : The proteins belonging to the Bcl family of apoptosis regulators have four domains, BH1, BH2, BH3 and BH4, each one of them characterised by a consensus pattern. Bcl-2 and Bcl-x block apoptosis, and Bax, Bak, and the BH3-only proteins are proapoptotic. Mouse, pig and rat apoptosis-regulators Bcl-x have princoms of the BH2 pattern.

There are 709 princoms of the BH-2 pattern (PS 01258) in SwissProt, while the expected number of occurrences is 357 ± 18.9 ($p < 10^{-50}$). One of the BH-2 princoms is found in position 44-55 of BCLX-Mouse (Q64373) and in equivalent position of rat, pig and human BCLX.

The BH-3 pattern (PS 01259) has 998 princoms whith an expected number of 907.8 ± 30.1. In this case the statistical signification is then very poor (1 %). One of the princoms of BH-3 is the segment 37-51 of the Rana catesbeiana -ch ain hemoglobin, corresponding to the residues 43-57 of Human -ch ain. (Tam, Gray and Riggs 1986). This sequence comprises the helices C and D ; four of its amino acids (phenylalanine 43, phenylalanine 44, leucine 48 and leucine 57) are highly conserved in all hemoglobins and chains, and myoglobins. This shows that the princoms of BH-3 is present in all hemoglobins, and that our program did not detect these molecules because in position 48, instead of [VI] there is a M and in position 44, where [PSAT] was expected, appears a G.

The Bcl family of apoptotic regulatory proteins, the membrane-spanning domains of colicin and dyphteria toxin, share the same myoglobin fold. Although the 3-dimensional structure of the -ch ain of Rana catesbiana Hb has not yet been directly determined, it may be safely assumed that it has the myoglobin fold. Segment 43-57 adopts a helical secondary structure in the -ch ain of human, bovine, equine and avian hemoglobins. Although the segment is predominantly an -h elix, in some species it also has a short 3-helical (3/10 helix) stretch, generally separated from the -h elix by an hydrogen bounded turn. The princoms of segment

43-57 in Rana castebiana is the BH3 domain (the death domain) of apoptosis-regulatory proteins. The X-ray structure of the rat apoptosis-regulatory Bcl-x shows that the fifteen amino acids that form the BH3 domain signature (86-100) form an - helix.

DISCUSSION

Duplications and inversions are characteristic genomic features and play a central role in the evolution of chromosomal architecture. Large size, low-copy repeats with high-sequence identity (several kb to mb duplicons) lead to deletions, duplications, inversions, and inverted duplications [Ji et al. 2000]. Contemporary mosaic proteins are the result of the iteration of small size domains (duplicated gene segments of up to 1 kb). Many of these domains preserve their characteristic motifs (**m**) and signatures (**s**) as well as their three dimensional (3D) structure [2]. A substantial amount of **m** and **s** are cysteine-rich, and the constancy of the positions of cysteine in **m** and **s** allows for the classification of known proteins in families and superfamilies, and to identify new members belonging to them [Brandon and Tooze 1999, Creighton 1993].

While studying the patterns of cysteine-**s** present in several families of autacoid peptides we became aware of the fact that in cysteine-rich regions alternate with threonine and/or alanine -rich regions. This clustered distribution of cysteines, threonines and alanines is also found in vertebrate and invertebrate membrane glycoproteins, mucins, metalloenzymes of the extracellular matrix, proteoglycans, DNA-binding proteins, nuclear membrane proteins, and viral capsides. Since threonine and alanine are encoded by the inverse complementary codons of cysteine, we asked whether the threonine/alanine-rich regions were in fact inversions of the cysteine-**s**. (their inverse complementary homologs).

To answer this question we applied Hidden Markov Models (HMM) in combination with the statistical tool R'HOM [Muri et al 1998]. We first studied the DNA encoding two small homologous proteins, MUA1-XENLA [sw P10667] and MUC1-XENLA [sw Q05049], each of which contains three cysteine-$\mathbf{s}_{\text{TREFOIL}}$, bracketed by threonine/alanine-rich regions. The HMM study of the encoding DNA showed that the threonine/alanine-rich regions are in fact the inverse complementary homologs (the princoms) of the cysteine-$\mathbf{s}_{\text{TREFOIL}}$ [Goldstein et al. 2000].

These trefoil peptides can be described, therefore, as mosaics made up by the linear combination of direct and inverse gene segments (princoms pairs). The analysis of the amino acid sequences of other peptides containing cysteine-**s** revealed that they also have princoms pairs – e.g. the prepropeptides of six endothelins and Zn^{2+} finger knot proteins of the classes 1, 2, 4, 4* (half of the type 4 **s**), and 5. In other cases, we found the princoms of a given cysteine-**s** in a different polypeptide. For example, the somatomedin molecule does not contain its princoms,

---

[2] We call a ***motif*** (m) a continuous sequence of amino that characterizes a protein or a domain acids– e.g. the collagen motif GPP. A ***signature*** (s) is a characteristic sequence in which only certain amino acids are invariant–e.g. the endothelin/corasolin signature $C[X]C[X]_5C[X]_3C$, where X can be any amino acid.

but the inverse complementary homolog of the somatomedin cysteine-**s** is found in 39 different proteins. In other cases, princoms are found both in the molecule containing its inverse complementary homolog and in totally unrelated molecules. For example, the princoms of mixed cysteine/histidine-**s** of the human transcription factor *Krab* [sw O14628], a Zn^{2+} finger knot protein, is present in the *Krab* protein and in other 138 proteins in the data bank.

Our findings give raise to several structural questions. Do all the highly homologous princoms have the same secondary structure in solution? If they do, will they conserve it when inserted in their respective host proteins or will the adopt a new secondary structure as a function of the context provided by the host protein? In fact, the existence of highly homologous princoms in different types of proteins is an "experiment of nature" for testing the generality of the findings of Milnor and Kim [Milnor & Kim, 1996] concerning the importance of context-dependent effects in protein folding. The other relevant structural question is whether a given sequence and its princoms have the same secondary structure. We have shown that the princoms of an -h elical of a hemoglobin -ch ain partially preserves the helical character. We are presently investigating the generality of this finding.

Genomes evolve by acquiring new sequences through horizontal genetic transfers, transposition events, and rearrangements (duplications and inversions), and divergence [Lewin 2000, Lawrence and Ochman 1998, Ochman, Lawrence, Groisman, 2000]. The widespread occurrence of palindromes and their biological relevance in genetic regulation and RNA structure and function, indicates that inversions played a crucial role in the diversification of the portfolio of biological opportunities at the polynucleotide level. Chromosomes are mosaics of ancestral and horizontally transmitted sequences [Ochman, Lawrence, Groisman, 2000].

The study of princoms allows for a new method to detect lateral transfer of small genetic fragments. This would provide a more refined appraisal of the tempo of evolution and reflect the multiple sources of genetic material from which a given protein family derives its contemporary structure.

FIGURE 1

MVKASGIPIALGVWGLCWSLATVNSVPLTSAHGNVTEGESGTKPEADVIE
QCSDGWSFDATTLDDNGTMLFFKDEFVWKSHRGIREL**ISERWKNFIGPVD**
AAFRHGHTSVYLIKGDKVWVYTSEKNEKVYPKSLQDEFPGIPFPLDAAVE
CHRGECQDEGILFFQGNRKWFWDLTTGTKKERSWPAVGNCTSALRWLGRY
YCFQGNQFLRFNPVSGEVPPGYPLDVRDYFLSCPG*RGHR*SSHRNSTQHGH
ESTRCDPDLVLSAMVSDNHGATYVFSGSHYWRLDTNRDGWHSWP**IAHQWP**
QGPSTVDAAFSWEDKLYLIQDTKVYVFLTKGGYTLVNGYPKRLEKELGSP
PVISLEAVDAAFVCPGSSRLHIMAGRRLWWLDLKSGAQATWTELPWPHEK
VDGALCMEKPLGPNSCSTSGPNLYLIHGPNLYCYRHVDKLNAAKNLPQPQ

Rabbit hemopexin. Underlined: princoms. Bold : princoms*

TABLE 1

	ProSite		Group B		ratio
A	773,6	6,05 %	46,2	5,09 %	0,84
R	631,7	4,94 %	55,5	6,11 %	1,24
N	441,3	3,45 %	24,6	2,71 %	0,79
D	651,2	5,09 %	40,6	4,47 %	0,88
C	811,2	6,34 %	142,0	15,64 %	2,47
Q	275,6	2,15 %	14,8	1,63 %	0,76
E	532,1	4,16 %	26,4	2,91 %	0,70
G	1519,1	11,88 %	133,1	14,66 %	1,23
H	453,7	3,55 %	28,5	3,14 %	0,88
I	715,7	5,60 %	36,7	4,04 %	0,72
L	927,8	7,25 %	52,5	5,78 %	0,80
K	515,7	4,03 %	45,4	5,00 %	1,24
M	538,1	4,21 %	25,5	2,81 %	0,67
F	597,1	4,67 %	36,2	3,99 %	0,85
P	524,2	4,10 %	29,5	3,25 %	0,79
S	771,3	6,03 %	51,8	5,70 %	0,95
T	575,1	4,50 %	33,1	3,65 %	0,81
W	238,9	1,87 %	20,8	2,29 %	1,23
Y	479,9	3,75 %	25,3	2,79 %	0,74
V	815,3	6,38 %	39,6	4,36 %	0,68
total	12789	100,00 %	908	100,00 %	

Statistics and frequencies of indiviudual amino acids in ProSite and ProSite*. The occurrence of each amino acid in the complete ProSite and in ProSite*A (the subset of group A, the 93 ProSite **m** and **s** having a highly significant number of princoms). When a position is degenerate, generating a k-letter bracket, we assigned a value of 1/k to each member of the bracket. The last column corresponds to the ratio of the frequency in ProSite* with respect to the frequency in ProSite.

TABLE 2

	SwissProt	ProSite	Group B
a	28,2 %	24,3 %	21,7 %
t	24,6 %	27,0 %	27,4 %
g	24,1 %	27,4 %	30,5 %
c	23,1 %	21,3 %	20,4 %
g+c	47,2 %	48,7 %	50,9 %

Comparison of the nucleotide composition (a, t, g, c) of the whole Swiss Prot (column 1), ProSite (column 2), and Prosite*A.

TABLE 3

	SwissProt	Human	Yeast	Enzyme	PDB
(1.96 s.d.)	273	69	72	58	37
(5 s.d.)	93	19	16	8	1

Number of ProSite motifs having a number of princoms that exceeds the expected value by more than 1.96 s.d. and 5 s.d. All these results have a signification $p < 10^{-10}$, except the 37 for PDB/1.96 s.d. where p = 0,10.

TABLE 4

P02997	Escherichia coli	ELONGATION FACTOR TS (EF-TS).
Q43894	Haemophilus influenzae.	ELONGATION FACTOR TS (EF-TS).
Q38913	Saccharomyces cerevisiae	FAD SYNTHETASE
P50907	Wolbachia pipientis	CELL DIVISION PROTEIN FTSZ
P45485	Wolbachia sp	CELL DIVISION PROTEIN FTSZ
Q10719	Saccharomyces pombe	CELL FUSION PROTEIN FUS1
P01868	Mouse	IG GAMMA-1 CHAIN C REGION
P01869	Mouse	IG GAMMA-1 CHAIN C REGION
P20058	Rabbit	HEMOPEXIN PRECURSOR
O29490	Archaeoglobus fulgidus	PROBABLE TRANSLATION IF-2
P38249	Saccharomyces cerevisiae	EUKARYOTIC TRANSLATION IF-3
P29681	Drosophila melanogaster	20-HYDROXYECDYSONE
Q06738	Arabidopsis thaliana	DESSICATION-RESPONSIVE PROTEIN
O23676	Arabidopsis thaliana	MAGO NASHI PROTEIN HOMOLOG
O51737	Borrelia burgdorferi	DNA MISMATCH REPAIR PROTEIN
P33238	Domestic duck	INTERFERON-INDUCED GTP-BINDING
Q90597	Chicken	INTERFERON-INDUCED GTP-BINDING
P33937	Escherichia coli	PERIPLASMIC NITRATE REDUCTASE PREC
P36608	Caenorhabditis elegans	NEURONAL CALCIUM SENSOR 1
Q09711	Saccharomyces pombe	HYPOTHETICAL CALCIUM-BINDING
Q08637	Enterococcus hirae	V-TYPE SODIUM ATP SYNTHASE
P27341	Sulfolobus acidocaldarius	TRANSCRIPTION ANTITERMINATION
Q42667	Citrus limon	PHENYLALANINE AMMONIA-LYASE
P05738	Saccharomyces cerevisiae	60S RIBOSOMAL PROTEIN L9-A
P51401	Saccharomyces cerevisiae	60S RIBOSOMAL PROTEIN L9-B
P48119	Cyanophora paradoxa	DNA-DIRECTED RNA POLYMERASE BETA
P12954	Saccharomyces cerevisiae	ATP-DEPENDENT DNA HELICASE
P45740	Bacillus subtilis	THIAMINE BIOSYNTHESIS
P20985	Vaccinia virus	PROTEIN A6
P33633	Escherichia coli	PROTEIN IN SRMB-UNG INTERGENIC
Q28295	Dog	VON WILLEBRAND FACTOR PRECURSOR
Q57624	Methanococcus jannaschii	GLUTAMYL-TRNA AMIDOTRANSFERASE
Q57692	Methanococcus jannaschii	HYPOTHETICAL PROTEIN MJ0240
O58012	Pyrococcus horikoshii	HYPOTHETICAL PROTEIN PH0274
Q57968	Methanococcus jannaschii	HYPOTHETICAL PROTEIN MJ0548
P57992	Drosophila melanogaster	YEMANUCLEIN-ALPHA
Q04693	Drosophila melanogaster	HYPOTHETICAL
P46327	Bacillus subtilis	HYPOTHETICAL

List of the proteins containing princoms of ProSite motif PS0113 (C1q).
ProSite motif : F X_5 [ND] X_4 [FYWL] X_6 F X_5 G X Y X [FY]
Princoms : [EIKV] X [EK] X [IV] X [APST] X_5 [EK] X_6 [EIKPQV] X_4 [IV] X_5 [EK]

TABLE 5

P01868 mus musculus	IG GAMMA-1 CHAIN C REGION
P01869 mus musculus	IG GAMMA-1 CHAIN C REGION (MEMBRANE BOUND)
P20759 rattus norvegicus	IG GAMMA-1 CHAIN C REGION
P20760 rattus norvegicus	IG GAMMA-2A CHAIN C REGION
P01863 mus musculus	IG GAMMA-2A CHAIN C REGION, AALLELE
P01865 mus musculus	IG GAMMA-2A CHAIN C REGION, MEMBRANE-BOUND
P01857 homo sapiens	IG GAMMA-1 CHAIN C REGION
P01870 oryctolagus cuniculus	IG GAMMA CHAIN C REGION
P01859 homo sapiens	IG GAMMA-2 CHAIN C REGION
P01860 homo sapiens	IG GAMMA-3 CHAIN C REGION
P20761 rattus norvegicus	IG GAMMA-2 CHAIN C REGION
P01862 cavia porcellus	IG GAMMA-2 CHAIN C REGION
Q02223 homo sapiens	B-CELL MATURATION PROTEIN
P01861 homo sapiens	IG GAMMA-4 CHAIN C REGION
P15566 tachypleus gigas	COAGULOGEN
P02681 tachypleus tridenta	COAGULOGEN
P15265 mus musculus	SPERM MITOCHONDRIAL CAPSULE SELENOPROTEIN
Q96353 brassica napus	METALLOTHIONEIN-LIKE PROTEIN TYPE 2
P30570 triticum aestivum	ZINC-METALLOTHIONEIN CLASS II
P30569 triticum aestivum	ZINC-METALLOTHIONEIN CLASS II
Q40158 lycopersicon escule	METALLOTHIONEIN-LIKE PROTEIN TYPE 2 B
P03997 carcinoscorpius	COAGULOGEN
P02804 cricetulus griseus	METALLOTHIONEIN-I
P01866 mus musculus	IG GAMMA-2B CHAIN C REGION
P01867 mus musculus	IG GAMMA-2B CHAIN C REGION
Q38805 arabidopsis thaliana	METALLOTHIONEIN-LIKE PROTEIN 2B
P56168 brassica juncea	METALLOTHIONEIN-LIKE PROTEIN TYPE 2
P56172 brassica juncea	METALLOTHIONEIN-LIKE PROTEIN TYPE 2
Q39269 brassica rapa ssp. p	METALLOTHIONEIN-LIKE PROTEIN
P56170 brassica juncea	METALLOTHIONEIN-LIKE PROTEIN
P02803 rattus norvegicus	METALLOTHIONEIN-LIKE PROTEIN
Q42258 arabidopsis thaliana	EC PROTEIN HOMOLOG 3
P80290 oryctolagus cuniculus	METALLOTHIONEIN-LIKE PROTEIN
P18055 oryctolagus cuniculus	METALLOTHIONEIN-IIA
Q42377 arabidopsis thaliana	EC PROTEIN HOMOLOG 2
P93746 arabidopsis thaliana	EC PROTEIN HOMOLOG
P43390 actinidia chinensis	METALLOTHIONEIN-LIKE PROTEIN TYPE 2
Q42494 brassica rapa	METALLOTHIONEIN-LIKE PROTEIN TYPE 2
P25860 arabidopsis thaliana	METALLOTHIONEIN-LIKE PROTEIN
P33654 streptomyces cacaoi	HYPOTHETICAL 14.2 KDA PROTEIN
P43396 coffea arabica	METALLOTHIONEIN-LIKE PROTEIN 1
P14425 stenella coeruleoalba	METALLOTHIONEIN-II
P04459 gallus gallus	KERATIN, SCALE
P18563 cavia porcellus	INTEGRIN BETA-6
Q52106 acinetobacter calcoaceticus	MERCURIC TRANSPORT PROTEIN
P41927 yarrowia lipolytica	METALLOTHIONEIN-I
P11844 homo sapiens	GAMMA CRYSTALLIN A
P20762 rattus norvegicus	IG GAMMA-2C CHAIN C REGION
P15229 buthus sindicus	SMALL TOXIN

List of proteins having a protein sequence highly homologous to the princoms contained in IgC-1c (P01868 and P01869)

REFERENCES

Bize, L., Muri, F., Samson, F., Rodolphe, F., Ehrlich, S.D., Prum, B., Bessieres, P. 1999. Searching gene transfers on *Bacillus subtilis* using hidden Markov models. In *Recomb99: Proceedings of the 3rd. Annual International Conference on Computational Molecular Biology* (eds. Istrail, S., Pevzner, P., and Waterman, M.), ACM Press, New York, NY.

Brandon, C. and Tooze, J. 1999. *Introduction to protein structure*. 2nd edition. New York : Garland Publishing Inc.

Creighton, T.E. 1993. *Proteins: structures and molecular properties*. 2nd edition. W.H. Freeman and Company, New York, NY.

Faber H.R., Groom C.R., Baker H.M., Morgan W.T., Smith A., and Baker E.N. 1995 . 1.8-A crystal structure of the C-terminal domain of rabbit serum haemopexin. *Structure* **3**: 551-559.

Goldstein, D.J., Muri, F., Saragueta, P., and Prum, B. 2000. Inverse complementary homologues of short cysteine signatures. *C.R. Acad. Sci. Paris/Life Sciences* **323**: 167-172.

Jain, R., Rivera, M.C. & Lake, J.A. (1999). “Horizontal gene transfer among genomes: the complexity hypothesis.” Proc. nac. Acad. Sci. 96:3801-3806.

Komano, T. (1999) “Shufflons: multiple inversion systems and integrons.” Annu. Rev. Genet. 33: 171-91.

Lawrence, J.G., and Ochman, H. 1998. Molecular archeology of the *Escherichia coli Escherichia coli* genome. *Proc. Natl. Acad. Sci. USA* **95**: 9413-9417.

Milnor, D.L. Jr and Kim, P. S. 1996 Context-dependent secondary structure formation of a designed protein sequence. *Nature* 300: 730-734.

Morgan W.T., Muster P., Tatum F., Kao S.-M., Alam J., and Smith A. 1993. Identification of the histidine residues of hemopexin that coordinate with heme-iron and of a receptor-binding region. *J. Biol. Chem*. **268**:6256-6262.

Morgunova, E., Tuuttila, A., Bergmann, U., Isupov, M., Lindqvist, Y., Schneider, G., and Tryggvason, K. 1999. Structure of human pro-matrix metalloproteinase-2: activation mechanism revealed. *Science* **284**:1667-1670.

Muri-Majoube, F., Prum, B. 2001 Une approche statistique de l'analyse des génomes. *La Gazette des Mathématiciens* **89**:63, 98

Ochman, H., Lawrence, J.G., and Groisman, E.A. 2000. Lateral gene transfer and

the nature of bacterial innovation. *Nature* **405**:299-304.

Ohno, S. 1970. *Evolution by gene duplication*. Springer-Verlag, New York, NY.

Ohno, S. 1985. The notion of primordial building blocks in constructing genes and transcriptional and processing errors due to the random occurrence of oligonucleotide signal sequences. *Adv. Exp. Med. Biol.* **190**:627-636.

Pasek, S. Etude du nombre d'occurrences d'inverses complémentaires de motifs protéiques dans les banques de protéines. Mémoire IUP 2. Laboratoire Statistique et Génome, U. CNRS 8071, Genopole, Évry 2001.

Pearson, W.R. and Lipman, D.J. 1988. Improve tools for biological comparison. *Proc. Natl. Acad. Sci. USA* **85**:2444-2448.

Romero, D. and Palacios, 1997. “Gene amplification and genomic plasticity in prokaryotes.” Annu. Rev. Genet. **31**:91-111.

Romero D, Martinez-Salazar J, Ortiz E, Rodriguez C, Valencia-Morales E. (1999). “Repeated sequences in bacterial chromosomes and plasmids: a glimpse from sequenced genomes.” Res Microbiol **150**:735-43.

Sadler, J.E. (1998) “Biochemistry and genetics of von Willebrand factor.” Annu. Rev. Biochem. **67**:395-424.

Snedecor G.W. and Cochran W.G. 1980, Statistical Methods, 7th edition, Ames Iowa : Iowa State University Press.

Tam L. T. George P. Gray and Austen F. Riggs 1986, J. of Biological Chemistry 261: 8290-8904.

Wellner D., Cheng K.C., and Mueller-Eberhard U. 1988. N-terminal amino acid sequences of the hemopexins from chicken, rat and rabbit. *Biochem. Biophys. Res. Commun*. **155**:622-625.

Chapitre 21

seq++ : analyzing biological sequences with a range of Markov-related models

Application note de Miele, Bourguignon, Robelin, Nuel, et Richard publiée en 2005 dans *Bionformatics*. seq++ est une librairie de programmation écrite en C++ qui permet la manipulation de séquences biologiques (format FASTA, tout type d'alphabet — y compris alphabet des codons par exemple) et de modèles markoviens (phasé ou non, Markov simple ou parcimonieux, calcul de la loi stationnaire par algèbre linéaire, etc ...).

BIOINFORMATICS APPLICATIONS NOTE Vol. 21 no. 11 2005, pages 2783–2784
doi:10.1093/bioinformatics/bti389

Sequence analysis

seq++: analyzing biological sequences with a range of Markov-related models

Vincent Miele*, Pierre-Yves Bourguignon, David Robelin, Grégory Nuel and Hugues Richard

UMR CNRS 8071 Statistique et Génome, 523 place des Terrasses, 91000 Evry, France

Received on October 28, 2004; revised on February 1, 2005; accepted on March 9, 2005
Advance Access publication March 17, 2005

ABSTRACT

Summary: The seq++ package offers a reference set of programs and an extensible library to biologists and developers working on sequence statistics. Its generality arises from the ability to handle sequences described with any alphabet (nucleotides, amino acids, codons and others). seq++ enables sequence modelling with various types of Markov models, including variable length Markov models and the newly developed parsimonious Markov models, all of them potentially phased. Simulation modules are supplied for Monte Carlo methods. Hence, this toolbox allows the study of any biological process which can be described by a series of states taken from a finite set.

Availability: Under the GNU General Public Licence at http://stat.genopole.cnrs.fr/seqpp

Contact: miele@genopole.cnrs.fr

INTRODUCTION

A considerable range of genomic data can be modelled as sequences of characters taken from various alphabets. Obvious candidates are the nucleic acids or protein sequences. However, RNA or protein secondary structures can similarly be described using character strings. This paper presents seq++ , a C++ software library, aiming to be a reference environment for studying the statistical properties of these sequences using a comprehensive set of Markov models. It already implements most of the classical methods, as well as some more recent ones, and offers an extensible framework to experiment with new models, which can be included in subsequent releases of the library.

MODELS AND FEATURES

The key to the flexibility of seq++ is its independence from any given alphabet: the algorithms implemented in seq++ handle strings of tokens. These tokens, which can be several characters long are enumerated in an alphabet file also allowing the definition of synonymous token groups. For instance, studying amino acid composition properties of protein sequences is possible by grouping amino acids according to their physical or chemical properties and assigning them a suitable label.

The algorithms themselves focus on Markov chains (MC) sequence modelling, including phased models. These models can be useful when phased emission heterogeneity occur in the observations. For example, it is well known that the third nucleotide of a codon has different occurrence patterns than the two preceding ones. This can be taken into account by fitting one transition matrix (which is the matrix determining emission probabilities for the observations) per phase to yield more accurate results (Borodovsky *et al*., 1995).

Models provided by seq++ include variable length Markov chains (VLMC) (Bühlmann and Wyner, 1999) and parsimonious Markov chains (PMC) [Bourguignon and Robelin, 2004; P.Y. Bourguignon, 2005 (submitted for publication)]. Let $Y_{i,0<i\leq l}$ be the set of tokens in a sequence of length l modelled by a d-order Markov model. In VLMC, the prediction of Y_i (the token at position i) can be determined by the preceding words of variable length $(Y_{i-d'}\cdots Y_{i-1}, d' \leq d)$. Using words of length less than d can considerably reduce the number of parameters and increase the quality of the model adjustment. The same motivations underlie the use of PMC, where predictors with identical emission probabilities are grouped into motifs representing degenerated token words with possible gaps (Fig. 1). Nevertheless such improvements in the model accuracy can require computational costs.

Therefore, methods are implemented in the library for Markovian transition matrix estimation, stationary distribution calculus, word probabilities, total variation distance between two Markovian matrices, likelihood and Bayesian information criterion (BIC) calculus. The efficiency of eigenproblems computation is ensured by the use of Arnoldi algorithms (Lehoucq *et al*., 1996). These methods aim, for example, to contribute to a motif-detection algorithm (Bulyk, 2003): given a model based on a selection of known motifs of interest and a background model on a target sequence, a sliding window approach can be developed based on the ratio of the likelihoods of the 'site' model and the 'background' model (high scores corresponding to potential sites; Fig. 2).

In addition, the programs `estim_m`, `estim_vlm` and `estim_pm` are also released in seq++ to perform a set of calculus associated with models MC, VLMC and PMC, respectively. Moreover, the program `simul_m` simulates a sequence according to a Markovian matrix (or p matrices for p phases) previously estimated. When working on homogeneous sequences, biologists are often interested in the P-value of an observation. This P-value is frequently impossible to calculate analytically, thus it can be estimated using simulated control sequences. As a result seq++ provides an efficient environment for Monte Carlo methods.

*To whom correspondence should be addressed.

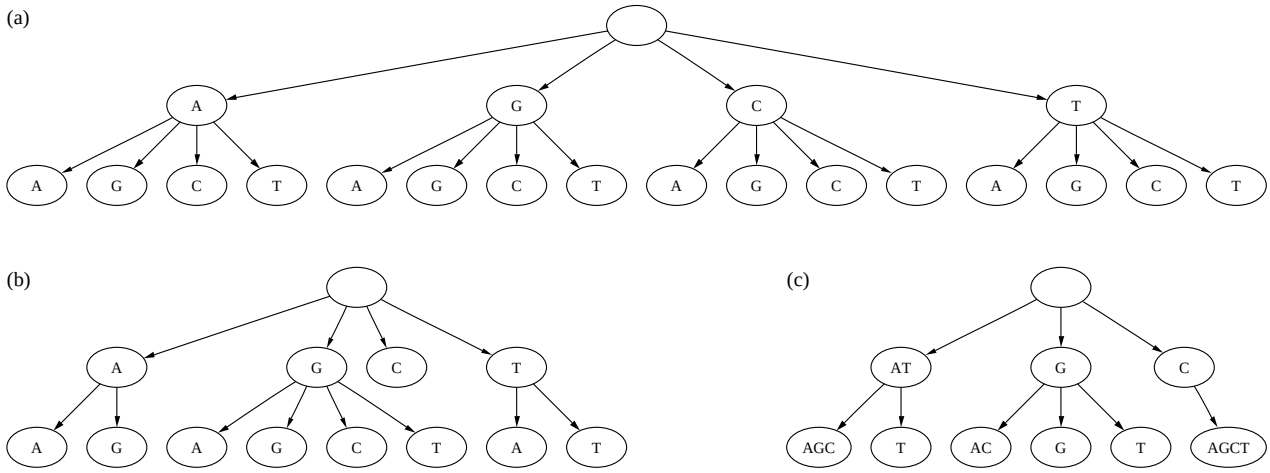


Fig. 1. A Markovian model can be represented by a tree: the root corresponds to the letter Y_t to be predicted, the possible values of Y_{t-1} are on the first level, those of Y_{t-2} on the second and so on. (**a**) represents an MC of order 2. In a VLMC, branches can be cut: as in (**b**), when $Y_{t-1} = C$, the law of Y_t does not depend on Y_{t-2}. In a PMC, subtrees can be merged: as in (**c**) the two trees below $Y_{t-1} = A$ and $Y_{t-1} = T$ generate a unique tree. In each case, the number of predictors is equal to the number of leaves in the tree.

```
ord = 2;
SequenceSet mot("motifs.fna","dna",ord);
nphase = mot(0).tell_length();
PhasedMarkov m_motif(mot, nphase);
SequenceSet set("target.gb","dna",ord);
Sequence & seq = set(0);
Markov  m_backg(seq);
for(t=nphase-1;t<seq.tell_length();t++){
   a = m_motif.proba(seq.t-nphase+1,t);
   b = m_backg.proba(seq.t-nphase+1,t);
   cout<<t<<" score: "<<a/b<<endl;}
```

Fig. 2. Code example for motif detection on DNA, where `m_motif` and `m_backg` are the models of order `ord` estimated on the known motifs and the target sequence, respectively. `proba` returns the probability of the word observed between positions `t-nphase+1` and `t`.

DESIGN AND AVAILABILITY

The object-oriented design of seq++ allows for further evolution. A module dedicated to the Mixture Transition Distribution (MTD) model (Lebre, 2004) is planned for future seq++ releases. To our knowledge, seq++ is the only available library for Markovian sequence analysis. A similar project, libsequence (Thornton, 2003), is dedicated to single nucleotide polymorphism analysis. The GHMM library (http://ghmm.org) for hidden Markov models may be a valuable alternative for various problematics on heterogeneous sequences.

The package is written in ANSI C++ and developed on $\times 86$ GNU/Linux systems with GCC 3.4. It has been successfully tested with Intel ICC 8.0, on Sun systems using GCC 3.3 and Apple Mac OSX systems with GCC 3.1. Compilation and installation are compliant with the GNU standard procedure. The library is free and available at http://stat.genopole.cnrs.fr/seqpp. Online documentation is also available. Software using seq++ (Robelin *et al*., 2003) dedicated to DNA bioinformatics can also be accessed online. seq++ is licensed under the GNU General Public License (http://www.gnu.org/licences.html).

ACKNOWLEDGEMENTS

We are grateful to M.Baudry for the hardware support, M.Hoebeke and P.Nicolas for the programming advice.

REFERENCES

Borodovsky,M. *et al*. (1995) Detection of new genes in a bacterial genome using Markov models for three gene classes. *Nucleic Acids Res.*, **25**, 3554–3562.

Bourguignon,P.Y. and Robelin,D. (2004) Modèles de Markov parcimonieux, *Actes de JOBIM*.

Bühlmann,P. and Wyner,A.J. (1999) Variable length Markov chains. *Ann. Stat.*, **27**, 480–513.

Bulyk,M.L. (2003) Computational prediction of transcription-factor binding site locations, *Genome Biol.*, **5**, 201.

Lebre,S. (2004) Estimation de modèle MTD, *Evry University Internal Report*.

Lehoucq,R.B., Sorensen,B. and Yang,C. (1996) ARPACK user's guide: solution of large-scale eigenvalue problems by implicitly restarted Arnoldi methods, *Rice University Technical Report*.

Robelin,D. *et al*. (2003) SIC: a tool to detect short inverted segments in a biological sequence. *Nucleic Acids Res.*, **31**, 3669–3671.

Thornton,K. (2003) libsequence, a C++ class library for evolutionary genetic analysis. *Bioinformatics*, **19**, 2325–2327.

Bibliographie

M. Abramowitz et I.A. Stegun. *Handbook of Mathematical Functions*. Wiley-Interscience Publication, 1972.

R. Arratia, L. Goldstein, et L. Gordon. Poisson approximation and the Chen-Stein method. *Stat. Sci.*, 5(4) :403–434, 1990.

K. Atteson. Calculating the exact probability of language-like patterns in biomolecular sequences. In AAAI Press, editor, *Sixth International Conference on Intelligent Systems for Molecular Biology*, pages 17–24, 1998.

A.D. Barbour, L. H. Y. Chen, et W. L. Loh. Compound Poisson approximation for nonnegative random variables via Stein method. *Ann. Probab*, 20 :1504–1527, 1992.

A. Berchtold. Estimation in the mixture transition distribution model. *J. Time Series Analysis*, 22 :379–397, 2001.

S. Bocs, S. Cruveiller, D. Vellenet, G. Nuel, et C. Medigue. AMIGene : Annotation of Microbial Genes. *Nucleic Acids Res.*, 31(13) :3723–3726, 2003.

P. Y. Bourguignon, M. Guedj, F. Képès, C. Matias, G. Nuel, N. Omont, et B. Prum. Recherche de gènes impliqués dans une maladie (genset-serono). *Matapli*, 2004.

P.Y. Bourguignon. *Parcimonie dans les modèles markoviens, application à l'analyse des séquences biologiques*. PhD thesis, Université d'Évry, 2006.

P. Buhlmann. Model selection for variable length Markov chains and tunning the context algorithm. *Annals of the Institute of Statistical Mathematics*, 52 :287–315, 2000.

P. Buhlmann et A. Wyner. Variable length Markov chains. *Annals of Statistics*, 27, 1999.

M. Chrochemore et C. Hancart. *Handbook of Formal Languages, Volume 2, Linear Modeling : Background and Application*, chapter Automata for Matching Patterns. Springer-Verlag, Berlin, 1997.

O. Chrysaphinou et S. Papastavridis. A limit theorem on the number of overlapping appearances of a pattern in a sequence of independant trials. *Proba. Theory Relat. Fields*, 79(1) :129–143, 1988.

E. Cinlar. *Introduction to Stochastic Processes*. Prentice Hall, 1975.

R. Cowan. Expected frequencies in DNA sequences using Whittle's formula. *Journal of Applied Probabilties*, 28 :886–892, 1991.

M. Crochemore et V. T. Stefanov. Waiting time and complexity for matching patterns with automata. *Info. Proc. Letters*, 87 :119–125, 2003.

M. Cuturi et J.-Ph. Vert. The context-tree kernel for strings. *journal a preciser*, 2005.

F. Den Hollender. *Large Deviations*. American Mathematical Society, 2000.

EMBOSS. The European Molecular Biology Open Software Suite, 2006. URL `http://emboss.sourceforge.net/`. Version 4.0.0.

FSA. Finite State Automata Utilities, 2006. URL `http://www.let.rug.nl/~vannoord/Fsa/`. Version 6.

J. C. Fu. Distribution theory of runs and patterns associated with a sequence of multi-state trials. *Statistica Sinica*, 6(4) :957–974, 1996.

J. C. Fu et M. V. Koutras. Distribution theory of runs : a Markov chain approach. *J. Amer. Statist. Assoc*, 89 :1050–1058, 1994.

J. C. Fu et W. Y. W. Lou. *Distribution Theory of Runs and Patterns and Its Applications : A Finite Markov Chain Approach*. World Scientific, Singapore, 2003.

M. X. Geske, A. P. Godbole, A. A. Schaffner, A. M. Skolnick, et G. L. Wallstorm. Compound Poisson approximation for word patterns under Markovian hypothesis. *J. Appl. Prob.*, 32 :877–892, 1995.

R. Giegerich et S. Kurtz. From Ukkonen to McCreight and Weiner : A Unifying View of Linear-Time Suffix Tree Construction. *Algorithmica*, 19(3) :331–353, 1997.

D. J. Goldstein, C. Fondrat, F. Muri, G. Nuel, P. Saragueta, A. S. Tocquet, et B. Prum. Short inverse complementary amino acid sequences generate protein complexity. *C. R. Biol.*, 2003.

Graphviz. Graph Visualization Software, 2006. URL `http://www.graphviz.org/`. Version 2.8.

M. Guedj, E. Della-Chiesa, F. Picard, et G. Nuel. Computing power in case-control association studies and application to meta-statistics. *Ann. Hum. Genet.*, 2006a.

M. Guedj, D. Robelin, M. Hoebeke, M. Lamarine, J. Wojcik, et G. Nuel. Detecting local high-scoring segments : a first-stage approach for genome-wide association studies. *Stat. App. Genet. Mol. Biol.*, 2006b. Accepté.

M. Guedj, J. Wojcik, E. Della-Chiesa, G. Nuel, et K. Forner. A fast, unbiased and exact allelic test for case-control association studies. *Hum. Hered.*, 61(4) :210–221, 2006c.

J. E. Hopcroft. *The Theory of Machines and Computations*, chapter An n log n algorithm for minimizing the states in a finite automaton, pages 189–196. Academic press, 1971.

J. E. Hopcroft, R. Motwani, et J. D. Ullman. *Introduction the automata theory, languages, and computation, 2nd edition*. ACM Press, New York, 2001.

R.A. Johnson. *Statistics : principles and methodes (5th ed)*. Wiley, 2006.

S. Karlin et H.M. Taylor. *A first course in Stochastic Processes*. Academic Press, 1975.

J. Kleffe et M. Borodovsky. First and second moment of counts of words in random text generated by Markov chains. *Comp. Applic. Biosci.*, 8 :433–441, 1992.

I. Kontoyiannis et S. P. Meyn. Large deviations asymptotics and the spectral theory of multiplicatively regular markov processes, 2003a. URL `http://citeseer.ist.psu.edu/kontoyiannis03large.html`.

I. Kontoyiannis et S. P. Meyn. Spectral theory and limit theorems for geometrically ergodic markov processes. *Ann. Appl. Probab.*, 13 :304–362, 2003b.

S. Lèbre et P.Y. Bourguignon. An EM algorithm for estimation in the mixture transition distribution model. *J. Time Series Analysis*, volume a preciser, 2005.

E.L. Lehmann. *Nonparametrics*. Springer, 2006.

E.L. Lehmann et J.P. Romano. *Testing Statistical Hypothesis*. Springer, 2005.

T. Leslie. Efficient Approaches to Subset Construction. Master's thesis, Computer Science, University of Waterloo, 1995.

libstree. A generic suffix tree library, 2006. URL `http://www.cl.cam.ac.uk/~cpk25/libstree/`. Version 0.4.2.

W. Y. W. Lou. On runs and longest run tests : A method of Finite Markov chain imbedding. *J. Am. Statist. Assoc.*, 91(436) :1595–1601, 1996.

V. Maout. *Expérience de programmation générique sur des structures non-séquentielles : les automates*. PhD thesis, Université de Marne la Vallée, 2003.

E. M. McCreight. A Space-Economical Suffix Tree Construction Algorithm. *Journal of the ACM*, 23(2) :262–272, 1976.

S.P. Meyn et R.L. Tweedie. *Markov chain and Stochastic Stability*. Springer, 1993.

MGA. Multiple Genome Aligner, 2003. URL `http://bibiserv.techfak.uni-bielefeld.de/mga/`.

V. Miele, P. Y. Bourguignon, D. Robelin, G. Nuel, et H. Richard. seq++ : analyzing biological sequences with a range of Markov related models. *Bioinformatics*, 21(11) : 2783–2784, 2005.

MUMmer. Ultra-fast alignment of large-scale DNA and protein sequences, 2004. URL `http://mummer.sourceforge.net/`. Version 3.0.

P. Nicodème, B. Salvy, et P. Flajolet. Motifs statistics. *Theor. Comput. Sci.*, 28(2) : 593–617, 2002.

J. Norris. *Markov chains*. Cambridge University Press, 1997.

G Nuel. Prédiction de distances phénotypiques à partir de données moléculaires pour la distinction variétale. Master's thesis, Université Paris XI, 1998.

G. Nuel. *Grandes déviations et chaînes de Markov pour l'étude des occurrences de mots dans les séquences biologiques*. PhD thesis, Université d'Evry Val d'Essonne, 2001.

G. Nuel. LD-SPatt : Large Deviations Statistics for Patterns on Markov Chains. *J. Comput. Biol.*, 11(6) :1023–1033, 2004.

G. Nuel. S-SPatt : simple statistics for patterns on Markov chains. *Bioinformatics*, 21 (13) :3051–3052, 2005.

G. Nuel. Cumulative distribution function of a geometric Poisson distribution. *J. Stat. Comp. and Sim.*, 2006a. Accepté.

G. Nuel. Distribution of patterns on Markov chains : a unified approach using deterministic finite state automata. *J. App. Proba.*, 2006b. Soumis.

G. Nuel. Effective p-value computations using Finite Markov Chain Imbedding (FMCI) : application to local score and to pattern statistics. *Algo. Mol. Biol.*, 1(5), 2006c.

G. Nuel. Numerical Solutions for Patterns Statistics on Markov Chains. *Stat. App. Gen. Mol. Biol.*, 2006d. Accepté.

G. Nuel. Pattern statistics on Markov chains and sensitivity to parameter estimation. *Algo. Mol. Biol.*, 2006e. Accepté.

G. Nuel, C. Baril, et S. Robin. Varietal Distinctness Assisted by Molecular Markers : a Methodological Approach. In *Acta Horticulturae : International Symposium on Molecular Markers for Characterizing Genotypes and Identifying Cultivars in Horticulture*, volume 546, pages 65–71, 1999.

G. Nuel, S. Robin, et C. Baril. Predicting distances using a linear model : the case of varietal distinctness. *Journal of Applied Statistics*, 28(5) :607–621, 2001.

P. A. Pevzner, M. Y. Borodovski, et A. A. Mironov. Linguistic of nucleotide sequences : the significance of deviation from mean statistical characteristics and prediction of frequencies of occurrence of words. *J. Biomol. Struct. Dyn.*, 6 :1013–1026, 1989.

W. H. Press, S. A. Teukolsky, W. T. Vettering, et B. P. Flannery. *Numerical recipes in C*. Cambridge University Press, 1997.

B. Prum, F. Rodolphe, et E. de Turckheim. Finding words with unexpected frequencies in DNA sequences. *J. Royal Statistical Society*, 57 :205–220, 1995.

P. Pudlo. *Estimations précises de grandes déviations et applications à la statistique des séquences biologiques*. PhD thesis, Université Claude Bernard - Lyon I, 2004.

A. Raftery et A. Berchtold. The Mixture Transition Distribution for high order markov chains and non gaussian time series. *Statistical Science*, 17 :328–356, 2002.

R. Rebolledo. Central limit theorem for local martingales. *Z. für Wahrscheinlichkeitstheorie*, 62 :269–286, 1980.

G. Reinert, S. Schbath, et M. Waterman. Probabilistic ans statistical properties of words, an overview. *J. Comp. Biology*, 7 :1–46, 2000.

G. Reinert, S. Schbath, et S Waterman. *Lothaire : Applied Combinatorics on Words*, chapter Probabilistic and Statistical Properties of Finite Words in Finite Sequences. Cambridge University Press, 2005.

P. Rice, I. Longden, et A. Bleasby. EMBOSS : The European Molecular Biology Open Software Suite. *Trends in Genetics*, 16(6) :276–277, 2000. URL `http://emboss.sourceforge.net/`.

H. Richard et G. Nuel. SPA : Simple web tool to assess statistical significance of DNA patterns. *Nucleic Acids Research*, 31(13) :3679–3681, 2003.

J. Rissanen. A universal data compression system. *IEE, Transaction on Information Theory*, 29, 1983.

RMES. Recherche de Mots ExceptionnelS, 1999. URL `http://genome.jouy.inra.fr/ssb/rmes/`.

S. Robin. A compound Poisson model for word occurrences in DNA sequences. *J. Roy. Stat. Soc. Ser. C*, 51 :437–451, 2002.

S. Robin, J.-J. Daudin, H. Richard, M.-F. Sagot, et S. Schbath. Occurrence probability of structured motifs in random sequences. *J. Comp. Biol.*, 9 :761–773, 2002.

S. Robin et J.J. Daudin. Exact distribution of word occurrences in a random sequence of letters. *J. App. Prob.*, 36 :179–193, 1999.

S. Robin et J.J. Daudin. Exact distribution of the distances between any occurrences of a set of words. *Ann. Inst. Statist. Math.*, 2000.

S. Robin et S. Schbath. Numerical comparison of several approximations of the word count distribution in random sequences. *J. Comp. Biol.*, 8 :349–359, 2001.

E. Roquain et S. Schbath. Efficient compound poisson approximation for the number of occurrences of multiple words in markov chains, 2006. submitted.

M. Régnier. A unified approach to word occurrence probabilities. *Discrete applied mathematics*, 104(1) :259–280, 2000.

M. Régnier et W. Szpankowski. On pattern frequency occurrences in a Markovian sequence. *Algorithmica*, 22(4) :631–649, 1998.

S. Schbath. Compound Poisson approximation of word counts in DNA sequences. *ESAIM, PS*, 1 :1–16, 1995.

S. Schbath. An efficient statistic to detect over- and under- represented words in DNA sequences. *J. Comp. Biol.*, 4 :189–192, 1997.

D.J. Sheskin. *Handbook of parametric and nonparametric procedures (3rd ed)*. CRC Press, 2003.

SPatt. Statistics for Patterns, 2006. URL `http://stat.genopole.cnrs.fr/spatt/`. Version 1.2.0.

V. T. Stefanov et A. G Pakes. Explicit distributional results in pattern formation. *Ann. Appl. Prob.*, 7 :666–678, 1997.

V. T. Stefanov et A. G Pakes. Explicit distributional results in pattern formation II. *Austral. and New Zealand J. Statist.*, 41 :79–90, 1999.

W.J. Stewart. *Introduction to the Numerical Solution of Markov Chains*. Princeton University Press, 1994.

Z. Talata. *Model selection via information theory for context trees and Markov random fields*. PhD thesis, University of Budapest, 2005.

K. Thompson. Regular Expression Search Algorithm. *Communication of the ACM*, 11 : 419–422, 1968.

E. Ukkonen. On-line construction of suffix trees. *Algorithmica*, 14(3) :249–260, 1995.

J. van Helden, B André, et J. Collado-Vides. Extracting Regulatory Sites from the Upstream Region of Yeast Genes by Computational Analysis of Oligonucleotide Frequencies. *J. Mol. Biol.*, 281 :827–842, 1998.

N. Vergne. *Chaine de Markov régulées pour l'analyse de séquences biologiques*. PhD thesis, Université d'Évry, 2006.

P. Weiner. Linear pattern matching algorithm. In *14th Annual IEEE Symposium on Switching and Automata Theory*, pages 1–11, 1973.

P. Whittle. Some distribution and moment formulae for Markov chains. *JRSS B*, 17 : 235–242, 1955.

F.M.J. Willems, Y.M. Shtarkov, et T.J. Tjalkens. The context tree weighting method : basic properties. *IEEE Transaction and Information Theory*, volume a preciser :653–664, 1995.

X. Zhao, H. Huang, et T.P. Speed. Finding short DNA motifs using permuted Markov models. *J. Computational Biology*, 12 :894–906, 2005.