Méthodes de théorie des jeux pour la prédiction de la structure 3D de l'ARN

Alexis Lamiable

UNIVERSITÉ DE VERSAILLES SAINT-QUENTIN EN YVELINES
École Doctorale Sciences et Technologies de Versailles – STV

Laboratoire Parallélisme, Réseaux, Systèmes et Modélisation
PR*i*SM, UMR 8144

THÈSE DE DOCTORAT
DE L'UNIVERSITÉ DE VERSAILLES SAINT-QUENTIN-EN-YVELINES
Spécialité : informatique

Présentée par :
Alexis LAMIABLE

Pour obtenir le grade de Docteur de l'Université de Versailles
Saint-Quentin-en-Yvelines

Méthodes de théorie des jeux pour la prédiction de la structure 3D de l'ARN

Soutenue le 9 décembre 2011

Directeurs de thèse :
Dominique BARTH : Professeur, UVSQ
Alain DENISE : Professeur, Université Paris-Sud 11

Rapporteurs :
Fabrice LECLERC : CR CNRS, Université de Nancy 1
Denis TRYSTRAM : Professeur, ENSIMAG

Examinateurs :
Jean-Michel FOURNEAU : Professeur, UVSQ
Fabrice JOSSINET : Maître de Conférences, IBMC
Franck QUESSETTE : Maître de Conférences, UVSQ

Numéro national d'enregistrement :

Table des matières

1 Introduction

> We must find RNA structure
> before we give up and return to
> viscosity and bird watching.
>
> James WATSON, 1955 [1]

L'ARN (acide ribonucléique) a longtemps été relégué au rôle d'intermédiaire passif entre l'ADN, dédié au stockage de l'information génétique, et les protéines, dédiées aux fonctions métaboliques. Cependant, depuis la découverte, dans les années 1980, des propriétés catalytiques de l'ARN, de nouvelles fonctions biologiques sont régulièrement associées à cette molécule, et quatre grandes avancées dans sa compréhension ont été récompensées par des prix Nobel en 1972, 1989, 2006 et 2009. On pense désormais que les ARN non-codants (ARNnc, ou *ncRNA* en anglais) sont quatre fois plus nombreux que les ARN codants chez les mammifères [2, préface]. Jusqu'à récemment, l'essentiel des molécules d'ARNnc connues étaient de petites molécules, de quelques centaines de nucléotides maximum, à l'exception du ribosome ou de la ribonucléase P. Cependant, ces dernières années, de nombreuses grosses molécules, aux fonctions encore mal identifiées, ont été découvertes [3, 4]. Leurs structures tridimensionnelles ne sont pas encore connues, et pourraient fournir des indices quant à leurs fonctions biologiques.

Les ARN non codants connus ont essentiellement des rôles de régulation de l'expression des gènes et apparaissent de plus en plus importants pour comprendre les différences entre espèces. De là naît le besoin d'étudier les relations entre la structure des molécules d'ARN et leurs fonctions ; cela passe par la nécessité de comprendre les mécanismes de repliement de l'ARN. Une meilleure compréhension de ces mécanismes de repliement permettrait en outre de faciliter la conception de nouvelles molécules d'ARN, en permettant de choisir une séquence se repliant dans la forme que l'on souhaite obtenir.

Par ailleurs, les méthodes expérimentales de détermination de structure (cristallographie et RMN) étant longues et coûteuses, un modèle de prédiction de structure à partir d'informations plus simples à obtenir (par exemple la séquence) serait souhaitable. Ces prédictions, si elles étaient automatiques et rapides, pourraient être utilisées dans les approches à haut débit de recherche d'ARNnc dans les génomes. À ce jour, les approches existantes de prédiction de structure 3D, simulant le replie-

ment au niveau atomique ou au niveau du nucléotide, sont limitées à de petites molécules, de l'ordre d'une centaine de nucléotides, alors que les plus grosses molécules connues en contiennent plusieurs milliers. L'augmentation de la puissance de calcul des ordinateurs ne suffira pas à rendre ces approches appliquables aux grosses molécules, et de nouvelles méthodes sont requises pour atteindre cet objectif.

Dans cette thèse, nous développons une nouvelle approche de prédiction de structure à gros grain. En effet, une connaissance précise de la position de chaque atome, ou même de chaque nucléotide, n'est pas toujours nécessaire. Dans de nombreux cas, en particulier pour inférer une fonction biologique, une connaissance approximative de la forme de la molécule est tout aussi utile, par exemple en indiquant quelles parties de la molécule sont suceptibles d'interagir avec des ligands. Par ailleurs, nous avançons l'hypothèse qu'une bonne prédiction de la forme à gros grain d'une molécule permetrait également d'améliorer les prédictions à l'échelle atomique. Bien que la reconstruction des molécules à cette échelle dépasse le cadre de cette thèse, nous proposons au chapitre 11 quelques éléments étayant cette hypothèse.

En utilisant une représentation à gros grain, nous espérons pouvoir aborder la prédiction des structures de grosses molécules, mais aussi améliorer les prédictions des structures de taille moyenne. Une représentation à gros grain a l'avantage immédiat de réduire le nombre d'éléments à considérer dans une simulation, ce qui permet, à puissance de calcul égale, de considérer des structures plus grosses. Cependant, notre représentation n'a pas pour seul but de réduire la complexité des calculs, mais aussi de se placer à un niveau plus pertinent (pour la prédiction du repliement) que celui de l'atome ou du nucléotide : celui des hélices Watson-Crick et des jonctions, appelé structure secondaire (voir chapitre 2). À cette échelle, nous isolerons des propriétés des constituants de la molécule qui nous permettront d'en calculer une représentation tridimensionnelle sans essayer de simuler le processus de repliement.

Nous supposons connue la structure secondaire de la molécule. Lorsque seuls une séquence ou un alignement de séquences homologues sont connus, il est possible de prédire une structure secondaire avec différents outils que nous présenterons au chapitre 3. Notre approche est constituée de deux étapes. Dans un premier temps, nous isolons chaque constituant de la structure secondaire : hélices et jonctions entre hélices (aussi appelées boucles internes). Nous classifions ces jonctions en familles topologiques, ces familles définissant une forme « idéale » tridimensionnelle pour les jonctions. Cela nous donne un ensemble de « morceaux » de molécule. La deuxième étape consiste à recoller ces morceaux bout à bout, ce qui produit un prototype de structure tridimensionnelle ne contenant que des interactions de structure secondaire, puis à déterminer si d'éventuelles interactions à longue distance provoquent un repliement de cette structure sur elle-même. La figure 1.1 résume ce processus de prédiction.

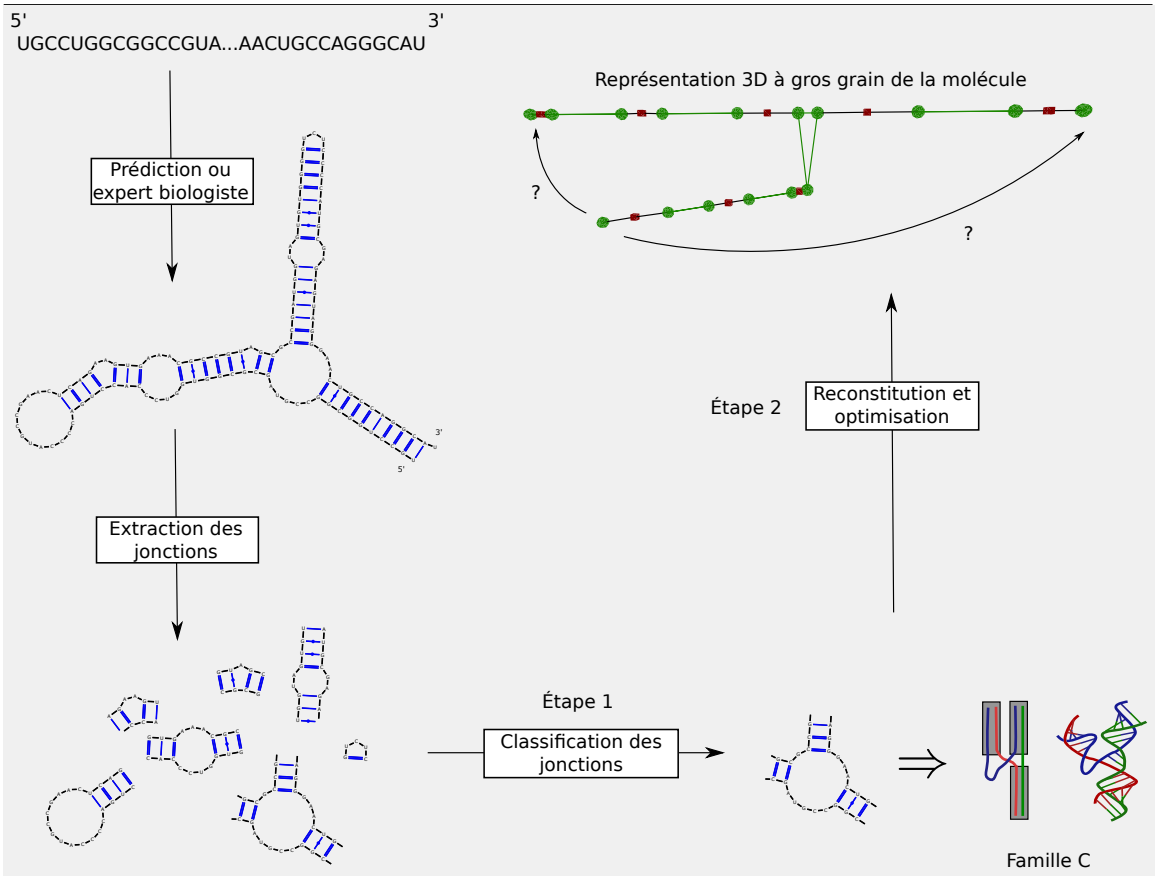


FIGURE 1.1: Processus de prédiction

La première partie de ce manuscrit est consacrée à l'état de l'art. Le chapitre 2 présente la molécule d'ARN et les grandes lignes de ce qui est connu de son processus de repliement. Nous décrivons ensuite, dans le chapitre 3, les différentes approches de prédiction de structure. Nous abordons d'abord les méthodes de prédiction de structure secondaire, sur la qualité desquelles nous nous reposons pour construire notre approche, puis les méthodes de prédiction de structure tridimensionnelle, avec lesquelles nous devrons comparer nos résultats.

La partie II s'intéresse à la première étape de notre méthode : la classification automatique d'éléments locaux de la structure secondaire (les jonctions) en familles topologiques. Le chapitre 5 présente la classification des jonctions à trois branches, qui constitue le travail le plus abouti, et le chapitre 6 aborde la classification des autres types de jonctions.

La partie III est consacrée à la deuxième étape de notre méthode : l'assemblage des éléments locaux et leur repliement afin de déterminer la forme globale de la molécule. Le chapitre 8 présente l'assemblage des jonctions – une fois classifiées en familles topologiques – en un plongement initial dans l'espace, ainsi qu'un algorithme modélisant le repliement de ce plongement initial en fonction d'éventuelles interactions longue distance. Les deux chapitres suivants s'intéressent à la

détermination automatique de ces interactions longue distance par optimisation d'une fonction de coût ; le chapitre 9 aborde une approche d'optimisation globale par un algorithme évolutionniste, et le chapitre 10 présente une approche à base de théorie des jeux, où chaque élément local de la structure est égoïste et optimise sa propre fonction de coût. Enfin, nous comparons au chapitre 11 nos prédictions avec celles des approches concurrentes.

Pour finir, au chapitre 12, nous présenterons nos conclusions et les différentes perspectives d'évolution du travail décrit dans ce manuscrit.

Première partie

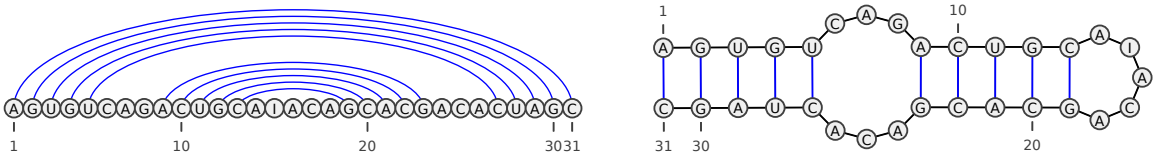
État de l'art

2 L'ARN

Résumé Dans ce chapitre, nous présentons la molécule d'ARN, ses différents niveaux de structure, et ce qui est connu actuellement de son processus de repliement.

2.1 Causes du repliement de l'ARN

Alors que les molécules d'ADN sont constituées de deux brins complémentaires, appariés l'un à l'autre, la plupart des molécules d'ARN sont simple-brin. Les nucléotides de ce brin sont néanmoins capables de s'apparier à d'autres nucléotides et de former des liaisons Watson-Crick (WC), comme dans l'ADN, ou bien d'autres types de liaisons moins solides. Les appariements de nucléotides augmentent la stabilité des molécules ; on ne rencontre donc jamais dans la nature de molécules linéaires comme les ARN messagers sont généralement représentés dans les livres. Tous les ARN contiennent des appariements, et donc se replient dans l'espace. Ainsi, sur l'ARN de trente-et-un nucléotides suivant, les traits bleus indiquent un appariement possible, et cet appariement induit un repliement et la formation d'hélices :



On peut observer de nombreux types d'appariements entre nucléotides en dehors des liaisons Watson-Crick canoniques qui relient les nucléotides A avec les U, et les G avec les C. Ces appariements non canoniques (ou *tertiaires*) ont une influence importante sur le repliement des molécules. La nomenclature *Leontis-Westhof* [5] les classifie en fonction des faces entrant en jeu (Watson-Crick, Hoogsteen et Sucre) et de l'orientation des nucléotides (Cis ou Trans) (voir figure 2.1). Chaque type d'appariement possède une mesure d'encombrement (ou *isostérie*) qui lui est propre, ainsi qu'une énergie caractérisant la solidité de la liaison, les appariements les plus solides étant les liaisons Watson-Crick.

En dehors des appariements entre nucléotides, d'autres types d'interactions entrent en jeu et contribuent à la stabilité d'une molécule : les interactions d'empile-

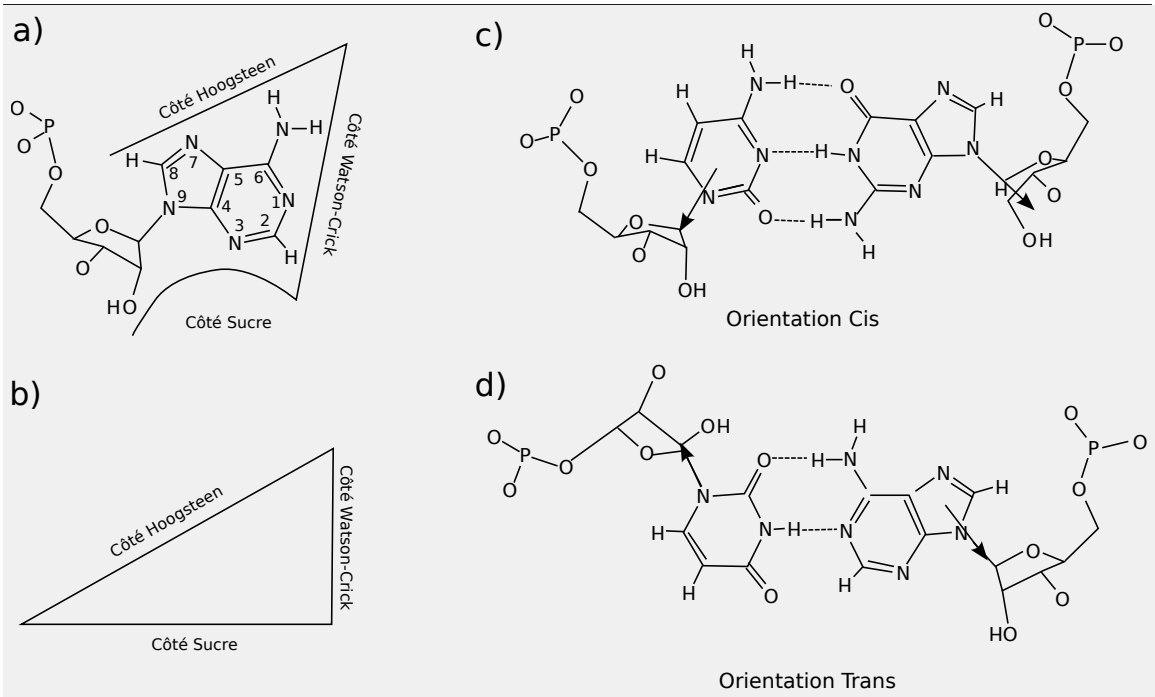


FIGURE 2.1: Appariements possibles de nucléotides. a) une base (ici une purine) présente trois côtés lui permettant d'interagir avec d'autres bases. b) représentation schématique d'un nucléotide. c) appariement d'une paire de bases en orientation Cis, et d) en orientation Trans, selon l'orientation de la liaison glycosidique. (Figure provenant de [5].)

ment de paires de bases, et les interactions entre une base et le phosphate d'une autre base.

Dans les molécules connues, près de 46 % des nucléotides ne font pas partie de régions appariées en liaisons WC [6]. Dans les ARN ribosomaux, environ 20 % des nucléotides forment au moins une liaison non-WC, et environ 21 % ne forment aucune liaison avec un autre nucléotide [7].

2.2 Un repliement hiérarchique

Lorsqu'on étudie la structure des protéines ou des ARN, on distingue trois niveaux de structure : la séquence est appelée *structure primaire*, certaines structures locales stables résultant de liaisons Watson-Crick canoniques forment la *structure secondaire*, et les autres types d'interactions plus faibles forment la *structure tertiaire*. Enfin, on peut ajouter l'information de la position précise de chaque atome, sous le nom de *structure 3D*. Dans le cas de l'ARN, la structure secondaire est constituée des liaisons Watson-Crick (A-U, G-C), ainsi que des liaisons G-U, dites *wobble*. On exclut parfois les pseudonœuds de la structure secondaire. Les figures 2.5 à 2.8 à la

fin de ce chapitre donnent différentes représentations d'une molécule d'ARN.

Dans les protéines, les énergies stabilisant les éléments de structure secondaire sont du même ordre que celles intervenant dans les liaisons tertiaires ; dans l'ARN, les énergies mises en jeu par les liaisons Watson-Crick canoniques sont beaucoup plus grandes que celles des liaisons tertiaires. Les contacts entre nucléotides se font de manière stochastique, mais les liaisons Watson-Crick dominant les autres liaisons, il est possible de considérer que la structure secondaire se forme en premier, et les liaisons tertiaires ensuite [8]. Cette hypothèse est confirmée par le fait que les deux étapes (secondaire puis tertiaire) de formation de structure peuvent être séparées expérimentalement en ajustant la concentration en Mg^{2+} [6].

Nous avons mentionné que le processus de repliement était stochastique, cependant l'ordre de formation des hélices WC et des liaisons tertiaires n'est pas arbitraire. Les bases complémentaires entrent en contact aléatoirement, mais une unique paire de bases n'est pas une configuration stable ; il faut plusieurs paires de bases consécutives pour fermer une boucle et former une tige-boucle (*hairpin loop*). De petites tige-boucles se forment d'abord, puis les hélices s'agrandissent en formant éventuellement des renflements (*bulges*) et boucles internes, avant de se rejoindre en formant des jonctions [8] (voir figure 2.2). Enfin, les liaisons tertiaires relient les boucles et autres régions non-appariées de la molécule et lui donnent une forme compacte.

2.3 Un repliement modulaire

On peut distinguer plusieurs niveaux de modularité dans les molécules d'ARN. Le niveau le plus trivial est celui de la séquence (la structure primaire), où l'unité modulaire est le nucléotide. En passant à la structure secondaire, nous pouvons considérer les hélices Watson-Crick comme des unités modulaires. Puis, dans la structure tertiaire, nous pouvons distinguer des motifs 3D récurrents constitués de brins non-appariés, d'hélices, ou des deux.

Les motifs 3D sont modulaires en ce sens que leur forme est indépendante du contexte dans lequel ils se trouvent [7, 9]. Ces modules ont la particularité d'être hiérarchiques, les modules de haut niveau étant constitués de motifs de niveau inférieur. Des modules 3D complexes peuvent également être constitués de modules 3D plus petits. Initialement, les motifs récurrents ont été identifiés « à la main », mais il est possible de les identifier automatiquement à partir des graphes de structures tertiaires [10].

Les jonctions, ou boucles multiples, entre hélices WC forment une catégorie de modules largement étudiée, et qui nous intéresse particulièrement, car elles sont à la base de l'approche que nous proposons dans cette thèse. Une jonction à n branches, ou n-jonction, est un ensemble de n hélices reliées entre-elles par des brins non appariés. On pourrait penser que la forme des jonctions dépend essentiellement

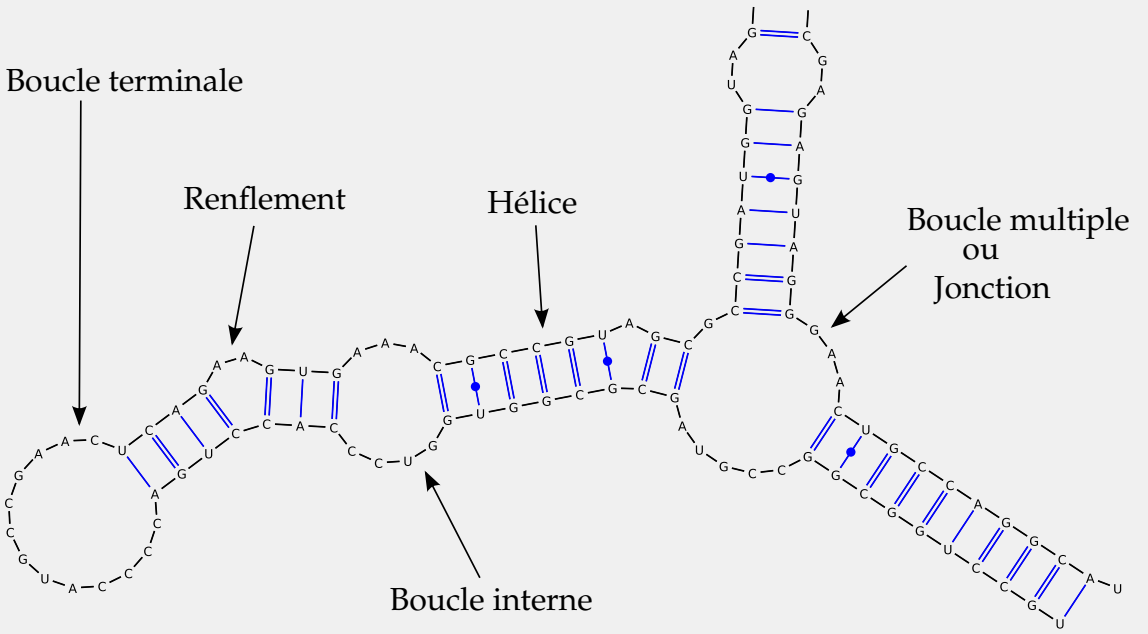


FIGURE 2.2: Nomenclature des éléments de structure secondaire. Une boucle terminale et l'hélice qui la précède sont également appelés une tige-boucle (*hairpin-loop*). Par souci de généralisation, nous nous éloignerons de cette nomenclature et appellerons les boucles internes et les renflements des jonctions à deux branches, et les boucles terminales des jonctions à une branche.

de leur contexte global, et si c'était le cas, elles ne pourraient être étudiées en tant que motifs autonomes. Cependant, de nombreuses études [10, 11, 12, 13] ont montré que les n-jonctions possèdent une structure 3D en grande partie déterminée localement par leur structure secondaire, au moins pour $n < 4$. L'influence de la séquence locale sur la structure des boucles est étudiée dans [14], et des taux de prédiction positifs sur la structure du backbone des boucles internes sont obtenus en n'utilisant que cette information.

Les 2-jonctions (ou boucles internes) sont les mieux étudiées. Elles comportent plusieurs familles topologiques (K-turn, C-loop, Hook-turn, ...) qui peuvent être identifiées automatiquement à partir des structures tertiaires [10, 11]. La figure 2.3 présente quelques-uns de ces motifs. Les 3-jonctions peuvent être séparées en trois familles topologiques, dont la forme semble être déterminée localement [12] ; les liens entre leur forme et la fonction des molécules est étudié dans [15]. Laing et co-auteurs [13] tentent d'effectuer une classification similaire des n-jonctions pour $n \geq 4$, bien que les effectifs des classes soient trop réduits pour que l'on puisse y voir des motifs récurrents. Une caractéristique importante des jonctions est que l'on y observe généralement des empilements d'hélices, qui s'arrangent selon un même axe dans l'espace. Ces empilements ne sont pas des appariements de base, mais apportent de la stabilité aux molécules ; leur prédiction à partir de critères énergétiques est étudiée dans [16]. Nous reviendrons sur la classification

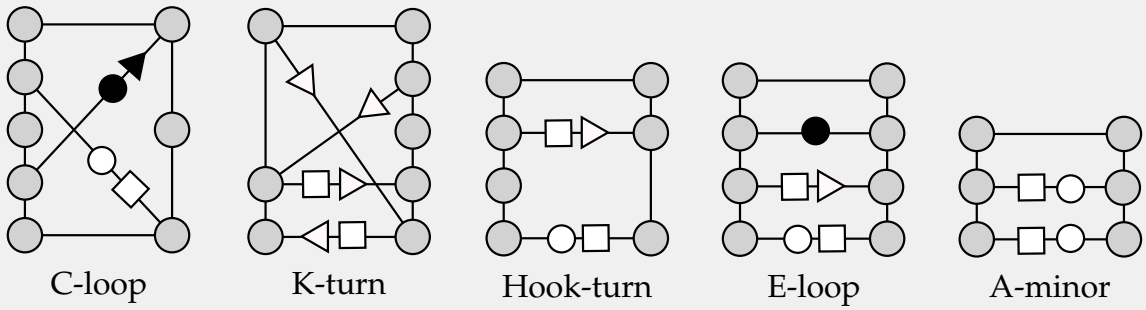


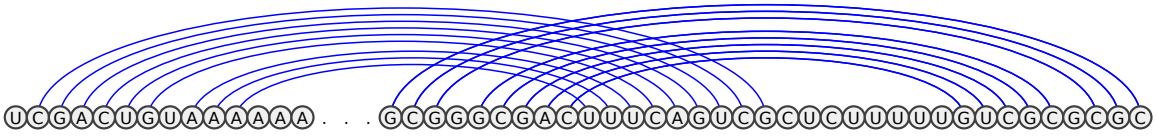
FIGURE 2.3: Exemples de motifs de 2-jonctions. Les disques gris représentent les nucléotides, les lignes verticales le squelette phosphate, et les lignes horizontales ou diagonales les interactions entre nucléotides. Les interactions sont identifiées suivant la nomenclature Leontis-Westhof [5]. Dans la structure secondaire, les interactions comportant carrés, triangles ou disques ne sont pas présentes ; par conséquent, tous ces motifs paraissent identiques lorsque l'on ne dispose que de la structure secondaire.

des jonctions en familles dans la partie II, où nous proposerons des méthodes de classification automatique.

Enfin, plusieurs bases de données de motifs et jonctions existent. FR3D [17] s'intéresse aux motifs de moins de vingt nucléotides et permet de rechercher des motifs 3D dans les molécules de sa base. RNAJunction [18] propose une base de données de n-jonctions ($2 \leq n \leq 9$) extraites automatiquement.

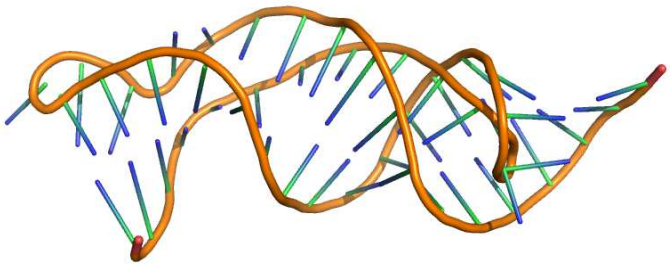
2.4 Les pseudonœuds

Parmi les lisaisons Watson-Crick, on distingue le cas des pseudonœuds, qui se produisent lorsqu'un brin reliant deux hélices ou une boucle terminale établit des liaisons Watson-Crick canoniques avec une autre partie de la molécule. Dans ce cas, la structure secondaire n'est pas arborescente : lorsqu'elle est représentée linéairement, les arcs se croisent :



Ces pseudonœuds posent problèmes aux bioinformaticiens, car la plupart des algorithmes sont plus simples sur des structures arborescentes. Par conséquent, ils sont souvent retirés des structures secondaires et considérés comme faisant partie de la structure tertiaire ; beaucoup d'algorithmes de prédiction de structure prédisent uniquement des structures sans pseudonœuds. Cependant, les pseudonœuds ont un rôle biologique important, et certaines molécules dépendent de leur présence pour accomplir leur fonction biologique, par exemple la partie ARN de la télomérase [19], représentée dans cette section. Ils ont une influence forte sur la

forme globale des molécules, car ils induisent un repliement en 3D dans lequel les brins s'emmêlent, sans toutefois former un vrai nœud topologique :



2.5 Existe-t-il une seule forme pour une molécule ?

Nous avons vu que le repliement de l'ARN était un processus stochastique, dépendant des mouvements aléatoires de la molécule dans son environnement. La forme « finale » de la molécule découle de la stabilité de chacune de ses liaisons. Au fil du temps, des liaisons se forment et se défont, et seules les configurations les plus stables persistent. Toutefois, le repliement dépend aussi de la composition de l'environnement : plus la température est élevée, moins les liaisons chimiques sont stables ; de même, la concentration en ions du solvant influe sur la stabilité des liaisons [8, 20]. On ne peut donc parler de « la » structure d'une molécule qu'en fonction d'un environnement précis ; c'est ainsi que certains ARN entrent en jeu dans la régulation des gènes en fonction des changements de l'environnement. Par exemple, les riboswitches peuvent changer de forme en fonction de la présence d'un ligand, ce qui leur donne de nouvelles propriétés catalytiques [21].

Cependant, même lorsque l'environnement est constant, le repliement peut former des configurations méta-stables qui sont difficiles à défaire pour parvenir à la configuration finale de la molécule. Sortir de ces bassins d'attraction énergétiques peut demander de gros changements conformationnels, et prendre beaucoup de temps [20]. La probabilité P_s de rencontrer une molécule dans une structure donnée s est donnée par la distribution de Boltzmann :

$$P_s = \frac{e^{-E_s/k_b.T}}{Z}, \text{ avec } \quad Z = \sum_s e^{-E_s/k_b.T}$$

Dans cette équation, E_s désigne l'énergie libre associée à la structure s, T la température et k_b la constante de Boltzmann, et Z est appelée la *fonction de partition*. La figure 2.4 représente un exemple de distribution de Boltzmann des structures secondaires d'une molécule. Dans cet exemple, une molécule a 72 % de chances de se trouver dans la structure d'énergie libre minimale, mais également 28 % de chances de se trouver sous une autre forme.

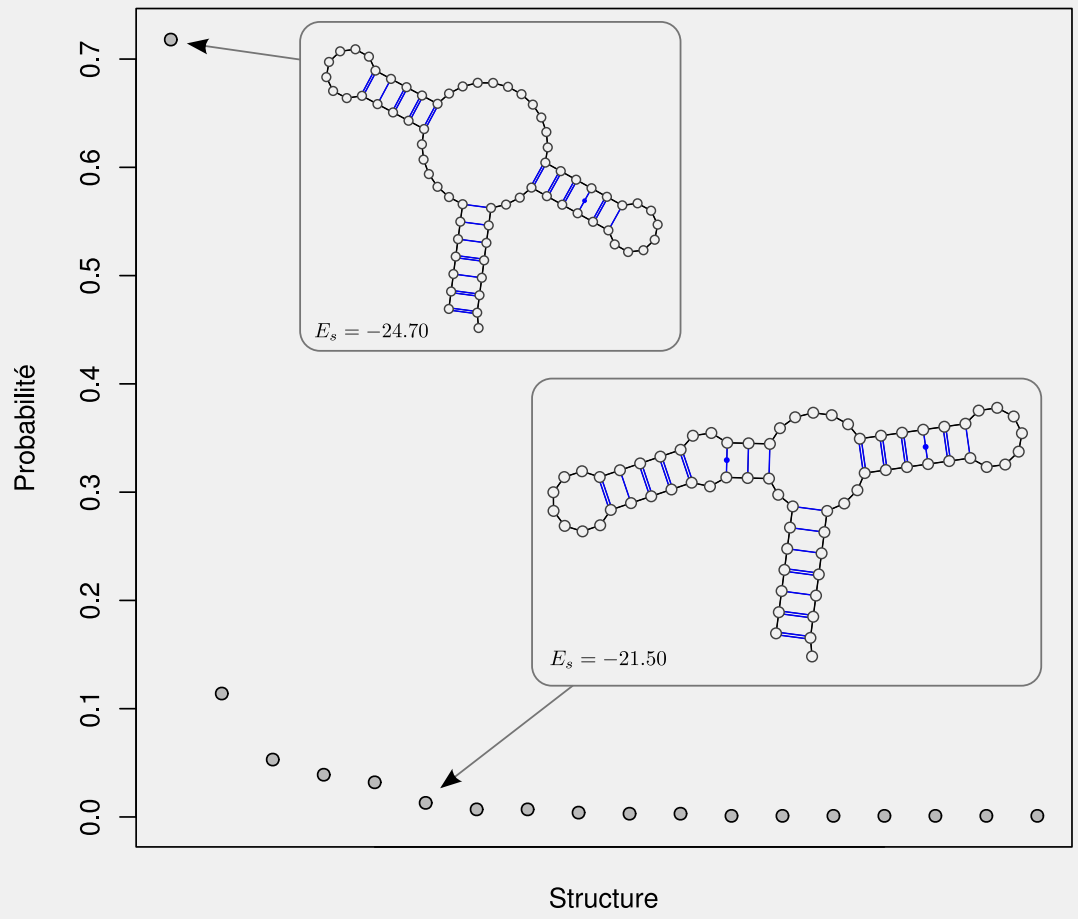


FIGURE 2.4: Exemple de distribution de Boltzmann des structures d'un ARN. Ici, la structure d'énergie libre minimale domine (probabilité de 72 %), mais dans certains cas la distribution peut être plus aplatie, et la probabilité de rencontrer d'autres structures que celle d'énergie libre minimale peut être élevée.
Cet exemple est généré par le logiciel Sfold à partir de la séquence de l'ARN d'identifiant PDB 1U8D. Sfold est un logiciel de prédiction de structure secondaire (voir chapitre suivant) ; son modèle d'énergie est incomplet, l'exemple donné ici est donc théorique.

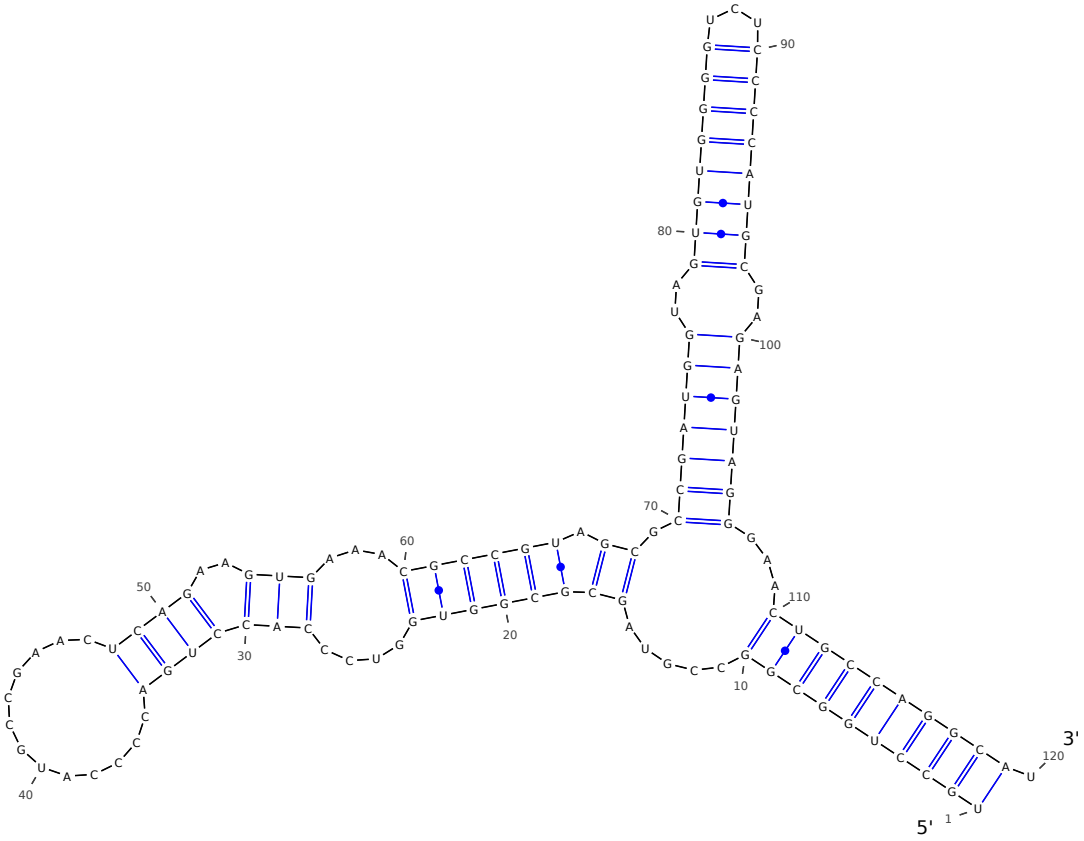


FIGURE 2.5: Structure secondaire de l'ARN ribosomal 5S chez E. Coli

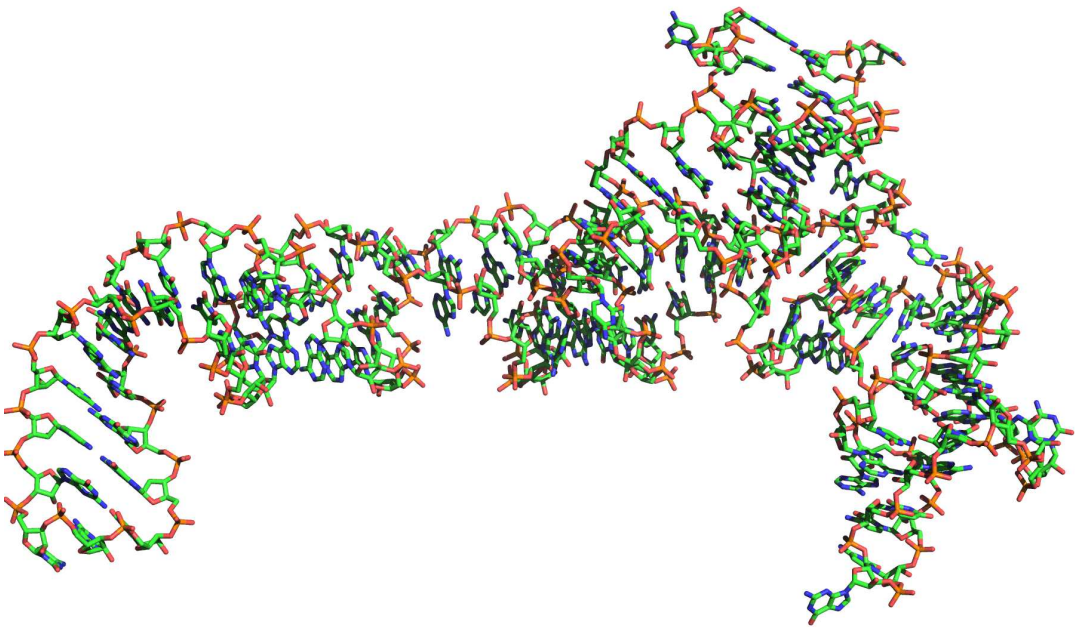


FIGURE 2.6: Structure tertiaire de l'ARN ribosomal 5S chez E. Coli, représentation en bâtons.

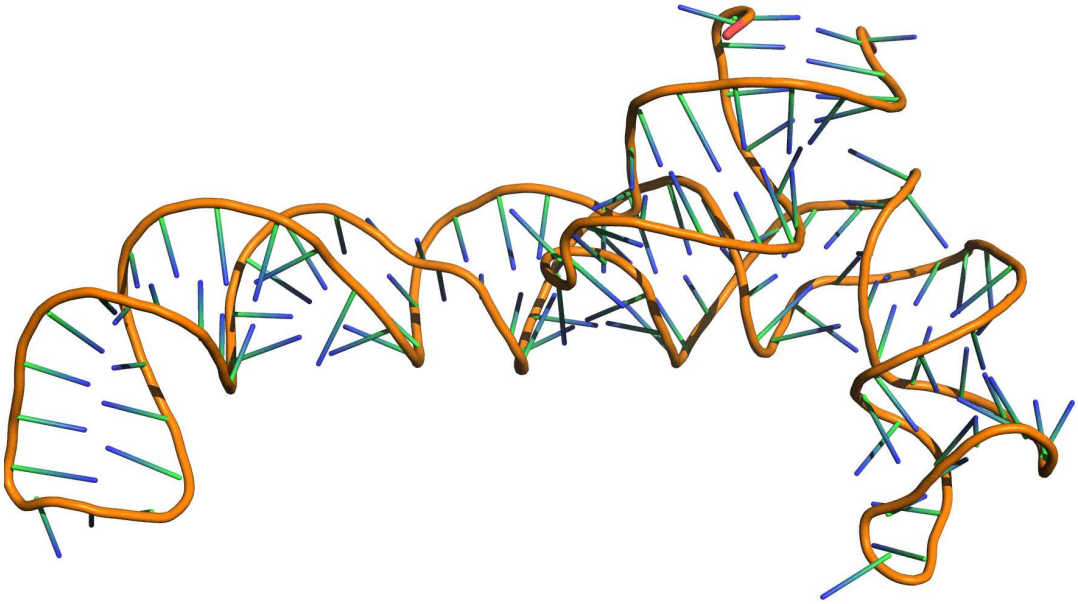


FIGURE 2.7: Structure tertiaire de 5S, représentation stylisée. Le squelette phosphate est en orange, les sucres en vert et les bases en bleu.

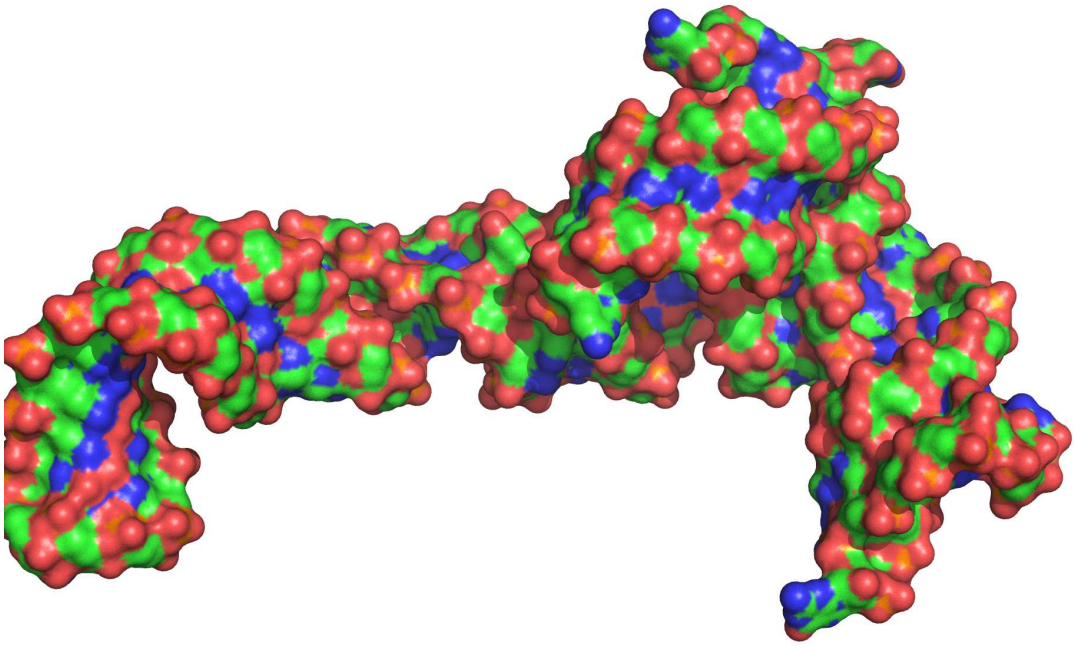


FIGURE 2.8: Structure tertiaire de 5S, représentation de la surface de la molécule.

3 La prédiction de structure

Résumé Dans ce chapitre, nous présenterons l'état de l'art de la prédiction de structure de l'ARN. Nous distinguerons d'une part les approches prédisant la structure secondaire et celles prédisant la structure tertiaire ou 3D, et d'autre part les approches reposant sur une seule séquence et celles nécessitant des alignements de séquence.

3.1 Introduction

Comme décrit dans au chapitre 2.1, le repliement de l'ARN se fait en deux étapes : structure secondaire, puis structure tertiaire. Certaines approches de prédiction s'intéressent à la structure secondaire, d'autres à la structure tertiaire, qui est généralement considérée comme plus difficile à prédire. Laing et Schlick [22] proposent une revue récente des différentes approches de prédiction *in silico*.

Par ailleurs, on peut distinguer deux types d'approches, en fonction des données nécessaires à leur fonctionnement. L'idéal pour une méthode de prédiction de structure serait de pouvoir partir d'une séquence d'ARN dont on ne connait rien et d'obtenir une structure tertiaire complète. Ce sont les approches *simple séquence*. Malheureusement, les approches actuelles de ce type produisent une variété de structures candidates proches de la structure optimale (par rapport à une fonction d'énergie) mais largement différentes entre elles, et aucun moyen de choisir entre ces candidates. L'autre approche, dite *comparative* (ou *par homologie*) est utilisable lorsque l'on connaît déjà d'autres molécules homologues à celle dont on veut prédire la séquence. Elle permet d'utiliser les informations connues sur ces molécules (que ce soit leurs structures ou simplement leurs séquences) pour faciliter la prédiction.

3.2 Structure secondaire

3.2.1 Approche simple séquence

L'algorithme de Nussinov [23], formulé en 1978, utilise la programmation dynamique pour maximiser le nombre d'appariements. Il s'intéresse uniquement aux paires de bases et, s'il est possible de pondérer les types de paires de bases, il ne tient pas compte de la thermodynamique ou de la cinétique de repliement, et ne prédit donc pas la structure réelle des molécules.

Une approche répandue de prédiction de structure secondaire à partir d'une seule séquence consiste à calculer un minimum d'énergie libre (*minimum free energy* ou MFE). Le premier algorithme utilisant cette approche a été celui de Zuker-Stiegler, Mfold [24] avec un modèle d'énergie de « plus proche voisin » (*nearest neighbour*), dont les paramètres ont été déterminés expérimentalement. Dans ce modèle, au lieu de considérer que la stabilité de la molécule provient des paires de bases, on considère qu'elle vient des couples de paires de bases voisines. On associe donc une énergie libre à chaque couple de deux paires de bases empilées possibles : AA/UU, GG/CC, AU/UA, UA/AU, etc. À ces couples de paires de bases, on ajoute les boucles terminales et les boucles internes, ce qui permet d'attribuer des énergies libres à tous les constituants de la structure secondaire. Mfold se charge ensuite de trouver la structure minimisant la somme des énergies libres de ses constituants à l'aide d'un algorithme de programmation dynamique. Alors que les premières versions ne produisaient que la structure optimale, les versions suivantes ont apporté la possibilité de générer également une liste de structure sous-optimales [25].

RNAfold [26] du *Vienna RNA Package* suit la même approche et utilise les mêmes paramètres d'énergie libre. Il est accompagé de RNAsubopt, qui garantit, contrairement à Mfold, de générer l'ensemble des structures sous optimales possibles, donc l'intégralité de l'espace de repliement de la molécule.

Dans cet espace de repliement, on peut définir comme notion de voisinage une différence d'une seule paire de bases. Il existe alors des vallées constituées de minimums locaux aux seins desquelles on observe des structures proches et plus ou moins stables. Le problème de la prédiction de structure n'est pas simplement de trouver l'optimum global, mais plutôt de pouvoir mesurer la probabilité de rencontrer la molécule sous une forme en particulier. Sfold [27] produit un échantillon des structures possibles respectant leur probabilité d'apparition, mais ne garantit donc pas la présence de la structure optimale dans l'échantillon.

La génération d'une longue liste de structures candidates confronte le biologiste à une surcharge d'information difficile à analyser. Différentes approches tentent de réduire la quantité de structures candidates proches afin de ne conserver que celles qui sont fondamentalement différentes. RNAshape [28] propose une représentation hiérarchique à gros grain des structures secondaires proche de notre graphe squelette. Cette représentation ignore les longueurs des hélices, les bulges, ou la taille

des boucles internes, et permet de regrouper sous la même "forme" de nombreuses structures présentant de petites variations. Le logiciel produit alors une liste des formes possédant de bons scores d'énergie libre, ce qui limite considérablement le nombre de candidats.

CONTRAfold [29] s'éloigne des modèles physiques reposant sur des paramètres thermodynamiques déterminés expérimentalement. Cette méthode utilise un modèle probabiliste appelé *modèle conditionnel log-linéaire* pour apprendre les paramètres à partir d'un jeu de données, et offre des résultats de qualité similaire aux méthodes citées ci-dessus.

Les approches citées jusqu'à maintenant ignorent les pseudonœuds, pour des raisons de complexité ; les structures sans pseudonœud sont arborescentes (une boucle d'une branche n'interagit pas avec une autre branche), et de nombreux problèmes sont plus simples à résoudre dans les arbres que dans les graphes en général. Toutefois, d'autres méthodes proposent d'inclure les pseudonœuds. Déterminer la structure de plus faible énergie libre comprenant des pseudonœuds est NP-complet [30], mais les pseudonœuds ont été répartis en plusieurs classes, et des approches polynomiales incluant certaines classes sont possibles. Pknots [31] est la première méthode permettant de prédire des structures secondaires avec pseudonœuds, mais sa complexité en temps de $O(n^6)$ la rend difficilement appliquable à de grosses molécules.

CyloFold [32] propose une approche à gros grains permettant de prédire des pseudonœuds sans limitation de classe. Une liste d'hélices possibles est générée, puis les hélices représentées par des cylindres sont tirées et placées aléatoirement dans l'espace afin de respecter des contraintes de distance et de collisions. Plusieurs simulations sont effectuées, et la configuration ayant la meilleure énergie libre est conservée. Notons que, malgré l'utilisation de cylindres et un placement dans l'espace, il s'agit bien de prédiction de structure 2D.

Par ailleurs, Y. Ponty et C. Saul [33] proposent un modèle combinatoire à base d'hypergraphes permettant de formuler et analyser de manière unifiée de nombreux algorithmes de prédiction de structures avec pseudonœuds.

Enfin, d'autres approches tentent de modéliser la cinétique de repliement, ce qui permet de tenir compte du fait qu'un ARN commence à se replier avant même d'avoir été entièrement transcrit. C'est le cas de Kinwalker [34] et Kinefold [35].

3.2.2 Approche comparative

L'approche comparative repose sur le fait que la structure d'une molécule est mieux conservée que sa séquence au cours de l'évolution. Par conséquent, pour une paire de bases $(i \cdot j)$, si le nucléotide i mute, pour que la paire de bases soit conservée, il faut qu'il y ait une mutation compensatoire du nucléotide j. Ainsi, dans l'alignement de trois séquences suivant, les parties grisées co-varient afin de rester complémentaires :

Séquence 1	G	C	C	U	U	C	G	G	G	C
Séquence 2	G	A	C	U	U	C	G	G	U	C
Séquence 3	G	G	C	U	U	C	G	G	C	C

Cette co-variation peut être utilisée pour inférer l'existence d'une paire de bases subissant une pression de sélection à cette position. Les molécules correspondant à chaque séquence ont des structures légèrement différentes les unes des autres, mais partageant beaucoup de paires de bases, qui peuvent être identifiées par ces co-variations. Par conséquent, si nous disposons d'un bon alignement de séquences, nous pouvons obtenir de nombreuses informations sur les structures des molécules concernées.

L'approche comparative utilise un ensemble de séquences homologues, et tente de calculer pour chacune d'elle sa structure *in vivo*, ainsi qu'une structure nommée *consensus*, qui n'existe pas *in vivo*, mais qui représente un dénominateur commun aux structures individuelles trouvées.

Gardner et Giegerich [36] proposent une comparaison des méthodes de prédiction de structure secondaire comparatives disponibles en 2004, et distinguent trois approches, qu'ils nomment plans A, B et C (voir figure 3.1). Nous reprenons cette nomenclature et présentons les principales approches.

Plan A

Cette approche consiste à utiliser un outil d'alignement de séquences, par exemple ClustalW [37], puis à calculer un repliement à partir de cet alignement. RNAalifold [38] calcule une structure consensus à partir de l'alignement reçu en entrée en utilisant une variante de l'algorithme de Zuker-Stiegler [24] incluant des scores de covariation.

Pfold [39] utilise une grammaire stochastique non-contextuelle pour calculer une distribution de probabilités à priori sur les structures d'un ARN. L'alignement de séquences fournit un arbre phylogénétique, qui permet de calculer les probabilités à postériori de ces structures.

Plan B

L'algorithme de Sankoff [40] fusionne l'alignement de séquences, la comparaison phylogénétique et l'algorithme de repliement de Nussinov, en les regroupant en une seule équation de récurrence. Le cas général est de complexité $O(n^{3m})$ en temps et $O(n^{2m})$ en espace, où n est la taille de la plus longue séquence et m le nombre de séquences. En posant des contraintes de plausibilité biologique, Sankoff ramène la complexité en temps à $O(n^3 k^m)$, où k est une constante petite par rapport à n.

Des implémentations simplifiées de l'algorithme de Sankoff permettent de réduire sa complexité. Foldalign [41] réduit sa complexité à $O(n^2 \lambda^2 \delta^2)$ en imposant des limites sur le nombre de séquences (deux), la taille λ des motifs structurels

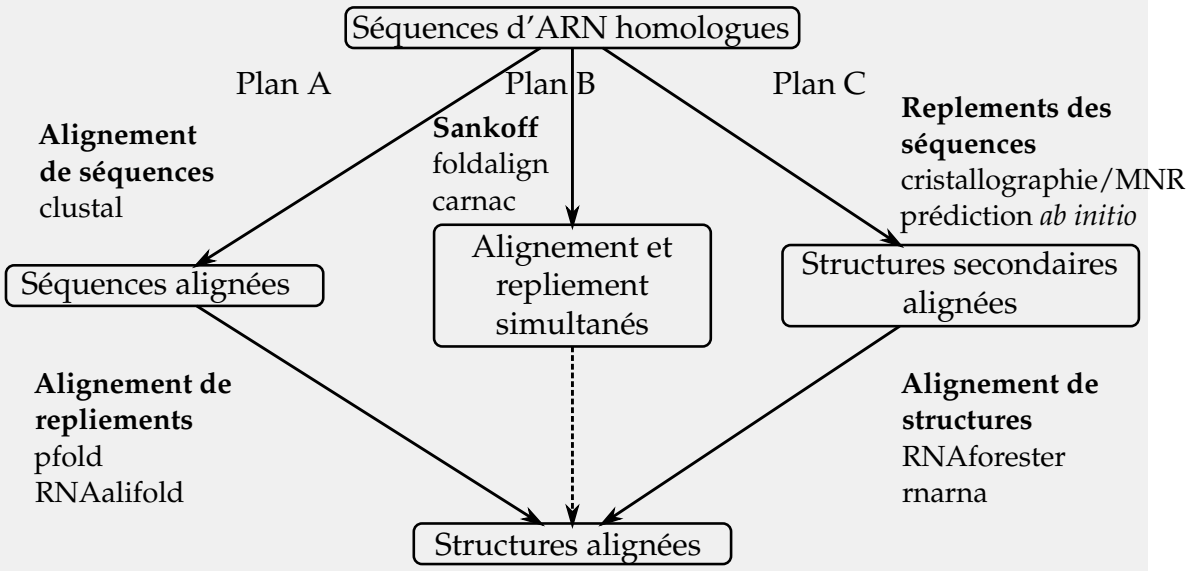


FIGURE 3.1: Prédiction de structure secondaire, approches comparatives. Le plan A consiste à aligner les séquences, puis calculer un repliement à partir de cet alignement multiple ; ce plan nécessite une bonne similarité de séquence. Le plan B corrige en partie ce problème, et regroupe les étapes d'alignement et de repliement, mais nécessite des ressources considérables. Enfin, le plan C est utilisable quand les séquences sont peu similaires, car il n'en tient pas compte. Des structures homologues sont obtenues, puis alignées par des algorithmes d'alignement de structures. Figure inspirée de [36].

recherchés et sur la taille δ des sous-séquences comparées. Dynalign [42], quant à lui, simplifie l'algorithme en limitant le nombre de séquences à deux, la distance maximale δ entre deux nucléotides alignés, ainsi que la taille des boucles internes. La complexité devient alors $O(n^3\delta^3)$.

Carnac [43] recherche d'abord dans les séquences de bonnes tiges candidates (en termes d'énergie et de covariation), puis applique une méthode semblable à celle de Sankoff, mais sur une séquence de tiges potentielles au lieu d'une séquence de nucléotides. L'équation de récurrence donne une complexité élevée, $O(n^6)$, mais Carnac implémente diverses optimisations, dont les auteurs estiment qu'elles font passer la complexité à environ $O(n^4)$.

Plan C

Enfin, lorsque les séquences sont trop peu similaires pour pouvoir utiliser des critères de co-variation, il est encore possible de compter sur le fait que la structure est plus conservée que la séquence. Dans cette approche, chaque séquence est repliée séparément, au moyen d'un logiciel de repliement simple-séquence, puis les structures obtenues sont alignées. RNAforester [44, 45] fait partie du *Vienna RNA*

Package, et permet de calculer des alignements globaux ou locaux de structures secondaires arborescentes (donc sans pseudonœuds).

Alors que RNAforester ne fait que des alignements de structure, Marna [46] aligne simultanément sur des critères de similarité de séquence et de structure. La complexité d'un alignement de n séquences de longueur l, possédant chacune e structures candidates est de $O(e^2n^2l^2) + O(n^3l^2)$.

3.2.3 Comparatifs

La table 3.1 recence les complexités en temps et en espace des principales méthodes citées ici. La figure 3.2 reprend quelques résultats de l'étude de performances de [36] ; en bref, les approches comparatives sont plus efficaces que les méthodes simple-séquence, et d'autant plus efficaces que les séquences sont similaires. Toutefois, la plupart de ces méthodes ne prédisent pas encore les pseudonœuds.

Méthode	Pseudo-nœuds	Séquences	Complexité	
			temps	mémoire
Nussinov	non	1	$O(n^3)$	$O(n^2)$
Mfold	non	1	$O(n^3)$	$O(n^2)$
CONTRAfold	non	1	$O(n^3)$	$O(n^2)$
Pknots	oui	1	$O(n^6)$	$O(n^4)$
CyloFold	oui	1	$\sim O(n^{4.47})$	
Kinwalker	non	1	$\sim O(n^{4.64})$	
RNAalifold	non	m	$O(mn^2 + n^3)$	$O(n^2)$
Sankoff	non	m	$O(n^{3m})$	$O(n^{2m})$
Foldalign	non	2	$O(kn^2)$	$O(k'n^2)$
Dynalign	non	2	$O(kn^3)$	
Carnac	non	2	$O(n^6)$	$O(n^4)$

TABLE 3.1: Complexité des algorithmes de prédiction de structure secondaire. Le nombre n désigne le nombre de nucléotides de la séquence, m le nombre de séquences dans le cas des approches comparatives, et k désigne une constante (différente d'un algorithme à l'autre) qui peut "cacher" une complexité découlant de paramètres ajustables. Le détail de ces paramètres est donné dans le corps du texte.

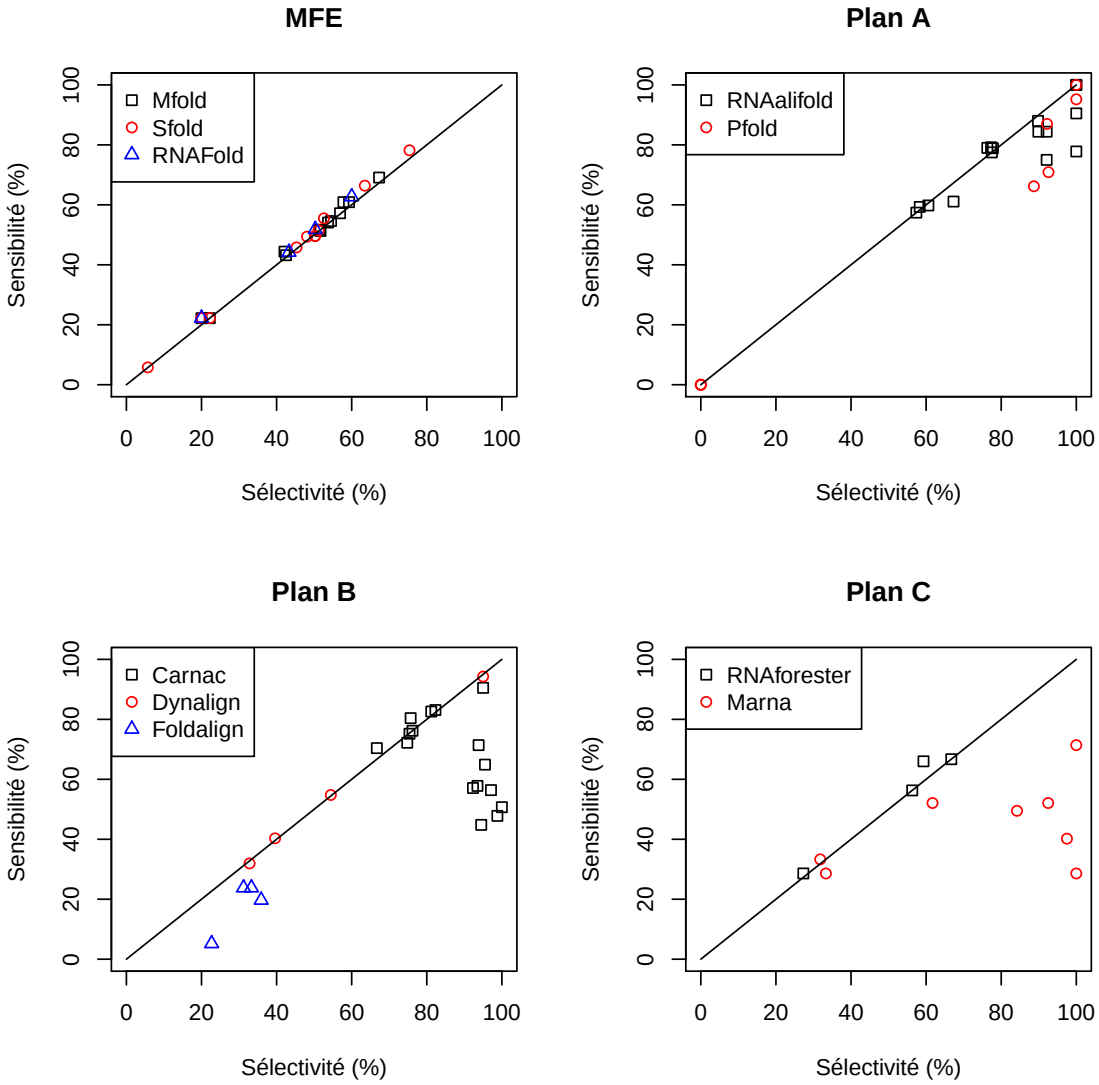


FIGURE 3.2: Performances des algorithmes de prédiction de structure secondaire comparés dans [36], sur quatre molécules (S. cerevisiae PHE-tRNA, E. Coli RNase P et les deux sous-unités du ribosome d'E. Coli). La sélectivité (ou PPV) et la spécificité sont calculées sur les taux de prédiction des paires de bases individuelles (Voir annexe B.1.1 pour les définitions). Pour les méthodes Mfold et Sfold, trois structures candidates ont été retenues pour chaque molécule, chaque point correspond donc à un couple (molécule, structure candidate). Pour les méthodes comparatives, plusieurs points peuvent correspondre à la même molécule, selon le jeu d'alignements utilisé, et selon que la structure consensus a été re-repliée avec RNAfold ou non (voir [36] pour les détails).

3.3 Structure tertiaire

3.3.1 Présentation des logiciels

Il y a deux manières d'aborder la prédiction de structure tertiaire d'une molécule : soit *ab initio* à partir de sa séquence, en modélisant l'ensemble du processus de repliement, soit en supposant que nous connaissons déjà la structure secondaire de la molécule. Dans ce dernier cas, la première approche consiste à construire les représentations 3D des hélices, qui sont des motifs très réguliers, puis à essayer de placer ces hélices dans l'espace et les relier entre-elles par des brins non appariés. RNA2D3D [47] et Assemble [48] offrent des interfaces graphiques permettant d'effectuer cela de manière interactive. Malheureusement, ces outils interactifs nécessitent une connaissance d'expert.

Les approches automatisées fonctionnent pour la plupart soit à partir de la séquence, soit en ajoutant des contraintes de structure secondaire, ce qui améliore sensiblement leurs performances. iFoldRNA [49, 50] et NAST [51] modélisent le processus de repliement par des simulations de dynamique moléculaire à gros grain. iFoldRNA représente les nucléotides par trois sphères représentant le groupement phosphate, le sucre et la base. Une fonction de potentiel représente les différents types d'interactions, et le voisinage énergétique du système est exploré en effectuant plusieurs simulations parallèles à différentes températures. NAST représente les nucléotides par une seule sphère, et utilise une fonction de potentiel calculée à partir des structures ribosomales connues ; NAST nécessite une structure secondaire et, si possible, des informations de contacts tertiaires en entrée.

FARNA [52] s'inspire de la méthodologie Rosetta utilisée pour la prédiction de structures de protéines. Les nucléotides sont représentés par une sphère placée au centre de la base. Une base de données de fragments 3D de trois nucléotides tirés de structures ribosomales est utilisée pour reconstituer les structures au moyens d'une méthode de Monte-Carlo. La fonction optimisée tient compte des torsions, collisions et potentiels d'appariement ou d'empilement de bases.

Enfin, le couple MC-Fold / MC-Sym [53] utilise une base de données de motifs 3D de couples de paires de bases empilées et de boucles internes provenant de 500 structures cristallographiées. MC-Fold utilise ces motifs et une fonction d'énergie libre pour prédire des structures secondaires, puis MC-Sym construit un modèle atomique de ces structures. Un algorithme de Monte Carlo est utilisé pour insérer les motifs de la base de donnée dans les structures prédites.

3.3.2 Comparaison

Dans cette section, nous reprenons une partie des résultats obtenus par [22] sur 41 molécules. Leur première observation est que, si iFoldRNA et FARNA ont proposé des prédictions de structures pour toutes ces molécules, MC-Fold/MC-Sym et NAST n'y sont pas toujours parvenu. En règle générale, MC-Fold/MC-Sym

offre de meilleures prédictions, mais ne parvient pas à produire de prédiction dans la moitié des cas étudiés. La figure 3.3 présente les performances relatives de MC-Fold/MC-Sym et FARNA lorsqu'on leur demande un ensemble de cinq structures sous-optimales, sur un jeu de données de six structures comportant des boucles internes et des 3-jonctions. On y observe de bons résultats de MC-Fold/MC-Sym sur de petites molécules ne comportant que des boucles internes, mais les performances se dégradent quand la complexité des molécules augmente (la base de données de motifs utilisée ne comporte que des 2-jonctions).

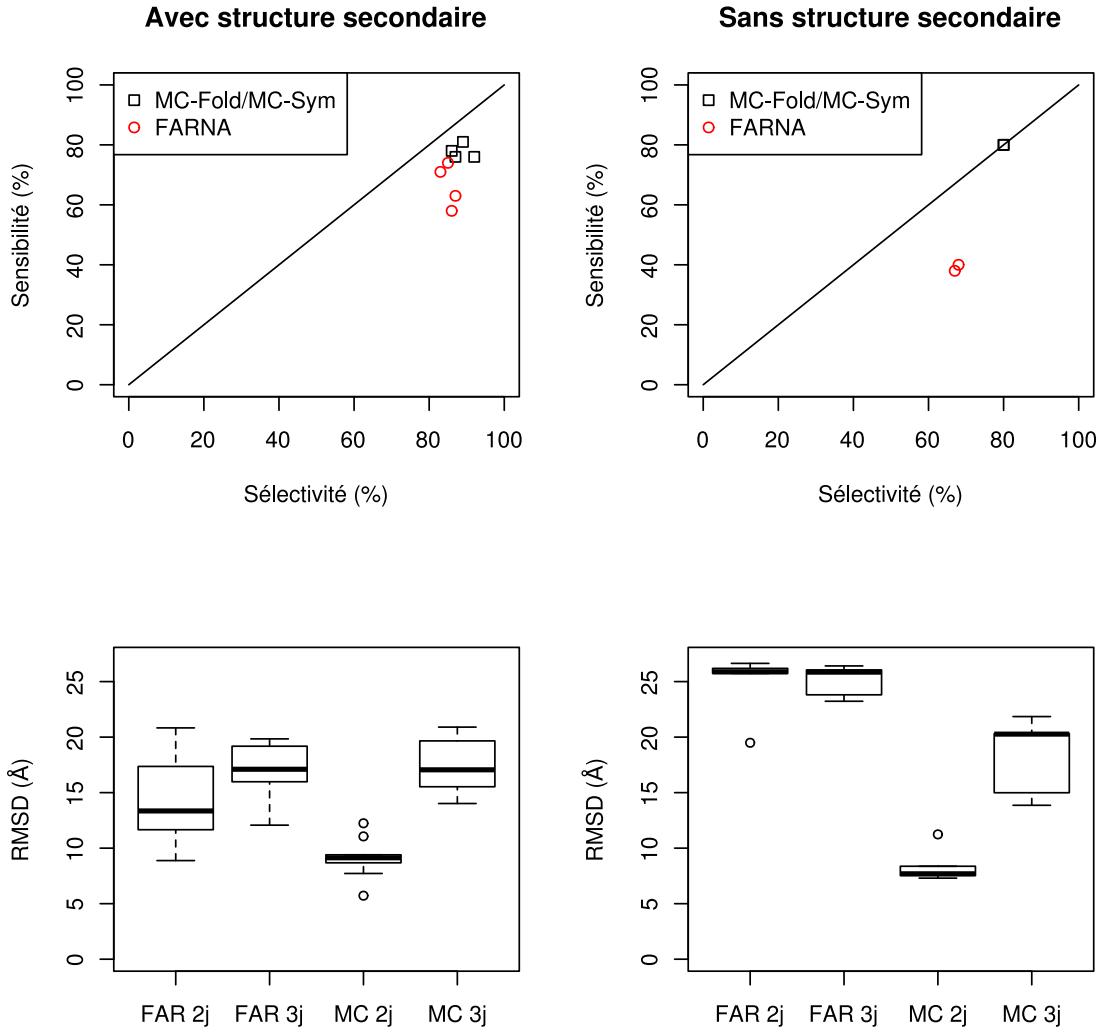
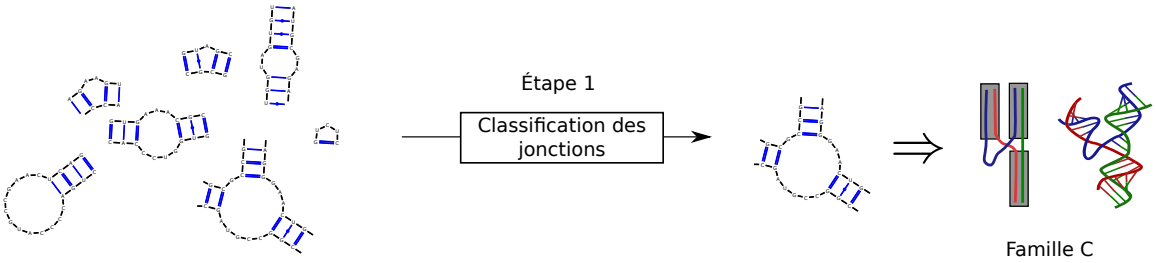


FIGURE 3.3: Performances de MC-Fold/MC-Sym et FARNA étudiés dans [22], sur six petites molécules. La colonne de gauche présente les performances lorsque les logiciels reçoivent en entrée une structure secondaire, et celle de droite présente leurs performances lorsqu'ils ne reçoivent en entrée qu'une séquence. La première ligne présente, de manière similaire à la figure 3.2, la sélectivité et sensibilité de la détection d'interactions (secondaires ou tertiaires). La deuxième ligne représente les distributions des RMSD selon la méthode et selon que la molécule ne contenait que des boucles internes (2j) ou également une 3-jonction (3j). MC-Fold/MC-Sym est meilleur que FARNA sur les molécules de structure simple (2j), et également bien meilleur lorsque seule la séquence est disponible.

Deuxième partie

Classification des éléments locaux

4 Introduction



Cette partie s'intéresse à la première étape de notre processus de repliement : la classification en familles topologiques des jonctions entre hélices. Si la forme en trois dimensions d'une hélice Watson-Crick est bien définie, celle des jonctions l'est moins. Deux hypothèses seraient envisageables : soit une jonction donnée a une forme bien définie, déterminée localement et qui serait la même si l'on transposait cette jonction dans une autre molécule, soit la jonction est « souple », et sa forme est définie par le contexte de la molécule. La nature modulaire [1] des 2-jonctions est bien connue, et il a également été montré dans [12] que des caractéristiques locales des 3-jonctions permettent de manière fiable de déterminer leur forme.

Nous nous proposons de montrer qu'une classification en un petit nombre de familles topologiques peut être déterminée automatiquement, à partir uniquement de la structure secondaire, pour les 3-jonctions (chapitre 5), puis nous proposerons des approches de classification similaires pour les 2- et 4-jonctions, et évoquerons la difficulté d'en faire de même pour les jonctions de plus haut degré (chapitre 6). Nous ne considérerons pas une éventuelle classification des 1-jonction, les boucles terminales des hélices : si ces dernières peuvent être structurées, leur structure n'influence pas les angles relatifs des hélices et ne nous concerne pas dans une approche à gros grain.

1. Jonction modulaire : qui a la même forme indépendamment de la molécule dans laquelle elle se trouve ; voir section 2.3 page 9.

5 Jonctions à trois branches

5.1 Introduction

Le cas particulier des jonctions entre trois hélices a été étudié par Aurélie Lescoute et Éric Westhof dans [12]. Dans cet article, les auteurs étudient trente-trois jonctions réparties entre dix molécules, et les regroupent en trois familles en fonction de leur forme en 3D. Ces trois familles sont représentées schématiquement dans la figure 5.1. Dans les jonctions étudiées, les auteurs observent toujours un empilement vertical entre deux hélices, l'angle formé par la troisième par rapport à cet empilement déterminant la famille.

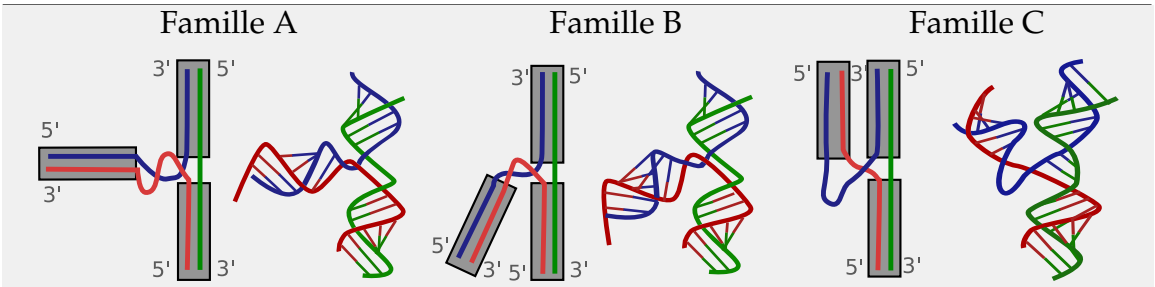


FIGURE 5.1: Les trois familles topologiques définies dans [12]. Les brins sont orientés en fonction de l'ordre de transcription ; ainsi, les familles B et C ne sont pas interchangeables par permutation horizontale du dessin.

La topologie de chacune de ces familles est due aux différentes interactions non canoniques qui interviennent entre les hélices ou entre les hélices et les autres nucléotides de la jonction. Lescoute et Westhof montrent dans [12] que certains critères locaux de séquence et structure secondaire sont spécifiques à certaines familles, et peuvent aider à déterminer à quelle famille appartient une jonction sans avoir besoin d'observer sa structure cristallographiée. En particulier, ils observent que les deux hélices reliées par le brin non apparié le plus court sont généralement celles qui s'empilent verticalement, et que c'est toujours le cas si un des trois brins non appariés est de longueur zéro. Toutefois, la frontière entre les trois familles est floue, et le choix d'une famille découle d'une estimation à partir des différents critères, estimation laissée à la charge du biologiste.

Dans ce chapitre, nous développons une approche automatisée permettant de prédire la famille topologique d'une jonction à partir de sa structure secondaire. Nous montrons que, parmi neuf possibilités, notre approche choisit la bonne configuration dans 64% des cas. Nous montrons également que la précision des prédictions est nettement améliorée par l'ajout d'une information d'homologie, c'est à dire d'un ensemble de structures ou de séquences homologues à la séquence d'entrée. Un article concernant ce travail a été publié dans [54].

5.2 Données

5.2.1 Jeux de données

Nous avons distingué trois jeux de données : premièrement, le jeu LW, contenant les jonctions étudiées dans l'article de Lescoute et Westhof [12] ; c'est sur ce jeu de données que nous avons conçu notre méthode de classification. Deuxièmement, le jeu FR3D comporte des jonctions extraites automatiquement à partir de la base de données du même nom [17] (voir la section 5.2.2). Ce jeu de données nous a permis de valider l'approche. Enfin, le jeu de données Tout englobe la totalité des jonctions dont nous disposons. La table 5.1 indique la composition de ces jeux de données.

Jeu de données	A	B	C	Total
LW	10	7	16	33
FR3D	16	13	24	53
Tout	26	20	40	86

TABLE 5.1: Composition des jeu de données en fonction des familles A, B et C. Le jeu Tout est l'union des jeux LW et FR3D.

5.2.2 Extraction des jonctions

À partir de fichiers PDB [55] provenant de la base de données non-redondante FR3D [17], nous avons extrait la structure secondaire des molécules en utilisant RNAView [56]. Nous avons ensuite retiré les pseudonœud à l'aide de K2N [57], en utilisant les paramètres par défaut. Enfin, nous avons extrait les jonctions à trois branches, en définissant ces jonctions comme des jonctions reliant trois hélices, en considérant comme dans [12] qu'une hélice doit comporter aux moins deux paires de bases Watson-Crick consécutives.

Parmi toutes les jonctions extraites, nous avons conservé chaque instance de jonctions homologues, pourvu qu'elles diffèrent en séquence ou en structure secondaire. Deux jonctions sont homologues si elles se trouvent à la même position dans les structures tertiaires de deux molécules homologues. C'est le cas, par exemple,

des jonctions 1J5E_001 et 2AVY_29 que l'on trouve, respectivement, dans les ARN ribosomaux 16S de *T. thermophilus* et *E. coli*. Deux jonctions homologues sont « la même » jonction au sens biologique, mais pas du point de vue de nos critères de classification. Par conséquent, nous voulons pouvoir vérifier si elles sont classifiées dans la même famille malgré leurs différences de structure. De plus, nous avons étudié la manière dont nous pouvions utiliser cette information d'homologie pour améliorer la qualité de nos prédictions (voir section 5.3.4).

Nous avons exclu deux jonctions de nos données : la jonction provenant de 3E5C (SAM-III riboswitch) à cause d'un ligand changeant sa conformation, et une jonction de 2A64 (RNase P bactérienne) parce qu'un de ses brins non appariés forme un pseudonœud avec une autre partie de la molécule. Ceci pris en compte, nous disposons de 86 jonctions possédant des structures secondaires différentes. Si nous regroupons ces jonctions par homologie, nous obtenons 39 classes de jonctions homologues. La table 5.7, page 43, liste les 26 classes contenant plus d'une jonction.

5.3 Méthode de prédiction

5.3.1 Processus de prédiction

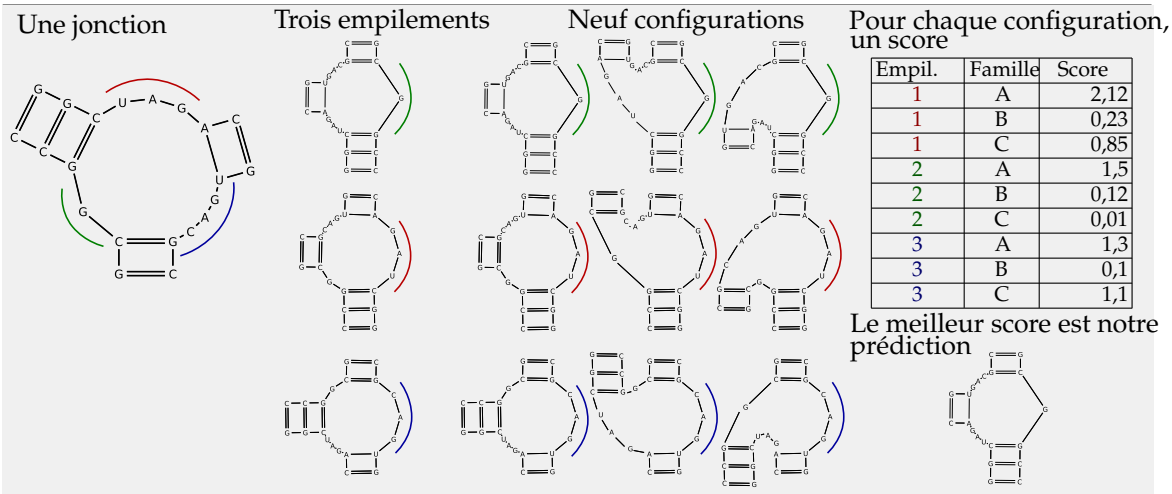


FIGURE 5.2: Processus de prédiction. Une jonction peut avoir trois empilements possibles, et chaque empilement peut ensuite appartenir à trois familles, ce qui nous donne neuf configurations. Nous calculons un score pour chaque configuration, et celle obtenant le meilleur score est notre prédiction pour la jonction étudiée.

La figure 5.2 donne une vision d'ensemble du processus de prédiction. Pour une 3-jonction donnée, trois empilements sont possibles et, pour chaque empilement, il y a trois possibilités de familles – A, B ou C. Notre méthode de prédiction doit donc choisir entre neuf configurations possibles. Nous allons attribuer un score à

chacune de ces configurations ; ce score est noté s_c pour une configuration c donnée. Le score s_c est une combinaison linéaire de quatre scores partiels, notés $s_{c,i}$ avec $i \in \{1, 2, 3, 4\}$ et détaillés dans la section suivante. Chacun de ces scores partiels est calculé à partir d'un critère spécifique ; en effet, selon [12], la classification des 3-jonctions dépend d'un petit nombre de paramètres. Enfin, nous choisissons la configuration possédant le meilleur score global comme notre prédiction.

5.3.2 Calcul des scores partiels

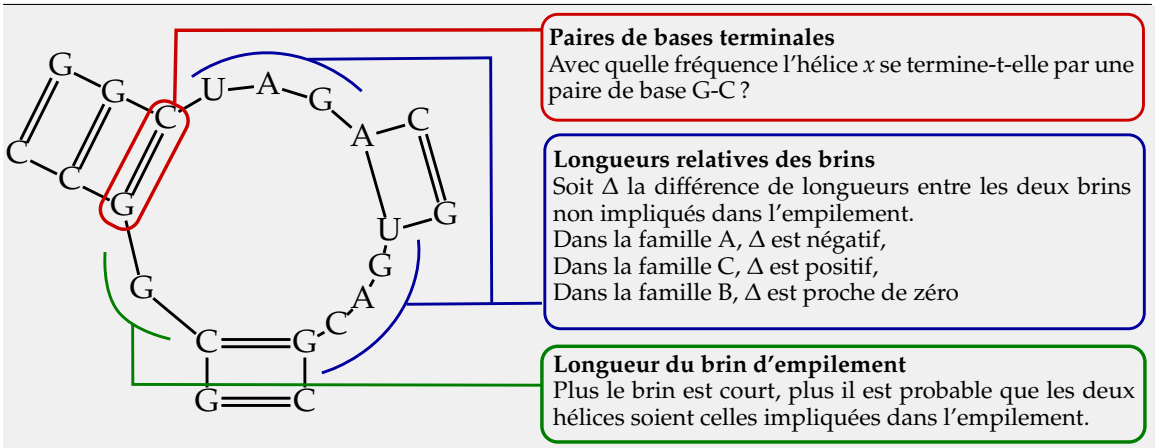


FIGURE 5.3: Résumé des critères de prédiction considérés.

Nous considérons des critères de séquence et de structure secondaire, et excluons tout critère de structure tertiaire ; pour chaque critère, nous calculons un score. Ces critères sont résumés dans la figure 5.3, et détaillés ci-dessous. Le pseudo-code correspondant à chaque calcul de score est donné à l'appendice A, page 125.

Longueur du brin d'empilement, $s_{c,1}$

Lorsque deux hélices sont empilées co-axialement, il s'agit en général de celles reliées par le brin le plus court. Nous définissons donc le score $s_{c,1}$ de sorte que pour une configuration donnée, plus le brin reliant les deux hélices empilées est court, meilleur est le score. Si ce brin est de longueur zéro, nous attribuons à la configuration un score encore plus élevé, car il n'y a aucun contre-exemple à cette règle dans le jeu de données LW.

Soit x_i la longueur du brin non apparié i, et s_{x_i} le score pour le test d'empilement lorsque l'on essaye d'empiler sur le brin i. La formule utilisée est alors la suivante :

$$s_{x_i} = \frac{1}{1 + x_i} + \sum_{j \neq i} \frac{1}{1 + \max(0, x_i - x_j)} \quad \text{si} \quad x_i > 0 \tag{5.1}$$

$$s_{x_i} = 10 \text{ si } x_i = 0 \tag{5.2}$$

En laissant de côté le cas particulier où $x_i = 0$, expliquons la construction de l'équation 5.1. La première partie, $1/(1 + x_i)$, dépend uniquement de la longueur du brin i ; plus le brin est long, plus sa valeur est petite. Ensuite, nous comparons la longueur du brin i avec les longueurs des autres brins. Lorsqu'un brin j est plus court que i, nous ajoutons $1/(1 + x_i - x_j)$ à s_{x_i} ; autrement, nous ajoutons 1. Le but de cette expression est de représenter le fait qu'il n'est pas identique d'avoir une longueur de 7 lorsque les autres brins sont de longueur 1 ou de longueur 6.

Par exemple, soient a, b et c les longueurs des trois brins non appariés d'une jonction, et supposons que $0 < a \leq b \leq c$. Dans ce cas, les scores sont :

$$\begin{aligned} s_a &= \tfrac{1}{1+a} + 1 + 1 \\ s_b &= \tfrac{1}{1+b} + \tfrac{1}{1+b-a} + 1 \\ s_c &= \tfrac{1}{1+c} + \tfrac{1}{1+c-a} + \tfrac{1}{1+c-b} \end{aligned}$$

Ainsi, si $a = 6, b = 7$ et $c = 8$, nous avons :

$$\begin{aligned} s_a &= \tfrac{1}{7} + 1 + 1 \simeq 2{,}14 \\ s_b &= \tfrac{1}{8} + \tfrac{1}{1+1} + 1 \simeq 1{,}63 \\ s_c &= \tfrac{1}{9} + \tfrac{1}{1+2} + \tfrac{1}{1+1} \simeq 0{,}94 \end{aligned}$$

Mais si $a = 1, b = 7$ et $c = 8$, nous avons :

$$\begin{aligned} s_a &= \tfrac{1}{2} + 1 + 1 = 2{,}5 \\ s_b &= \tfrac{1}{8} + \tfrac{1}{1+6} + 1 \simeq 1{,}27 \\ s_c &= \tfrac{1}{9} + \tfrac{1}{1+7} + \tfrac{1}{1+1} \simeq 0{,}74 \end{aligned}$$

Longueurs relatives des brins, $s_{c,2}$

Dans une configuration donnée, un des brins est impliqué dans un empilement. Appelons les deux autres brins haut et bas selon leur position (dans la figure 5.1, le brin haut est bleu, et le brin bas est rouge). Dans la famille A, le brin bas est plus court que le brin haut, et inversement dans la famille C. Dans la famille B, les deux brins sont à peu près de la même longueur. La figure 5.4 représente la distribution des longueurs respectives des brins haut et bas des jonctions du jeu de données LW.

Pour traduire cela en score, nous calculons Δ, la différence entre les longueurs de haut et bas. La distance entre Δ et 0 détermine le score d'une configuration donnée, comme indiqué dans la table 5.2.

Paires de bases terminant les hélices, $s_{c,3}$

Nous avons numéroté les hélices d'une jonction en fonction de l'ordre de transcription (la première hélice est celle contenant le nucléotide de plus petit identifiant). Nous avons ensuite calculé la fréquence d'apparition de chaque type de

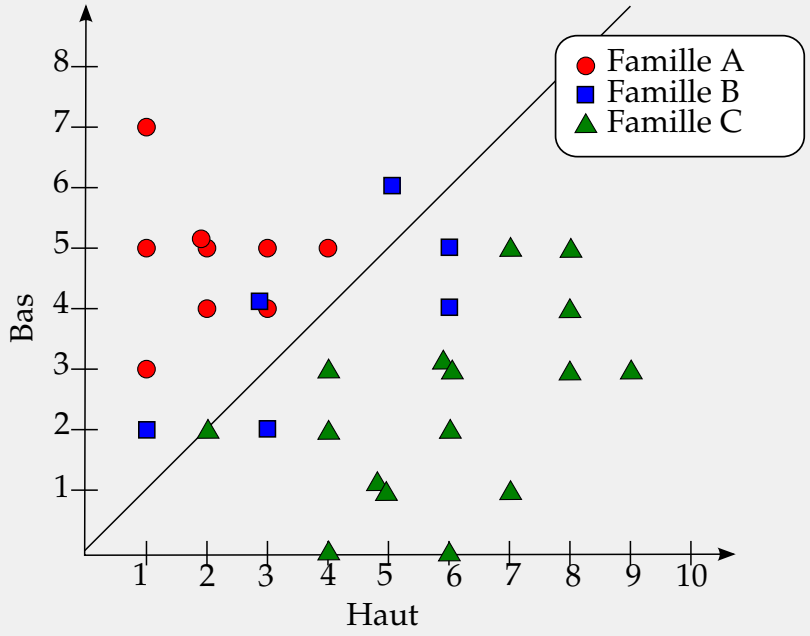


FIGURE 5.4: Longueurs relatives des brins haut et bas, selon la famille.

A	B	C
	5 si $\Delta = 0$ ou 1	
Δ	1 si $\Delta = 2$	$-\Delta$
	$3 - \Delta$ sinon	

TABLE 5.2: Le score $s_{c,2}$ selon les longueurs des brins haut et bas.

paire de base dans les hélices de chaque famille, et construit une matrice de score pour chacune de ces hélices, ce qui nous donne trois matrices, M_1, M_2 et M_3. La table 5.3 présente une de ces matrices de score. Pour une configuration $c = (f,e)$ donnée, correspondant à une famille f et un empilement e, dont les hélices se terminent par les paires de base h_1, h_2 et h_3, le score $s_{c,3}$ est la somme des scores $M_1[f, h_1] + M_2[f, h_2] + M_3[f, h_3]$.

Décrivons maintenant comment nous transformons une matrice d'occurrences (5.3 a) en matrice de scores (5.3 b). La transformation se fait par colonnes, et toutes les colonnes sont traitées de la même manière ; nous allons donc nous intéresser à une colonne d'une matrice. Nous voulons transformer la colonne x d'une matrice d'occurences en la colonne y d'une matrice de scores.

Soit s la somme des k rangées de la colonne x :

$$s = \sum_{0 \leq i \leq k} x_i$$

a)

A	B	C	
0	0	2	AU
5	4	2	CG
4	2	7	GC
1	1	1	UA
0	0	0	GU
0	0	3	UG

b)

A	B	C	
−0,20	−0,20	−0,04	AU
0,40	0,49	−0,04	CG
0,28	0,14	0,36	GC
−0,08	−0,03	−0,12	UA
−0,20	−0,20	−0,20	GU
−0,20	−0,20	0,04	UG

TABLE 5.3: Tables d'occurrences (a) et de scores (b) pour les différentes paires de bases possibles de l'hélice 1, selon la famille. Par exemple, si la configuration évaluée est (*empilement* x, *famille A*), et si nous avons une première hélice se terminant en U-A, nous ajoutons −0,08 au score de cette configuration.

Nous calculons alors les cellules y_i ainsi :

$$y_i = (x_i - \frac{s}{k}) \times \frac{k}{s(k-1)} \tag{5.3}$$

La première partie de l'équation 5.3 compare la valeur de la cellule courante x_i à la moyenne des cellules de x. La sconde partie de l'équation est un facteur de normalisation. Pour un exemple de calcul, voir la table 5.3. Notons, dans cette table, que la somme des valeurs de chaque colonne est nulle.

Jeu d'apprentissage	28 jonctions
Jeu de test	56 jonctions
Nombre d'essais	100
Pourcentage de réussite moyen	62,45%
Écart type	4,95
Pourcentage de réussite avec les matrices tirées de LW	64,0%

TABLE 5.4: Qualité des matrices de paires de bases terminales dérivées de [12], et dépendance de ces matrices au jeu d'apprentissage. En calculant ces matrices sur un tiers du jeu de données, choisi aléatoirement, nous obtenons un taux de réussite similaire à celui que nous obtenions avec les matrices d'origine.

On peut se demander si les jonctions utilisées pour le calcul des matrices de score pour les paires de bases terminales sont représentatives de la distribution générale des paires de bases terminales. En utilisant le jeu de données complet, nous avons sélectionné aléatoirement un tiers des jonctions pour calculer ces matrices, et testé les prédictions sur le reste des données. Nous avons reproduit ce processus cent fois. Les résultats, dans la table 5.4, montrent que les jonctions provenant de LW sont légèrement plus représentatives que des jonctions sélectionnées aléatoirement, mais que la méthode fonctionne bien avec n'importe quel ensemble d'une trentaine de jonctions.

Critère bonus, $s_{c,4}$

Ce critère prend en compte certaines possibilités de formation de liaisons tertiaires, et se base sur les observations mentionnées dans [12, page 86]. Les bonus sont les suivants :

- Famille A : le premier critère bonus envisage la possibilité d'une interaction de type A-mineur entre le brin le plus long et l'hélice du bas. Nous attribuons à la configuration évaluée un bonus de $+1$ s'il y a un A sur ce brin. Le second critère bonus vérifie si l'hélice du bas peut se terminer par une interaction non-Watson-Crick contenant un A ($+1$) ou, encore mieux, un A et un G ($+1{,}5$).
- Famille B : nous n'avons pas de critère bonus pour cette famille, ce qui la rend plus difficile à prédire que les autres (voir plus loin).
- Famille C : dans cette famille, un des brins est souvent « structuré comme une tige-boucle (utilisant le motif standard *U-turn*), et souvent terminée par une paire de base. » Nous recherchons une telle possibilité en comptant les A et les U de ce brin, et en ajoutant un bonus de $\frac{a+b}{2}$ à cette configuration si $a > b$.

5.3.3 Calcul de score

Soit $s_{c,i}$ le score pour la configuration c et le test i. Pour un test donné, les neuf configurations possibles peuvent être ordonnées selon leur score ; soit $r_{c,i}$ le rang de la configuration c pour le test i. De plus, un poids w_i est attribué à chaque test. Le score total d'une configuration c est alors déterminé par la formule suivante :

$$s_c = \frac{\sum_i w_i s_{c,i}}{\sum_i r_{c,i}} \tag{5.4}$$

La configuration qui obtient le score s_c le plus élevé est notre prédiction pour cette jonction.

Nous divisons la somme des scores par la somme des rangs afin de nous assurer que, pour avoir un score total élevé, une configuration doit être bonne sur de nombreux tests, par rapport aux autres configurations possibles. Par exemple, pour une configuration donnée, si la somme de ses scores partiels est 42, et ses rangs sont 1, 3, 3, et 3, elle obtient un score total s_c de 4,2, mais si ses rangs sont 1, 1, 1 et 1, elle obtient un score s_c de 10,5. Ainsi, un bon score dans un test ne compense pas des scores médiocres dans les autres tests.

Le choix des pondérations des tests a été effectuée empiriquement, en explorant toutes les combinaisons de scores entre 0 et 4 par incréments de 0,5 sur le jeu de données LW, et en retenant celles fournissant les meilleures prédictions. Un certain nombre de choix de coefficients étaient équivalents sur ce jeu de données. Parmi ceux-ci, nous avons sélectionné ceux qui donnaient de bons résultats également sur le jeu de données Tout. Enfin, parmi les choix restants, nous avons choisi le plus simple, correspondant à des poids de 1 pour tous les tests, sauf le test des paires de bases terminales, qui a un poids de 2.

5.3.4 Calcul de score tenant compte des informations d'homologie

Considérons maintenant le problème pour lequel les données d'entrée ne sont pas une seule jonction, mais un ensemble de n jonctions homologues $J_1, \ldots, J_n$. En tant que jonctions homologues, elles sont supposées appartenir toutes à la même famille topologique.

Dans ce cas, nous calculons un score global à cet ensemble, comme suit. Soit $s_{c,i}(k)$ le score pour la jonction J_k, la configuration c et le test i. Alors, nous posons :

$$S_{c,i} = \frac{\sum_k s_{c,i}(k)}{n} \tag{5.5}$$

et le score total global pour une configuration c est :

$$S_c = \frac{\sum_i w_i S_{c,i}}{\sum_i r_{c,i}} \tag{5.6}$$

En d'autres termes, nous calculons, pour chaque test, la moyenne des scores de chaque jonction, et nous calculons ensuite le score final comme nous le faisions précédemment.

5.4 Résultats et discussion

5.4.1 Résultats de prédiction

Jeu de données	Taille	Première position	Trois premières positions
LW	33	20 (60,6%)	31 (93,4%)
FR3D	53	35 (66,0%)	44 (83,0%)
Tout	86	55 (64,0%)	75 (87,2%)

TABLE 5.5: Qualité des prédictions sur les différents jeux de données. La configuration correcte obtient le meilleur score dans 64% des cas. Dans 87% des cas, la configuration correcte est parmi les trois meilleurs scores (sur neuf au total).

Pour les trois jeux de données décrits à la section 5.2.1, nous avons calculé combien de fois nous prédisions la réponse correcte en première position, ou dans les trois premières positions parmi les neuf possibilités (voir table 5.5). Choisir aléatoirement une configuration placerait la réponse correcte en première position 11% du temps, et dans les trois premières positions 33% du temps.

Nous prédisons la configuration correcte en première position 64% du temps, et dans les trois premières positions 87% du temps. Cela montre que nous pouvons exclure de manière fiable six configurations sur neuf.

En observant ces résultats, on pourrait penser que le critère de longueurs de brins prédit correctement l'empilement (excluant ainsi six possibilités), et que la difficulté réside dans le choix de la famille. Autrement dit, que les trois meilleurs scores correspondent aux familles A, B et C pour un même empilement. Ce n'est pas le cas ; que la bonne réponse soit en seconde ou troisième position n'implique pas que les trois premières configurations aient le même empilement. Par conséquent, disposer de connaissances supplémentaires concernant l'empilement pourrait nous aider à choisir parmi ces cas ; c'est ce que nous allons étudier dans les deux prochaines sections.

5.4.2 Méthode de Tyagi et Mathews pour prédire l'empilement

Nous avons utilisé l'approche décrite par Tyagi et Mathews [16] pour prédire l'empilement. Un défaut de cette méthode est qu'elle ne peut fonctionner que sur des jonction ayant relativement peu de bases non appariées, car un brin doit contenir moins de deux nucléotides non appariés pour que le modèle *nearest-neighbor* puisse y prédire un empilement ; cette méthode ne peut donc prédire correctement l'empilement de certaines de nos données.

Dans notre application, cette approche prédit correctement huit empilements (25% de tous les empilements) dans le jeu de données LW, et ne prédit pas les vingt-cinq autres empilements. Ces vingt-cinq faux négatifs sont dus à trois causes :

- la limitation mentionnée précédemment fait que pour certaines jonctions, la méthode ne pouvait pas prédire le bon empilement
- de mauvaises prédictions de la méthode, mentionnées comme faux négatifs par Tyagi et Mathews
- des différences dans nos définitions d'un empilement (voir la discussion dans [16, page 944]) font que certains de nos faux positifs sont considérés par Tyagi et Mathews comme de vrais négatifs.

Pour les huit cas positifs, nous prédisions déjà l'empilement correct avec notre simple critère de longueurs de brins. Utiliser les prédictions d'empilement de Tyagi et Mathews n'améliore pas nos prédictions, ce qui suggère que notre critère simple est « suffisant » en première approximation.

5.4.3 MC-Fold pour prédire les interactions tertiaires

Notre approche ne prend pas en compte les interactions tertiaires. Cependant, des logiciels de prédiction de structure secondaire comme MC-Fold [53] incluent certains types d'interactions tertiaires dans leurs prédictions. Afin de vérifier si cela pouvait nous aider, nous avons utilisé MC-Fold pour tenter de prédire les interactions tertiaires au sein de nos jonctions. Nous avons pris les jonctions du jeu de données LW, contraint leurs interactions Watson-Crick afin que MC-Fold ne puisse les modifier, et appliqué MC-Fold sur ces jonctions.

La première observation est que nous ne retrouvons la jonction « réelle », avec ses interactions tertiaires, dans les dix premiers candidats renvoyés par MC-Fold, seulement dans un cas, car la jonction de départ était très contrainte au départ et contenait peu d'interactions tertiaires. En règle générale, d'une structure candidate à l'autre, les interactions tertiaires sont très variées, mais rarement proches de la structure réelle. Par conséquent, nous ne pouvons attendre de MC-Fold qu'il nous produise une structure tertiaire parfaite pour nos jonctions. Cependant, il est possible que ses préditions nous fournissent suffisamment d'indices pour nous aider à en déterminer l'empilement, ce qui faciliterait notre prédiction.

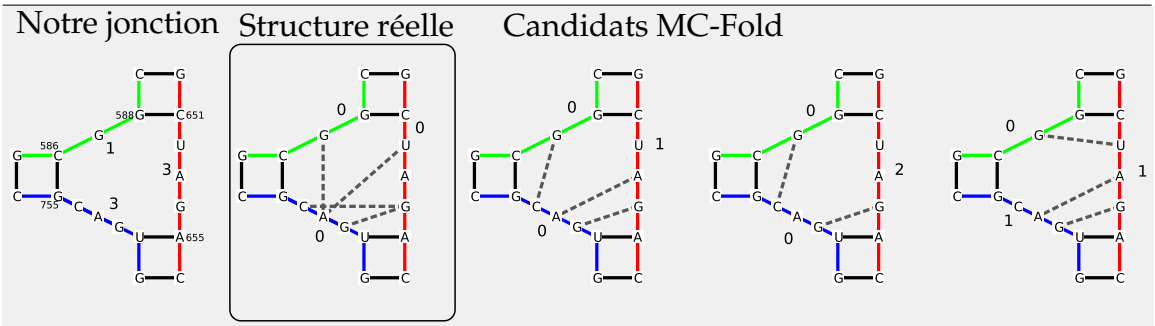


FIGURE 5.5: La structure secondaire de la jonction 1J5E_001, sa structure tertiaire extraite d'une structure cristallographiée, et trois des premières structures candidates produites par MC-Fold. Les nombres indiquent les longueurs des brins non appariés que nous avons utilisé pour calculer les scores d'empilement. Nous considérons que les prédictions de MC-Fold nous aident lorsque la longueur du brin d'empilement est raccourcie en comparaison des longueurs des autres brins. Dans l'exemple présenté ici, la longueur du brin rouge est réduite de 3 à 2 ou 1, mais la longueur du brin vert est toujours réduite à zéro, produisant ainsi un score d'empilement bien plus élevé pour ce brin.

Nous avons pris vingt jonctions de LW sur lesquelles nous avons appliqué MC-Fold, et nous avons observé manuellement les trois meilleurs candidats pour chacune d'elles. Au lieu de se restreindre à des hélices composées d'interactions Watson-Crick, nous avons supposé que les interactions tertiaires pouvaient faire parties des hélices en les « prolongeant. » Nous avons alors vérifié si l'utilisation de ces hélices prolongées permettait d'augmenter le score d'empilement de la configuration correcte, relativement aux autres configurations (voir figure 5.5). Lorsque c'était le cas, nous disons que l'information tertiaire nous « aide », que notre prédiction finale soit correcte ou non.

Sur les vingt jonctions étudiées, utiliser l'information tertiaire réelle (issue des données de cristallographie) nous a aidé six fois ; utiliser le premier candidat de MC-Fold nous a aidé une fois, et utiliser l'un des trois premiers candidats nous a aidé trois fois, en supposant que nous sachions quel candidat choisir parmi les trois. Un problème récurrent est que MC-Fold produit souvent plusieurs brins de longueur zéro, prédisant ainsi plusieurs très bons candidats pour un empilement.

Ces résultats suggèrent que MC-Fold n'améliore pas notre approche en nous aidant sur le critère de longueurs de brins. Il pourrait peut-être nous aider si nous introduisions des critères tertiaires dans notre approche, ce que nous n'avons pas fait ; cependant, il nous faudrait toujours pouvoir choisir entre les différents candidats produits par MC-Fold, ce qui n'est pas possible de manière automatique sans données supplémentaires.

Famille	PPV	Spécificité	Sensibilité
A	0,60	0,67	0,85
B	0,44	0,94	0,20
C	0,83	0,58	0,74

TABLE 5.6: Valeur prédictive (*Positive predicting value, PPV*), spécificité et sensibilité de notre classificiation, en considérant des tests binaires tels que « est-ce que cette jonction appartient à la famille A ? » sur le jeu de données Tout. Un choix aléatoire parmi les trois familles donnerait une PPV de 0,33, une spécificité de 0,66 et une sensibilité de 0,33.

Dans la table 5.6, nous considérons notre méthode de classification comme des tests binaires oui/non qui déterminent si une jonction appartient à une famille donnée. Nous indiquons la valeur prédictive (PPV), la spécificité et la sensibilité de ces tests, dont nous rappelons les définitions dans l'appendice B.1.1, page 131.

Les résultats sont bons pour les familles A et C avec, en particulier, une PPV beaucoup plus élevée que ce que l'on pourrait attendre par un choix aléatoire. La haute spécificité et basse sensibilité du test pour la famille B indique que nous avons rarement tort, mais aussi rarement raison ; autrement dit, nous prédisons rarement qu'une jonction est en famille B. Il s'agit de la moins bien définie des trois familles, et nous manquons de critères discriminants la concernant. Cette famille n'a pas de score bonus $s_{c,4}$, mais essayer de compenser cela en lui ajoutant un bonus fixe ou en ignorant le score bonus complètement n'améliore pas les prédictions dans l'ensemble. Notons que la famille C est la plus répandue, et la famille B la moins représentée.

5.4.4 Amélioration fournie par l'utilisation d'information d'homologie

Clusters de notre jeu de données

Nous nous sommes également demandés si des jonctions homologues mais de structure secondaire différente étaient prédites dans la même configuration par notre méthode (voir section 5.3.4). Nous avons regroupé les jonctions par homologie en trente-neuf clusters, dont vingt-six comportent plus d'une jonction ; ces derniers clusters sont représentés dans la table 5.7. Dans huit de ces clusters (31%), une jonction est prédite dans une mauvaise configuration bien que d'autres soient

Id	Jonctions	Fam.	Sep.
1	1J5E_001, 2AVY_29	A	**1/2**
2	1J5E_002, 2AVY_32	A	**1/2**
3	1J5E_003, 2AVY_35	A	2/2
4	1J5E_004, 2AVY_47	A	**1/2**
5	1J5E_006, 2AVY_40	B	0/2
6	1J5E_008, 2AVY_3	C	2/2
7	1J5E_011, 2AVY_49	C	2/2
8	1J5E_007, 2AVY_42	B	2/2
9	1S72_001, 2AW4_5, 2ZJR_1	A	3/3
10	1S72_002, 2AW4_10, 2ZJR_4	A	3/3
11	1S72_003, 2AW4_50, 2ZJR_47	A	3/3
12	1S72_004, 1FG0_7, 2AW4_51	A	**2/3**
13	2ZJR_101, 2AW4_109	B	2/2
14	1S72_007, 2AW4_28, 2ZJR_24	B	0/3
15	1S72_008, 2AW4_52, 2ZJR_49	B	0/3
16	1S72_010, 2AW4_21, 2ZJR_0	C	3/3
17	1S72_011, 2AW4_18, 2ZJR_10	C	**2/3**
18	1S72_012, 2AW4_27, 2ZJR_21	B	0/3
19	1S72_013, 2AW4_99, 2ZJR_92, 1FG0_14	C	4/4
20	1HC8_001, 2AW4_44, 1S72_47, 1QA6_0, 1MMS_0, 2ZJR_41	C	6/6
21	1S72_005, 2AW4_86, 2ZJR_82	B	3/3
22	1S72_014, 2AW4_0, 2ZJR_106, 1UN6_0	C	**1/4**
23	1E8O_001, 1MFQ_001, 1LNG_0	C	**2/3**
24	1U8D_001, 1Y26_0, 2EET_0	C	**2/3**
25	1MME_001, 2QUS_0, 2OEU_0	C	3/3
26	3D2G_0, 2HOJ_0	B	0/2

TABLE 5.7: Liste des clusters d'homologie. Chaque ligne contient un cluster de jonctions homologues (les clusters ne contenant qu'une jonction ne sont pas représentés). La colonne « Sep. » indique combien de jonctions sont prédites correctement lorsqu'elles sont prises séparément ; cette valeur est en gras pour les clusters *contradictoires*. Les clusters correctement prédits par la fonction de score global S_c sont représentés avec un fond gris. Notons qu'il n'y a qu'un seul cluster contradictoire qui n'a pas de fond gris (c'est à dire : qui soit mal prédit), le numéro 23.

correctement prédites. Appelons ces clusters des clusters *contradictoires*. Dans les dix-huit clusters restants, soit toutes les jonctions sont bien prédites, soit elles sont toutes mal prédites. Nous pouvons remarquer que les cinq clusters pour lesquelles toutes les jonctions sont mal prédites appartiennent à la famille B, ce qui confirme que notre classification manque de critères discriminants pour cette famille.

Dans la table 5.7, les lignes à fond gris représentent une bonne prédiction pour le cluster lorsque la fonction de score global S_c définie section 5.3.4 est utilisée. Sept des huit clusters contradictoires sont maintenant bien prédits ; le dernier cluster

est désormais entièrement mal prédit. La comparaison des résultats selon que l'on utilise ou non l'information d'homologie est donnée dans la table 5.8.

	Sans homologie	Avec homologie
Toutes les jonctions correctes	17	24
Certaines jonctions correctes	8	0
Aucune jonction correcte	5	6
Nombre de jonctions correctes	50	57

TABLE 5.8: Utiliser l'information d'homologie améliore les prédictions dans la majorité des clusters lorsque des jonctions homologues n'étaient pas toujours prédites de la même manière, mais ne nous aide pas lorsque toutes les jonctions homologues étaient mal prédites.

Utilisation d'un grand nombre d'alignements

Dans la section précédente, nous avons utilisé un petit nombre de jonctions homologues, et pour chacune d'entre elles, nous possédions une structure secondaire déduite de la structure tertiaire cristallographiée. Cependant, le cas le plus courant est de posséder un alignement de séquences issu de bases de données d'ARN, fournissant une structure secondaire consensus que l'on espère proche de la structure secondaire réelle. Comment notre approche se comporte-t-elle dans ce cas ?

Pour le savoir, nous avons utilisé des alignements provenants du *Comparative RNA Web Site* [58]. Pour chaque jonction ribosomale J de notre jeu de données, nous avons sélectionné 500 séquences alignées, utilisé la structure secondaire de J pour ces alignements, et appliqué notre méthode de prédiction en utilisant le calcul de score global S_c décrit en 5.3.4. Nous nous sommes assurés que le nombre de séquences choisi était assez grand pour être représentatif de l'ensemble des séquences disponibles dans les alignements.

Sur les clusters de jonctions ribosomales, selon la structure utilisée lorsque nous avions plusieurs choix possibles, deux ou trois des cinq clusters contradictoires sont désormais bien prédits.

5.5 Conclusion

Nous avons montré qu'à partir de la structure secondaire d'une 3-jonction, sans connaissance du contexte, il était possible de déterminer sa forme approximative avec 64% de réussite et que, de plus, la connaissance de plusieurs structures hologues améliorait grandement le taux de réussite. Cela conforte l'hypothèse selon

laquelle la forme des jonctions est déterminée localement. Nous allons maintenant nous intéresser aux autres types de jonctions, et montrer qu'il est également possible de classifier automatiquement au moins les 2- et 4-jonctions, qui, avec les 3-jonctions constituent l'essentiel des jonctions existantes.

6 Autres jonctions

6.1 Jonctions à deux branches

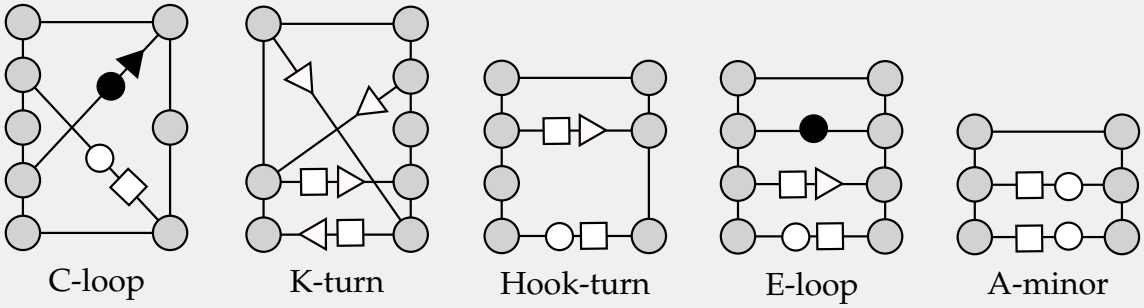


FIGURE 6.1: Exemples de motifs de 2-jonctions. Les disques gris représentent les nucléotides, les lignes verticales le squelette phosphate, et les lignes horizontales ou diagonales les interactions entre nucléotides. Les interactions sont identifiées suivant la nomenclature Leontis-Westhof [5]. Dans la structure secondaire, la taille des boucles peut varier et les interactions comportant carrés, triangles ou disques ne sont pas présentes ; par conséquent, tous ces motifs paraissent identiques lorsque l'on ne dispose que de la structure secondaire.

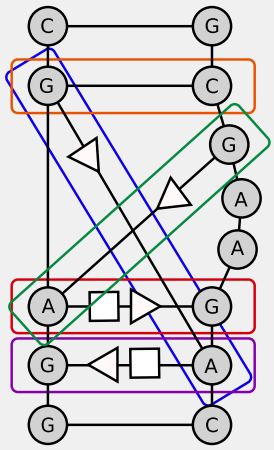
La figure 6.1 présente quelques représentations schématiques de familles récurrentes de 2-jonctions, appelées aussi *motifs* ou *motifs internes*. Ces motifs sont définis par les interactions tertiaires qui les composent, et autorisent une certaine variabilité de séquence et de structure en dehors de ces interactions. Par exemple, dans le cas du motif C-loop de la figure, le nucléotide isolé du brin de droite pourrait être remplacé par deux nucléotides sans que le motif n'en soit affecté.

Si toutes les variations de séquence étaient autorisées, nous ne pourrions pas espérer distinguer les différents types de 2-jonctions à partir uniquement de leurs structures secondaires, qui peuvent être identiques. Cependant, Lescoute et Westhof ont montré [11] que les mutations au sein des motifs C-loop et K-turn sont soumises à une pression de sélection afin de respecter des contraintes d'isostérie et maintenir les mêmes interactions tertiaires. Ces contraintes d'isostérie définissent des classes de mutation équivalentes pour chaque type d'interaction tertiaire, mais toute mutation sortant de ces classes risque de détruire l'interaction, et donc le motif. La conservation de structure introduit un biais statistique dans les fréquences

d'apparition des nucléotides à différentes positions d'un motif. Nous présentons ici une approche simple exploitant ce biais statistique pour tenter de prédire l'appartenance d'une jonction à une famille de motifs.

6.1.1 Méthode

Nous reprenons le jeu de données de Lescoute et Westhof [11], constitué de onze motifs K-turn et quatre motifs C-loop cristallographiés, ainsi que d'alignements de séquences d'ARNr 16S et 23S correspondants à des milliers d'espèces d'archées, bactéries et eucaryotes. Dans leur article, les auteurs présentent une structure consensus pour chaque motif, identifient les interactions tertiaires caractéristiques de ces motifs, et effectuent des statistiques d'apparition des différents nucléotides pour ces interactions. Nous reprenons ces statistiques en ignorant les distinctions entre les différents règnes, qui sont mineures.



K-turn consensus

	A	C	G	U
A				5,7
C			67,8	
G		17,8		
U	1,9		6,8	

	A	C	G	U
A	2,3			
C	7,9			
G	88,0	0,5	1,4	
U				

	A	C	G	U
A	12,9			
C	17,4		1,1	0,4
G	47,0			0,4
U	18,2			3,0

	A	C	G	U
A	12,1	1,7	80,0	0,8
C				
G	0,4	0,6	0,8	
U	2,5		0,4	1,3

	A	C	G	U
A	4,2	0,6	73,2	6,0
C			0,6	
G		0,6		1,2
U	1,8		11,9	

FIGURE 6.2: Structure tertiaire consensus du motif K-turn, et les matrices de fréquences d'apparition des nucléotides pour chaque interaction caractéristique du motif. Les chiffres sont des pourcentages, et la somme de chaque tableau vaut 100.

La figure 6.2 présente le cas du motif K-turn, avec sa structure tertiaire consensus dont les interactions caractéristiques sont identifiées par des cadres de couleur, et les matrices de fréquences de nucléotides pour chacune de ces interactions. nous pouvons y lire, par exemple, que l'interaction « rouge », de type Hoogsteen-Sucre, est constituée d'un G et d'un A dans 80% des cas. Une matrice similaire est calculée pour le motif C-loop. Pour prédire l'appartenance d'une 2-jonction à une de ces familles de motifs, nous considérerons ces fréquences comme des probabilités.

Pour classifier une 2-jonction donnée, nous ne disposons que de sa structure secondaire. Nous ne savons pas où se trouvent les différentes interactions tertiaires spécifiques au motif considéré, car il peut y avoir des insertions ou suppressions de nucléotides par rapport au motif consensus. Nous allons donc essayer de pla-

cer chaque interaction tertiaire (chaque boîte de couleur) à chaque emplacement possible. Cela nous donne un certain nombre de configurations possibles et, pour chaque configuration, nous calculons un score qui est la somme, pour chaque interaction tertiaire caractéristique, de la probabilité que cette interaction soit constituée de ces paires de bases. Soit $i = (a, b)$ une interaction tertiaire d'un motif constituée des nucléotides a et b, et P_i sa matrice de probabilités ; le score $S_{f,c}$ correspondant à une famille f dans la configuration c est :

$$S_{f,c} = \sum_{i=(a,b)} P_i[a, b]$$

Le score correspondant au fait qu'une 2-jonction soit dans la famille de motifs f est le score maximal parmi toutes les configurations possibles pour cette famille f :

$$S_f = C_f \max_c(S_{f,c})$$

Afin que les scores obtenus soient comparables entre différents motifs, nous devons procéder à une normalisation. Le motif K-turn est constitué de cinq interactions tertiaires caractéristiques, son score est donc compris entre 0 et 5. Nous multiplierons ce score par $C_f = 20$. Le motif C-loop comporte quatre interactions tertiaires, nous multiplierons donc son score par $C_f = 25$ pour le ramener également entre 0 et 100. Enfin, nous comparons les deux scores S_{k-turn} et S_{c-loop}. Si l'un des deux est supérieur à 50 et supérieur à l'autre score, alors nous prédisons que la jonction est dans la famille correspondante. Sinon, nous ne prédisons rien. De plus, si l'écart entre les deux scores est inférieur à 10, nous avertissons l'utilisateur que la prédiction est ambigüe. Ces seuils de score ne résultent pas d'un apprentissage mais de l'interprétation de ces valeurs comme des pourcentages de probabilité.

Considérons une 2-jonction possèdant, dans sa partie centrale non appariée par des liaisons Watson-Crick, i nucléotides sur le brin de gauche et j nucléotides sur l'autre, et un motif caractérisé par m interactions tertiaires. Combien d'opérations devrons-nous effectuer pour calculer le score de ce motif ?

Chaque interaction peut relier n'importe lequel des i nucléotides de gauche à n'importe lequel des j nucléotides de droite. Nous avons donc mij configurations possibles. Nous pouvons également prendre la jonction dans l'autre sens, c'est à dire inverser le brin de gauche et le brin de droite, ce qui double le nombre de configurations. Pour chaque configuration, nous effectuons $O(m)$ opérations, ce qui nous donne au total $O(m^2ij)$ opérations pour un motif. Pour un motif donné, m est constant, et dans une molécule réelle, la taille des 2-jonctions est bornée et petite ; le calcul du score d'un motif peut donc être considéré en $O(1)$.

6.1.2 Résultats

La table 6.1 présente les résultat de prédictions de cette méthode appliquée aux quatorze jonctions Kink-turns et C-loops cristallographiées de [11]. Neuf d'entre

Jonction	Score K-turn	Score C-loop	Prédiction
23S KT-46	67,05	25,26	K-turn
23S KT-7	100,00	25,85	K-turn
23S KT-38	100,00	50,00	K-turn
23S KT-15	46,03	50,00	C-loop (?)
23S KT-42	87,28	50,85	K-turn
16S KT-23	100,00	50,00	K-turn
23S KT-58	65,56	25,85	K-turn
23S KT-94/95	94,01	41,77	K-turn
23S KT-4/5	87,74	27,90	K-turn
23S KT-77/78	41,18	27,37	Aucune
23S C-96	40,29	70,27	C-loop
23S C-50	14,12	41,77	Aucune
23S C-38	41,78	41,77	Aucune
16S C-15	14,17	25,54	Aucune

TABLE 6.1: Résultats de prédictions sur les C-loop et les K-turns de [11].

elles (64,3%) sont correctement prédites, une est mal prédite (mais l'utilisateur est prévenu que les deux scores sont proches), et quatre (28,6%) ne sont prédites dans aucune famille.

6.1.3 Perspectives

Cette approche fonctionne bien sur le cas publié des C-loops et K-turns, mais il existe de nombreuses familles de 2-jonctions. Rien ne s'oppose en principe à la généralisation de notre approche à d'autres motifs. Nous n'avons pas approfondi cette piste car, dans l'optique d'une prédiction de structure à gros grain, les 2-jonctions peuvent en général être approximées comme un prolongement d'hélice. Il serait néanmoins intéressant de poursuivre ce travail si nous voulions redescendre ensuite à un niveau d'échelle plus détaillé.

6.2 Jonctions à quatre branches

6.2.1 Définition des familles

Les jonctions comportant plus de trois branches sont nettement plus rares que les 1-, 2- ou 3-jonctions. Ainsi, sur le jeu de données utilisé au chapitre 8, si les 2- et 3-jonctions représentent la moitié des jonctions, les 4-jonctions en représentent moins de 5% (voir table 8.2 page 65). Sur si peu de données, il est risqué de définir des familles, et encore plus risqué de proposer une méthode de classification. Laing et Schlick [59] définissent neuf familles topologiques, représentées dans la figure 6.3, mais certaines familles ne comportent, si l'on exclut les jonctions homologues, qu'un individu (familles X, cW et π) ou deux (famille cX). Par ailleurs, les familles H et π sont très proches, la famille π possédant un empilement de moins, compensé par des interactions tertiaires. En prenant une définition étendue de l'empilement coaxial, comme nous l'avions fait pour les 3-jonctions, ces deux familles sont identiques.

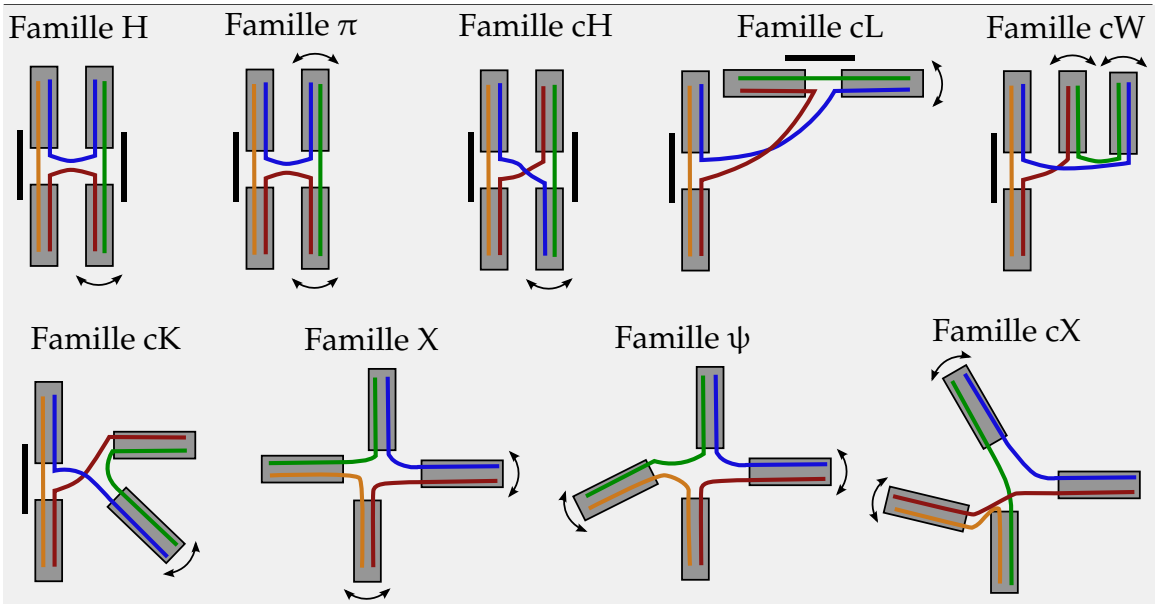


FIGURE 6.3: Les neuf familles topologiques définies dans Laing et Schlick [59]. Les traits noirs indiquent un empilement coaxial entre deux hélices, et les doubles flêches indiquent une flexibilité de l'hélice.

6.2.2 Méthode de prédiction

Avant la publication de [59], nous avions tenté également d'établir une méthode de classification automatique des 4-jonctions en familles topologiques. Nous allons la rapprocher de cette classification. Disposant de trop peu d'exemplaires de 4-jonctions, nous ne sommes pas intéressés à des paramètres trop précis de séquence

ou structure secondaire, car nous courrions le risque de caractériser une jonction et pas une famille de jonctions. Le seul critère que nous avons considéré est donc celui de la longueur des brins non appariés reliant les hélices, en gardant à l'esprit que l'acquisition de nouvelles données devrait nous permettre de raffiner cette classification.

Nous disons qu'un brin est *court* (c) s'il comporte zéro ou un nucléotide non apparié, ou s'il est plus court qu'un des autres brins d'au moins quatre nucléotides ; nous disons qu'il est *long* (l) sinon. Puisque nous ne considérons que le critère de longueur, nous pouvons caractériser une 4-jonction par l'énumération des longueurs de ses brins dans l'ordre de transcription, par exemple (clcl) pour la 4-jonction de l'ARN de transfert, ou (cccc) pour la 4-jonction de la RNase P de type B (1NBS). Une première observation nous conduit à noter que lorsqu'il y a des brins courts, nous avons un empilement entre les hélices qu'ils relient, mais que si plusieurs brins courts se suivent, par exemple (ccll), il ne peut y avoir empilement que sur l'un des brins. Nous proposons les règles suivantes :

1. Si deux brins courts sont opposés, par exemple (c̲lc̲l) ou (c̲cc̲l), il y a deux empilements parallèles sur ces brins.
2. Si nous avons une alternance (clcl), et si les brins longs sont de taille 2 ou 3, nous sommes en famille cH (les deux brins longs se croisent, les deux courts sont empilés).
3. Sinon, si les deux brins longs sont très longs, nous sommes en famille cL, la famille de l'ARNt (les deux brins longs sont suffisamment longs pour permettre ce genre de repliement).
4. Si nous avons au moins trois brins courts, nous sommes en famille H (les deux brins non empilés sont trop courts pour se croiser).
5. Dans le cas (cccc) l'ambiguité sur le choix des empilements est levée par l'ordre de transcription, et nous empilons selon les brins soulignés : (c̲cc̲c̲)
6. Si aucun critère n'est décisif, alors nous ne proposons aucune classification.

6.2.3 Résultats

Les règles données ci-dessus découlent d'une observation de la représentation 3D des jonctions, et semblent rétrospectivement logiques. Voyons la qualité de leurs prédictions si nous les appliquons aux 4-jonctions de [59], en regroupant les familles π et H. La table 6.2 recense ces prédictions.

Si nous considérons qu'une bonne prédiction est une prédiction positive dans la bonne famille, 44% des jonctions sont correctement prédites. L'absence de prédiction alors que la jonction est dans une des familles H, cH ou cL ou la prédiction dans une mauvaise famille concernent 20% des jonctions, les 36% restants étant constitués d'absences *correctes* de prédiction : nous ne prédisions aucune des familles H, cH ou cL et la jonction n'était effectivement dans aucune de ces familles.

Jonction	Molécule	Famille	Prédiction
1NBS	Rnase P B-type	H	H ou rien (*)
1U9S	Rnase P A-type	H	H ou rien (*)
2A25	Rnase P A-type	H	H
1S72 1827	23S rRNA	H	rien
1M5O	Hairpin ribozyme	H	H
2AW4 1443	23S rRNA	cH	H ou rien (*)
1S72 2678	23S rRNA	cH	cH
1KH6	HIV IRES domain	cH	cH
2AVY 141	16S rRNA	cH	rien
3F2Q 7	FMN Riboswitch	cH	rien
2GIS 7	SAM I Riboswitch	cL	cL
2J01 1269	23S rRNA	cL	cL
1EHZ	tRNA Phe	cL	cL
2JOO 568	16S rRNA	cL	cL
1S72 1452	23S rRNA	cK	rien
2AW4 2284	23S rRNA	cK	rien
2AVY 18	16S rRNA	cK	rien
2AVY 141	16S rRNA	cK / cH	cH
2J01 1665	23S rRNA	cW	rien
1S72 42	23S rRNA	ψ	rien
2AW4 267	23S rRNA	ψ	rien
2J01 1832	23S rRNA	ψ	rien
2AW4 600	23S rRNA	X	cH
2IHX	Sarcoma Virus Domain	cX	rien
2AVY 942	16S rRNA	cX	rien

(*) : le résultat dépend de la définition d'hélice Watson-Crick. Si une paire de bases est une hélice, alors nous prédisons en famille H, sinon, nous ne prédisons rien.

TABLE 6.2: Prédictions sur les 4-jonctions

Autrement dit, dans 80% des cas, notre méthode indique correctement si une jonction appartient ou non à une de ces trois familles, mais dans 36% des cas la réponse est « non », et nous ne savons pas à quelle famille la jonction appartient.

6.2.4 Discussion et perspectives

Ces quelques règles déterminées expérimentalement fonctionnent bien pour classifier une jonction appartenant aux familles H, cH ou cL. La plupart de ces

règles semblent logiques, et représentent de simples contraintes de distances liées à la longueur des brins qui constituent la jonction. Une règle, cependant, est surprenante : la règle 5, qui dit que, si nous avons le choix des brins d'empilement, nous devons empiler sur le deuxième et le quatrième brin. Pourquoi cette règle serait-elle justifiée ? Bien qu'elle fonctionne sur notre jeux de données, nous ne disposons que de cinq jonctions en famille H, il se pourrait donc qu'il s'agisse d'une coïncidence. Cependant, sur l'ensemble du jeu de données, et pas seulement sur les jonctions de la famille H, il existe une nette tendance à ce que l'empilement se fasse dans cet ordre (hélices 1 et 4, hélices 2 et 3). Laing et Schlick [59, p. 553] suggèrent que cette tendance pourrait être due au sens de rotation (droit) des hélices d'ARN. Il se pourrait aussi qu'elle soit due à un ordre de formation des hélices : le repliement commençant pendant la transcription, les hélices 2 et 3 auraient plus de chances de se former, et donc de s'empiler, avant les hélices 1 et 4 ; ces dernières n'auraient ensuite plus le choix et s'empileraient entre elles.

Dans les cas des familles qui sont hors de portée de notre méthode, nous ne sommes pas nécessairement complètement démunis d'informations concernant la forme des jonctions. Ainsi, si un seul brin est court, alors il y a généralement un empilement sur ce brin, les positions du reste des hélices restant indéterminées. Cependant, certains cas semblent très particuliers et difficilement caractérisables, comme celui de la famille cW, qui ressemble beaucoup en 2D à une famille cL. Pour ces familles, s'il s'agit bien de familles et pas de cas dépendant du contexte, il faudrait plus d'exemples afin d'en dériver de nouveaux critères.

Enfin, remarquons une particularité de la famille ψ. Dans cette famille, il n'y a pas d'empilement entre des hélices reliées par un brin, mais il y a un empilement coaxial entre deux hélices opposées. Si nous prenons le cas de la jonction 1S72_42, et que nous considérons l'empilement central, l'hélice de droite semble s'arranger selon une famille B de 3-jonction (ce qui est cohérent avec les longueurs de brins), et celle de gauche en famille A.

6.3 Jonctions à plus de quatre branches

Si les 4-jonctions sont assez rares, nous avons vu qu'il était néanmoins possible de les regrouper en familles de topologie similaire. Les jonctions entre plus de quatre hélices sont encore plus rares et représentent moins de 4% des jonctions présentes dans les bases de données. Laing et co-auteurs [13] ont analysé ces jonctions en fonction de leur forme mais, en dehors de quelques 5-jonctions, un seul exemplaire de chaque forme est disponible, il est donc impossible de parler de familles topologiques. Il est par ailleurs difficile de déterminer si la structure des plus grosses jonctions est déterminée localement, ou si elle dépend en grosse partie du contexte.

Dans ces grosses jonctions, nous retrouvons toujours des empilements co-axiaux le long des brins les plus courts. Nous pouvons observer également des arrange-

ments parallèles d'hélices, cette fois sans empilement mais côte à côte. D'autre part, parmi les 5-jonctions, nous pouvons identifier certaines ressemblances avec des familles de 4-jonctions dans lesquelles une hélice se serait insérée sur un brin, et dont l'angle serait plus souple. Dans les jonctions de plus haut degré, il est souvent possible de reconnaitre certains arrangements similaires à des repliements de 3-jonctions. Cependant, l'œil a souvent tendance à trouver des motifs là où il n'y en a pas, et nous nous abstiendrons de proposer de classification avant d'avoir suffisamment de jonctions de formes similaires pour étayer nos intuitions par des statistiques.

Dans la suite de notre travail, les jonctions de degré élevé seront considérées comme indéterminées. Nous répartirons les hélices uniformément sans supposer d'empilement ou d'alignements entre elles, et nous compenserons en partie l'erreur introduire en rendant ces jonctions moins rigides quant à l'angle de leurs hélices. Cependant, nous verrons que la plus grosse source d'amélioration de notre approche serait d'améliorer notre connaissance de ces jonctions.

6.4 Conclusion

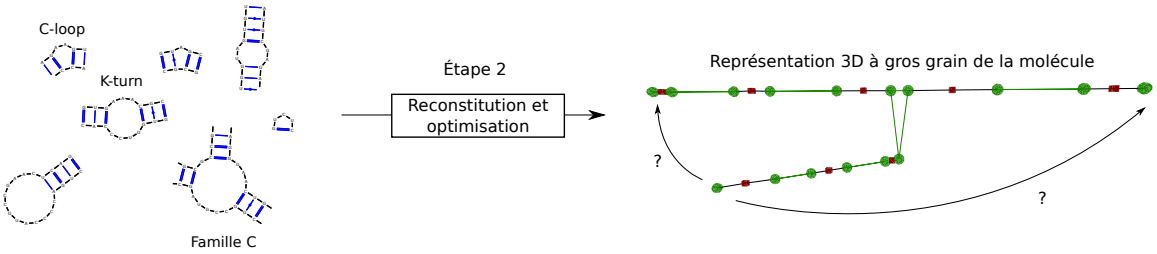
Dans ce chapitre, nous avons proposé des méthodes de prédiction pour les 2- et les 4-jonctions, donnant de bons résultats sur un sous-ensemble des familles observées. Dans le cas des 4-jonctions, notre approche est limitée par le manque de représentants dans les familles pour lesquelles nous ne proposons pas de classification automatique, et qui sont presques exclusivement représentées dans les ribosomes. La situation est identique pour les jonctions de plus de quatre hélices, et nécessite, pour être résolue, que plus de grosses molécules soient cristallographiées.

Bien que nos classifications ne soient pas complètes, la conclusion majeure des deux derniers chapitres est qu'il est possible, dans le cas des familles comportant suffisamment de représentants, de déterminer automatiquement si une jonction est dans cette famille, à partir uniquement de sa structure secondaire. Il est donc plausible d'espérer que, pour les familles restantes, l'apport de nouveaux représentants permettent d'introduire une classification automatique similaire.

Troisième partie

Détermination de la forme globale par optimisation

7 Introduction



Dans cette partie, nous présenterons la deuxième étape de notre méthode de prédiction : l'obtention de la forme globale de la molécule à partir des formes des éléments locaux qui la composent. Cette reconstruction s'effectue elle-même en deux temps : la création d'une structure prototype, que nous nommerons *plongement initial* à partir des formes idéales de chaque élément de la structure secondaire (hélices et jonctions), puis un repliement de ce plongement initial par l'ajout d'interactions de type tertiaire entre des éléments distants.

Le raisonnement derrière cette approche est le suivant : chaque jonction a une forme idéale en 3D, qui est déterminée localement par la structure secondaire de cette jonction. Nous pouvons donc mettre bout-à-bout les jonctions et hélices de la molécule pour obtenir une forme globale en 3D. Cependant, cette forme est soumise à des déformations qui peuvent mettre en contact différentes parties, qui sont alors suceptibles d'interagir entre elles et de former des interactions entrant en jeu de la formation de la structure tertiaire. Nous construisons donc la forme globale de la molécule comme une forme initiale découlant de la structure secondaire, à laquelle est appliqué un repliement induit par les éventuelles interactions formant la structure tertaire. La construction de cette forme initiale et la manière de la replier après l'ajout d'interactions sont décrites au chapitre 8.

Nous avons mentionné l'ajout, par rapport à la structure secondaire, d'interactions longue-distance (c'est-à-dire entre éléments éloignés dans la structure secondaire) qui produisent un repliement de la molécule. Ces interactions ne sont en général pas connues en entrée, et il nous faut les découvrir automatiquement. Nous proposerons deux méthodes pour y parvenir, l'une optimisant une fonction de coût globale à partir d'un algorithme évolutionniste (chapitre 9), l'autre recherchant un équilibre de Nash dans un jeu (chapitre 10). Enfin, au chapitre 11, nous comparerons nos prédictions, suivant ces deux méthodes, avec les approches concurrentes : iFoldRNA, FARNA et MC-Fold.

8 Modélisation

Résumé Dans ce chapitre, nous décrivons comment, à partir de la classification des jonctions d'une molécule en familles topologiques décrite dans la partie précédente, nous construisons un prototype de forme globale. Nous décrivons également comment nous pouvons replier ce prototype pour tenir compte d'informations de structure tertiaire (des interactions longue-distance), et nous définissons une fonction de coût qui sera utilisée dans la suite de ce mémoire pour la découverte d'interactions longue-distance par optimisation.

8.1 Introduction

Notre approche repose sur deux hypothèses : premièrement, que la structure secondaire de la molécule se forme avant sa structure tertiaire (voir section 2.2), et deuxièmement, que la forme des jonctions est essentiellement déterminée localement (voir la partie II). Ces hypothèses justifient la construction par étapes que nous effectuons. Réciproquement, nous pouvons considérer notre approche comme une mise à l'épreuve de ces hypothèses.

Nous construisons un *graphe squelette* (8.2), représentation à gros grain de la structure secondaire de la molécule comme un assemblage d'hélices et de jonctions. À l'aide de statistiques sur les dimensions connues de ces hélices et jonctions dans les structures cristallographiées (8.3), nous effectuons un plongement initial de ce graphe squelette dans l'espace (8.4). Puis, nous modélisons l'ajout d'interactions à longue distance, impliquées dans la formation de la structure tertiaire, comme un repliement de ce plongement initial pour respecter les contraintes de distance de ces nouvelles interactions (8.5). Enfin, nous définirons une fonction de coût représentant l'écart, dans un graphe, entre les positions de ses sommets et les positions idéales telles que définies par notre algorithme de repliement (8.6).

Dans les chapitres suivants, nous nous intéresserons à la découverte des interactions longue distance à l'aide des outils définis dans ce chapitre, en particulier la fonction de coût.

8.2 Graphe squelette

Notre approche visant à obtenir une prédiction à gros grain de la structure 3D des molécules, notre modèle se place à un plus haut niveau que les structures secondaires et tertiaires habituelles. Nous modélisons une molécule d'ARN M par un graphe $S = (V, E)$ (voir appendice B.2 pour les définitions) avec $V = J \cup H$ et $E = E_s \cup E_l$, appelé *graphe squelette*, dans lesquel un sommet $j \in J$ représente une jonction et un sommet $h \in H$ une hélice. À un sommet $s \in V$, nous pouvons donc associer un ensemble de nucléotides $M[s]$ de M.

Une arête $(j, h) \in E_s$ relie les sommets j et h si ces éléments sont adjacents dans la structure secondaire de la molécule, et peut être assimilée à une « demi-hélice ». Une arête de E_s relie nécessairement un sommet de J et un sommet de H, par conséquent le sous-graphe de S induit par E_s est biparti ; il est également arborescent si l'on exclut les pseudonœuds de la structure secondaire. Enfin, une arête $(s_1, s_2) \in E_l$ relie s_1 et s_2 si au moins un nucléotide de $M[s_1]$ et un nucléotide de $M[s_2]$ sont impliqués dans une interaction tertiaire longue-distance.

La figure 8.1 donne un exemple de structure et son graphe squelette associé.

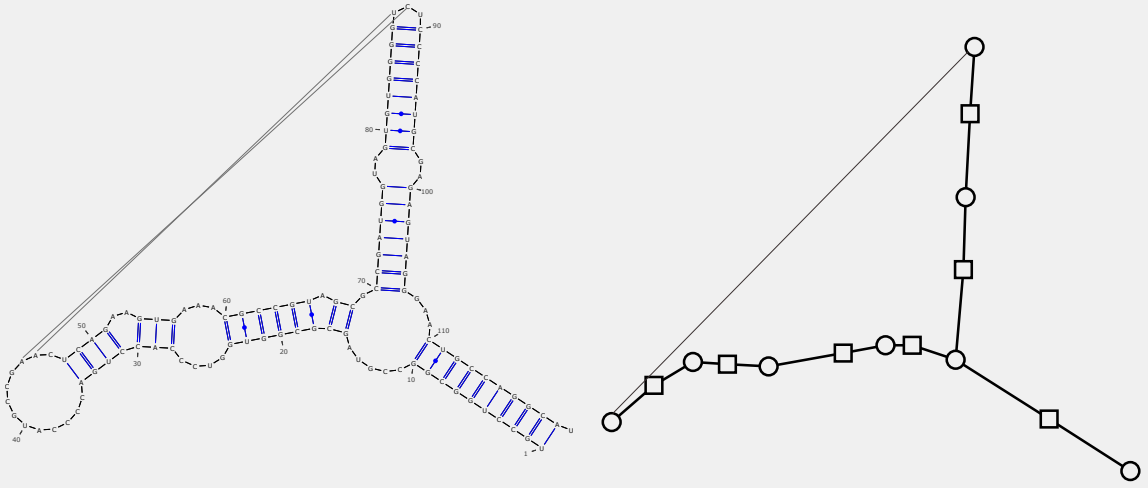


FIGURE 8.1: Structure secondaire d'un ARN et son graphe squelette associé (à droite). Les jonctions sont représentées par des cercles et les hélices par des carrés. Les arêtes grises représentent des interactions longue-distance (E_l).

On considère généralement qu'une molécule est constituée d'hélices, de jonctions, mais aussi d'autres types de motifs comme les boucles internes (voir figure 2.2 page 10). Dans un souci de généralisation, nous considérons les renflements et les boucles internes comme des 2-jonctions, et les boucles terminales comme des 1-jonctions. Par conséquent, chaque nucléotide de M est associé soit à une hélice soit une jonction de S.

Le plongement d'un graphe squelette dans l'espace se fait en associant des coordonnées à trois dimensions à ses sommets. Cependant, alors que les hélices ont

une forme relativement régulière et peuvent être approximées par des cylindres, les jonctions ont un encombrement variable qui dépend du nombre de nucléotides qu'elles contiennent et du nombre d'hélices qu'elles relient, ainsi que des interactions tertiaires qu'elle contient. Au lieu de représenter une n-jonction par un seul point de l'espace, nous lui associerons n points $p_1, \ldots, p_n$, correspondant chacun au point de départ d'une des n hélices qui composent cette jonction (voir figure 8.2). Nous appellerons ces points les *poignées* de la jonction. Lors de transformations géométriques, ces points évolueront ensemble comme un corps solide ; dans le graphe, une jonction est toujours représentée par un seul sommet, les poignées ne servant qu'aux calculs d'angle et de distance. Lorsque nous aurons besoin de considérer la position de la jonction dans l'espace, nous utiliserons le centre géométrique de ses poignées.

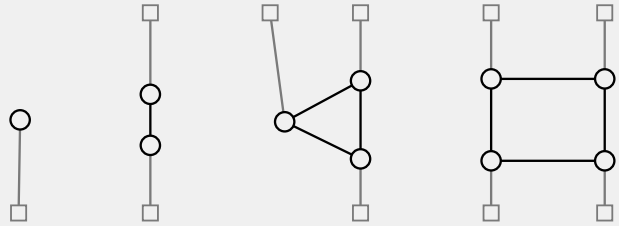


FIGURE 8.2: Exemples de poignées de jonctions pour une 1-jonction, 2-jonction, 3-jonction et 4-jonction.

8.3 Analyse des données réelles

Les paramètres géométriques des constituants structures d'acides nucléiques ont été bien caractérisés [60]. Cependant, notre représentation à gros grain s'accomode mal de ces paramètres, qui se situent à un niveau d'échelle plus bas : celui des nucléotides, ou des parties qui les composent (base, sucre et groupement phosphate). Nous devons calculer les valeurs des paramètres analogues dans notre représentation à gros grain du graphe squelette des molécules.

Afin de déterminer les paramètres moyens de longueurs et d'angles correspondant aux différents types de sommets, nous avons calculé les distributions de ces paramètres sur les structures cristallographiées de six molécules, dont la liste est donnée dans la table 8.1. Ces molécules ont été choisies pour couvrir l'essentiel des tailles de molécules présentes dans les bases de données à ce jour. Nous avons construit le graphe squelette de ces structures et placé ses sommets dans l'espace selon les coordonnées cristallographiées. Les poignées des jonctions correspondent à des paires de bases faisant parties d'hélices ; nous avons placé une poignée à l'emplacement où l'axe de l'hélice coupe le plan formé par la paire de bases. Les sommets correspondants aux hélices sont ensuite placés au centre du segment

reliant les deux poignées concernées, comme décrit sur la figure 8.3. Enfin, nous avons effectué des statistiques sur les graphes obtenus.

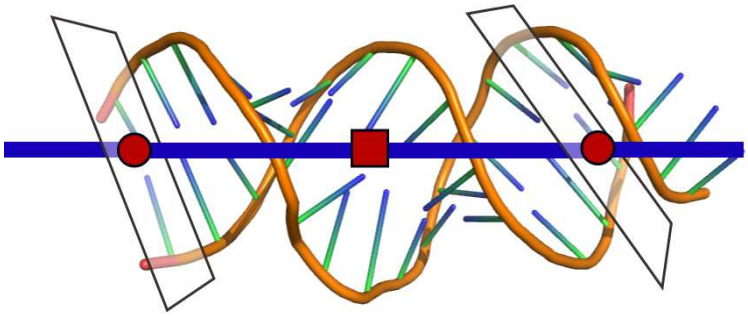


FIGURE 8.3: Plongement du squelette pour une hélice. L'axe de l'hélice est représenté en bleu, les plans définis par les nucléotides en blanc, et les sommets construits en rouge.

Identifiant	Nucléotides	Sommets	Description
2ZJR	3003	327	Ribosome 23S *D. radiodurans*
1J5E	1545	185	Ribosome 16S *T. thermophilus*
2A64	417	37	RNase P bactérienne
1NBS	155	21	RNase P bactérienne
1MFQ	127	19	SRP *H. sapiens*
1E8O	49	7	Alu domain

TABLE 8.1: Identifiant PDB, nombre de nucléotides, nombre de sommets dans le graphe squelette et description des molécules utilisées pour le calcul des paramètres moyens de longueurs et d'angles.

8.3.1 Hélices

Sur notre jeu de données, l'essentiel des hélices contient entre trois et six paires de bases, la valeur médiane étant de 4 (voir figure 8.4). Les hélices sont les motifs les plus réguliers de la structure d'ARN, et comportent donc peu de variabilité. En calculant la régression linéaire de la distribution des longueurs d'hélices en fonction de leur nombre de nucléotides, nous obtenons une approximation de la longueur « idéale » $D_h(n)$ d'une hélice de n paires de bases par la fonction $f(n) = 2{,}64n - 2{,}62$ Å.

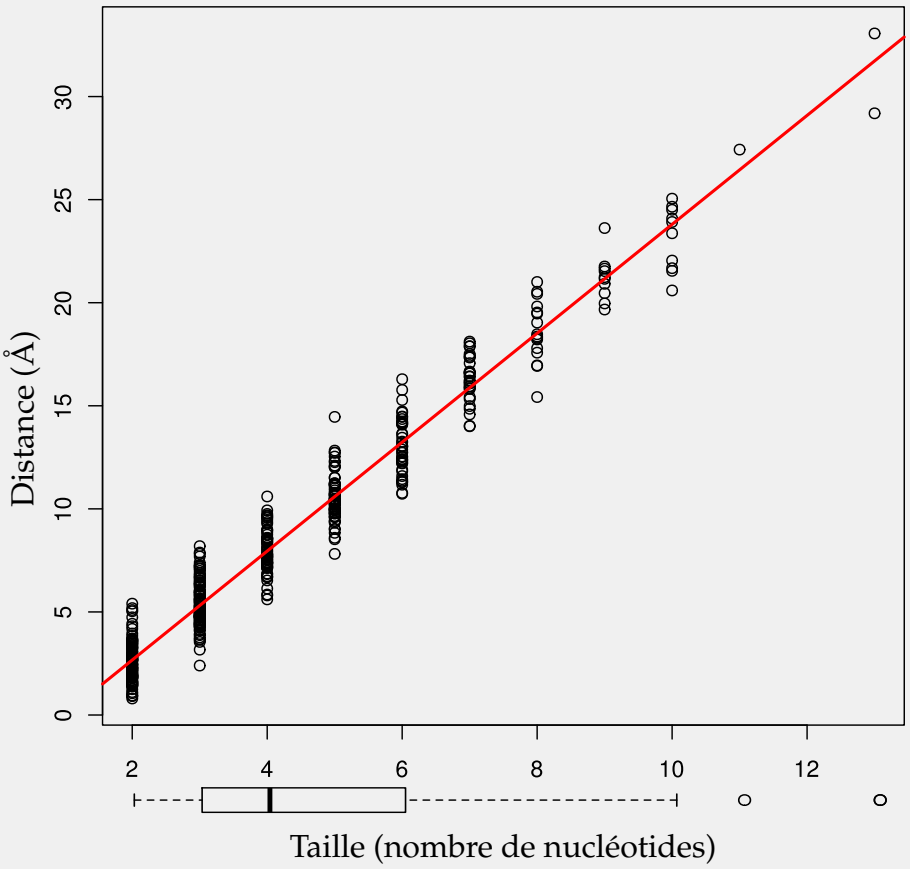


FIGURE 8.4: Distance entre les extrémités des hélices en fonction de leur taille, c'est à dire en fonction du nombre de nucléotides qu'elles contiennent sur un brin. En rouge, la régression linéaire correspondante, d'équation $y \simeq 2{,}64x - 2{,}62$. La distribution des hélices en fonction de leur taille est représentée par une boîte à moustaches sous l'axe des abscisses.

Hélices	Nombre	Pourcentage
1	182	39%
2	193	42%
3	48	10%
4	21	4,6%
5+	17	$< 4\%$

TABLE 8.2: Caractéristiques des jonctions de notre jeu de données

8.3.2 Jonctions

La table 8.2 indique la composition de notre jeu de données en jonctions. La majorité des jonctions (81%) sont des 1- ou 2-jonctions, et les jonctions comportant cinq hélices ou plus représentent seulement 3,5% du jeu de données.

Beaucoup de paramètres peuvent être mesurés sur une n-jonction : l'écart entre ses poignées, les angles entre ses hélices adjacentes, ou encore son enveloppe convexe. Cependant, nous disposons de trop peu de n-jonctions avec $n > 3$ pour que ces mesures soient significatives. Nous nous contenterons donc, pour les jonctions comportant un grand nombre d'hélices, d'une représentation approximative ne tenant pas compte du nombre de nucléotides que comprend la jonction.

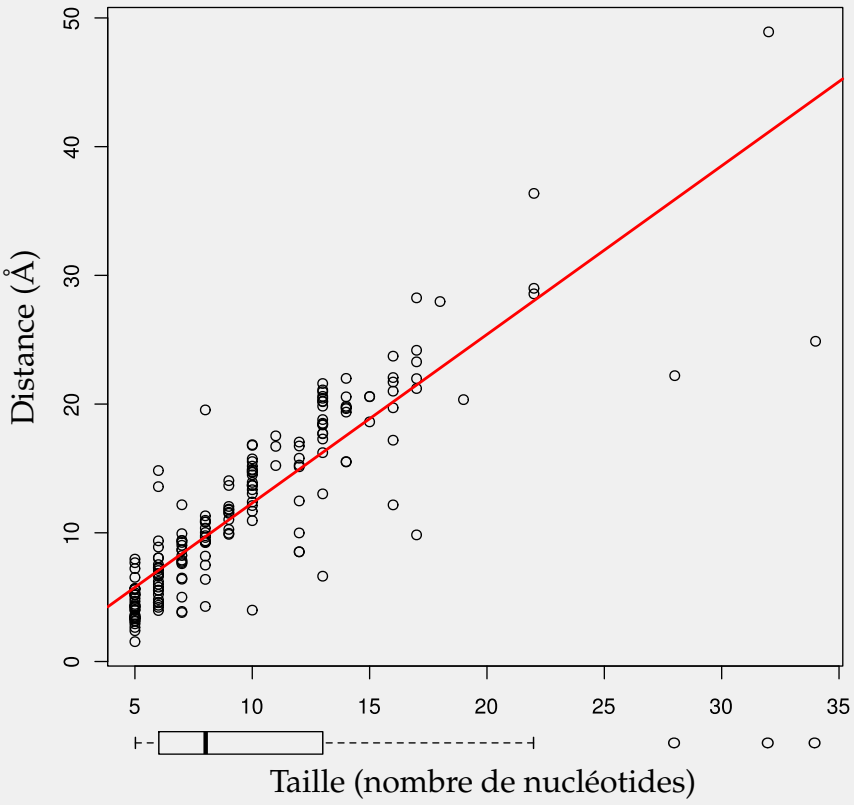


FIGURE 8.5: Statistiques de longueurs des jonctions à deux branches. La longueur est calculée comme la distance entre les deux poignées (*ie.* extrémités) des jonctions. En rouge, la régression linéaire correspondante, d'équation $y \simeq 1{,}31x - 0{,}79$.

Dans le cas particulier des 2-jonctions, toutefois, nous pouvons mesurer la distance entre les deux extrémités de la jonction, en fonction du nombre de nucléotides qu'elle comporte (sans tenir compte de la classification des 2-jonctions en motifs). La figure 8.5 représente cette distribution. En comparant une 2-jonction comportant $2n$ nucléotides avec une hélice comportant n paires de bases, nous pouvons constater une forte similitude entre les deux distributions. Du point de vue de la structure secondaire, une 2-jonction semble vide, mais elles contiennent en réalité des interactions non-canoniques et conservent généralement la forme approximative d'une hélice, une grande partie de la stabilité de cette structure en hélice découlant des empilements verticaux entre les nucléotides.

8.3.3 Interactions longue distance

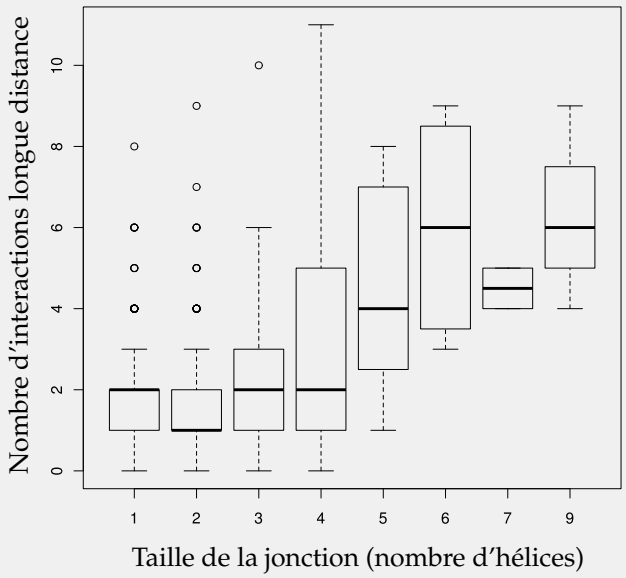


FIGURE 8.6: Statistiques de nombre d'interactions longue distance par jonction. Voir appendice B.1.3 page 132 concernant l'interprétation des boîtes à moustache.

Nous définissons une interaction longue-distance comme une interaction de type tertiaire entre deux sommets qui ne sont pas directement reliés par une arête de type « structure secondaire ». La figure 8.6 représente le nombre d'interactions longue distance par n-jonction, en fonction de n. Nous pouvons nous attendre à ce que les jonctions comportant de nombreuses hélices occupent un volume plus grand dans l'espace, et soient impliquées dans plus d'interactions longue distance, et cela semble effectivement être le cas. Cependant, même les 1- ou 2-jonctions sont parfois impliquées dans de nombreuses interactions longue distance. Il est à noter que la plupart (91%) des jonctions de notre jeu de données sont des 1-, 2- ou 3-jonctions, comme indiqué dans la table 8.2.

En considérant une jonction comme un seul sommet, nous faisons abstraction de ce qu'elle contient ; une 1-jonction peut aussi bien comporter 3 que 20 nucléotides, et il semble raisonnable de supposer que sa capacité à établir des interactions dépendra de ce paramètre. La figure 8.7 représente la distance (en Å) entre deux sommets reliés par une interaction longue-distance, en fonction de la taille (en nombre de nucléotides) des sommets impliqués. À l'échelle du nucléotide, une interaction tertiaire est nécessairement courte ; cependant, à notre échelle, un sommet ponctuel peut représenter un ensemble de nucléotides occupant un volume important. Cette figure nous indique que, dans notre représentation à gros grain, la plupart des interactions longue-distance relient des sommets espacés d'entre 10 et 30 Å. Si nous nous intéressons aux ordres de grandeurs des différents éléments de structure, nous

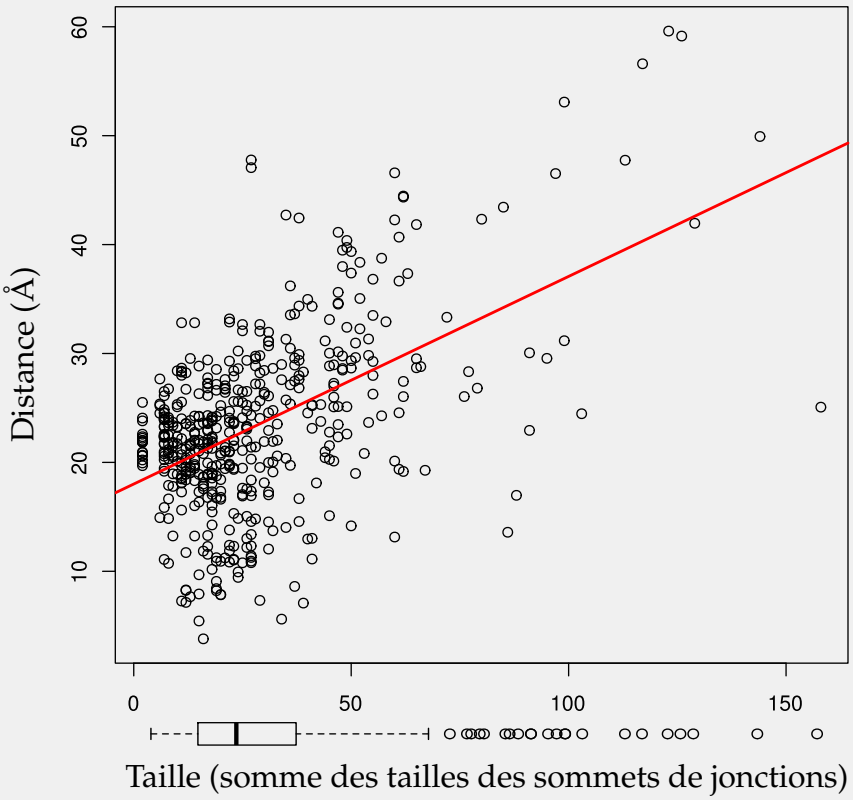


FIGURE 8.7: Statistiques de longueurs des interactions longue distance. En abscisse, nous considérons le nombre de nucléotides des jonctions qui interagissent, en tant que mesure grossière de leur encombrement. En rouge, la régression linéaire correspondante, d'équation $y \simeq 0{,}19x + 18{,}01$.

pouvons observer ainsi que les interactions longue distance seront en général *plus longues que les hélices*, ce qui était inattendu.

8.4 Plongement initial

8.4.1 Construction

Ayant déterminé les distances et angles moyens associés aux différents éléments composant un graphe squelette, nous pouvons en effectuer un plongement dans l'espace en commençant par n'importe quel sommet, placé à l'origine d'un repère, puis en ajoutant chaque autre élément bout à bout. La construction ne sera pas ambiguë, sauf si le graphe squelette comporte plus d'une composante connexe. Ce cas ne sera pas considéré. Nous nous contenterons de traiter chaque composante connexe séparément, sans essayer de prédire de quelle manière elles peuvent ensuite interagir les unes avec les autres. Dans le cas où nous mentionnons des molécules comportant plusieurs ARN (par exemple la grosse sous-unité du ribosome, 2ZJR), nous nous intéressons à la plus grosse composante connexe (23S).

Représentation des jonctions

Les n-jonctions sont représentées par n poignées, comme indiqué dans la figure 8.2 (page 63). Toutefois, dans les structures réelles, les positions de ces poignées sont variables. Nous adopterons une représentation idéalisée des jonctions. Les 2-jonctions seront représentées naturellement comme un segment dont la longueur dépend du nombre k de nucléotides contenus dans la jonction, selon l'équation $longueur = 1{,}31k - 0{,}79$ déterminée précédemment (voir figure 8.5).

Les 3-jonctions seront représentées par des triangles isocèles de base 3 Å et hauteur 9 Å, quel que soit le nombre de nucléotides contenus dans les brins de la jonction. Les deux hélices faisant partie de l'empilement seront positionnées coaxialement, et l'angle de la troisième hélice par rapport à l'axe de l'empilement sera de 90°, 160° ou 10° selon que la jonction est en famille A, B ou C, respectivement.

Les 4-jonctions en famille H ou X seront représentées comme un rectangle allongé de hauteur 2 Å et largeur 9 Å, les côtés les plus courts représentant les empilements.

Enfin, étant donné le manque d'instances de n-jonctions pour $n \geq 5$, nous avons préféré nous abstenir de tenter de les caractériser avec précision. Elles seront donc représentées par des polygones réguliers de n sommets (de diamètre 8 Å), les hélices adjacentes étant réparties en étoile autour d'elles. Par ailleurs, dans la suite, nous supposerons que ces positionnements sont plus « mous » que ceux des 3- ou 4-jonctions.

Angles entre les plans des jonctions

Dans notre représentation simplifiée, chaque jonction se trouve placée dans un plan. Les jonctions sont reliées entre elles par des hélices, ce qui signifie une rotation des paires de bases autour de l'axe de l'hélice. Par conséquent ces plans ne sont en général pas parallèles entre eux. L'angle de rotation d'une hélice d'ARN est bien connu (environ 33° par paire de bases), cependant, notre modélisation est à trop gros grain pour nous permettre de connaitre l'angle de départ des hélices. Connaitre cet angle nécessiterait de connaitre la forme des brins non appariés qui constituent l'hélice, ce qui est hors de portée de notre approche. Par conséquent, nous devons renoncer à placer correctement les plans des jonctions les uns par rapport aux autres ; la suite de notre approche, qui consiste à améliorer le plongement initial, devra prendre en compte cette lacune et autoriser des rotations de jonctions.

8.4.2 Comparaison avec les structures réelles

Afin de comparer notre plongements initiaux avec les structures réelles, nous utiliserons l'algorithme de Kabsch [61] pour calculer la RMSD (*Root mean square deviation*, voir appendice B.1.2 page 131) entre le plongement réel et le plongement initial du graphe squelette. Les points pris en compte pour le calcul de RMSD sont les positions des sommets hélices et les centres des sommets jonctions. Les résultats

pour les molécules sur lesquelles nous avons effectué nos statistiques sont présentés dans la table 8.3.

Identifiant	Sommets	RMSD (Å)	Jonctions
2ZJR	327		976554444444433333333333333333
1J5E	185		75444433333333333
2A64	37	21,18	64333
1NBS	21	10,29	43
1MFQ	19	6,10	3
1E8O	7	3,53	3

TABLE 8.3: RMSD entre le plongement initial et le plongement réel. La colonne « Jonctions » recense les n-jonction avec $n > 2$; chaque jonction est représentée par un chiffre, ainsi, la molécule 1NBS comporte-t-elle une 4-jonction et une 3-jonction.

Dans cette table, la complexité des molécules est indiquée de deux façons : la colonne « sommets » indique le nombre de sommets du graphe, et la colonne « jonctions » indique le nombre et le type de jonctions de taille supérieure à 2. Ainsi, la molécule 1NBS comporte-t-elle une 4-jonction et une 3-jonction. Il s'agit d'un meilleur indicateur de complexité que le nombre de sommets car, si nous pouvons espérer un bon plongement initial d'une molécule ne comportant que des 1-, 2- ou 3-jonctions, les jonctions de taille plus grande sont sujettes à plus de variations. Ainsi, notons que la structure secondaire de la molécule 1NBS est quasiment un sous-graphe de celle de la molécule 2A64. La différence de RMSD entre les plongements initiaux de ces deux molécules est largement due à la présence d'une 6-jonction dans la seconde, que nous avons représenté comme un hexagone sur un plan alors qu'elle a en réalité une forme plus complexe dans l'espace.

La RMSD entre le plongement initial et la forme réelle n'est pas donnée pour les molécules 1J5E et 2ZJR (deux sous-unités ribosomales). En effet, ces molécules sont beaucoup plus complexes que les autres, et notre implémentation n'est pas capable d'en construire un plongement initial au moment de la rédaction de ce manuscrit. Il ne s'agit pas d'une limitation de la méthode, mais d'un problème d'implémentation. Par ailleurs, notons que les sous-unités du ribosome contiennent de nombreuses 4-jonctions n'étant dans aucune des familles H, cH ou cL, et donc sortant du champ de notre classification.

8.5 Interactions longue distance et repliement

Lorsque nous effectuons notre plongement initial, nous n'avons connaissance que de la structure secondaire de la molécule et d'une classification des jonctions en familles topologiques, mais aucune information sur les interactions longue distance

qui peuvent entrer en jeu de la formation de sa structure tertiaire. Si nous avons connaissance de certaines de ces interactions, par exemple si un expert nous indique leur existence, nous pouvons les ajouter dans le graphe. Cependant, il est possible qu'elles relient des sommets qui sont, dans notre plongement initial, très éloignés ; pour que ces interactions aient lieu, il faut donc qu'il y ait un repliement de la molécule sur elle-même afin de respecter des contraintes de distance. Dans cette section, nous présentons un algorithme modélisant ce repliement. Pour l'instant, nous supposons que l'on nous donne une liste correcte d'interactions longue-distance. Dans les chapitres suivants, nous nous intéresserons à la découverte automatique de ces interactions.

8.5.1 Présentation de l'algorithme

Pour modéliser ce repliement, nous nous inspirons d'un algorithme de dessin de graphe modélisant le graphe comme un objet physique constitué de particules (les sommets) et de ressorts reliant ces particules (les arêtes). L'application itérative de forces de répulsion entre les particules et d'attraction entre les ressorts jusqu'à obtention d'un équilibre permet d'obtenir un dessin de graphe où les sommets ne se chevauchent pas et où les arêtes ont une taille relativement uniforme [62, 63]. L'algorithme 1 présente une version générique de cette approche.

Dans la version présentée ici, tous les sommets se déplacent simultanément, mais d'autres variantes incluent le déplacement du sommet dans la première boucle **pour**. La fonction d'amorti est décroissante en fonction du temps, afin d'accélérer les premiers déplacements et d'éviter les oscillations infinies. Les fonctions de répulsion et d'attraction couramment utilisées sont la loi de Coulomb, pour simuler la répulsion entre deux particules chargées électriquement, et la loi de Hooke, pour simuler des ressorts :

$$\text{Coulomb} = k_e \frac{q_1 q_2}{d} \tag{8.1}$$

Où k_e est une constante, q_1 et q_2 les charges électriques des deux particules, et d la distance qui les sépare.

$$\text{Hooke} = -k_r(D - d) \tag{8.2}$$

Où k_r est une constante dépendant du ressort, d la distance qui sépare les deux sommets reliés par le ressort, et D la taille idéale (c'est à dire : à l'équilibre) de ce ressort.

Dans notre application de cette méthode, nous introduisons plusieurs types de ressorts, décrits ci-dessous et représentés sur la figure 8.8.

Algorithme 1 : Dessin de graphe par application de forces

Entrées :
Un graphe squelette $G = (V, E)$
Un ensemble de ressorts R
Un (petit) seuil d'énergie ϵ
Une limite de temps *temps_max*
Sorties :
Une position dans $\mathbb{R}^3$ pour chaque sommet de V

```
1  temps = 0;
2  tant que énergie > ε et temps < temps_max faire
3      pour chaque sommet s ∈ V faire
4          s.forces = (0, 0, 0);
5          pour chaque autre sommet v ∈ V faire
6              s.forces = s.forces + repulsion(s, v)
7          fin
8          pour chaque ressort r ∈ R faire
9              s.forces = s.forces + attraction(s, r)
10         fin
11     fin
12     pour chaque sommet s ∈ V faire
13         s.position = s.position + s.forces × amorti(temps);
14         énergie = énergie + s.énergie
15     fin
16     temps = temps + 1;
17 fin
```

Structure secondaire

Les arêtes $e \in E_s$, qui représentent des demi-hélices dans le graphe squelette, sont modélisées par un ressort $r_{s,e}$ dont la distance idéale D dépend du nombre de nucléotides de la demi-hélice ; tous les ressorts $r_{s,-}$ ont la même constante k_s. Pour une demi-hélice de n nucléotides et de longueur d, en notant $D_h(n)$ la longueur « idéale » d'une hélice de n nucléotides de long, la force exercée par ce type de ressort est :

$$F_{r_{s,e}} = -k_s(D_h(n) - d)$$

Interactions longue-distance

Les arêtes $e \in E_l$ représentent les interactions longue-distance, et sont modélisées par un ressort $r_{l,e}$; tous les ressorts $r_{l,-}$ ont la même constante k_l. Les interactions longue distance sont toutes à peu près de la même taille, tous les $r_{l,-}$ devraient donc avoir la même distance idéale D, cependant nous voulons prendre

en compte le fait que certaines jonctions occupent un plus gros volume que d'autres, et nous faisons donc entrer le nombre de nucléotides des jonctions dans le calcul de D.

Soit une arête $e \in E_l$ de longueur d, impliquant deux sommets s_1 et s_2. Notons n_i le nombre de nucléotides contenus dans s_i si s_i est une jonction, 1 sinon, et notons $D_l(n)$ la longueur moyenne d'une interaction longue distance impliquant n nucléotides. La forme de ce ressort est :

$$F_{r_{l,e}} = -k_l(D_l(n_1 + n_2) - d)$$

Contraintes d'angle

À ces contraintes de distance s'ajoutent des contraintes d'angle. Afin d'inciter les hélices à être droites tout en autorisant un peu de flexibilité, nous ajoutons un ressort r_h entre les sommets représentant une hélice et le centre du segment reliant les deux jonctions adjacentes. Enfin, les jonctions sont classifiées en familles, et chaque famille a ses « préférences » en termes d'angles entre les hélices. Pour chaque jonction, nous relions ses hélices à une position idéale par un ressort r_j. Ces deux types de ressorts ont une longueur idéale D nulle, la force qui leur est associée est donc :

$$F_{r_h} = -k_h(D - d) = k_h d$$

$$F_{r_j} = -k_j(D - d) = k_j d$$

Par ailleurs, afin de compenser l'imprécision de nos prédictions d'angles pour les jonctions entre plus de quatre hélices, nous divisons la valeur de la force par deux pour ces jonctions.

Répulsion

Lors de nos expérimentations, la répulsion entre les sommets n'a pas démontré son utilité ; il n'y a d'ailleurs pas d'objection dans notre modèle à ce que des sommets occupent la même position, puisqu'ils représentent des structures en partie creuses et pouvant s'imbriquer.

L'algorithme 1 a une complexité de $O(n^2 + m)$, où n est le nombre de sommets du graphe et m son nombre d'arêtes (en supposant le nombre d'itérations borné). En supprimant cette étape de répulsion, nous pouvons ramener la complexité de l'algorithme à $O(m)$. Cette optimisation sera particulièrement utile dans les chapitres suivants, où le nombre d'arêtes m sera en $O(n)$; en effet, nous autoriserons chaque sommet à se choisir un voisin en plus des un ou deux voisins qu'il possèdera déjà, ce qui bornera le nombre d'arêtes à $3n$. Notre algorithme de repliement sera donc linéaire dans ce cadre (mais pas dans le cadre général où le nombre d'arêtes maximum possible est $n(n-1)$).

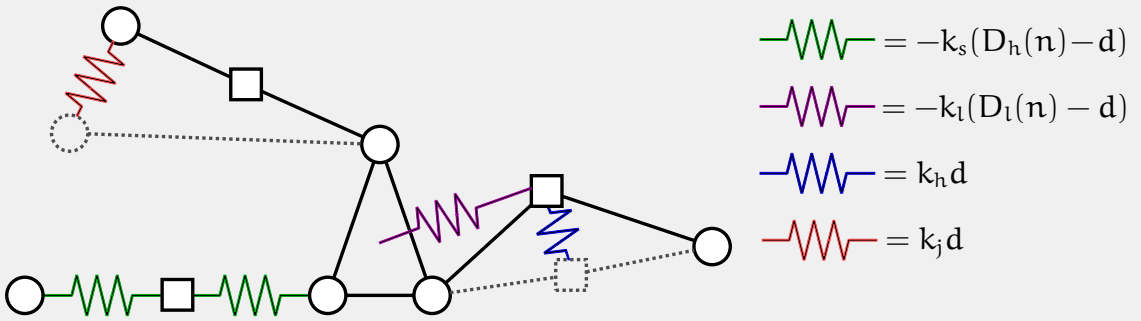


FIGURE 8.8: Les différents types de ressorts utilisés. Dans tous les cas, d représente la distance entre les extrémités du ressort. Dans le cas des ressorts verts (demi-hélices du squelette), n désigne le nombre de paires de bases de la demi-hélice représentée. Dans le cas des ressorts violets (interactions longue distance), n désigne le nombre de nucléotides associés aux jonctions impliquées ; notons le fait que ces ressorts sont reliés aux centres des jonctions, la notion de poignée n'étant pas pertinente ici. Par souci de clareté, tous les ressorts verts ne sont pas représentés.

Rotation des jonctions

Le type d'algorithme de dessin de graphe que nous utilisons ici considère chaque sommet comme une particule réduite à un point. Cependant, nos jonctions ont des poignées dont l'orientation est importante ; nous ne pouvons pas espérer obtenir un repliement proche de la réalité en n'effectuant que des translations à partir de la position initiale. Il nous faut donc ajouter une étape de rotation des jonctions sur elle-mêmes (c'est à dire autour de leur centre de gravité) pour tenter de s'aligner avec leurs voisins.

Pour une jonction J_1 donnée, nous effectuons une rotation à chaque étape de l'algorithme de repliement, et après avoir effectué les déplacements dus aux ressorts. Pour chaque jonction J_2 voisine de J_1, nous calculons la distance entre J_1 et J_2 et, si cette distance est plus grande qu'un certain seuil, nous effectuons une rotation d'une fraction de l'angle séparant les deux jonctions. Cette fraction d'angle est petite afin d'éviter des rotations trop brusques, et diminue avec le temps, de manière similaire au calcul de l'amorti dans l'algorithme 1. L'algorithme 2 décrit cette rotation.

L'algorithme 3 décrit notre algorithme de repliement final, une fois l'étape de répulsion supprimée et l'étape de rotation ajoutée.

Valeurs des paramètres

À partir de l'analyse des données cristallographiées décrire dans la section 8.3, nous avons déterminé les valeurs suivantes des paramètres de longueur :

$$\begin{aligned} D_h(n) &= 2{,}64n - 2{,}62 \quad \text{(taille idéale d'une hélice de } n \text{ paires de base)} \\ D_l(n) &= 0{,}19n + 18{,}01 \quad \text{(taille idéale d'une interaction longue distance)} \end{aligned}$$

Algorithme 2 : Rotation des jonctions

Entrées :
Un sommet de jonction J
Un seuil de distance δ
Un temps t correspondant au temps dans l'algorithme de repliement

pour chaque *jonction* J_2 *voisine de* J **faire**
    a = J_2.position;
    b = J.position_idéale_pour(J_2);
    si *distance(a, b)* $> \delta$ **alors**
        α = angle $\widehat{aJb}$;
        amorti = 100 + t/10;
        J.rotation(α/amorti)
    fin
fin

Par ailleurs, nous avons déterminé expérimentalement les valeurs des autres paramètres :

$k_s = 2$ (raideur des ressorts modélisant les hélices)
$k_l = 2$ (raideur des ressorts modélisant les interactions longue-distance)
$k_h = 2$ (raideur des ressorts incitant les hélices à rester droites)
$k_j = 0{,}8$ (raideur des ressorts modélisant les angles préférés des jonctions)
$c_l = 0{,}5$ (diminution de l'attraction des interactions longue-distance selon la taille des jonctions impliquées)

8.5.2 Adéquation de l'algorithme

Afin de valider l'adéquation de cet algorithme pour modéliser le repliement induit par l'ajout d'interactions longue distance, nous devons vérifier, d'une part, qu'il rapproche une structure « mal repliée » de la structure réelle, et d'autre part, qu'il est neutre lorsqu'on l'applique à la structure réelle. Il est impossible d'atteindre parfaitement ces deux objectifs, et nous devons effectuer un compromis qui ne déforme pas trop les structures réelles, tout en imposant des contraintes fortes sur nos plongements initiaux.

Nous avons pris le plongement réel des graphes squelettes des molécules munis de leurs interactions longue distance réelles et, dans un cas, nous l'avons laissé tel quel, dans l'autre cas, nous avons appliqué notre algorithme de repliement. La table 8.4 présente la RMSD entre ces deux graphes. Idéalement, cette RMSD devrait toujours être nulle, mais elle ne peut pas l'être car notre modèle est nécessairement imparfait. Il faut noter que ces mesures de RMSD donnent également une limite maximale à la qualité de notre prédiction : nous ne pouvons raisonnablement espérer obtenir une meilleure RMSD sur nos prédictions que ce que nous obtenons en repliant la structure réelle.

Algorithme 3 : Notre algorithme de repliement

Entrées :
Un graphe squelette $G = (V, E)$
Un ensemble de ressorts R
Un (petit) seuil d'énergie ϵ
Un seuil de distance avant une rotation δ
Une limite de temps *temps_max*
Sorties :
Une position dans $\mathbb{R}^3$ pour chaque sommet de V

```
1  temps = 0;
2  tant que énergie > ε et temps < temps_max faire
3      pour chaque sommet s ∈ V faire
4          s.forces = (0, 0, 0);
5          pour chaque ressort r ∈ R faire
6              s.forces = s.forces + attraction(s, r)
7          fin
8      fin
9      pour chaque sommet s ∈ V faire
10         s.position = s.position + s.forces × amorti(temps);
11         si s est une jonction alors
12             rotation(s, δ, temps);
13         fin
14         énergie = énergie + s.énergie
15     fin
16     temps = temps + 1;
17 fin
```

Identifiant	Sommets	RMSD après repliement (Å)
2ZJR	347	7,83
1J5E	185	6,11
2A64	37	3,99
1NBS	21	3,00
1MFQ	19	2,18
1E8O	7	1,58

TABLE 8.4: Influence de l'algorithme de repliement sur une structure déjà bien repliée.

La table 8.5 compare la RMSD du plongement initial des molécules avec celle obtenue en ajoutant les interactions longue distance réelles et en appliquant l'algorithme de repliement. Il s'agit du résultat que nous obtiendrions si nous avions connaissance des interactions réelles, donc d'une estimation du meilleur résultat

Identifiant	Sommets	RMSD (Å)	
		Sans interactions	Avec interactions
2A64	37	21,40	19,82
1NBS	21	10,29	8,39
1MFQ	19	6,10	5,97
1E8O	7	3,53	3,80

TABLE 8.5: Le repliement nous rapproche-t-il de la forme réelle ? À gauche, la RMSD entre le plongement initial et le plongement réel ; à droite, la RMSD entre le plongement initial muni des interactions longue-distances réelles, replié, et le plongement réel.

que nous pouvons espérer obtenir en essayant de découvrir ces interactions (ce qui sera le sujet des chapitres suivants).

De manière prévisible, sur de petites molécules, l'algorithme de repliement ne nous aide pas. Les molécules 1MFQ et 1E80 sont, dès le départ, dans la bonne position, et ne font d'interaction longue-distance qu'avec leurs voisins proches. L'algorithme de repliement est utile à partir du moment où il permet de rapprocher des sommets qui étaient éloignés dans le plongement initial.

Identifiant	Sommets	RMSD (Å)	
		Sans interactions	Avec interactions
1NBS C	21	10,29	8,39
1NBS A	21	17,28	8,56
1NBS B	21	26,53	17,28
1MFQ C	19	6,10	5,97
1MFQ A	19	14,99	7,33
1MFQ B	19	23,78	13,20

TABLE 8.6: RMSD avec et sans les interactions réelles, selon la classification des 3-jonctions. La classification correcte est en gras. La famille A est celle qui forme un angle droit avec l'empilement, c'est donc naturellement celle qui est la plus facile à « corriger » en C.

Jusque là, nous avons supposé connaître les familles topologiques de chaque jonction du squelette. Il est également intéressant de mesurer la qualité de notre plongement initial et de l'algorithme de repliement dans le cas où nous aurions fait des erreurs de classification. La table 8.6 présente la même comparaison de RMSD avec et sans interactions longue-distance, pour deux molécules, selon la classification que nous avons attribuée à la 3-jonction qu'elles contiennent. Dans les

deux cas, nous avons affaire à une famille C. La famille la plus proche, en terme d'angles, est la famille A ; il n'est pas surprenant que ce soit elle qui obtienne la meilleure RMSD avant et après l'ajout des interactions longue-distance, car c'est celle qui nécessite le moins de déformation pour être « contrainte » à prendre la forme d'une famille C pour satisfaire ses interactions longue-distance. Cependant, même lorsque les jonctions sont par erreur classifiées en famille B, l'ajout des interactions réelles permet d'obtenir une bonne RMSD. On peut donc espérer pouvoir ainsi rattraper en partie les erreurs de classification lors de l'étape de découverte des interactions longue-distance.

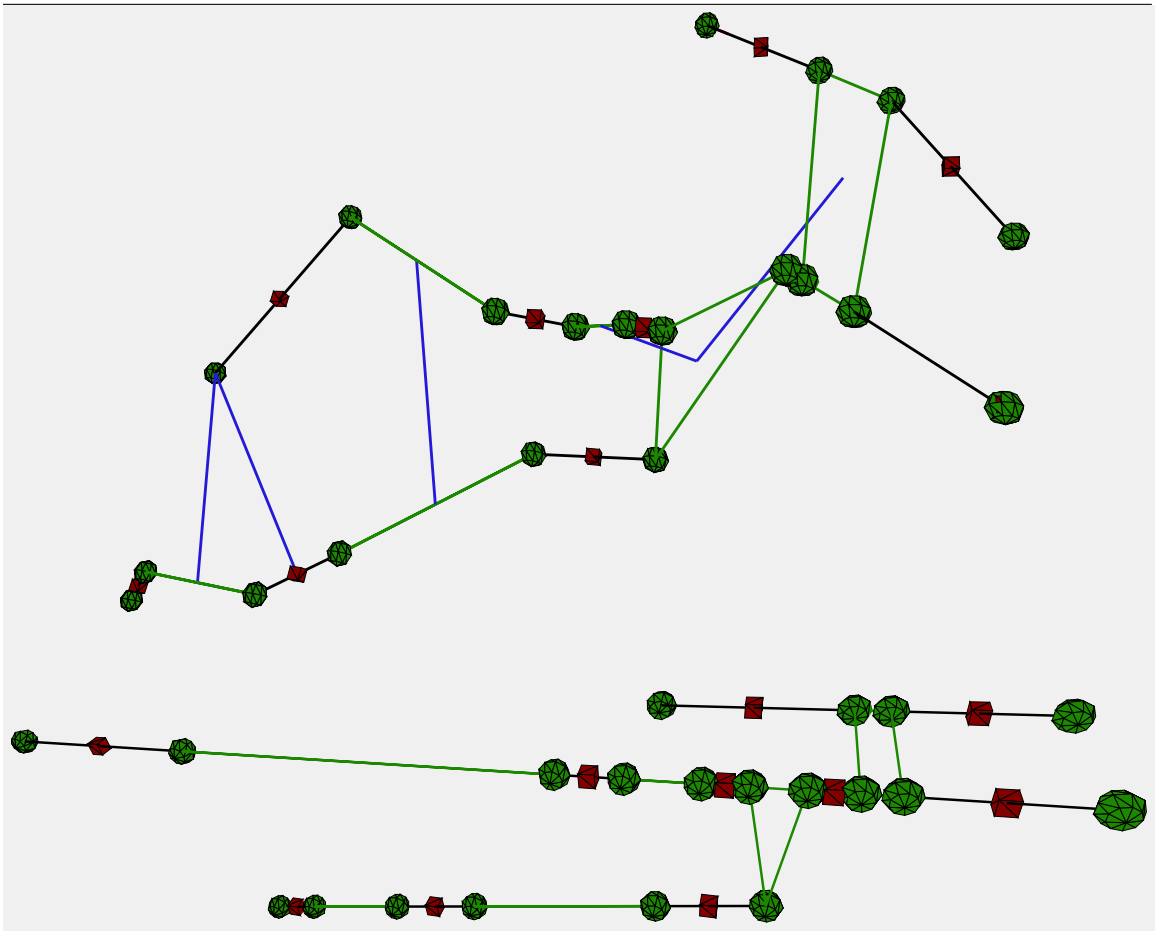


FIGURE 8.9: Forme réelle (en haut) et plongement initial (en bas) de la molécule 1NBS, qui comporte une 3-jonction en famille C et une 4-jonction en famille H. Les jonctions sont les sphères vertes reliées par des arêtes vertes ; les hélices sont les cubes rouges. Les arêtes noires relient les éléments adjacents dans la structure secondaire, et les arêtes bleues représentent les interactions longue distance.

La RMSD permet de comparer les qualités relatives de différents plongements, mais ne permet pas facilement de visualiser la qualité globale du résultat. La figure 8.9 représente le plongement réel et le plongement initial du squelette de

la molécule 1NBS. Nous pouvons constater quelques erreurs de longueurs sur les 2-jonctions de la partie gauche, et dans l'ensemble des angles trop « droits », mais la forme globale de la molécule est reconnaissable.

8.6 Fonction de coût

Dans les chapitres suivant, nous allons étudier différentes méthodes dont l'objectif est de déterminer un ensemble d'interactions à longue distance fournissant une bonne stabilité à la molécule. Nous considérerons une approche d'optimisation globale et une approche d'optimisation égoïste reposant sur la théorie des jeux ; dans les deux cas, nous aurons besoin de définir une fonction de coût, chargée de représenter l'intensité des forces s'appliquant sur chaque sommet.

Alors que dans notre algorithme de repliement, les forces exercées par les ressorts sur un sommet s'additionnaient pour produire un vecteur de déplacement de ce sommet, nous allons maintenant considérer la somme des intensité de ces forces. Ainsi, deux ressorts identiques appliquant des forces de sens opposés sur un sommet ne s'annulent plus, mais s'ajoutent et augmentent le coût payé par le sommet pour rester dans sa position. Pour un sommet s donné, soit $R(s)$ l'ensemble des ressorts qui s'appliquent sur s, et pour $r \in R(s)$ un ressort donné, soit k_r sa raideur, d_r sa longueur et D_r sa longueur idéale. Alors, la somme F_r des intensités des forces dues aux ressorts s'appliquant sur s est :

$$F_r(s) = \sum_{r \in R(s)} |-k_r(D_r - d_r)|$$

À cela, nous ajoutons une répartition des coûts subis par les voisins $V(s)$ de s, selon un cœfficient C_r. En effet, si une interaction longue-distance entre s et un autre sommet provoque une déformation pénalisant un voisin de s, nous voulons que s paye une partie de ce coût, afin qu'il se « rende compte » de la déformation.

$$F_v(s) = C_r \sum_{v \in V(s)} F_r(v)$$

Si notre fonction de coût est correctement ajustée, lorsqu'un graphe squelette est dans son plongement initial, alors les coûts F_r et F_v doivent être nuls pour tous les sommets. Ajouter des interactions longue-distance provoque un repliement, donc une déformation par rapport à cette situation idéale ; si les interactions longue-distance n'étaient pas récompensées, il n'y aurait intérêt pour un sommet à en créer. Nous modélisons le gain de stabilité obtenu par la création d'interactions longue-distance en attribuant simplement une réduction de coût L pour chaque interaction concernant un sommet :

$$F_l(s) = \sum_{\forall x,(s,x) \in E_l} L$$

La fonction de coût $F(s)$ globale pour un sommet est alors :

$$F(s) = F_r(s) + F_v(s) - F_l(s) \tag{8.3}$$

Enfin, il est parfois plus naturel de considérer que des joueurs maximisent une fonction de gain plutôt qu'ils ne minimisent une fonction de coût, bien que le problème soit équivalent. Nous utiliserons alors la fonction de gain suivante :

$$Gain(s) = 10 - F(s)$$

8.7 Conclusion

Dans ce chapitre, nous avons défini une représentation à gros grain de la forme des molécules d'ARN, nommée *graphe squelette*, dont les sommets représentent les hélices et jonctions entre hélices de la structure secondaire. Nous avons mesuré sur un échantillon de molécules cristallographiées les paramètres moyens de distance entre ces sommets, et fonction de leurs caractéristiques (longueur des hélices, taille des jonctions). Cela nous a permis de formuler une méthode permettant, à partir d'un graphe squelette non positionné, de créer un plongement dans l'espace de ce graphe respectant ces paramètres moyens, nommé *plongement initial*. Cependant, ce plongement initial est construit à partir d'un graphe squelette représentant la structure secondaire de la molécule ; il ne tient pas compte d'éventuelles interactions tertiaires entre des parties de la molécule qui sont éloignées sur la structure secondaire. Pour tenir compte d'interactions longue-distance, nous avons proposé un algorithme permettant de replier le plongement initial afin de respecter les contraintes de distance introduites par ces interactions.

Nous avons modélisé le repliement de la molécule lors de l'ajout d'interactions longue distance, mais dans un contexte de prédiction de structure, ces interactions ne nous sont pas données en entrée. Nous allons consacrer les deux chapitres suivant à la découverte automatique d'interactions longue distance. Nous avons défini dans ce chapitre une fonction de coût (et une fonction de gain analogue) mesurant l'inconfort d'un sommet comme une fonction de son déplacement par rapport à sa position initiale et du gain de stabilité obtenu par l'établissement de nouvelles interactions. Le chapitre 9 utilisera cette fonction dans une approche d'optimisation globale, et le chapitre 10 l'utilisera dans une modélisation par un jeu, où chaque sommet essaye de maximiser son propre gain.

9 Optimisation globale

Résumé Dans ce chapitre, nous développons une approche d'optimisation globale par algorithme évolutionniste, dans le but de déterminer automatiquement les interactions longue-distance minimisant la somme des coûts des sommets du graphe squelette d'une molécule.

9.1 Introduction

Nous avons opté pour l'utilisation d'un algorithme évolutionniste [64, 65] afin d'optimiser la somme des coûts des différents sommets. Cette classe d'algorithmes s'inspire du processus de sélection naturelle décrit par Charles Darwin [66]. Une population de solutions est générée aléatoirement, des règles (ou *opérateurs*) de reproduction sont définies, et une pression de sélection est créée afin de favoriser la reproduction des solutions de meilleur coût ; le schéma général de l'algorithme que nous avons utilisé est décrit dans l'algorithme 4.

L'efficacité d'un algorithme évolutionniste dépend de la définition des opérateurs de *fitness*, *croisement* et *mutation*. Le fitness décrit l'adéquation d'un individu à son environnement ; le croisement détermine comment un nouvel individu est créé à partir de ses parents, et la mutation détermine des modifications aléatoires qui peuvent se faire sur les individus afin de créer de la diversité qui n'était pas présente dans la population de départ.

9.2 Définition du problème et des opérateurs

9.2.1 Problème

En entrée du problème, nous avons un graphe squelette $G = (V, E)$, doté de son plongement initial dans l'espace. Ce graphe G possède n sommets, possédant chacun sa propre fonction de gain g telle que définie au chapitre précédent. Nous souhaitons optimiser la fonction F définie comme la somme des gains des sommets de G :

$$F = \sum_{s \in V} g(V)$$

Algorithme 4 : Schéma d'algorithme évolutionniste simple

Entrées : Un graphe squelette $G = (V, E)$, un ensemble de ressorts R, un (petit) seuil d'énergie ϵ, une limite de temps *temps_max*.
Sorties : Un choix $s \in V$ de partenaire d'interaction pour chaque sommet.

```
1  temps = 0;
2  faire
3      moyenne = 0;
4      pour chaque individu i de P faire
5          i.fitness = fitness(i);
6          moyenne = moyenne + i.fitness;
7      fin
8      moyenne = moyenne / taille(P);
9      candidats = {i ∈ P : i.fitness ≥ moyenne};
10     pour chaque individu i tel que i.fitness < moyenne faire
11         parent1 = tirage_uniforme(candidats);
12         parent2 = tirage_uniforme(candidats);
13         i = croisement(parent1, parent2);
14         mutation(i);
15     fin
16     temps = temps + 1;
17 tant que temps < temps_max ;
```

9.2.2 Individu

Nous définissons un individu comme un ensemble I d'arêtes de type « interaction longue-distance » entre des sommets de G. Cependant, afin de limiter le nombre de possibilités, nous limitons à n le nombre d'arêtes de I, en autorisant chaque sommet à « se choisir » zéro ou un partenaire d'interaction. Formulé autrement, un individu est une liste de n choix parmi n possibilités. Un sommet u a pour choix possibles $C_u = \{(u, v) : v \in V \wedge v \neq u\} \cup \{\emptyset\}$, et un individu I est un élément de $C_0 \times C_1 \times \ldots \times C_n$.

Nous représenterons un individu comme un tableau de n cellules, la cellule i contenant l'identifiant du sommet partenaire de i, ou $\emptyset$ si i n'a pas de partenaire :

Sommet	1	2	3	...	n
Partenaire	$\emptyset$	4	1	...	$\emptyset$

9.2.3 Fitness

Le fitness d'un individu est la somme des gains de chaque sommet de G, une fois le repliement induit par $G \cup I$ effectué. Cependant, nous désirons interdire certaines configurations biologiquement improbables. Par exemple, nous ne souhaitons pas que deux hélices ou deux boucles internes séparées par une hélice puissent

établir des interactions, même si les distances entre ces sommets dans l'espace le permettent. Pour ce faire, nous considérons qu'une « branche » du squelette est une chaîne composée d'hélices et de 1- ou 2-jonctions, et nous pénalisons les interactions entre deux sommets d'une branche, ou entre deux hélices, d'un malus de -100.

9.2.4 Croisement

Soit un sommet $u \in V$, et un individu I. Nous noterons $I[u]$ l'arête (ou $\emptyset$) correspondant au choix de u dans l'individu I. Nous définissons un opérateur de croisement entre deux parents I_1 et I_2, créant un nouvel individu, qui consiste à prendre, pour chaque sommet $u \in V$, soit $I_1[u]$ avec une probabilité de $1/2$, soit $I_2[u]$ avec une probabilité de $1/2$. Par exemple :

Sommet	1	2	3	4	5	6
I_1		4		3	$\emptyset$	
I_2	$\emptyset$		1			$\emptyset$
I	$\emptyset$	4	1	3	$\emptyset$	$\emptyset$

9.2.5 Mutation

Lors de la mutation d'un individu, chaque sommet a 2% de chances de choisir un nouveau partenaire (ou $\emptyset$). Le taux de mutation peut être ajusté selon que l'on souhaite augmenter la diversité ou conserver les individus ayant un bon fitness.

9.3 Résultats

Nous avons utilisé une population de 200 individus. Vingt individus ont été générés sans interactions longue-distance, et l'opérateur de mutation est appliqué sur dix-sept d'entre eux. Les autres individus sont générés aléatoirement. Ce biais nous sert à nous assurer la présence dans la population de départ d'au moins quelques individus « assez bons » du point de vue de la fonction de fitness, et accélérer ainsi la convergence. Nous appliquons ensuite notre algorithme évolutionniste sur cette population, en bornant le nombre de générations à 1000 ; en pratique, la convergence est atteinte avant la génération 500.

La table 9.1 recense les résultats de l'optimisation globale pour différentes molécules, avec ou sans erreurs de classification. Le gain total est représenté avant et après optimisation. Les molécules ont été choisies afin de représenter plusieurs tailles et niveaux de structuration, allant d'une seule 3-jonction (pour la molécule 1E8O) à une 6-jonction, une 4-jonctions et trois 3-jonctions, pour 2A64. L'ARN de transfert, qui sert souvent d'étalon des méthodes de prédiction de structure, n'a pas été considéré, car il est uniquement constitué d'une 4-jonction (et de quatre 1-jonctions adjacentes). Une fois classifié en famille cL (voir section 6.2 page 51),

Identifiant	Sommets	Gain total		RMSD (Å)		
		Initial	Optimisation	Initial	Optimisation	Réel
1E8O C	7	51,6	68,9	3,53	3,48	3,80
1E8O A	7	52,2	69,6	5,89	6,00	4,66
1E8O B	7	51,1	70,0	7,86	8,17	7,08
1MFQ C	19	79,4	150,9	6,10	6,88	5,97
1MFQ A	19	82,2	154,8	14,99	7,57	7,33
1MFQ B	19	77,8	144,8	23,78	23,69	13,20
1NBS C	21	122,0	207,1	10,29	9,38	8,39
1NBS A	21	124,8	210,0	17,28	8,86	8,56
1NBS B	21	119,8	206,0	26,53	28,30	17,28
2A64 CAC	37	204,1	311,8	21,40	23,22	19,82
2A64 AAA	37	206,1	323,4	21,36	23,40	19,67
2A64 CCC	37	189,4	319,9	25,61	26,88	24,74

TABLE 9.1: Résultats de l'optimisation globale. Pour chaque molécule, nous considérons différentes classifications pour les jonctions qui la composent ; la ligne en gras indique la classification correcte. La colonne « Réel » indique la RMSD obtenue si nous munissons le graphe des interactions réelles.

sa forme est déterminée, mais nous ne pouvons pas revendiquer cela comme une bonne « prédiction », dans la mesure où la réponse est contenue dans la classification.

La RMSD entre le graphe squelette prédit et le graphe squelette réel est donnée avant et après optimisation ; nous indiquons aussi, dans la colonne « réel », la RMSD que nous obtiendrions si nous trouvions les interactions qui se produisent dans la réalité. Dans tous les cas, l'algorithme optimise bien la fonction de coût, mais cela ne se traduit pas systématiquement par une réduction de la RMSD comme nous pourrions l'espérer. Dans les cas où les jonctions sont correctement classifiées, notre algorithme ne produit pas de variation significative de la RMSD. Cela s'explique par le fait que notre plongement initial est assez bon lorsque la classification est correcte, et difficilement améliorable. Dans ce cas, les erreurs du plongement initial sont dues en grande partie à l'imprécision inhérente à notre représentation des jonctions comme polygones fixes, et ne sont pas corrigeables par optimisation.

Remarquons également que la valeur de RMSD après optimisation est parfois meilleure que la valeur que nous aurions avec les interactions réelles (colonne de droite). En effet, notre fonction de gain étant un modèle imparfait de la réalité, il peut exister des choix d'interactions produisant un meilleur gain que les interactions constatées dans les structures cristallographiées. Cependant, au mieux, les différences constatées sont de l'ordre de 0,4 Å, ce qui n'est pas significatif à cette

échelle.

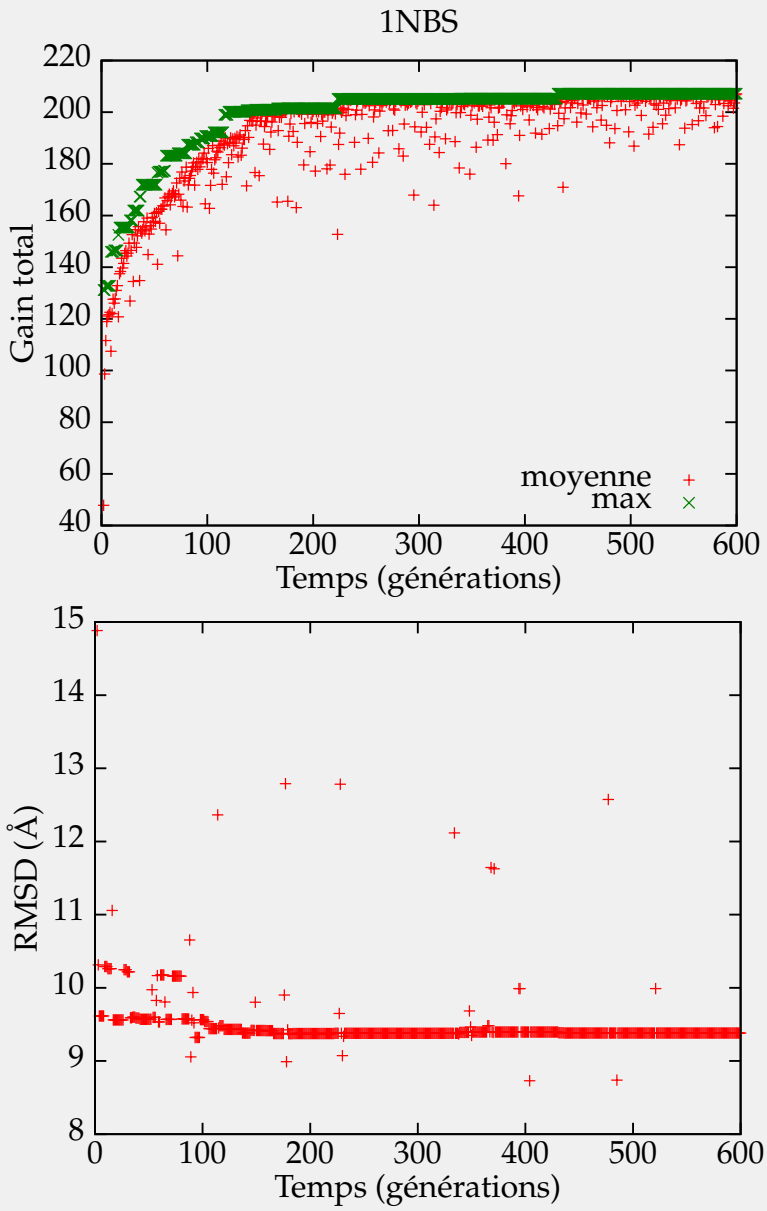


FIGURE 9.1: Gain total et RMSD obtenue à partir du plongement initial de la molécule 1NBS.

La figure 9.1 donne le détail de l'évolution de l'optimisation au fil des générations pour la molécule 1NBS, correctement classifiée.

Notre classification de départ étant imparfaite, il est intéressant de savoir si l'optimisation permet rétrospectivement de corriger certaines erreurs. Nous avons donc également utilisé de mauvaises classifications pour les 3-jonctions des molécules de la table 9.1. Nous pouvons constater, dans le cas des molécules 1NBS

et 1MFQ, que les jonctions classifiées par erreur en famille A sont correctement remplacées en famille C, mais qu'une classification en famille B n'est pas modifiée. Le détail de l'évolution de l'algorithme pour les deux mauvaises classifications de 1MFQ sont données en figure 9.3. Le fait qu'une famille A puisse plus facilement se transformer en famille C qu'une famille B s'explique par leurs formes respectives : dans la famille A, la troisème hélice est perpendicculaire à l'empilement, donc « à mi-chemin » entre une famille B et une famille C. Elle pourra donc se transformer en l'une ou l'autre selon les possibilités d'interactions offertes par son voisinage dans l'espace.

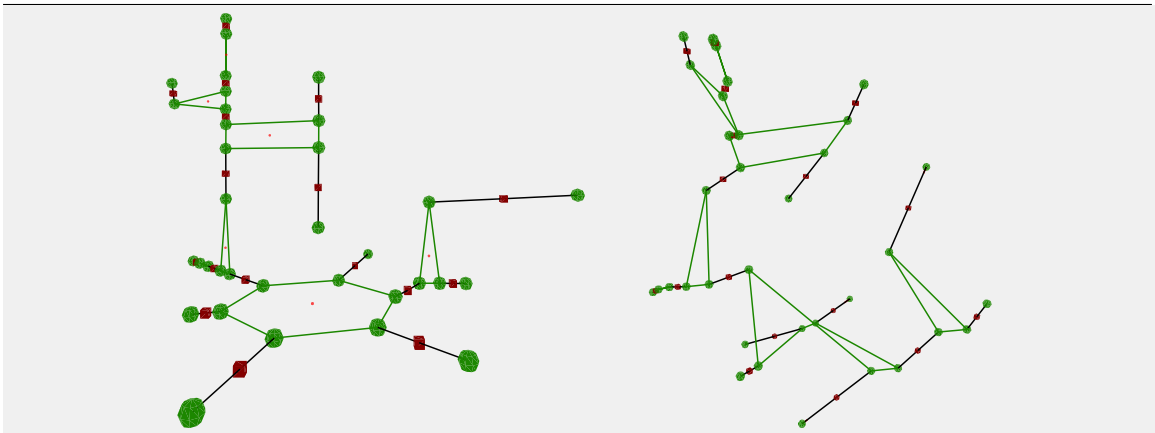


FIGURE 9.2: Plongement initial et plongement réel du squelette de 2A64.

Enfin, nous pouvons constater que, dans le cas de la molécule 2A64, l'optimisation de la fonction de gain globale conduit systématiquement à une augmentation de la RMSD. Une comparaison (figure 9.2) entre le plongement initial (à gauche) et le plongement réel (à droite) nous permet d'identifier que la plus grosse partie des erreurs de positionnement sont dues à l'utilisation d'un hexagone pour représenter la 6-jonction (en bas) et à une mauvaise rotation du plan de la 3-jonction de gauche. Si l'on exclue du calcul de RMSD la 6-jonction et les branches adjacentes, la RMSD du plongement initial passe de 21 Å à moins de 10 Å. Améliorer les résultats nécessiterait donc une meilleure prédiction de la structure des 6-jonctions.

9.4 Conclusion

Dans ce chapitre, nous avons présenté un algorithme génétique optimisant la fonction de gain définie au chapitre précédent. Nous avons montré que cette optimisation ne conduit pas à une amélioration significative de la RMSD entre notre prédiction et la structure cristallographiée si les jonctions étaient bien classifiées, mais qu'elle permet de corriger certaines erreurs de classification. Dans le chapitre suivant, nous allons présenter une nouvelle méthode d'optimisation reposant

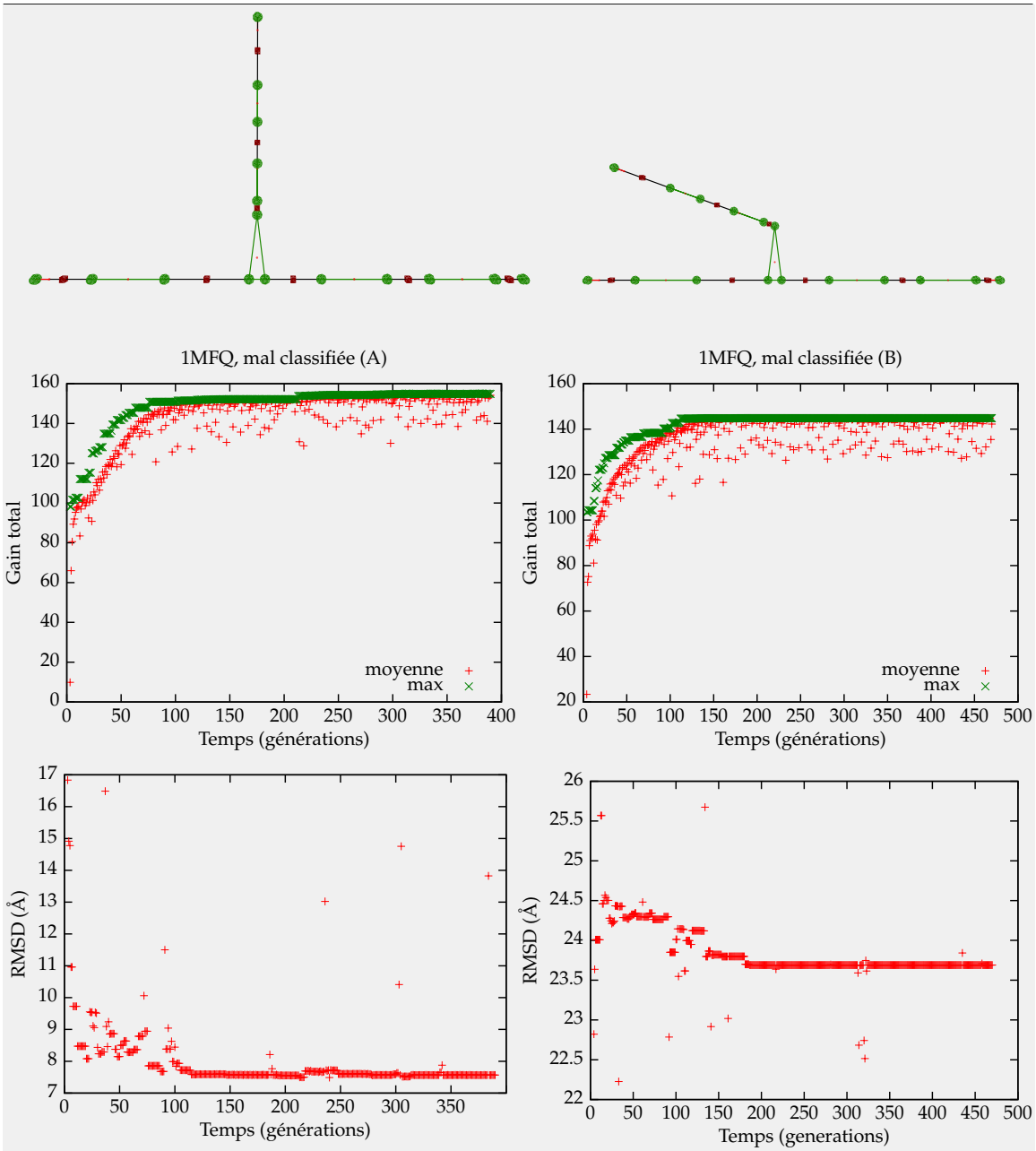


FIGURE 9.3: Gain total et RMSD obtenue à partir d'un plongement initial où la 3-jonction de la molécule 1MFQ est mal classifiée. Les quatre premières générations, presque totalement aléatoires, sont omises, car leur gain est très négatif. Lorsque la jonction est classifiée en famille A, le repliement se fait dans le bon sens ; dans le cas de la famille B, le repliement se faire sur l'autre hélice de l'empilement, et notre procédure n'améliore pas la RMSD.

sur la théorie des jeux. Au lieu d'une optimisation globale, celle-ci décrira une optimisation locale, égoïste, où chaque sommet essaye de maximiser son gain.

10 Modélisation par un jeu

Résumé Dans ce chapitre, nous proposons une alternative à l'optimisation globale : la recherche d'équilibres de Nash dans un jeu où les sommets du graphe squelette sont les joueurs. Nous étudions plusieurs algorithmes heuristiques de recherche d'équilibre.

10.1 Introduction

Alors que, dans le chapitre précédent, nous considérions que les différents acteurs (les sommets du graphe squelette) œuvraient pour le « bien commun », c'est à dire pour la stabilité globale de la molécule, dans ce chapitre nous allons présenter une approche où chaque acteur est égoïste, et optimise sa propre fonction de gain. Cette approche a pour but d'être plus réaliste et d'éviter les situations où, pour favoriser la stabilité globale, un algorithme peut choisir de pénaliser fortement certains acteurs. *In vivo*, la présence d'interactions tertiaires est due à une stabilité locale accrue, sans considération pour la stabilité globale de la molécule, que nous considérons une propriété émergente découlant des interactions individuelles ; c'est ce que nous essayons de capturer au moyen d'une optimisation égoïste.

L'outil permettant d'étudier les interactions entre acteurs égoïstes est la théorie des jeux, introduite en 1944 par John Von Neumann et Oskar Morgenstern [67]. Nous donnerons ici une description générale de ses concepts.

Un *jeu* est un processus regroupant plusieurs joueurs possédant chacun un ensemble de stratégies possibles, ainsi qu'une fonction de gain (ou d'*utilité*, en termes économiques). Les joueurs, ayant effectué des choix de stratégie, obtiennent un gain dépendant des choix de tous. Le concept d'*équilibre de Nash*, introduit en 1950 par John Nash [68], décrit une situation stable où chaque joueur, individuellement, n'a pas intérêt à changer son choix de stratégie (il faut qu'au moins deux joueurs changent de stratégie pour que l'un d'entre eux puisse augmenter son gain). Un équilibre de Nash peut être *pur* (un choix unique de stratégie par joueur) ou *mixte* (un choix d'un vecteur de probabilités de choix de stratégies par joueur). La recherche d'un équilibre de Nash est généralement l'objectif de la modélisation d'un problème par un jeu. Dans ce chapitre, nous présenterons une modélisation du problème de la découverte d'interactions longue-distance sous forme de jeu et nous étudierons quelques approches visant à atteindre des équilibres de Nash par

apprentissage et répétition du jeu.

10.2 Modélisation

Soit $G = (V, E)$ un graphe squelette. Les joueurs sont les sommets de V, et les stratégies du joueur i consistent à choisir un autre joueur et d'établir une interaction longue-distance avec lui, ou bien de ne choisir aucun joueur :

$$S_i = \{(i, j) : j \in V \wedge j \neq i\} \cup \{\emptyset\}$$

Chaque joueur dispose donc de $|V|$ stratégies possibles. Une fois les stratégies choisies, nous disposons de entre 0 et $|V|$ arêtes de type interaction longue-distance à ajouter à G. Une fois ces arêtes ajoutées, le plongement initial de G est replié suivant l'algorithme 3 (page 76), puis le gain des joueurs est calculé sur ce plongement replié. La fonction de coût des joueurs est la fonction F décrite à l'équation 8.3, page 80.

Posons $n = |V|$. Dans G, il y a $n - 1$ arêtes de type « structure secondaire », car la structure secondaire est un arbre ; il y a entre 0 et n arêtes de type « interaction longue distance », donc le nombre d'arêtes de G est en $O(n)$. Par conséquent, le calcul de la fonction de gain d'un joueur est en $O(n)$, et nous pouvons calculer les gains de tous les joueurs en une passe. De plus, avant de calculer les gains des joueurs, nous devons replier le placement initial du graphe, ce qui nécessite également $O(n)$ opérations (voir la remarque du paragraphe « répulsion », page 73). Au final, l'ensemble de la procédure de repliement et calcul de gains est en $O(n)$.

Cette fonction de gain fonctionne comme une boîte noire ; autrement dit, même si un joueur connait les stratégies $s' \in S_{-i}$ choisies par ses adversaires, sa fonction de gain F_i ne permet pas de calculer la meilleure réponse b : $F_i(b, s') = \max_{s \in S_i} F_i(s, s')$ sans calculer toutes les valeurs $F_i(s, s')$. De plus, pour un graphe à n sommets, il existe n^n combinaisons de stratégies possibles, ce qui rend impossible le pré-calcul du gain de toutes les combinaisons de stratégies pour les graphes de plus de quelques sommets. Cela limite nos possibilités de calcul analytique d'un équilibre de Nash, et nous oriente vers des approches d'apprentissage par répétition du jeu, que nous décrivons à la section suivante.

10.3 Découverte d'un équilibre de Nash

Tous les jeux ne possèdent pas d'équilibre de Nash pur, mais tous possèdent au moins un équilibre mixte [68]. Calculer un équilibre mixte dans un jeu quelconque est PPAD-complet, mais probablement pas NP-complet [69]. PPAD (*Polynomial Parity Argument on Directed graphs*) est une classe de complexité, introduite par Christos Papadimitriou, qui regroupe un certain nombre de problèmes pour lesquels il existe une preuve d'existence de solution non constructive basée sur un

Algorithme 5 : Schéma général

Entrées :
– $\forall j$, P_j un vecteur de probabilité sur les stratégies de j
– une fonction de gain (qui peut être distincte pour chaque joueur)

```
1 tant que l'algorithme n'a pas convergé faire
2     pour chaque joueur j faire
3         actions[j] = tirage(j)
4     fin
5     coûts = fonction_de_gain(actions);
6     pour chaque joueur j faire
7         mise_a_jour(P_j, coûts[j])
8     fin
9 fin
```

argument de parité. On pense que les problèmes de PPAD sont difficiles [70], mais il est possible que PPAD = P tout en ayant P ≠ NP ; l'appartenance à PPAD n'est donc pas une indication de difficulté aussi forte que le fait d'être NP-complet. Quoi qu'il en soit, nous ne disposons pas à ce jour d'un algorithme polynomial permettant de calculer un équilibre de Nash pour un jeu quelconque (il existe des algorithmes polynomiaux pour certaines classes restreintes de jeux).

Lorsque des humains jouent à un jeu dont ils ne maitrisent pas bien le fonctionnement, et donc pour lequel ils ne savent pas calculer une stratégie optimale, ils essayent de se rapprocher de cette stratégie optimale par essais successifs. Différentes approches tentent de simuler ce processus pour apprendre un équilibre de Nash ; nous allons en étudier quelques unes.

10.3.1 Algorithme de Sastry

Sastry et co-auteurs [71] ont proposé un algorithme simple d'apprentissage pour lequel ils ont démontré des résultats de convergence quel que soit le jeu. L'algorithme 5 donne le schéma général de la méthode de Sastry, dans lequel, tant que l'algorithme n'a pas convergé, les joueurs tirent une stratégie au hasard selon leur vecteur de probabilité, observent leur gain, et mettent à jour leur vecteur de probabilité en fonction de ce gain.

L'algorithme 6 décrit le tirage de stratégie pour chaque joueur dans l'algorithme de Sastry. Il s'agit ici d'un simple tirage selon un vecteur de probabilités propre à chaque joueur, mais nous verrons par la suite une modification de cet algorithme de tirage de stratégie.

Enfin, le cœur de la méthode de Sastry est dans l'algorithme 7, qui décrit la mise à jour du vecteur de probabilité d'un joueur en fonction de son gain. Si un joueur a choisi une stratégie *action*, et s'il a obtenu un gain *gain*, alors nous multiplions la probabilité de rejouer *action* par ce gain et par un facteur $b \in \mathbb{R}^+$ très petit.

Algorithme 6 : Tirage de stratégie (Sastry)

Entrées : P_j le vecteur de probabilité des m stratégies de j
Sorties : Un choix de stratégie

```
1 aléa = random(0, 1);
2 valeur = P_j[1];
3 pour i de 1 à m faire
4     si aléa ≤ valeur alors
5         retourner i
6     valeur = valeur + P_j[i+1]
7 fin
```

Algorithme 7 : Mise à jour du vecteur de probabilités (Sastry [71])

```
1  c = 0;
2  pour chaque stratégie s faire
3      si P_j[s] > 0 alors
4          c = c+1
5      fin
6  fin
7  x = b × gain × (1 − P_j[action]);
8  P_j[action] = P_j[action] + x;
9  pour chaque stratégie s faire
10     si P_j[s] > 0 ET s ≠ action alors
11         P_j[s] = P_j[s] - x/c;
12     fin
13 fin
```

Nous ajoutons le résultat à la probabilité de rejouer *action*, et nous le retranchons (uniformément réparti) aux probabilités de jouer les autres stratégies.

Sastry et co-auteurs [71] montrent que, si b tend vers 0, l'algorithme converge vers un équilibre de Nash, sans que l'on ait besoin de supposer que la fonction de gain des joueurs possède des propriétés particulières. Les joueurs ne connaissent pas eux-même leur propre fonction de gain, et la découvrent au cours des répétitions du jeu.

Détaillons le calcul de x, à la ligne 7 de l'algorithme, x désignant l'augmentation de la probabilité de rejouer la même action que celle que le joueur vient de choisir. Cette augmentation dépend naturellement du gain obtenu ; plus celui-ci est élevé, plus le joueur aura envie de rejouer la même action. Le facteur b sert à ralentir les modifications au vecteur de probabilité, afin de ne pas converger trop rapidement vers une stratégie pure. Enfin, le facteur $1 - P_j[\text{action}]$ sert à ralentir la convergence lorsque la probabilité se rapproche de 1, : si un joueur obtient un gain élevé avec une stratégie qu'il jouait rarement, il va beaucoup augmenter sa probabilité de

Algorithme 8 : Tirage Monte-Carlo

Entrées :
P_j le vecteur de probabilité des m stratégies de j
X un entier
Sorties :
Un choix de stratégie

```
1  pour chaque stratégie s de j faire
2      moyenne = 0;
3      actions[j] = s;
4      pour i de 1 à X faire
5          pour chaque autre joueur k faire
6              actions[k] = tirage(P_j)
7          fin
8          gains = fonction_de_gain(actions);
9          moyenne = moyenne + gains[j] ;
10     fin
11     moyenne = moyenne / X;
12 fin
```

la rejouer, mais si cette stratégie était déjà souvent choisie, alors sa probabilité n'augmentera qu'un peu.

10.3.2 Sastry + Monte-Carlo

Un inconvénient notable de l'algorithme de Sastry est qu'il nécessite beaucoup d'itérations avant de converger : le facteur b doit être très petit pour que l'algorithme converge bien vers un équilibre de Nash, mais plus b est petit, plus la convergence est lente. Ces itérations prennent peu de temps chacune, mais ne sont pas parallélisables, car la mise à jour du vecteur de probabilités des joueurs dépend du résultat du calcul de la fonction de gain.

Nous proposons une modification de l'algorithme dans laquelle chaque itération est beaucoup plus coûteuse, en conjecturant que le nombre d'itérations nécessaire à la convergence est généralement beaucoup plus faible. Nous ne pourrons toujours pas effectuer plusieurs itérations en parallèle, mais nous pourrons paralléliser chaque itération. Ainsi, en utilisant suffisamment de processeurs, le coût supplémentaire de chaque itération peut être compensé. L'algorithme 8 décrit une nouvelle manière de choisir une stratégie, en effectuant pour chaque stratégie X simulations, et en choisissant la stratégie ayant produit le meilleur gain moyen.

Si nous évaluons la complexité des algorithmes en nombre d'appels à la fonction de gain, l'algorithme 5 nécessitait $O(n)$ appels. Avec notre algorithme de choix de stratégie modifié, chacun des n joueurs effectue X simulations, chaque simulation coûtant $O(n)$ appels à la fonction de gain. Une itération conduit donc à $O(Xn^2)$

appels à la fonction de gain. Cependant, chaque simulation peut être traitée séparément ; si nous disposons de p processeurs, le coût d'une itération est de $O(\frac{Xn^2}{p} + n)$. Une carte graphique nous donnant accès, à ce jour, à des centaines de processeurs, il est envisageable en pratique que, pour certains jeux, p soit du même ordre de grandeur que Xn ; dans ces cas là, nous pouvons maintenir une complexité en $O(n)$ pour une itération. Nous vérifierons dans la suite de ce chapitre que cet algorithme semble bien converger plus rapidement (en nombre d'itérations) que l'algorithme de Sastry.

Choix de la taille de l'échantillon

Dans la présentation de l'algorithme, nous avons supposé la taille de l'échantillon constante, et reçue en entrée. Dans ce cas, notre algorithme est très proche de l'algorithme SFP présenté ci-dessous, dans lequel la taille de l'échantillon peut varier d'une itération à l'autre, mais est identique pour chaque joueur. Dans notre algorithme, il est possible d'ajuster la taille de l'échantillon pour chaque joueur, par exemple augmentant la taille de l'échantillon tant que l'erreur type (l'écart-type de l'échantillon divisé par la racine carrée de la taille de l'échantillon) ne passe pas sous un certain seuil.

10.3.3 Algorithme « Sampled Fictitious Play »

L'algorithme que nous avons présenté, conçu comme une amélioration de l'algorithme de Satsry, s'avère très proche de l'algorithme *Sampled Fictitious Play* (jeu fictif échantilloné) introduit par Lambert et co-auteurs [72, 73]. *Sampled Fictitious Play* (SFP) est une modification de de l'algorithme classique de jeu fictif (*fictitious play*) afin de le rendre utilisable dans le cas de jeux où la fonction de gain n'est pas bien connue ou demande du temps de calcul.

Le jeu fictif est un algorithme introduit en 1951 par Brown [74] et Robinson [75]. Dans le jeu fictif, chaque joueur garde en mémoire la fréquence d'utilisation des stratégies P de ses adversaires, où $P_{i,j}$ est la fréquence d'utilisation de la stratégie j par le joueur i. À chaque itération du jeu fictif, un joueur choisit la stratégie qui est une meilleure réponse en supposant que ses adversaires choisiront leurs stratégies en respectant cette fréquence. Une meilleure réponse est définie comme faisant partie des stratégies qui produisent l'espérance de gain la plus forte.

En pratique, le jeu fictif est souvent trop coûteux pour être appliquable. En effet, dans le pire des cas, où aucun des $P_{i,j}$ n'est nul, calculer l'espérance de gain d'une stratégie peut nécessiter d'évaluer le gain de toutes les combinaisons de stratégies, ce qui est en $O(j^i)$. Le jeu fictif échantillonné remplace l'évaluation de l'espérance de gain des stratégies par le calcul de l'espérance de gain sur un échantillonage des stratégies possibles, tiré selon la matrice de probabilités P. L'algorithme 9 décrit cette procédure.

Algorithme 9 : Sampled Fictitious Play (SFP)

```
1 t = 0;
2 faire
3     E = tirage de k(t) choix de stratégies selon P_{i,j};
4     pour chaque joueur i faire
5         actions[i] = meilleure_réponse(E);
6     fin
7     mise_a_jour(P_{ij}, actions);
8     t = t + 1;
9 tant que t < temps_max ;
```

La taille de l'échantillon $k(t)$ dépend du temps de la simulation et augmente avec le temps. Dans le cas des jeux où tous les joueurs possèdent la même fonction de gain (ce qui n'est pas le cas de notre jeu), les auteurs démontrent que si $k(t) \geq \lceil C\sqrt{t} \rceil$, avec C une constante strictement positive, alors le processus converge vers un équilibre. Le calcul de meilleure réponse pour un échantillon E revient à calculer le gain moyen de chaque stratégie pour E, et prendre une des meilleurs. Enfin, la mise à jour de la matrice P se fait en incrémentant les indices correspondant à des stratégies choisies à cette itération, puis en normalisant la matrice pour que la somme de ses lignes soit égale à 1.

10.4 Comportement des algorithmes sur des jeux simples

Avant d'appliquer ces algorithmes à notre jeux, dans lequel la fonction de coût est complexe, nous allons comparer comparer leurs comportement sur des jeux simples dans lesquels les équilibres de Nash sont bien connus. Nous étudierons un jeu comportant un équilibre pur (le dilemme du prisonnier), un autre en comportant deux (la guerre des sexes), et enfin, un jeu ne comportant qu'un équilibre mixte (le jeu de « pierre-papier-ciseaux »).

10.4.1 Dilemme du prisonnier

Le *dilemme du prisonnier* fait partie des jeux types de la théorie des jeux. Dans ce jeu, deux complices d'un crime sont arrêtés par la police. Ils sont isolés, et à chacun d'eux on propose de dénoncer son camarade en échange d'une réduction de peine. Si Alice dénonce Bob et qu'il se tait, elle repart libre, et il écope de dix ans de prison. S'ils se dénoncent tous les deux, ils écopent chacun de quelques années de prison. Enfin, s'ils se taisent tous les deux, ils ne subiront qu'une peine de prison bien inférieure, pour d'autres délits. Une matrice de gains possible correspondant à ce jeu est représentée dans la table 10.1.

Selon un résultat contre-intuitif bien connu, il n'existe qu'un seul équilibre de Nash dans ce jeu, consistant pour les deux joueurs à se dénoncer mutuellement et

		Bob	
		Dénoncer	Se taire
Alice	Dénoncer	1, 1	10, 0
	Se taire	0, 10	5, 5

TABLE 10.1: Matrice de gains du dilemme du prisonnier. Dans une cellule, la première valeur désigne le gain d'Alice, et la deuxième valeur le gain de Bob.

gagner 1. Les joueurs pourraient gagner 5 s'ils se mettaient d'accord pour garder le silence, cependant, une fois cet accord conclu, chacun d'eux a intérêt à rompre sa promesse et augmenter ainsi son gain.

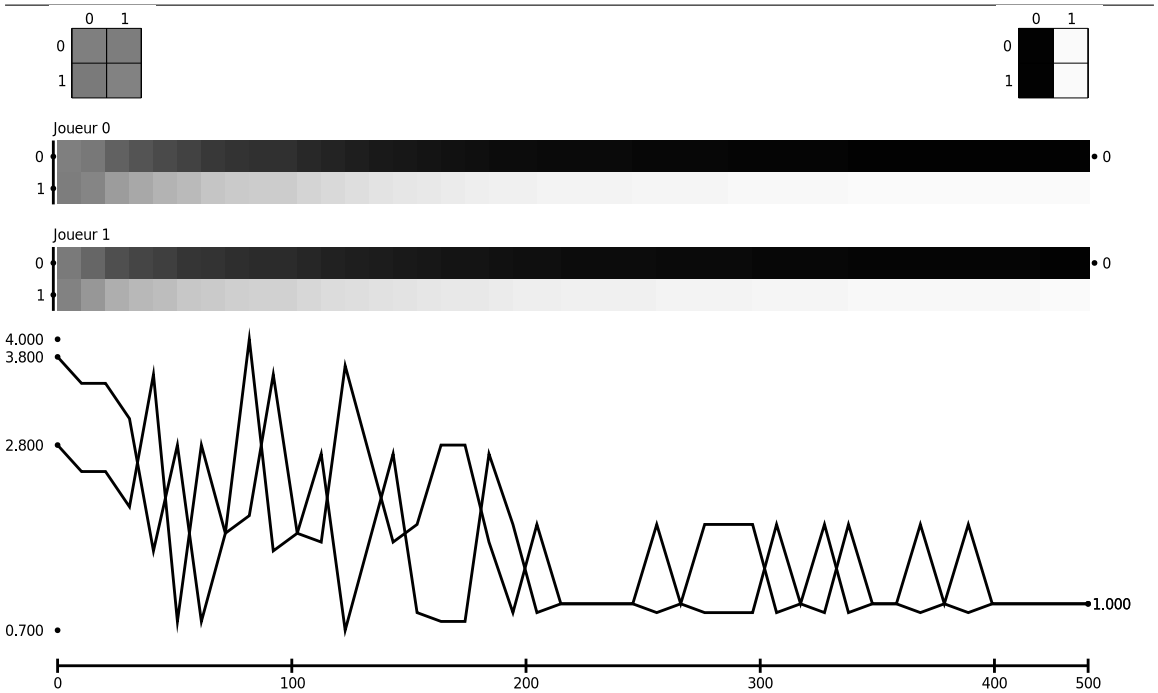


FIGURE 10.1: Dilemme du prisonnier, algorithme de Sastry, avec $b = 10^{-2}$. La figure représente, en haut à gauche, les vecteurs de probabilité des deux joueurs au début du jeu, et à droite, leur vecteur après que l'algorithme ait convergé ; au milieu, l'évolution des vecteurs de probabilité de chaque joueur au fil du temps ; en bas, l'évolution du gain des joueurs. On observe une convergence directe vers la stratégie 0 (dénoncer) pour les deux joueurs.

La figure 10.1 présente l'évolution de l'algorithme de Sastry sur ce jeu, pour une valeur $b = 10^{-2}$. La convergence se fait directement vers l'unique équilibre de Nash pour les trois algorithmes. Les figures correspondant aux algorithmes Monte-Carlo et SFP sont omises, leur convergence étant beaucoup plus rapide que pour l'algorithme de Sastry.

Cette figure est constituée de trois parties. La première, en haut, présente la matrice de probabilités de jeu de chaque stratégie au début du jeu (à gauche) et après la convergence (à droite). Chaque ligne de cette matrice correspond à un joueur, et chaque colonne correspond à un choix de stratégie. L'intérieur d'une cellule (i, j) est la probabilité que le joueur i joue la stratégie j ; plus la teinte de la cellule est noire, plus la probabilité est proche de 1. Ici, la stratégie 0 correspond à « dénoncer » et la stratégie 1 à « se taire ». La deuxième partie, au centre, représente l'évolution de ces mêmes vecteurs de probabilités, présentée pour chaque joueur. Cette évolution permet de visualiser les éventuelles oscillations de l'algorithme ; dans le cas du dilemme du prisonnier, la probabilité de jouer la stratégie 0 augmente de manière continue, il n'y a donc pas d'« hésitation » de l'algorithme. Enfin, la troisième partie de la figure représente l'évolution des gains des joueurs au fil du temps (c'est à dire des itérations).

10.4.2 Guerre des sexes

La guerre des sexes est un autre jeu type, à deux joueurs, décrivant une situation où Alice et Bob doivent décider de leur sortie du soir. Bob préfère aller à l'opéra, et Alice préfère assister à un match de football, mais ils préfèrent être ensemble et souhaitent donc se décider sur l'une des deux sorties. La table 10.2 décrit une fonction de gain associée à ce jeu.

		Bob	
		Opéra	Match
Alice	Opéra	2, 3	0, 0
	Match	1, 1	3, 2

TABLE 10.2: Matrice de gains de la guerre des sexes

Ce jeu possède deux équilibres de Nash purs, correspondant chacun à un compromis de la part d'Alice ou Bob. Rien dans la théorie des jeux ne permet de déterminer quel équilibre les joueurs vont choisir. Les figures 10.2 à 10.4 présentent les évolutions des algorithmes de Sastry, Monte-Carlo et SFP sur ce jeu, respectivement. Les temps de convergence varient d'une exécution à l'autre, mais dans tous les cas, Monte-Carlo converge nettement plus rapidement que Sastry, et SFP nettement plus vite que Monte-Carlo. Les trois algorithmes convergent indifféremment vers l'un ou l'autre des deux équilibres purs, du fait de la symmétrie du jeu. Contrairement au dilemme du prisonnier, nous pouvons observer des fluctuations dans l'évolution des vecteurs de probabilité. Ainsi, le joueur 0 commence par s'orienter vers la stratégie 1, qui est bonne si le joueur 1 choisit la même chose. Cependant, le joueur 1 se décide au fil des itérations pour la stratégie 0, et le joueur 0 s'aligne ensuite sur se choix.

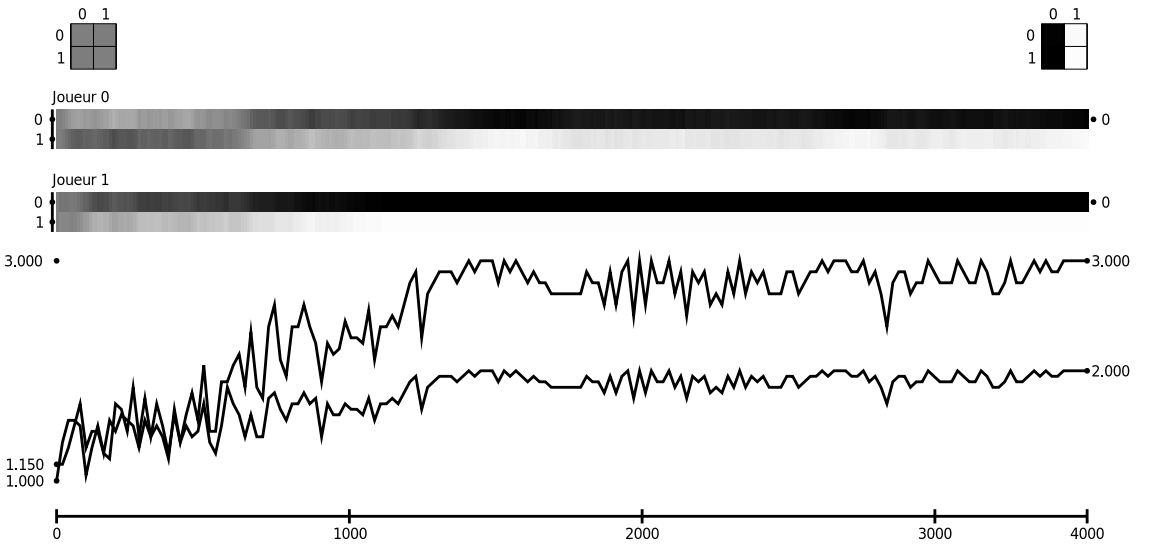


FIGURE 10.2: Guerre des sexes, algorithme de Sastry, avec $b = 10^{-2}$.

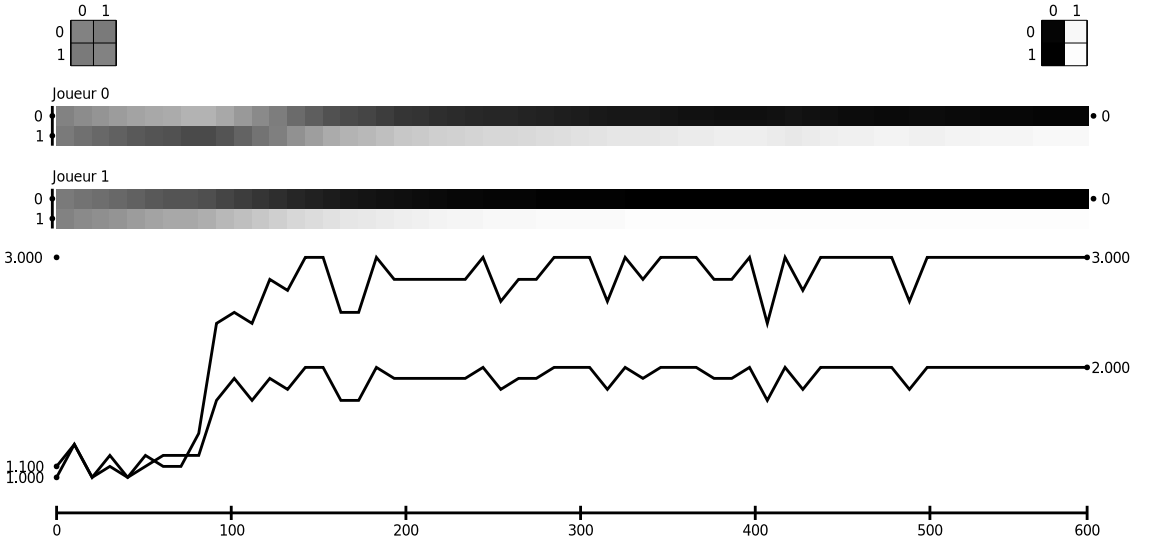


FIGURE 10.3: Guerre des sexes, algorithme Monte-Carlo, avec $b = 10^{-2}$.

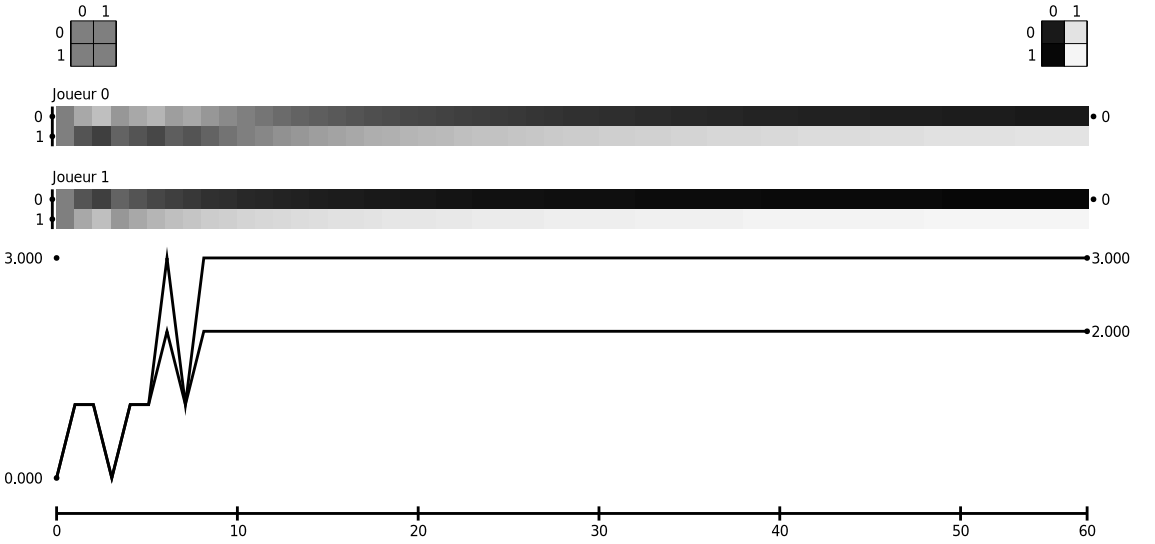


FIGURE 10.4: Guerre des sexes, algorithme SFP.

10.4.3 Pierre-papier-ciseaux

Le jeu de « pierre-papier-ciseaux » (ou *shifoumi*) est un jeu d'enfants où les deux joueurs doivent choisir – simultanément – une stratégie parmi « pierre », « papier » ou « ciseaux ». Les gains sont circulaires : la pierre gagne contre les ciseaux, mais perd contre le papier ; les ciseaux gagnent contre le papier mais perdent contre la pierre, etc. (voir table 10.3).

		Bob		
		Pierre	Papier	Ciseaux
Alice	Pierre	1, 1	0, 2	2, 0
	Papier	2, 0	1, 1	0, 2
	Ciseaux	0, 2	2, 0	1, 1

TABLE 10.3: Matrice de gains du pierre-papier-ciseaux

Il n'existe pas d'équilibre de Nash pur dans ce jeu, puisque chaque stratégie gagne contre une stratégie et perd contre une autre. Si Alice se décidait pour « pierre », Bob pourrait aussitôt choisir « papier » et gagner à chaque fois. La meilleure stratégie pour chaque joueur est de probabiliser son choix de manière uniforme, et d'avoir donc une probabilité $1/3$ de jouer chaque stratégie.

Si l'on initialise les vecteurs de probabilité uniformément, nous sommes dès le départ en situation d'équilibre et, en effet, les trois algorithmes restent dans cet équilibre. Il est plus intéressant de déterminer si nous pouvons retrouver l'équilibre à partir d'une situation de départ aléatoire. Nous avons donc initialisé les vecteurs de probabilité aléatoirement, et appliqué les algorithmes. Les figures 10.5 à 10.7 présentent les résultats.

Nous observons est qu'il est très difficile de faire converger l'algorithme de Sastry si la situation de départ est éloignée de l'équilibre. Le cas présenté à la figure 10.5 est caractéristique : Alice démarre avec une forte propension à jouer « pierre », par conséquent Bob renonce progressivement à jouer « ciseaux », qui est souvent perdant. Une fois qu'il a renoncé, la stratégie « papier » devient bien plus intéressante pour Alice, puisqu'elle ne risque plus qu'un match nul. Elle n'a par contre plus intérêt à jouer « pierre ». La probabilité que Bob tente à nouveau « ciseaux » est tellement faible que la situation ne s'inverse pas. Le problème se produit également avec des valeurs de b plus petites que 10^{-7}, et dans ce cas le temps de convergence augmente exponentiellement. Dans le cas des algorithmes Monte-Carlo et SFP, nous convergeons bien vers le bon équilibre de Nash mixte, avec des temps de convergence de l'ordre du millier d'itérations.

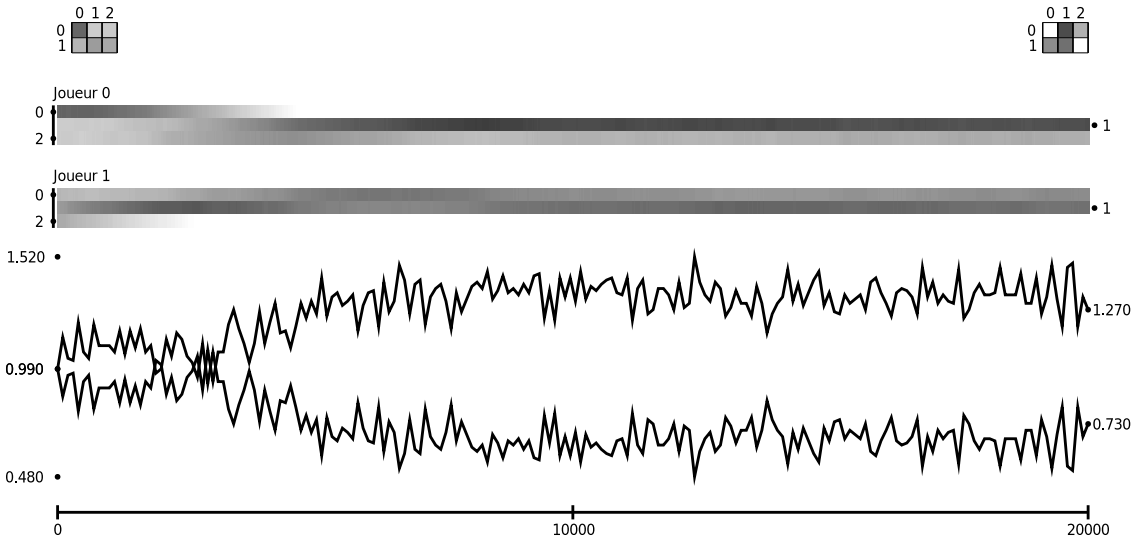


FIGURE 10.5: Pierre-papier-ciseaux, algorithme de Sastry, avec $b = 10^{-3}$.

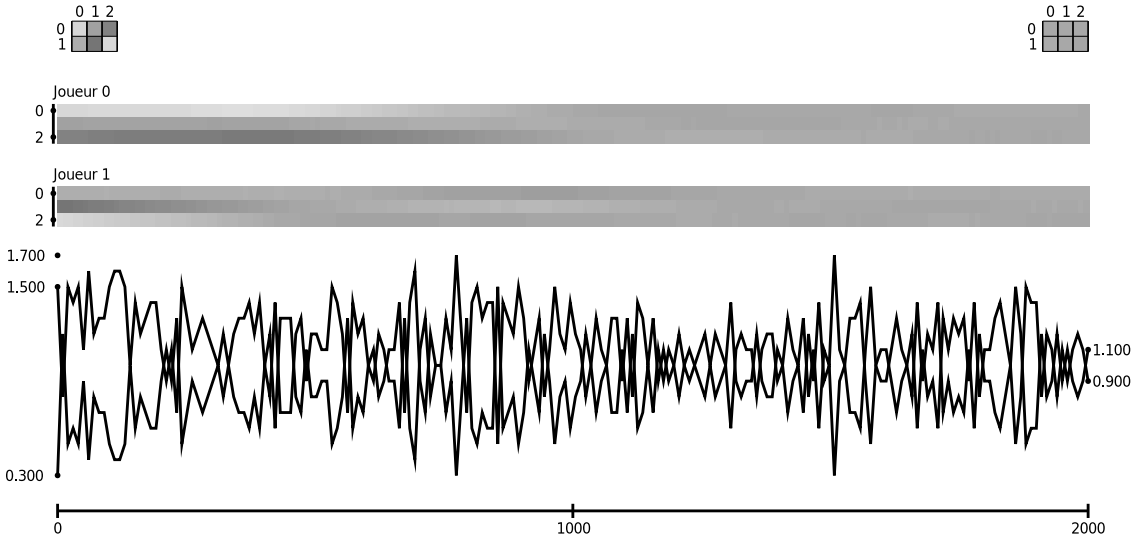


FIGURE 10.6: Pierre-papier-ciseaux, algorithme Monte-Carlo, avec $b = 10^{-3}$.

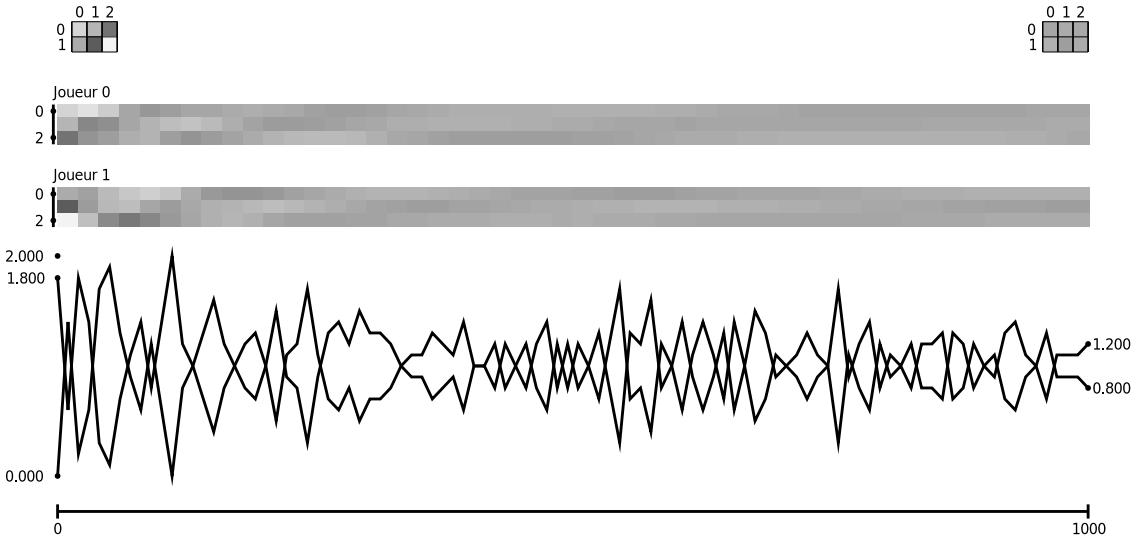


FIGURE 10.7: Pierre-papier-ciseaux, algorithme SFP.

10.5 Application au jeu du repliement d'ARN

Reprenons le jeu tel que défini à la section 10.2, page 90. Dans ce jeu, les joueurs sont les sommets du graphe squelette, et une action consiste à se choisir un partenaire et établir une interaction longue-distance avec lui (ou ne choisir aucun partenaire). Cependant, certains choix sont biologiquement peu probables ou peu significatifs du point de vue de notre modélisation à gros grain. Ainsi, nous ne considérerons pas comme possibles les interactions entre deux hélices, ou encore les interactions entre deux sommets proches dans la structure secondaire. Ces restrictions seront modélisées par l'initialisation du vecteur de probabilité des joueurs : les interactions jugées possibles se verront attribuer une probabilité uniforme, et les autres une probabilité nulle.

Nous avons appliqué nos différents algorithmes sur le jeu ainsi défini. En règle générale, obtenir que l'algorithme de Sastry converge vers un résultat sur ce jeu est difficile : soit la valeur du paramètre b est élevée, et l'on converge vers des solutions non pertinentes, soit la valeur de b est faible, et le temps de convergence rend cet algorithme inapplicable. Pour cette raison, nous ne présenterons ici que les résultats obtenus par l'algorithme Sastry+Monte-Carlo. La valeur de b choisie dépendra de la taille du graphe en entrée, car plus le graphe est gros et plus le gain potentiel d'un sommet est évelé. Ce gain, multiplié par b, doit rester petit par rapport à 1. Pour les graphes de moins de vingt sommets, nous avons pris $b = 0{,}01$, et pour ceux de plus de vingt sommets, $b = 0{,}005$.

La figure 10.8 présente l'évolution des vecteurs de probabilité et des gains des joueurs au cours de l'application de l'algorithme à la molécule 1MFQ lorsque la 3-jonction est classifiée en famille C. Au niveau des gains, nous pouvons constater qu'au dela d'une trentaine d'itérations, le gain moyen de chaque sommet n'évolue plus beaucoup. Cela ne signifie pas pour autant que l'algorithme ait convergé : nous observons des changements de stratégie survenant plus tard, par exemple pour les joueurs 5, 7 ou 13.

Notons également que le gain moyen de deux des joueurs reste bas tout au long de la simulation. Il s'agit du sommet correspondant à la 3-jonction, qui n'a pas de choix de stratégie intéressant dans cette molécule, et donc les déformations de ses voisins augmentent le coût, ainsi que d'un sommet dont les deux voisins ont fait des choix qui le pénalisent, mais dont les actions possibles ne permettent pas de modifier la situation.

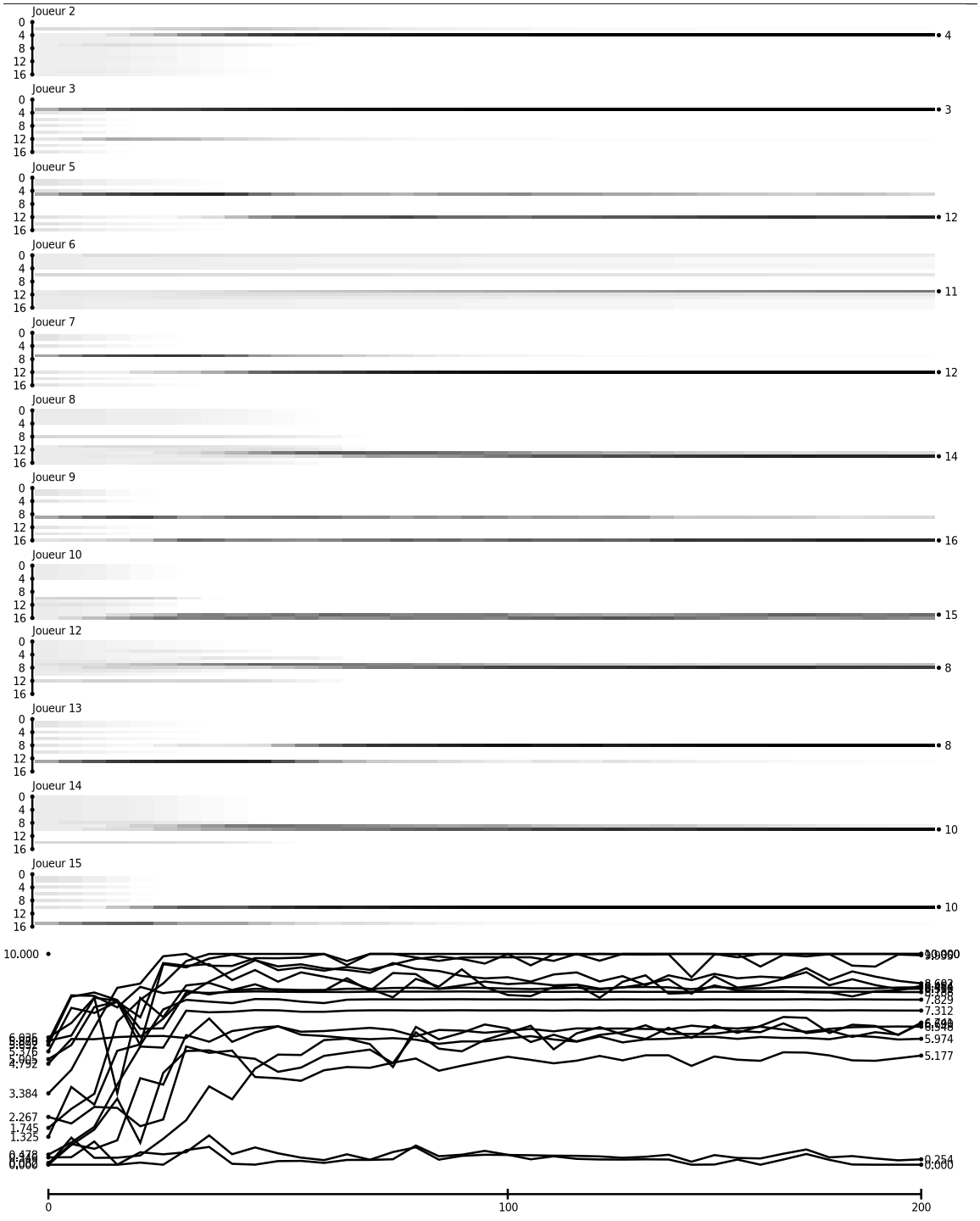


FIGURE 10.8: Évolution de l'algorithme Sastry+MonteCarlo sur la molécule 1MFQ avec la 3-jonction classifiée en C. Certains sommets dont la convergence est directe sont ommis de la figure. La convergence d'un sommet x vers un choix de stratégie x correspond au choix « ne pas faire d'interaction ».

Afin de comparer les résultats obtenus avec les structures cristallographiées, nous avons construit les structures candidates de la manière suivante : après convergence ou expiration du délai maximum, pour chaque sommet, nous avons choisi dans son vecteur de probabilités le choix de stratégie majoritaire et ajouté l'interaction correspondante. S'il n'y avait pas de choix majoritaire, nous n'avons pas ajouté d'interaction. Le graphe ainsi obtenu est ensuite comparé aux structures de référence, et les résultats sont présentés dans la table 10.4.

Dans certains cas, l'algorithme ne converge pas vers des choix de stratégies pures, c'est pourquoi nous prenons le choix majoritaire. Il serait également possible, au lieu du graphe majoritaire, de produire l'ensemble des graphes possibles, soit par échantillonage soit par énumération exhaustive, et d'ordonner les candidats par probabilité d'occurence. Une observation des vecteurs de probabilités « mixtes » nous montre qu'en cas d'hésitation sur un choix de partenaire d'interaction, les joueurs hésitent entre des sommets proches les uns des autres. Les différents graphes possibles sont donc assez proches dans les cas étudiés, mais générer une liste de candidats possibles est une perpective à explorer à l'avenir. Dans le travail présenté ici, nous nous sommes contenté de produire le candidat le plus probable.

Identifiant	Sommets	Gain total		RMSD (Å)		
		Initial	Jeu	Initial	Jeu	Réel
1E8O C	7	51,6	56,5	3,53	3,98	3,80
1E8O A	7	52,2	56,0	5,89	5,42	4,66
1E8O B	7	51,1	59,5	7,86	7,94	7,08
1MFQ C	19	79,4	115,7	6,10	5,65	5,97
1MFQ A	19	82,2	126,15	14,99	7,15	7,33
1MFQ B	19	77,8	136,6	23,78	22,32	13,20
1NBS C	21	122,0	145,9	10,29	9,71	8,39
1NBS A	21	124,8	108,4	17,28	16,63	8,56
1NBS B	21	119,8	147,1	26,53	28,13	17,28
2A64 CAC	37	204,1	250,9	21,40	23,58	19,82
2A64 AAA	37	206,1	261,4	21,36	24,14	19,67
2A64 CCC	37	189,4	241,9	25,61	26,75	24,74

TABLE 10.4: Résultats de l'optimisation par la théorie des jeux. Pour chaque molécule, nous considérons différentes classifications pour les jonctions qui la composent ; la ligne en gras indique la classification correcte. La colonne « Réel » indique la RMSD obtenue si nous munissons le graphe des interactions réelles.

La présentation de la table 10.4 est similaire à la table analogue du chapitre sur l'optimisation globale, la table 9.1, page 84. Là encore, lorsque la classification

des jonctions est correcte, le plongement initial est bon et l'étape d'optimisation améliore peu la RMSD, voire la dégrade légèrement. Lorsqu'une 3-jonction est mal classifiée en famille A au lieu de C, l'optimisation arrive parfois à corriger l'erreur (1MFQ-A), mais quand une 3-jonction est mal classifiée en famille B, la correction ne se fait pas. Dans ce cas, une autre configuration est trouvée, également crédible du point de vue de la classification erronée, mais incorrecte en réalité.

Identifiant	Sommets	Gain total		
		Initial	Global	Jeu
1E8O C	7	51,6	68,9	56,5
1E8O A	7	52,2	69,6	56,0
1E8O B	7	51,1	70,0	59,5
1MFQ C	19	79,4	150,9	115,7
1MFQ A	19	82,2	154,8	126,1
1MFQ B	19	77,8	144,8	136,6
1NBS C	21	122,0	207,1	145,9
1NBS A	21	124,8	210,0	108,4
1NBS B	21	119,8	206,0	147,1
2A64 CAC	37	204,1	311,8	250,9
2A64 AAA	37	206,1	323,4	261,4
2A64 CCC	37	189,4	319,9	241,9

TABLE 10.5: Comparaison des gains entre l'optimisation globale et l'optimisation par un jeu.

La table 10.5 présente côte-à-côte les résultats en termes de gains de l'optimisation globale (par un algorithme évolutionniste) et les résultats obtenus dans ce chapitre. Nous pouvons constater que la somme des gains est plus faible dans le cas d'une optimisation par un jeu que dans le cas d'une optimisation globale, ce qui est normal : dans ce chapitre, nous n'essayons pas d'optimiser le gain global. Comme nous l'avons vu précédemment dans ce chapitre, un équilibre dans un jeu ne représente pas nécessairement un gain global maximal, ni même un gain individuel élevé pour tous les sommets.

La table 10.6 compare quant à elle les résultats de RMSD. Malgré un gain global plus faible, la RMSD obtenue est en général proche, parfois moins bonne (1NBS-C) et parfois meilleure (1MFQ-C) que la RMSD obtenue par optimisation globale. Seule la structure pour 1NBS-A est nettement moins bonne que celle obtenue par algorithme évolutionniste (dans ce cas, l'erreur de classification n'est pas corrigée).

Il est intéressant de noter que, pour la molécule 1MFQ-C, la somme des gains obtenue par jeu est nettement plus faible que la somme des gains obtenue par

Identifiant	Sommets	RMSD (Å)		
		Initial	Global	Jeu
1E8O C	7	3,53	3,48	3,98
1E8O A	7	5,89	6,00	5,42
1E8O B	7	7,86	8,17	7,94
1MFQ C	19	6,10	6,88	5,65
1MFQ A	19	14,99	7,57	7,15
1MFQ B	19	23,78	23,69	22,32
1NBS C	21	10,29	9,38	9,71
1NBS A	21	17,28	8,86	16,63
1NBS B	21	26,53	28,30	28,16
2A64 CAC	37	21,40	23,22	23,58
2A64 AAA	37	21,36	23,40	24,14
2A64 CCC	37	25,61	26,88	26,75

TABLE 10.6: Comparaison des RMSD entre l'optimisation globale et l'optimisation par un jeu.

algorithme évolutionniste, mais que la RMSD est meilleure. Cela souligne que les deux méthodes ne sont pas deux variantes pour rechercher le même résultat, mais atteignent bien parfois des solutions distinctes. La figure 10.9 représente les deux prédictions : celle produite par l'algorithme évolutionniste, en haut, et celle résultant de notre recherche d'un équilibre dans un jeu, au centre, ainsi que la structure cristallographiée, en bas. Dans cet exemple, la prédiction produite par le jeu est plus proche de la structure cristallographiée, et contient moins d'interactions longue-distance que la prédiction résultant d'une optimisation globale.

10.6 Conclusion

Dans ce chapitre, nous avons présenté une nouvelle approche utilisant la théorie des jeux pour proposer une optimisation locale et égoïste par les sommets, chaque sommet essayant de maximiser son propre gain. Bien que la recherche d'équilibres de Nash soit un domaine moins bien maîtrisé que le domaine de l'optimisation globale, la qualité des prédictions est similaire à celle obtenue précédemment. Il ne s'agit cependant pas des mêmes solutions, trouvées de deux manières, mais bien de solutions correspondant à des objectifs différents. Dans un cas, l'objectif est commun à tous les sommets (maximiser la fonction de gain globale), ce qui peut conduire à pénaliser certains sommets pour le bien commun. Dans l'autre cas, l'objectif est local, et aucun des sommets n'a de raison d'accepter un gain plus faible que ce qu'il pourrait obtenir. La figure 10.9 montre que ces deux approches

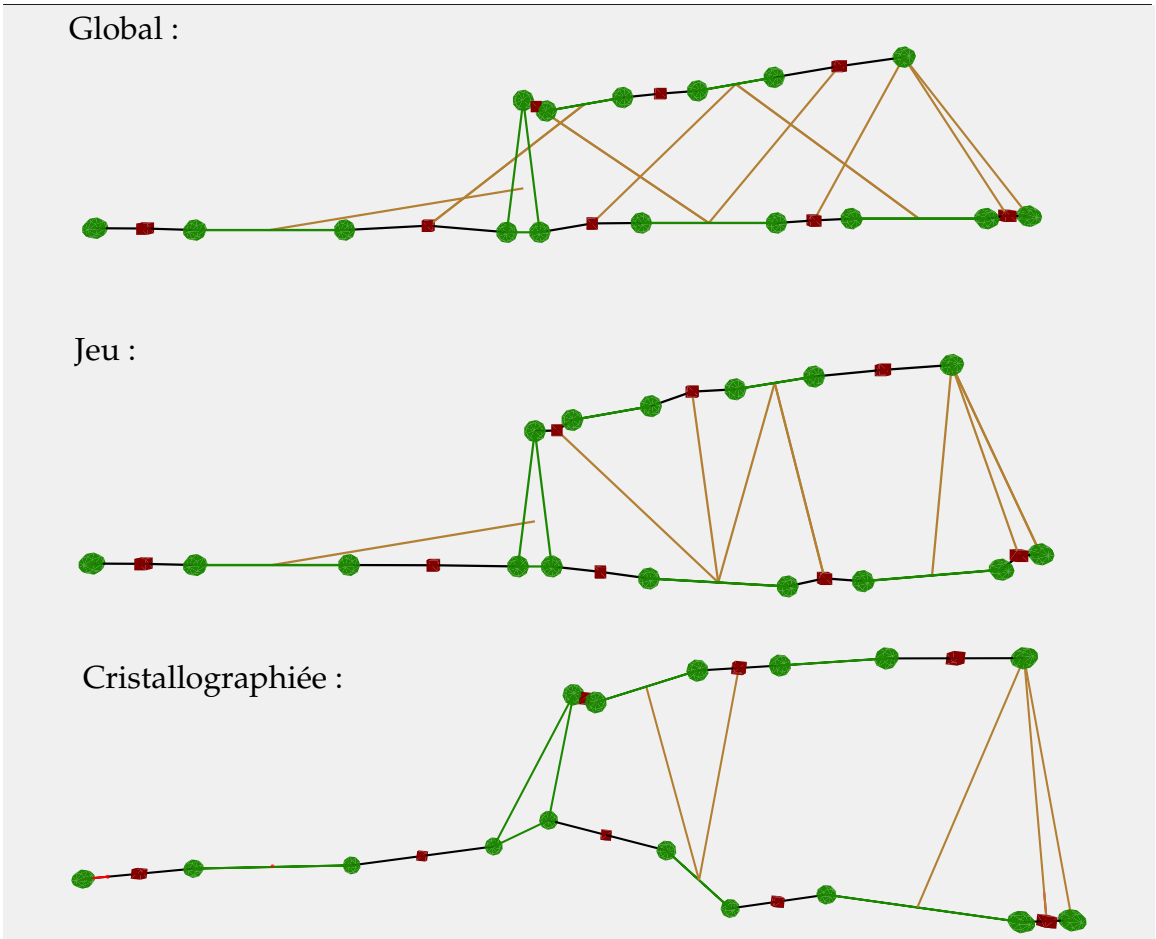


FIGURE 10.9: Comparaison entre global (en haut), jeu (au centre) et structure cristallographiée (en bas) pour 1MFQ avec sa 3-jonction classifiée en famille C. Les interactions longue-distance sont représentées en couleur ocre.

peuvent conduire à des résultats très différents.

Une optimisation locale et égoïste semble plus justifiée intuitivement que l'optimisation d'une fonction globale : rien ne pousse un sommet à accepter une situation défavorable pour lui si cela améliore la stabilité globale de la molécule. L'optimisation globale avec exploration des résultats sous-optimaux est souvent choisie car il s'agit de techniques bien connues. Il encourageant de constater que l'on parvient à de bons résultats également par une approche reposant sur la théorie des jeux.

11 Comparaison avec les approches existantes

Résumé Dans ce chapitre, nous comparons nos prédictions de structure avec les prédiction d'approches concurrentes : iFoldRNA, FARNA et MC-Sym.

11.1 Introduction

Comparer notre approche avec les approches pré-existantes est une tâche complexe, car le problème que nous essayons de résoudre n'est pas le même. Là où les approches concurrentes cherchent à produire une structure 3D à l'échelle de l'atome, nous cherchons à produire une structure à gros grain. Nous pouvons calculer une RMSD entre notre prédiction et la structure à gros grain induite par la forme 3D cristallographiée ; nous pouvons également calculer une RMSD entre les prédictions des autres approches et la structure cristallographiée ; cependant les deux RMSD ne sont pas comparables, car les graphes produits sont différents. L'idéal pour nous serait de calculer la RMSD entre notre plongement du graphe squelette d'une molécule et le plongement du même graphe découlant d'une prédiction concurrente. Cependant, le graphe squelette induit par la structure prédite par d'autres logiciels est parfois différent du graphe squelette induit par la structure réelle. Dans ces cas, n'ayant pas d'isomorphisme entre les graphes, il n'est pas possible de calculer leur RMSD. Lorsqu'il ne nous est pas possible de présenter une valeur de RMSD pertinente entre nos prédictions et les prédictions concurrentes, nous présenterons les RMSD comparant les prédictions avec le réel, et les accompagnerons de représentations graphiques afin d'aider à la comparaison.

D'autres mesures que la RMSD ont été conçues pour comparer les structures, notamment le cœfficient de corrélation de Matthews [76] ou l'indice de déformation [77], cependant ces mesures prennent en compte les interactions prédites, et non uniquement la coordonnées des sommets. Notre approche n'essayant pas de prédire précisément les interactions se produisant, mais uniquement les positions relatives des sommets, nous ne pouvons utiliser ces mesures. Une autre mesure, le profil de déformation [77], n'utilise pas la qualité des interactions prédites pour mesurer la qualité de la structure, mais elle nécessite par contre de pouvoir aligner correctement dans l'espace des éléments individuels de la structure. Cet alignement

est facile à effectuer sur des nucléotides (qu'on peut se représenter comme un objet plan avec un haut et un bas), mais est n'est pas bien défini sur notre graphe squelette : les 2-jonctions et les hélices, représentées par des segments, peuvent être alignées d'une infinité de manières (par rotation autour de l'axe), et les 3-jonctions ou plus peuvent être alignées de deux manières selon le choix d'orientation haut / bas. Par conséquent, bien que cette mesure soit imparfaite, nous nous contenterons de mesurer la RMSD entre nos prédictions et les structures cristallographiées.

Enfin, dernier écueil, la plupart des logiciels de prédiction *ab initio* de structure 3D fonctionnent mieux sur de petites molécules, pour lesquelles ils ont été conçus. L'article introduisant le logiciel iFoldRNA [49] s'intéresse ainsi aux molécules de cinquante nucléotides ou moins. Pour de plus grosses molécules, le temps d'exécution des logiciels devient rapidement prohibitif ou, si l'on préfère limiter le temps d'exécution, la qualité des résultats se dégrade en proportion. Notre approche, quant à elle, ne présente d'intérêt que sur les molécules comportant une ou plusieurs jonctions, donc au moins quelques centaines de nucléotides. Afin d'effectuer une comparaison, nous avons utilisé les logiciels concurrents pour prédire la structure de grosses molécules ; il nous faut cependant garder à l'esprit que cette application n'est pas représentative des performances de ces logiciels sur toutes les molécules.

11.2 iFoldRNA

Au moment de la rédaction de cette thèse, le logiciel iFoldRNA [49] n'est pas téléchargeable pour une utilisation locale, mais utilisable au moyen d'un serveur Web [1] mis à disposition par les auteurs. Les temps de simulation sont donnés à titre indicatif, mais ne sont donc pas comparables avec ceux des autres logiciels, qui ont été exécutés en local sur un ordinateur personnel. Nous les indiquons afin d'observer l'évolution du temps d'exécution en fonction de la taille des molécules. Par ailleurs, le nombre d'itérations de la simulation n'était pas modifiable par l'intermédiaire de l'interface web.

Nous avons pris la séquence de chaque molécule et demandé à iFoldRNA de prédire trois structures candidates à partir de cette séquence. La table 11.1 indique les résultats. La RMSD est calculée entre les représentations atomiques des structures prédites et de la structure cristallographiée. Cette RMSD a été calculée par le logiciel PyMol. La première valeur indique la RMSD entre tous les atomes, et le nombre entre parenthèses une RMSD raffinée en retirant successivement toutes les paires d'atomes alignés dont la distance était plus grande que deux fois l'écart-type (en général, moins de 10% des atomes). Cette seconde valeur permet d'identifier les cas où deux molécules peuvent dans l'ensemble bien s'aligner mais où certaines parties très différentes faussent la RMSD.

1. `http://troll.med.unc.edu/ifoldrna/`

iFoldRNA					
Molécule	Nucléotides	Candidat 1	Candidat 2	Candidat 3	Temps
1E8O	49	20,3 (18,8)	19,1	18,7	1 h 26
1MFQ	127	28,2 (25,3)	40,4 (39,8)	41,8 (41,3)	3 h 27
1NBS	155	30,1 (19,3)	29,1 (19,0)		4 h 06
2A64	417	pas de résultat			

TABLE 11.1: RMSD (en Å) entre la 3D réelle et les prédictions du logiciel iFoldRNA. L'alignement a été calculé par le logiciel PyMol. Les nombres entre parenthèse indiquent une RMSD raffinée, par exclusions successives des atomes les moins bien alignés.

De manière prévisible, la RMSD et le temps de simulation augmentent avec la taille des séquences soumises. Dans le cas de la molécule 2A64, la plus grosse molécule soumise, le serveur ne produit pas de structure candidate. Une valeur de RMSD ne donnant qu'une vision abstraite de la précision des prédictions, nous nous intéresserons surtout à la représentation graphique des molécules.

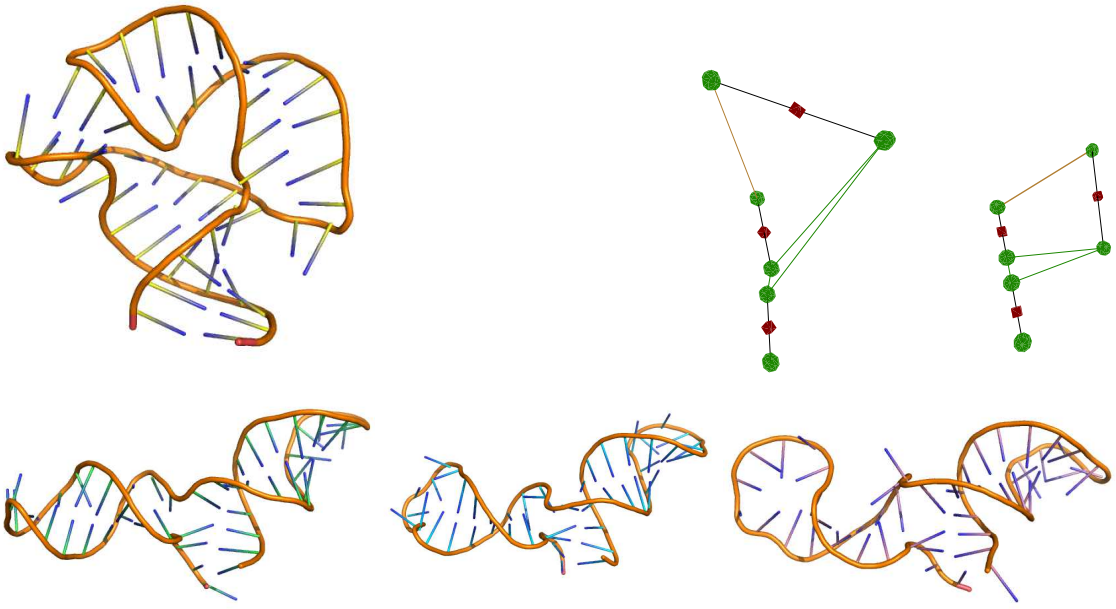


FIGURE 11.1: Comparaison entre la structure cristallographiée de la molécule 1E8O, en haut à gauche, son graphe squelette et notre prédiction, en haut à droite, et les trois candidats de iFoldRNA, en bas.

La figure 11.1 présente la forme cristallographiée et les trois structures candidates pour la molécule 1E8O, qui ne comporte qu'une 3-jonction. Nous pouvons

constater qu'iFoldRNA parvient à retrouver approximativement les hélices et la 3-jonction sans avoir reçu la structure secondaire en entrée ; cependant, l'angle entre les axes des hélices est faux, et les interactions entre les boucles terminales ne sont par conséquent pas prédites. En haut à droite sont représentés le graphe squelette réel et le graphe squelette que nous prédisons. Cette fois, l'angle entre les axes est plus proche de la réalité, mais la représentation de la 3-jonction par un triangle isocèle induit une erreur qui n'est pas rattrapable par optimisation.

Dans l'ensemble, pour cette molécule, les prédictions d'iFoldRNA sont assez éloignées de la structure cristallographiée. Notre graphe squelette, quant à lui, est proche du graphe squelette correspondant à la structure cristallographiée, et conduit à une meilleure prédiction de la forme globale de la molécule, à gros grain.

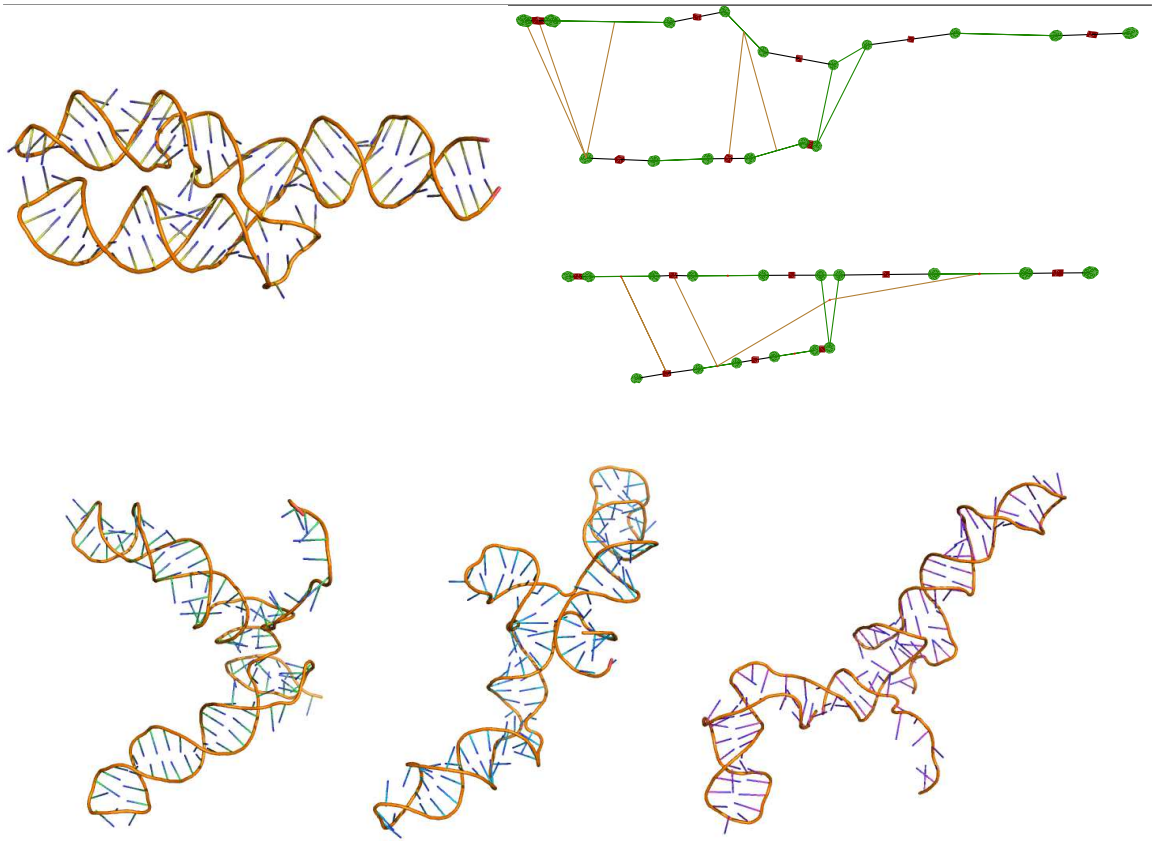


FIGURE 11.2: Comparaison entre la structure cristallographiée de la molécule 1MFQ, en haut à gauche, son graphe squelette et notre prédiction, en haut à droite, et les trois candidats de iFoldRNA, en bas.

La figure 11.2 présente les résultats pour la molécule 1MFQ, qui comporte une 3-jonction mais également plusieurs 2-jonctions. Cette fois, les trois structures candidates produites par iFoldRNA ont des formes plus diverses, mais aucune d'entre elles ne reproduit la forme caractéristique de la molécule, avec ses deux

axes hélicoïdaux parallèles. Notre graphe squelette, quant à lui, est très proche du graphe réel, hormis quelques angles entre les hélices et les 2-jonctions sur l'une des branches. Les arêtes correspondant à des interactions longue-distance (en orange) résultent ici d'une optimisation par algorithme génétique, mais n'influent pas sur la forme de la molécule dans cet exemple.

Ne pouvant fournir de structure secondaire en entrée à iFoldRNA, celui-ci est libre de former les structures secondaires qu'il désire. Par conséquent, il n'est pas aisé de comparer les graphes squelettes des structures candidates aux notres ; calculer une RMSD ne serait pertinent que si nous pouvions trouver un isomorphisme (à quelques sommets près) entre eux, ce qui n'est en général pas le cas. C'est pourquoi nous avons donné des représentations de la structure à l'échelle du nucléotide, et pas leurs graphes squelettes. Ces représentations graphiques permettent de confirmer visuellement que nos prédictions à gros grain sont plus proches de la structure cristallographiée que celle d'iFoldRNA.

Nous verrons cependant dans les sections suivante que les RMSD calculées sur des graphes squelettes sont proches de celles calculées sur les structures atomiques, bien que plus faibles. La table 11.1 fournit donc une estimation pessimiste mais significative des RMSD que nous obtiendrions si nous pouvions les calculer sur des graphes squelettes.

11.3 FARNA

FARNA [52] fait partie de la suite logicielle Rosetta [2] sous le nom « rna_denovo. » FARNA prend en entrée une séquence, mais il est également possible de l'obliger à produire certaines interactions, quitte à causer des ruptures dans le *backbone* de la molécule. Nous avons donc forcé la présence des interactions correspondant à la structure secondaire des molécules.

FARNA / rna_denovo (molécules)

Molécule	Nucléotides	Candidat 1	Candidat 2	Candidat 3	Temps
1E8O	49	16,7 (16,6)	17,9 (17,7)	20,0	50 s
1MFQ	127	30,5	31,9 (27,1)	27,9 (25,5)	3 min 10 s
1NBS	155	24,8 (24,0)	28,2 (25,1)	21,0	3 min 17 s
2A64	417	85,7	34,4 (30,5)	39,9 (37,8)	17 min 21 s

TABLE 11.2: RMSD (en Å) entre la 3D réelle et les prédictions du logiciel FARNA. L'alignement a été calculé par le logiciel PyMol.

La table 11.2 présente la RMSD entre les structures candidates produites par FARNA et la structure cristallographiée. Les résultats sont d'une qualité similaire

2. https://www.rosettacommons.org/software/academic/3.3/

à ceux d'iFoldRNA, malgré la connaissance de la structure secondaire ici. Cependant, les temps de calcul sont nettement plus bas, y compris pour les plus grosses molécules.

Puisque nous avons contraint la structure secondaire, nous devrions obtenir dans les structures candidates des structures secondaires proches de la réalité ; FARNA ne devrait être capable que de rajouter de nouvelles interactions, mais pas d'en enlever. Néanmoins, obliger FARNA à respecter des interactions le conduit éventuellement à casser la chaîne de *backbone* en plusieurs morceaux, ce qui peut conduire à une structure secondaire différente.

Nous avons extrait le graphe squelette correspondant aux structures prédites, et nous avons calculé la RMSD entre ce graphe squelette et le graphe squelette de la structure cristallographiée. La table 11.3 présente ces résultats, accompagnés des RMSD entre nos prédictions et la structure cristallographiée. Les chiffres en noir et gras indiquent que les graphes squelettes étaient identiques, et que la RMSD a pu être calculée entre tous les sommets ; les chiffres en gris indiquent que les graphes différaient légèrement et que la RMSD a été calculée sur les sommets qui restaient inchangés.

Nous pouvons constater que les RMSD entre squelettes sont proches des RMSD entre molécules ; passer de l'échelle atomique à l'échelle du squelette n'améliore pas sensiblement la RMSD, et ne fait que gommer les imperfections locales. Cela nous conforte dans la conviction que connaitre avec précision la position des sommets du graphe squelette devrait permettre également un bon placement à des niveaux plus détaillés, que ce soit à l'échelle du nucléotide ou à l'échelle de l'atome.

Une représentation visuelle étant plus parlante qu'une valeur moyenne, nous avons regroupé et aligné dans la figure 11.3 les squelettes correspondant, de gauche à droite, à la structure cristallographiée, à notre prédiction, et aux structures candidates produites par FARNA, pour la molécule 1NBS, qui comprend une 4-jonction et une 3-jonction. Bien que notre prédiction comporte des imperfections, en par-

FARNA / rna_denovo (squelettes)

		Nous		FARNA		
Molécule	Sommets	Global	Jeu	1	2	3
1E8O	7	3,48	3,98	11,31	11,22	16,71
1MFQ	19	6,88	5,65	27,99	31,66	24,67
1NBS	21	9,38	9,71	21,07	26,88	20,44
2A64	37	23,22	23,58	84,65	31,95	23,63

TABLE 11.3: RMSD (en Å) entre les squelettes correspondant à la structure cristallographiée et aux prédictions du logiciel FARNA. Les chiffres en noir correspondent à une RMSD entre tous les sommets, et les chiffres en gris à une RMSD sur une sélection de sommets.

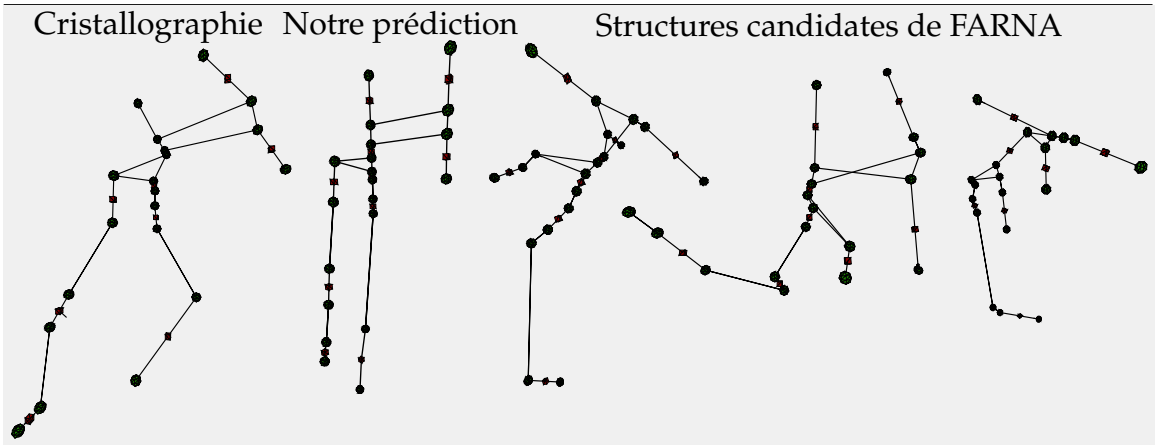


FIGURE 11.3: Comparaison des squelettes pour la molécule 1NBS et les prédictions de FARNA. De gauche à droite, les squelettes de la structure cristallographiée, de notre prédiction, et des trois structures candidates produites par FARNA.

ticulier le fait que ses branches soient « trop droites », elle est plus proche de la structure cristallographiée que les structures produites par FARNA, et donne une idée plus précise des positions relatives des différentes jonctions.

11.4 MC-Sym

MC-Sym [53] est disponible soit par l'intérmédiaire d'un serveur Web [3] soit sous la forme d'un logiciel téléchargeable. Afin de pouvoir comparer le temps d'exécution avec celui du logiciel FARNA, nous avons opté pour la version téléchargeable et une utilisation locale.

MC-Sym nécessite qu'on lui fournisse un script d'exécution, qui décrit les composants à assembler et les paramètres de simulation. Il existe deux manières de construire ces scripts d'exécution depuis le serveur Web : nous pouvons utiliser MC-Fold pour prédire une structure secondaire et appeler directement MC-Sym sur cette structure secondaire, ou bien fournir une structure secondaire au format *dotbracket*. La première approche est recommandée par les auteurs, car MC-Fold produit une structure secondaire comportant des interactions tertiaires à l'intérieur des boucles internes, ce qui réduit l'espace de conformations à explorer par MC-Sym, par rapport à une structure secondaire classique ne comportant que les interactions Watson-Crick canoniques.

Dans un premier temps, nous avons utilisé l'interface Web fournie pour construire les scripts correspondant à la structure secondaire des molécules, modifié le temps limite de simulation par défaut de trente minutes à une journée, et exécuté ces scripts. Malheureusement, MC-Sym ne parvient à un résultat que pour les plus petites molécules, et nous ne sommes pas parvenus à obtenir de structures pour

3. `http://www.major.iric.ca/MC-Pipeline/`

des molécules plus grosses que 1E8O (dont les résultats sont présentés dans les tables 11.4 pour les structures atomiques et 11.5 pour les squelettes).

MC-Sym					
Molécule	Nucléotides	Candidat 1	Candidat 2	Candidat 3	Temps
1E8O	49	15,1	17,5 (15,5)	14,6 (14,5)	31 min 5 s

TABLE 11.4: RMSD (en Å) entre la 3D réelle et les prédictions du logiciel MC-Sym. L'alignement a été calculé par le logiciel PyMol.

		MC-Sym				
		Nous		MC-Sym		
Molécule	Sommets	Global	Jeu	1	2	3
1E8O	7	3,48	3,98	9,85	10,93	8,67

TABLE 11.5: RMSD (en Å) entre la 3D réelle et les prédictions du logiciel MC-Sym. L'alignement a été calculé par le logiciel PyMol.

L'utilisation préalable de MC-Fold ne résout pas ces problèmes. Nous avons fourni à MC-Fold la séquence et la structure secondaire des molécules en entrée. Pour la molécule 1MFQ, MC-Fold ne produit pas de prédiction. Pour les molécules 1NBS et 2A64, malgré la présence en entrée de la structure secondaire, MC-Fold prédit une structure ne contenant aucune liaison, Watson-Crick ou autre.

Par conséquent, bien que le couple MC-Fold/MC-Sym soit plus performant que les logiciels précédemment cités sur des structures de moins de 100 nucléotides, il ne semble pas envisageable de s'en servir pour prédire les structures de plus grosses molécules.

11.5 Conclusion

Sur toutes les structures étudiées, l'approche que nous proposons prédit mieux, en termes de RMSD, les structures à gros grain que les méthodes existantes. Nous pourrions penser que cela est dû au fait que les autres méthodes n'essaient pas de prédire une structure à gros grain mais au niveau atomique, et que passer à une représentation à gros grain gommerait les imprécisions. Cependant, nous avons vu que faire les mesures sur le graphe squelette des prédictions concurrentes au lieu du graphe atomique n'apporte qu'une amélioration marginale de la RMSD. Notre approche obtient de meilleures RMSD non pas parce qu'elle manipule des graphes

ayant un plus petit nombre de sommets, mais bien parce qu'elle positionne mieux les différents constituants de la molécule les uns par rapport aux autres.

Quatrième partie

Conclusion

12 Conclusion

Au cours de cette thèse, nous avons développé une nouvelle approche de prédiction à gros grain de la structure tridimensionnelle de l'ARN. Cette approche permet d'aller au delà des limites de taille imposées par les méthodes habituelles simulant la dynamique de repliement en détail ; alors que ces dernières sont adaptées aux molécules de moins cent nucléotides, notre approche s'intéresse aux plus grosses molécules. Sur ces dernières, nos prédictions sont plus proches de la structure cristallographiée que les prédictions des logiciels concurrents.

Notre approche n'essaye pas de simuler la dynamique de repliement. Elle suppose que, à partir de la structure secondaire d'une molécule, constituée d'hélices et de jonctions entre hélices, il est possible de déterminer la forme de chaque constituant pris séparément, hors de son contexte dans la structure. Une fois chaque constituant classifié en fonction de sa famille topologique, nous reconstituons un prototype de forme pour cette molécule ne tenant pas compte d'éventuelles interactions de type tertiaire. Enfin, nous avons étudié deux approches permettant de raffiner cette forme prototype pour tenir compte d'une structure tertiaire, d'une part par optimisation globale d'une fonction de coût sur les sommets, et d'autre part par optimisation locale de chacun des sommets, au moyen d'algorithmes de théorie des jeux.

Dans la première partie, nous avons développé le processus de classification des jonctions en famille topologique. Nous avons donné une méthode de classification automatique complète pour les 3-jonctions. Cette méthode donne de bons résultats si l'on dispose d'une seule jonction, et peut être améliorée si l'on dispose d'un ensemble de jonctions homologues, ce qui est souvent le cas lorsque l'on cherche à déterminer la forme d'une molécule inconnue. Nous avons proposé des classifications automatiques partielles pour les 2-jonctions et les 4-jonctions, une partie de ces dernières n'étant pas classifiables dans la mesure où nous disposons de trop peu d'individus pour en extraire les critères représentatifs. Nous avons ensuite abordé le cas des jonctions comprenant plus de quatre hélices, pour lesquelles nous avons également trop peu de représentants à ce jour pour proposer une classification.

Les principales perspectives concernant cette partie sont d'ajouter de nouveaux représentants aux familles que nous connaissons afin d'en préciser les propriétés, notamment pour la famille B des 3-jonctions, qui est la plus mal caractérisée des trois familles. Nous savons que de nombreuses grosses molécules existent [3, 4].

L'obtention, par cristallographie ou RMN, des structures de certaines de ces molécules devrait permettre également de complétér la classification des 4-jonctions, voire de développer une classification des jonctions de degré plus élevé (6-, 7-, 8-jonctions...). Enfin, s'il s'avérait que les jonctions de degré élevé échappent à une classification sur des critères simples de structure secondaire, il devrait néanmoins être possible de caractériser une partie de leur forme, notamment l'empilement entre certaines hélices. Nous pourrions ensuite représenter les parties indéterminées de ces jonctions autrement, soit en relâchant les contraintes d'angle et de distance, soit en essayant de déterminer par simulation la forme des brins.

Dans la deuxième partie de cette thèse, nous avons décrit une modélisation à gros grain de la structure des ARN, sous la forme d'un graphe squelette dont les sommets représentent hélices et jonctions. En classifiant les jonctions de ce graphe squelette selon les méthodes décrites dans la première partie, nous avons proposé une méthode permettant d'obtenir un plongement de ce graphe dans l'espace. Bien que le graphe représente uniquement la structure secondaire de la molécule, nous avons montré que ce plongement initial était très proche des structures cristallographiées si les jonctions étaient bien classifiées.

Le graphe ainsi obtenu peut néanmoins être amélioré, soit parce qu'il nous manque des informations sur les éventuelles interactions entre des sommets éloignés dans la structure secondaire, soit parce que nous avons commis des erreurs de classification. Nous avons voulu capturer la possibilité pour la molécule, à partir de plongement initial, de se replier sur elle-même afin d'augmenter sa stabilité. Nous avons formulé une fonction de coût décrivant l'inconfort d'un sommet par rapport à sa position initiale et le gain de stabilité gagné par l'ajout d'interactions de type tertiaire. Nous avons ensuite proposé deux approches permettant de raffiner le plongement initial. La première, utilisant un algorithme évolutionniste, essaye d'optimiser une fonction de coût globale : la somme des coûts de tous les sommets. Cette approche donne de bons résultats sur des structures bien classifiées, et permet également de corriger certaines erreurs de classification. La deuxième approche, plus exploratoire, modélise le problème du repliement par un jeu, et tente d'obtenir un équilibre dans lequel chaque sommet essaye d'optimiser son coût de manière égoïste. Nous avons obtenu de cette manière des résultats de qualité similaire à ceux de l'optimisation globale, parfois meilleurs et parfois moins bons, ce qui nous laisse penser que cette approche est prometteuse.

Les perspectives concernant cette partie sont nombreuses. Tout d'abord, le plus gros des erreurs que nous avons observées est dû à des jonctions sortant du champ de notre classification, ce qui rejoint les perspectives évoquées précédemment. Par ailleurs, l'algorithme modélisant le repliement de la molécule est mal adapté à une représentation où les sommets sont des volumes et pas des points. Cet algorithme pourrait être amélioré pour mieux représenter les rotations des jonctions dans l'espace lors de leur repliement. La fonction de coût que nous utilisons a été conçue en grande partie de manière intuitive et empirique ; elle ne correspond pas nécessaire-

ment au meilleur choix, et pourrait probablement être améliorée, notamment dans la manière dont le coût d'un sommet influence le coût des sommets voisins. De plus, pour une utilisation pratique, une implémentation des algorithmes présentés utilisant les capacités de parallélisation des processeurs graphiques actuels fournirait une accélération notable, permettant de converger en quelques minutes au lieu de quelques heures, mais également d'explorer les résultats de convergence pour de plus petites valeurs du paramètre b, ce qui pourrait améliorer qualitativement les prédictions.

Enfin, bien que nous pensions qu'une représentation à gros grain des molécules est utile en elle-même, il est difficile de ne pas envisager de parvenir à l'échelle du nucléotide ou à l'échelle atomique. Nous pouvons distinguer deux grandes manières d'y parvenir. La première serait d'assembler, à partir de notre représentation à gros grain, les hélices aux bons emplacements, puis tenter de reconstituer les parties simple-brin par simulation. La deuxième serait d'intégrer à d'autres logiciels de prédiction de structure l'information que notre prédiction apporte, sous la forme, par exemple, de contraintes de distance entre des sommets que nous prédisons proches. Nous avons vu en effet que, en comparant nos prédictions avec celles des logiciels concurrents, ceux-ci étaient précis localement, mais avaient des difficultés à prédire les angles relatifs entre les hélices, et la forme générale des molécules. Nous pensons donc que les deux approches peuvent se compléter.

Cinquième partie

Appendices

A Pseudocode des tests de classification

Le pseudocode présenté dans cet appendice suit la syntaxe du langage Ruby. Le plus gros de cette syntaxe devrait être évident, sauf peut-être les points suivants :

- "%" signifie "modulo", "x += y" signifie "assigner x+y à x."
- l'instruction **case/when/else** est équivalente au **switch/case/default** du C
- "collection.each { |i| ... }" décrit un intérateur sur la collection, et nomme chaque élément i. Cette instruction est équivalente à "foreach i in collection" dans d'autres langages.

Les structures de données sont très simples : une *jonction* est composée de trois brins, et chaque brin est constitué de trois parties (première hélice, non apparié et seconde hélice). Une *configuration* est un couple (s, f) d'empilement $s \in \{1, 2, 3\}$ et de famille $f \in \{A, B, C\}$.

A.1 Empilement sur le plus court des brins

```
1  def compute_for(configuration)
2    a = junction.strand(0).unpaired.size
3    b = junction.strand(2).unpaired.size
4    c = junction.strand(1).unpaired.size
5
6    sa = 1.0/(1+a)
7    sb = 1.0/(1+b)
8    sc = 1.0/(1+c)
9    if (a<=b) then sb += 1.0/(1+b-a) else  sa += 1.0/(1+a-b) end
10   if (b<=c) then sc += 1.0/(1+c-b) else  sb += 1.0/(1+b-c) end
11   if (c<=a) then sa += 1.0/(1+a-c) else  sc += 1.0/(1+c-a) end
12
13   if (a<=b) then sa += 1 else sb += 1 end
14   if (b<=c) then sb += 1 else sc += 1 end
15   if (c<=a) then sc += 1 else sa += 1 end
16
17   sa = 10.0 if a == 0
18   sb = 10.0 if b == 0
19   sc = 10.0 if c == 0
20
```

```
21    return case (configuration.stacking)
22           when 0
23             sa
24           when 1
25             sb
26           when 2
27             sc
28           end
29  end
```

A.2 Longueurs relatives des brins non impliqués dans un empilement

```
1   def compute_for(configuration)
2     s1 = junction.strand((4-configuration.stacking)%3).unpaired.size
3     s2 = junction.strand((5-configuration.stacking)%3).unpaired.size
4     delta = s1 - s2
5     value = case configuration.family
6             when 'A'
7               delta
8             when 'B'
9               delta.absolute_value
10            when 'C'
11              -delta
12            end
13    value = case configuration.family
14            when 'A'
15              value
16            when 'B'
17              case value
18              when 0
19                5.0
20              when 1
21                5.0
22              when 2
23                1.0
24              else
25                3.0 - value
26              end
27            when 'C'
28              bonus = junction.strand((5-rotation)%3).unpaired.size - 5
29              bonus = 0 if bonus < 0
30              value + bonus
```

```
31            end
32    return if value > 10.0
33              10.0
34            elsif value < -10.0
35              -10.0
36            else
37              value
38            end
39  end
```

A.3 Paires de bases terminant les hélices

Ce test utilise trois matrices de score, une pour chaque hélice de la jonction. Les trois brins d'une jonction sont appelés V, R et B (pour *vert*, *rouge* et *bleu*), avec V étant le brin impliqué dans l'empilement, rouge le brin suivant (par ordre de transcription), et B le dernier brin. Ainsi, la matrice de la première hélice (constituée des brins V et B) est-elle nommée V5B3 (c-à-d. Vert-5', Bleu-3').

Nous décrivons d'abord le calcul du score, en supposant que nous ayons ces matrices. Nous décrirons ensuite comment construire les matrices à partir d'un ensemble de jonctions pré-classifiées.

A.3.1 Calcul du score

```
1  def compute_for(configuration)
2    column = case configuration.family
3             when 'A'
4               0
5             when 'B'
6               1
7             when 'C'
8               2
9             end
10
11   iV = (3-configuration.stacking)%3
12   iR = (4-configuration.stacking)%3
13   iB = (5-configuration.stacking)%3
14
15   v5 = junction.strand(iV).first_helix.last_nucleotide
16   r5 = junction.strand(iR).first_helix.last_nucleotide
17   b5 = junction.strand(iB).first_helix.last_nucleotide
18
19   v3 = junction.strand(iV).last_helix.first_nucleotide
20   r3 = junction.strand(iR).last_helix.first_nucleotide
```

```
21    b3 = junction.strand(iB).last_helix.first_nucleotide
22
23    column = case configuration.family
24             when 'A'
25                0
26             when 'B'
27                1
28             when 'C'
29                2
30             end
31
32    return R5V3[r5+v3][column] + B5R3[b5+r3][column] +
33           V5B3[v5+b3][column]
34  end
```

A.3.2 Construction des matrices de score

Pour construire les matrices de score, nous comptons combien de fois l'hélice numéro x se termine par une paire de bases spécifique, et ce pour chaque famille. Cela nous donne les trois matrices représentées dans la table A.1. Nous appliquons ensuite l'algorithme suivant à ces matrices, afin de construire les matrices représentées dans la table A.2.

hélice 1 :

A	B	C	
0	0	2	AU
5	4	2	CG
4	2	7	GC
1	1	1	UA
0	0	0	GU
0	0	3	UG

hélice 2 :

A	B	C	
1	0	0	AU
6	4	3	CG
3	2	5	GC
0	1	6	UA
0	0	1	GU
0	0	0	UG

hélice 3 :

A	B	C	
2	1	2	AU
7	3	8	CG
0	2	2	GC
0	1	3	UA
0	0	0	GU
1	0	0	UG

TABLE A.1: Décompte des paires de bases terminales, selon la famille, pour les trois hélices des jonctions.

```
1  def build(helix1, helix2, helix3)
2     k = 6;
3     R5V3, B5R3, V5B3 = Array.new(3) {Array.new(6) {Array.new(3)}}
4
5     0.upto(2) { |i|
```

```
6       s = 0
7       0.upto(k-1) { |j| s += helix1[j][i] }
8       0.upto(k-1) { |j| R5V3[j][i] = (k*helix1[j][i] - s)/(s*(k-1)) }
9       s = 0
10      0.upto(k-1) { |j| s += helix2[j][i] }
11      0.upto(k-1) { |j| B5R3[j][i] = (k*helix2[j][i] - s)/(s*(k-1)) }
12      s = 0
13      0.upto(k-1) { |j| s += helix3[j][i] }
14      0.upto(k-1) { |j| V5B3[j][i] = (k*helix3[j][i] - s)/(s*(k-1)) }
15    }
16  end
```

A	B	C	
-0.20	-0.20	-0.04	AU
0.40	0.49	-0.04	CG
0.28	0.14	0.36	GC
-0.08	-0.03	-0.12	UA
-0.20	-0.20	-0.20	GU
-0.20	-0.20	0.04	UG

A	B	C	
-0.08	-0.20	-0.20	AU
0.52	0.49	0.04	CG
0.16	0.14	0.20	GC
-0.20	-0.03	0.28	UA
-0.20	-0.20	-0.12	GU
-0.20	-0.20	-0.20	UG

A	B	C	
0.04	-0.03	-0.04	AU
0.64	0.31	0.44	CG
-0.20	0.14	-0.04	GC
-0.20	-0.03	0.04	UA
-0.20	-0.20	-0.20	GU
-0.08	-0.20	-0.20	UG

TABLE A.2: Matrices de score, selon la famille, pour les trois hélices.

A.4 Critères bonus

```
1   def compute_for(configuration)
2     bonus = 0
3     iV = (3-configuration.stacking)%3
4     iR = (4-configuration.stacking)%3
5     iB = (5-configuration.stacking)%3
6
7     case configuration.family
8     when 'A'
9       # bonus 1: an A in position 2,3,4 or 5 on strand iR
10      junction.strand(iR)[2..5].each { |nucleotide|
11        if nucleotide == 'A'
```

```
12              bonus += 1
13              break
14          end
15      }
16
17      # bonus 2
18      first_non_WC_v3 = junction.strand(iV).unpaired.last
19      first_non_WC_R5 = junction.strand(iR).unpaired.first
20      if first_non_WC_v3 == 'A'
21          bonus += 1
22          if first_non_WC_R5 == 'G'
23              bonus += 0.5
24          end
25      end
26
27    when 'C'
28        # bonus 3: do we have a AU, AAU, AUA sequence on the unpaired
29        # ''above'' strand, in position 1, 2, 3 or 4 after the last
30        # base-pair in 3' order?
31        first_non_WC_B3 = junction.strand(iB).unpaired.last
32        a = u = 0
33        (first_non_WC_B3-1).downto(first_non_WC_B3-4) { |nucleotide|
34          case nucleotide
35          when 'A'
36              a += 1
37          when 'U'
38              u += 1
39          end
40        }
41        bonus = if a > 0
42                    (a+u)/2
43                else
44                    0
45                end
46    end
47
48    return bonus
49  end
```

B Rappels et notions utiles

B.1 Rappels de statistiques

B.1.1 Tests et mesures de qualité des tests

Un test est une fonction booléenne qui a une entrée associe la réponse *oui* ou *non*. On appelle *vrai positif* (VP) une réponse *oui* correcte et *faux positif* (FP) une réponse *oui* alors qu'elle aurait dû être *non*. De même, on appelle *vrai négatif* (VN) une réponse *non* correcte et *faux négatif* (FN) une réponse *non* alors qu'elle aurait dû être *oui*. À partir de ces définitions, nous pouvons calculer plusieurs mesures de qualité pour un test :

$$\text{PPV} = \frac{\text{VP}}{\text{VP} + \text{FP}}$$
$$\text{Spécificité} = \frac{\text{VN}}{\text{VN} + \text{FP}}$$
$$\text{Sensibilité} = \frac{\text{VP}}{\text{VP} + \text{FN}}$$

La PPV (*positive predictive value*), ou *sélectivité*, indique quel pourcentage des réponses *oui* sont correctes. La spécificité indique quel pourcentage des réponses qui devraient être *non* sont effectivement *non*, c'est à dire à quel point le test est capable d'exclure les valeurs négatives. Enfin, la sensibilité indique quel pourcentage des réponses qui devraient être *oui* sont effectivement *oui*, c'est à dire à quel point le test est capable de reconnaitre les valeurs positives.

B.1.2 RMSD

La RMSD (*Root Mean Square Deviation*) est la mesure de distance la plus répandue pour comparer deux structures. Si l'on désire comparer un ensemble ordonné de n points X avec un ensemble ordonné de n points Y, la RMSD se calcule suivant la formule suivante :

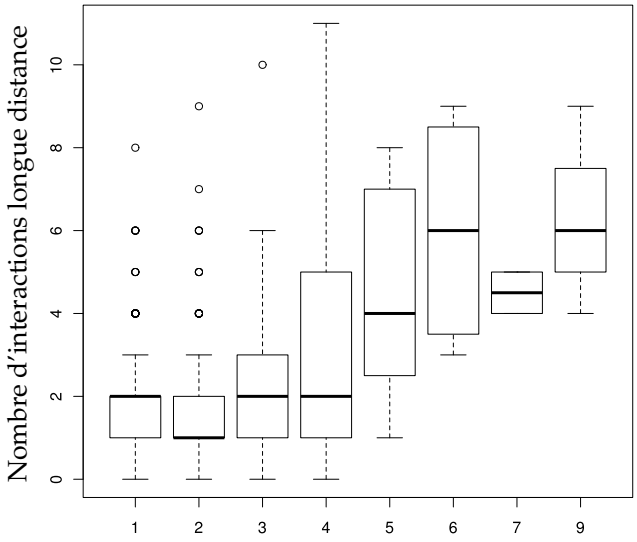
$$\text{RMSD}(X, Y) = \sqrt{\frac{\sum_{i=1}^{n} d(x_i, y_i)}{n}}$$

Où $d(x, y)$ désigne la distance euclidienne entre x et y.

Pour que la RMSD soit une bonne mesure de distance entre deux structures, il faut qu'à) chaque point x_i corresponde un point y_i, et il faut ques les deux structures soient alignées et orientées « le mieux possible » afin de minimiser la distance séparant les points associés. Plusieurs algorithmes existent, par exemple celui de Kabsch [61] afin de calculer une transformation minimisant la RMSD.

B.1.3 Boîtes à moustache

Les boîtes à moustache, ou *box plot* en anglais, sont une représentation graphique des principaux paramètres d'une série statistique. En voici un exemple, tiré du chapitre 8 :



Il s'agit d'un rectangle allant du premier quartile au troisième quartile, coupé par un trait en gras indiquant la médiane. Autrement dit, 75% des valeurs de la série sont dans le rectangle. Si t est la taille du rectangle, les « moustaches » en pointillé englobent les points se trouvant à moins de 1,5t du rectangle. Enfin, les cercles désignent les valeurs aberrantes (*outliers*).

B.2 Rappels de théorie des graphes

Un graphe [78, 79] $G = (V, E)$ est un ensemble de sommets V et une relation binaire E sur V dénommée « relation d'adjacence ». Les graphes sont généralement visualisés en représentant chaque sommet par un cercle, et en reliant par un segment les cercles correspondant aux sommets u et v si $(u, v) \in E$. Un graphe est dit « non orienté » si E est une relation symmétrique, et « orienté » sinon. Les couples $(u, v) \in E$ sont dénommés « arêtes » si G est non orienté, et « arcs » sinon.

On dit qu'un chemin existe entre u et v si, par fermeture transitive, u est en relation avec v, et la longueur de ce chemin est le nombre d'arêtes (ou arcs) qui

le composent. L'ensemble des sommets reliés les uns aux autres par des chemins forment une composante connexe du graphe. Enfin, un cycle est un chemin partant d'un sommet et revenant à ce sommet. Un graphe non orienté ne comportant pas de cycle est appelé « arbre », et l'absence de cycle fait que de nombreux problèmes sont plus simples à résoudre sur des arbres.

C Glossaire

Jonction

Dans la structure secondaire, certaines parties de la séquence d'ARN s'apparient et forment des hélices. Ces hélices sont reliées entre elles par des parties non appariées. Le point de rencontre entre des hélices est nommé *jonction*. Voir figure 2.2 page 10.

Interaction longue-distance

Une interaction longue-distance est une interaction entre deux parties d'une molécule qui sont éloignées dans la structure secondaire. Cependant, dans l'espace, la distance entre les deux parties reliées par cette interaction est nécessairement courte. Voir page 67. Dans le graphe squelette, les interactions longue-distance sont les arêtes appartenant à l'ensemble E_l (page 62).

Interaction tertiaire

Une interaction tertiaire entre deux nucléotides est une interaction n'appartenant pas à la structure secondaire, c'est à dire autre qu'une liaison phosphodiester, une liaison Watson-Crick canonique (A/U et G/C sur les faces Watson-Crick) ou une liaison *wobble* (G/U sur les faces Watson-Crick). Voir page 7.

Pseudonœud

Parmi les lisaisons Watson-Crick, on distingue le cas des pseudonœuds, qui se produisent lorsqu'un brin reliant deux hélices ou une boucle terminale établit des liaisons Watson-Crick canoniques avec une autre partie de la molécule. Voir section 2.4 page 11.

Structure secondaire

La structure secondaire d'une molécule est l'ensemble des interactions de type Watson-Crick canoniques (A/U et G/C), des liaisons *wobble* (G/U) et des liaisons phosphodiester du *backbone*. Comme c'est souvent le cas, dans ce manuscrit nous considérons que les pseudonœuds ne font pas partie de la structure secondaire. Voir section 2.2 page 8.

Structure tertiaire

La structure tertiaire d'une molécule est l'ensemble des interactions entre les nucléotides. Il s'agit de la structure secondaire, à laquelle on ajoute toutes les interactions qui n'en faisaient pas partie, et que l'on nomme donc *interactions tertiaires*. Il ne faut pas confondre structure tertiaire et structure tridimensionnelle (voir plus loin dans ce glossaire). Voir section 2.2 page 8.

Structure tridimensionnelle (3D)

La structure 3D d'une molécule est l'ensemble des coordonnées dans l'espace des atomes qui la constituent. La structure tertiaire (c-à-d. les interactions) peut être déduite de la structure 3D, mais construire la structure 3D à partir de la structure tertiaire n'est pas un problème résolu.

Table des figures

Liste des tableaux

Bibliographie

[1] J. D. Watson. Lettre à Francis Crick. February 1955. Cité page 1.

[2] Nils G. Walter Bateley, Sarah A. Woodson, and Robert T. *Non-Protein Coding RNAs*. Springer, 2008. Cité page 1.

[3] M. Guttman, I. Amit, M. Garber, C. French, M. F. Lin, D. Feldser, M. Huarte, O. Zuk, B. W. Carey, J. P. Cassady, M. N. Cabili, R. Jaenisch, R. S. Mikkelsen, T. Jacks, N. Hacohen, B. E. Bernstein, M. Kellis, A. Regev, J. L. Rinn, and E. S. Lander. Chromatin Signature Reveals over a Thousand Highly Conserved Large Non-Coding RNAs in Mammals. *Nature*, 458(7235) :223–227, February 2009. Cité pages 1 et 119.

[4] Z. Weinberg, J. Perreault, M. M. Meyer, and R. R. Breaker. Exceptional Structured Noncoding RNAs Revealed by Bacterial Metagenome Analysis. *Nature*, 462 :656–659, December 2009. Cité pages 1 et 119.

[5] N. B. Leontis and E. Westhof. Geometric Nomenclature and Classification of RNA Base Pairs. *RNA*, 7 :499–512, 2001. Cité pages 7, 8, 11 et 47.

[6] R. I. Dima, C. Hyeon, and D. Thirumalai. Extracting Stacking Interaction Parameters for RNA from the Data Set of Native Structures. *J. Mol. Biol.*, 347 :53–69, March 2005. Cité pages 8 et 9.

[7] L. Nasalean, J. Stombaugh, C. L. Zirbel, and N. B. Leontis. RNA 3D Strutural Motifs : Definition, Identification, Annotation, and Database Searching. In *Non-Protein Coding RNAs*, chapter 1. Springer, December 2009. Cité pages 8 et 9.

[8] I. Tinoco Jr. and C. Bustamante. How RNA Folds. *J. Mol. Biol.*, 293 :271–281, 1999. Cité pages 9 et 12.

[9] N. B. Leontis, A. Lescoute, and E. Westhof. Building Blocks and Motifs of RNA Architecture, The. *Current Opinion in Structural Biology*, 16 :279–287, 2006. Cité page 9.

[10] M. Djelloul and A. Denise. Automated Motif Extraction and Classification in RNA Tertiary Structures. *RNA*, 14(12) :1–9, 2008. Cité pages 9 et 10.

[11] A. Lescoute, N. B. Leontis, C. Massire, and E. Westhof. Recurrent Structural RNA Motifs, Isostericity Matrices and Sequence Alignments. *Nucleic Acids Research*, 33(8) :2396–2409, April 2005. Cité pages 10, 47, 48, 49 et 50.

[12] A. Lescoute and E. Westhof. Topology of Three-Way Junctions in Folded RNAs. *RNA*, 12 :83–93, 2006. Cité pages 10, 29, 31, 32, 34, 37 et 38.

[13] C. Laing, S. Jung, A. Iqbal, and T. Schlick. Tertiary Motifs Revealed in Analyses of Higher-Order RNA Junctions. *J. Mol. Biol.*, 393 :67–82, August 2009. Cité pages 10 et 54.

[14] C. Schudoma, A. Larhlimi, and D. Walther. Influence of the Local Sequence Environment on RNA Loop Structures, The. *RNA*, 17 :1247–1257, 2011. Cité page 10.

[15] M. De La Pena, D. Dufour, and J. Gallego. Three-Way RNA Junctions with Remote Tertiary Contacts : A Recurrent and Highly Versatile Fold. *RNA*, 15 :1949–1964, 2009. Cité page 10.

[16] R. Tyagi and D. H. Mathews. Predicting Helical Coaxial Stacking in RNA Multibranch Loops. *RNA*, pages 939–951, July 2007. Cité pages 10 et 40.

[17] M. Sarver, C. L. Zirbel, J. Stombaugh, A. Mokdad, and N. B. Leontis. FR3D : Finding Local and Composite Recurrent Structural Motifs in RNA 3D Structures. *Journal of Mathematical Biology*, 56 :215–252, 2008. Cité pages 11 et 32.

[18] E. Bindewald, R. Hayes, Y. Yingling, W. Kasprzak, and B. A. Shapiro. RNAJunction : A Database of RNA Junctions and Kissing Loops For Three-Dimensional Structural Analysis and Nanodesign. *Nucleic Acids Research*, 36 :392–397, January 2008. Cité page 11.

[19] J. L. Chen and C. W. Greider. Functional Analysis of the Pseudoknot Structure in Human Telomerase RNA. *PNAS*, 102(23) :8080–5, 2005. Cité page 11.

[20] D. Thirumalai and C. Hyeon. Theory of RNA Folding : From Hairpins to Ribozymes. In *Non-Protein Coding RNAs*, chapter 2. Springer, December 2009. Cité page 12.

[21] M. Mandal and R. R. Breaker. Gene Regulation by Riboswitches. *Nat Rev Mol Cell Biol*, 5 :451–463, June 2004. Cité page 12.

[22] C. Laing and T. Schlick. Computational Approaches to 3D Modeling of RNA. *J Phys- Condens Mat*, 22. Cité pages 17, 24 et 26.

[23] R. Nussinov, G. Piecznik, J. R. Griggs, and D. J. Kleitman. Algorithms for Loop Matchings. *SIAM Journal on Applied Mathematics*, 35 :68–82, 1978. Cité page 18.

[24] M. Zuker and P. Stiegler. Optimal Computer Folding of Large RNA Sequences Using Thermodynamics and Auxiliary Information. *Nucl. Acids Res.*, 9 :133–148, 1981. Cité pages 18 et 20.

[25] M. Zuker. On Finding All Suboptimal Foldings of an RNA Molecule. *Science*, 244 :48–52, 1989. Cité page 18.

[26] I. L. Hofacker, W. Fontana, S. Bonhoeffer, and P. F. Stadler. Fast Folding and Comparison of RNA Secondary Structures. *Monatshefte Fur Chemie*, 125 :167–188, 1994. Cité page 18.

[27] Y. Ding and C. E. Lawrence. A Statistical Sampling Algorithm for RNA Secondary Structure Prediction. *Nucl. Acids Res.*, 31 :7280–7301, 2003. Cité page 18.

[28] R. Giegerich, B. Voß, and M. Rehmsmeier. Abstract Shapes of RNA. *Nucl. Acids Res.*, 32(16) :4843–4851, 2004. Cité page 18.

[29] C. B. Do, D. A. Woods, and S. Batzoglou. CONTRAfold : RNA Secondary Structure Prediction Without Physics-Based Models. *Bioinformatics*, 22(14) :e90–e98, 2006. Cité page 19.

[30] R. B. Lyngsø and C. N. Pedersen. RNA Pseudoknot Prediction in Energy-Based Models. *J Comput Biol*, 7 :409–27, 2007. Cité page 19.

[31] E. Rivas and S. R. Eddy. A Dynamic Programming Algorithm for RNA Structure Prediction Including Pseudoknots. *J. Mol. Biol.*, 285(5) :2053–68, 1999. Cité page 19.

[32] E. Bindewald, T. Kluth, and B. A. Shapiro. CyloFold : Secondary Structure Prediction Including Pseudoknots. *Nucleic Acids Research*, 38 :368–372, 2010. Cité page 19.

[33] Y. Ponty and C. Sault. A Combinatorial Framework for the Design of (pseudoknotted) RNA Algorithms. September 2011. Cité page 19.

[34] M. Geis, C. Flamm, M. T. Wolfinger, I. L. Hofacker, M. Middendorf, C. Mandl, P. F. Stadler, and C. Thurner. Folding Kinetics of Large RNAs. *Journal of Molecular Biology*, (1) :160–173, 2008. Cité page 19.

[35] A. Xayaphoummine, T. Bucher, and H. Isambert. Kinefold Web Server for RNA/DNA Folding Path and Structure Prediction Including Pseudoknots and Knots. *Nucl. Acids Res.*, 33 :W605–10, 2005. Cité page 19.

[36] P. P. Gardner and R. Giegerich. A Comprehensive Comparison of Comparative RNA Structure Prediction Approaches. *BMC Bioinformatics*, 5(140), September 2004. Cité pages 20, 21, 22 et 23.

[37] J. Thompson, D. Higgins, and T. Gibson. CLUSTAL W : Improving the Sensitivity of Progressive Multiple Sequence Alignment Through Sequence Weighting, Positions-Specific Gap Penalties and Weight Matrix Choice. *Nucleic Acids Research*, 22 :4673–4680, 1994. Cité page 20.

[38] I. L. Hofacker, M. Fekete, and P. F. Stadler. Secondary Structure Prediction for Aligned RNA Sequences. *Journal of Molecular Biology*, 319(5) :1059–1066, 2002. Cité page 20.

[39] B. Knudsen and J. Hein. Pfold : RNA Secondary Structure Preditction Using Stochastic Context-Free Grammars. *Nucleic Acids Research*, 31(13) :3423–3428, 2003. Cité page 20.

[40] D. Sankoff. Simultaneous Solution of the RNA Folding, Alignment and Protosequence Problems. *SIAM Journal on Applied Mathematics*, 45(5) :810–825, 1985. Cité page 20.

[41] J. Hull Havgaard, R. B. Lyngsø, G. D. Stormo, and J. Gorodkin. Pairwise Local Structural Alignment of RNA Sequences with Sequence Similarity Less Than 40%. *Bioinformatics*, 21(9) :1815–1824, 2005. Cité page 20.

[42] D. H. Mathews and D. Turner. Dynalign : an Algorithm for Finding the Secondary Structure Common to Two RNA Sequences. *Journal of Molecular Biology*, 317(2) :191–203, 2002. Cité page 21.

[43] O. Perriquet, H. Touzet, and M. Daucher. Finding the Common Structure Shared by Two Homologous RNAs. *Bioinformatics*, 19(1) :108–116, 2003. Cité page 21.

[44] M. Höchsmann, Y. Töller, R. Giegerich, and S. Kurtz. Local Similarity of RNA Secondary Structures. *Proc of the IEEE Bioinformatics Conference*, pages 159–168, 2003. Cité page 21.

[45] M. Höchsmann. *Tree Alignment Model : Algorithms, Implementations and Applications for the Analysis of RNA Secondary Structures, The*. PhD thesis, 2005. Cité page 21.

[46] S. Siebert and R. Backofen. MARNA : Multiple Alignment and Consensus Structure Prediction of RNAs Based on Sequence Structure Comparisons. *Bioinformatics*, 21(16) :3352–3359, 2005. Cité page 22.

[47] H. M. Martinez, J. V. Maizel, and B. A. Shapiro. RNA2D3D : a Program for Generating, Viewing, and Comparing 3-Dimensional Models of RNA. *J Biomol Struct Dyn*, 25(6) :669–83, 2008. Cité page 24.

[48] F. Jossinet, T. E. Ludwig, and E. Westhof. Assemble : an Interactive Graphical Tool to Analyze and Build RNA Architectures at the 2D and 3D Levels. *Bioinformatics*, 2010. Cité page 24.

[49] S. Sharma, F. Ding, and N. V. Dokholyan. IFoldRNA : Three-Dimensional RNA Structure Prediction and Folding. *Bioinformatics*, 24(17) :1951–2, 2008. Cité pages 24 et 108.

[50] F. Ding, S. Sharma, P. Chalasani, V. V. Demidov, N. E. Broude, and N. V. Dokholyan. Ab Initio RNA Folding by Discrete Molecular Dynamics : from Structure Prediction to Folding Mechanisms. *RNA*, 14(6) :1164–73, 2008. Cité page 24.

[51] M. A. Jonikas, R. J. Radmer, A. Laederach, R. Das, S. Pearlman, D. Herschlag, and R. B. Altman. Coarse-Grained Modeling of Large RNA Molecules with Knowledge-Based Potentials and Structural Filters. *RNA*, 15(2) :189–99, 2009. Cité page 24.

[52] R. Das and D. Baker. Automated de Novo Prediction of Native-Like RNA Tertiary Structures. *PNAS*, 104(37) :14664–9, 2007. Cité pages 24 et 111.

[53] M. Parisien and F. Major. MC-Fold and MC-Sym Pipeline Infers RNA Structure from Sequence Data, The. *Nature*, 452(7183) :51–55, March 2008. Cité pages 24, 40 et 113.

[54] A. Lamiable, D. Barth, A. Denise, F. Quessette, S. Vial, and E. Westhof. Automated Prediction of Three-Way Junction Topological Families in RNA Secondary Structures. *Computational Biology and Chemistry*, 2012. Cité page 32.

[55] H. M. . Berman, J. Westbrook, Z. Feng, G. Gilliland, Bhat T. N., H. Weissig, I. N. Shindyalov, and P. E. Bourne. Protein Data Bank, The. *Nucleic Acids Research*, 28 :235–242, 2000. Cité page 32.

[56] H. Yang, F. Jossinet, N. B. Leontis, L. Chen, J. Westbrook, H. M. Berman, and E. Westhof. Tools for the Automatic Identification and Classification of RNA Base Pairs. *Nucl. Acids Res.*, 31(13) :3450–3460, July 2003. Cité page 32.

[57] S. Smit, K. Rother, J. Heringa, and R. Knight. From Knotted to Nested RNA Structures : A Variety of Computational Methods for Pseudoknot Removal. *RNA*, 14(3) :410–416, March 2008. Cité page 32.

[58] J. J. Cannone, S. Subramanian, M. N. Schnare, J. R. Collett, L. M. D'Souza, Y. Du, B. Feng, M. Lin, L. V. Madabusi, K. M. Müller, N. Pande, Z. Shang, N. Yu, and R. R. Gutell. Comparative RNA Web (CRW) Site : An Online Database of Comparative Sequence and Structure Information for Ribosomal, Intron, and Other RNAs., The. *BioMed Central Bioinformatics*, 3(2), 2002. Cité page 44.

[59] C. Laing and T. Schlick. Analysis of Four-Way Junctions in RNA Structures. *J. Mol. Biol.*, 390 :547–559, 2009. Cité pages 51, 52 et 54.

[60] W. K. Olson, M. Bansal, S. K. Burley, R. E. Dickerson, M. Gerstein, S. C. Harvey, U. Heinemann, X. J. Lu, S. Neidle, Z. Shakked, H. Sklenar, M. Suzuki, C. S. Tung, E. Westhof, C. Wolberger, and H. M. Berman. A Standard Reference Frame for the Description of Nucleic Acid Base-Pair Geometry. *J. Mol. Biol.*, 313 :229–237, 2001. Cité page 63.

[61] W. Kabsch. A Solution of the Best Rotation to Relate Two Sets of Vectors. *Acta Crystallographica*, 32(922), 1976. Cité pages 69 et 132.

[62] T. Kamada and S. Kawai. An Algorithm for Drawing General Undirected Graphs. *Information Processing Letters*, 31(1) :7–15, April 1989. Cité page 71.

[63] T. M. J. Fruchterman and E. M. Reingold. Graph Drawing by Force-Directed Placement. *Software – Practice and Experience*, 21(11) :1129–1164, November 1991. Cité page 71.

[64] J. H. Holland. Genetic Algorithms. *Scientific American*, July 1992. Cité page 81.

[65] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, 1989. Cité page 81.

[66] C. Darwin. *On the Origin of Species by Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life*. 1859. Cité page 81.

[67] J. Von Neumann and O. Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, 1944. Cité page 89.

[68] J. F. Nash Jr. Equilibrium Points in N-Person Games. *P.N.A.S*, 1950. Cité pages 89 et 90.

[69] C. H. Papadimitriou. Complexity of Finding Nash Equilibria, The. In *Algorithmic Game Theory*. Cambridge University Press, 2007. Cité page 90.

[70] C. H. Papadimitriou. On the Complexity of the Parity Argument and Other Inefficient Proofs of Existence. *Journal of Computer and System Sciences*, 48(3), June 1994. Cité page 91.

[71] P. S. Sastry, V. V. Phansalkar, and M. A. L. Thathachar. Decentralized Learning of Nash Equilibria in Multi-Person Stochastic Games with Incomplete Information. *IEEE Transactions on Systems, Man and Cybernetics*, 24(5) :769–777, May 1994. Cité pages 91 et 92.

[72] T. J. Lambert, M. Epelman, and R. L. Smith. A Fictitious Play Approach to Large-Scale Optimization. *Operations Research*, 53(3) :477–489, 2005. Cité page 94.

[73] M. Epelman, A. Ghate, and R. L. Smith. Sampled Fictitious Play for Approximate Dynamic Programming. *Computers & Operations Research*, 38 :1705–1718, February 2011. Cité page 94.

[74] G. W. Brown. *Iterative Solution of Games by Fictitious Play*. Wiley, 1951. Cité page 94.

[75] J. Robinson. An Iterative Method of Solving a Game. *Annals of Mathematics*, 54 :296–301, 1951. Cité page 94.

[76] B. Matthews. Comparison of the Predicted and Observed Secondary Structure of T4 Phage Lysozyme. *Biochimica Et Biophysica Acta (BBA) - Protein Structure*, 405(2) :442–451, 1975. Cité page 107.

[77] M. Parisien, J. A. Cruz, E. Westhof, and F. Major. New Metrics for Comparing and Assessing Discrepancies Between RNA 3D Structures and Models. *RNA*, 15 :1875–1885, 2009. Cité page 107.

[78] C. Berge. *Graphes Et Hypergraphes*. 1969. Cité page 132.

[79] R. Diestel. *Graph Theory*, volume 173 of *Graduate Texts in Mathematics*. Springer-Verlag, 2010. Cité page 132.