Approche algorithmique pour la prédiction de la structure secondaire des ARN

THÈSE

présentée et soutenue publiquement le 8 décembre 2003

pour l'obtention du

Doctorat de l'Université des Sciences et Technologies de Lille

(spécialité informatique)

par

Olivier PERRIQUET

Composition du jury

Président :	Rémi GILLERON, Professeur	Université de Lille III
Rapporteurs :	Serge DULUCQ, Professeur	LABRI, Université de Bordeaux I
	Daniel GAUTHERET, Professeur	TAGC INSERM, Université de la Méditerrannée
Examinateur :	Marie-France SAGOT, D.R.	INRIA Rhône-Alpes
Directeurs :	Max DAUCHET, Professeur	LIFL, Université des Sciences et Technologies de Lille
	Hélène TOUZET, Maître de conférences	LIFL, Université des Sciences et Technologies de Lille

UNIVERSITÉ DES SCIENCES ET TECHNOLOGIES DE LILLE
Laboratoire d'Informatique Fondamentale de Lille — UPRESA 8022
U.F.R. d'I.E.E.A. – Bât. M3 – 59655 VILLENEUVE D'ASCQ CEDEX
Tél. : +33 (0)3 20 43 47 24 – Télécopie : +33 (0)3 20 43 65 66 – email : direction@lifl.fr

Mis en page et composé avec LaTeX et la classe thlifl.

Remerciements

Je remercie en premier lieu mes directeurs de thèse pour l'expérience qu'ils m'ont transmise. J'ai le sentiment d'avoir scientifiquement mûri à leur contact. Hélène Touzet, qui m'a encadré quotidiennement, m'a appris par son exigence à donner le meilleur de moi-même : elle m'a enseigné ce qu'était un travail de qualité et je la remercie pour ce précieux savoir. Je remercie Max Dauchet pour son écoute et sa disponibilité malgré un emploi du temps chargé !

Je remercie Daniel Gautheret pour nos échanges agréables autour de l'ARN lors de mes séjours à Marseille. J'en garde un très bon souvenir. Je le remercie aussi pour avoir rapporté ma thèse et pour l'interêt qu'il porte à ce travail. Merci à Serge Dulucq d'avoir bien voulu rapporter ma thèse et à Marie-France Sagot pour avoir accepté de participer au jury comme examinatrice. Enfin merci à Rémi Gilleron d'avoir présidé ce jury.

Merci à l'équipe bioinfo pour son accueil. Celle du bureau d'à coté : Jean-Stéphane et sa bonne humeur intarissable, Hélène, Maude, Stéphane. Merci à l'équipe du b306 : Matthieu et particulièrement Martin, qui m'a aidé concernant bien des choses, techniques comme humaines. Bon courage à toi pour ta thèse. Merci à Jean-Paul Delahaye pour nos discussions fort interessantes qui commencent habituellement au coin d'un couloir sur une anecdote et finissent souvent dans son bureau sur des considérations théoriques.

Je remercie mes parents et mes deux petites sœurs, ainsi que mes amis de longue date (Thomas, Scott, Damien, Céline, ... et la clôture transitive ;–) qui ont su comprendre et accepter mes choix.

Un merci particulier à certaines personnes que j'ai rencontrées plus récemment sur ma route et qui m'ont éclairé lors de moments obscurs. Merci à Cédric Notredame pour m'avoir écouté et partagé avec moi son expérience humaine de la thèse. Merci à Prisca pour nos nuits blanches à parler de cinéma et de tant d'autres choses.

Merci à toi Aurélie pour ton insouciance. Tu donnes la touche de légèreté qui manque parfois à toutes ces choses très sérieuses.

(Bon, j'ai forcément oublié quelqu'un...)

Curieusement, l'atmosphère qui m'environnait durant ma thèse à (r)éveillé chez moi un goût pour d'autres types de recherches que je ne pourrai passer sous silence. Je remercie bluescreen et Julien Clauss, qui m'ont grandement aidé, chacun à leur manière, à mettre en place les préliminaires à ces futures recherches.

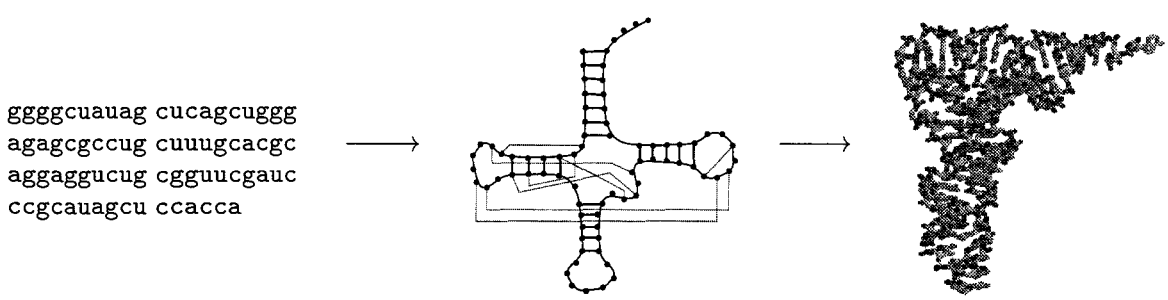
Table des matières

Introduction

De manière un peu simplifiée, on pourrait dire que dans les organismes vivants, l'information est *codée* linéairement, comme dans un ordinateur, puisque les trois types de molécules qui forment les briques structurelles de la vie – l'ADN, l'ARN et les protéines – sont des polymères linéaires. Cette analogie de codage conduit à ce que l'informatique soit un allié de poids concernant les problèmes génétiques. Mais elle s'arrête là, car ces molécules interagissent entre elles et se replient de manière compacte dans l'espace, épousant une *structure* qui joue un rôle essentiel.

Dans les rapports qu'entretiennent les trois types de polymères, l'ARN est très longtemps apparu comme une molécule intermédiaire, reléguant l'étude et la connaissance de sa structure à des questionnements de moindre importance. A l'heure actuelle, où le nombre de séquences et leur disponibilité via l'internet va croissante, et où notre connaissance des phénomènes génétiques se précise, on découvre de nouvelles familles d'ARN dont les fonctions très variées sont fondamentales à la vie cellulaire : ces classes d'ARN, qu'on appelle *non-codants* car ils ne sont pas traduits en proteines, adoptent en se repliant une structure qui semble conditionnée par leur séquence, ce qui les rend directement actifs dans la cellule.

La structure d'un ARN se modélise classiquement de façon hiérarchique en plusieurs niveaux, d'acuité croissante : la **structure primaire** (la séquence de l'ARN, qu'on peut voir simplement comme un mot sur l'alphabet à quatre lettres $\{a,u,c,g\}$), les **structures secondaire et tertiaire** (le graphe des appariements entre les bases a,u,c,g, éléments constitutifs de ce polymère), et enfin la **structure tridimensionnelle**.



La structure primaire — *Les structures secondaire et tertiaire* — *La structure tridimensionnelle* [1]

Structure d'un ARN de transfert, modélisée classiquement de façon hiérarchique.

1. *source de l'image* : `http://www.sc.fukuoka-u.ac.jp/~bc1/Biochem/NAfig/Phe-tRNA.gif`

Comme l'observation directe ou indirecte de la structure atomique de ces grosses molécules est lourde à mettre en œuvre et que ces séquences sont en grand nombre, la prédiction informatique de la structure d'un ARN à partir de sa séquence offre un intérêt indéniable.

La prédiction de la forme tridimensionnelle d'une molécule d'ARN reviendrait à déterminer les positions relatives de ses atomes dans l'espace. Un tel niveau de précision n'est accessible que pour de très petits ARN. L'espace des configurations à explorer devient tellement grand avec la taille de la séquence que n'importe quelle approche utilise nécessairement des informations intermédiaires – structures secondaire et tertiaire – pour le restreindre par contraintes. La structure secondaire, quant à elle, contient très souvent la majeure partie des appariements, tandis qu'elle est beaucoup moins coûteuse à prédire algorithmiquement que la structure tertiaire. Les méthodes classiques se sont donc logiquement concentrées sur la prédiction de structure secondaire. Notre contribution se situe à ce niveau : nous présentons une nouvelle approche à la prédiction de la structure secondaire, qui est mise en œuvre dans l'algorithme **CARNAC** [PTD03].

La prédiction de la structure secondaire des ARN est une affaire de *contexte d'application* : les différentes méthodes de prédiction se placent plus ou moins explicitement dans des conditions différentes. Deux grandes voies ont ainsi été explorées, donnant lieu à des méthodes qui s'appliquent dans des contextes presque opposés.

La première – l'approche **thermodynamique** – s'appuie sur l'hypothèse que la molécule *repliée* est dans son état d'énergie libre minimal. Ce principe thermodynamique donne lieu assez naturellement à un algorithme qui s'applique dans le cas où l'on cherche le repliement d'une unique séquence : le programme donne la structure qui minimise l'énergie libre de l'ARN selon un modèle énergétique très élaboré, mais néanmoins simplifié par rapport à la réalité. L'algorithme utilise des paramètres énergétiques obtenus principalement à l'aide d'expériences physiques sur l'ARN par Turner *et al.* [FKJ+86, JTZ89, MSZT99]. Les deux principales implémentations sont celle de Zuker (MFOLD [Zuk03, MSZT99]) et celle du paquetage de Vienne [HFS+94]. A cause des approximations faites, en particulier sur le modèle énergétique, l'énergie libre qui est calculée pour un ARN n'est pas exactement la sienne, et l'on peut dire qu'il se retrouve ainsi non pas dans son état minimal d'énergie libre *calculée*, mais dans un état proche. De ce fait, l'algorithme de Zuker fournit pour un ARN une collection de structures *sous-optimales*, sans qu'il soit possible de savoir laquelle est éventuellement la bonne.

La seconde approche – **l'analyse comparative** – prend en considération un ensemble de séquences provenant souvent d'organismes différents, mais partageant la même fonction, et donc *a priori* la même structure. Cette structure ayant été conservée au cours de l'évolution, les mutations ont subi une pression sélective dont la méthode tire parti pour inférer la structure commune : les séquences sont alignées, et les colonnes de cet alignement sont comparées deux à deux pour déceler les mutations compensatoires, ou *covariations*. La méthode est longtemps restée manuelle ; elle a donné lieu à une implémentation à l'aide d'outils linguistiques (grammaires stochastiques hors-contexte [ED94, DEKM98]). Pour l'appliquer, on comprend bien qu'il faut un alignement initial de qualité. C'est la difficulté majeure de cette approche, car les séquences s'alignent relativement bien si elles n'ont pas trop muté, or la méthode tire son information des mutations. Il faudra donc en général un grand nombre de séquences pour disposer d'assez d'informations mutationnelles.

En réalité, le problème de la prédiction de structure secondaire supporte mal une dichotomie aussi franche car les nouvelles séquences qu'on découvre ont cette particularité de ne pas être seules mais d'être assez souvent peu nombreuses dans leur famille. Or l'implémentation de la première approche souffre de ne pas s'étendre au cas où l'on aurait à notre disposition plusieurs séquences sans augmenter de manière drastique la complexité [San85], et la seconde méthode nécessite par contre un grand nombre de séquences. C'est pourquoi de nouvelles voies ont été explorées récemment pour prédire la structure dans ce contexte [JW99, KH99, TCGS98, CLM00].

CARNAC, la méthode que nous proposons, s'inscrit dans cette lignée nouvelle en procèdant des deux approches *classiques*, puisqu'elle utilise à la fois des informations thermodynamiques et des informations mutationnelles.

Sankoff a proposé il y a une vingtaine d'années une extension des récurrences sous-jacentes à l'algorithme de Zuker [San85] de façon à étendre celui-ci à deux séquences, mais cela conduit à un algorithme $O(n^4)$ en espace et $O(n^6)$ en temps, qui est inapplicable à la plupart des ARN. Plusieurs adaptations en ont été faites, qui s'appliquent à des cas particuliers [CM94, BMR95, GHS97, MT02, WZ99].

L'algorithme que nous proposons en est aussi une adaptation selon deux directions : d'une part nous replions des tiges au lieu d'apparier des nucléotides, d'autre part l'algorithme s'applique à un nombre quelconque de séquences dont il cherche la structure commune. Le partipris de CARNAC est de *ne pas surprédire* mais de donner des résultats dans lesquels on peut avoir confiance, quitte à manquer une partie de la structure. En particulier CARNAC ne prédit rien quand il n'y a pas de structure partagée, ce qui lui confère un pouvoir de *détection*. CARNAC se place dans le contexte moderne, où l'on a peu de séquences : l'algorithme est opérationnel dès que l'on a deux séquences, et la qualité des prédictions s'affine bien sûr lorsqu'on dispose d'un jeu plus important. Il prend en entrée une famille de quelques séquences *non alignées* – moins d'une dizaine jusqu'à plusieurs dizaines – dont la longueur doit être globalement homogène, mais peut varier de quelques dizaines de bases à quelques milliers.

Plan de lecture

Nous avons construit ce document en quatre parties.

Au **premier chapitre** nous rappelons les fondements biologiques nécessaires pour appréhender le problème de la prédiction de la structure des ARN. Nous parlons principalement de l'ARN en replaçant la molécule dans son environnement habituel – la cellule – et nous définissons la modélisation de sa structure. Dans les chapitres suivants, nous nous plaçons presque exclusivement au niveau de la *structure secondaire* de l'ARN, c'est pourquoi nous donnons dès ce chapitre les outils utiles à sa manipulation.

Au **chapitre 2**, nous posons le problème de la prédiction de structure. Nous expliquons rapidement pourquoi les algorithmes de prédiction se concentrent majoritairement sur la structure secondaire et tertiaire, puis nous présentons les deux grandes approches : l'approche thermodynamique, mise en œuvre dans l'algorithme de Zuker, et l'analyse comparative. Ces deux angles d'attaque nous amènent aux méthodes que nous nommons **hybrides** puisqu'elles utilisent les deux types d'information. En donnant une description de ces méthodes, nous ne

prétendons pas être exhaustifs, mais nous cherchons plutôt à montrer la diversité des approches, et à positionner ainsi la notre.

Le **chapitre 3** est consacré à la description de CARNAC, notre contribution au problème de la prédiction de structure secondaire. L'exposé est guidé par un exemple concret – des séquences de RNases P. Nous partons du constat que les deux approches classiques ne fonctionnent pas sur cet exemple, puis nous présentons la notre sur ces deux séquences, mise en œuvre dans l'algorithme CARNAC_2. Nous construisons un jeu de tiges potentielles pour chaque séquence en utilisant les paramètres d'énergie de Turner *et al.* [FKJ+86, JTZ89, MSZT99] et en tenant compte des mutations compensatoires. Le repliement commun s'obtient par l'algorithme de Sankoff appliqué aux tiges. Enfin nous décrivons le programme final CARNAC, qui s'applique à un nombre quelconque de séquences. Dans son principe, il fonctionne un peu à la manière de CLUSTALW [THG94] pour l'alignement multiple en formant un tournoi où les séquences sont repliées deux à deux par CARNAC_2 et en combinant ensuite ces repliements.

Nous présentons au **chapitre 4** les résultats obtenus avec CARNAC. À cette occasion nous les comparons avec ceux des autres méthodes (classiques et hybrides). Nous revenons d'abord sur l'exemple qui nous a concerné au chapitre précédent (RNases P), puis nous examinons le comportement de CARNAC et des autres méthodes sur un jeu d'ARN non-codants qui s'alignent particulièrement mal (Telomerase-cil). Nous illustrons le caractère universel de CARNAC en l'appliquant aux deux grandes familles bien connues d'ARN (ARN de transfert, ARN ribosomiques). Enfin nous montrons sur un ensemble de séquences d'ARN messagers et d'ARN viraux sa capacité à détecter la présence ou l'absence de structure.

Chapitre 1

L'ARN structuré

Dans chacune des cellules d'un organisme vivant, il existe trois grandes familles de macromolécules qui coopèrent et interagissent en se répartissant les rôles de manière spécifique : l'ADN, l'ARN et les protéines. L'ADN *stocke* l'information génétique, qui sera transmise fidèlement au cours des générations cellulaires, les protéines *expriment* cette information principalement au travers de leur structure en assurant la plupart des tâches nécessaires au bon fonctionnement de l'organisme, et l'ARN joue les ambassadeurs entre ADN et protéines.

En réalité les mandats de chacun ne sont pas si clairement définis, et certains ARN endossent les missions d'une protéine en adoptant eux-même une structure. Ce sont ces familles d'ARN, qu'on découvre de plus en plus nombreuses, qui vont nous concerner dans toute la suite.

La structure de tels types de grosses molécules, évidemment fort complexe, est classiquement modélisée de façon hiérarchique en différents "paliers", correspondant à des niveaux de précision croissants. C'est plus particulièrement le second niveau de description – la *structure secondaire* – qui nous intéressera.

Les notions que nous développons dans ce chapitre sont volontairement simplifiées, il ne s'agit pas d'un cours de biologie. Nous nous attachons plutôt à mettre en valeur le caractère spécifique de notre travail et à le replacer par rapport à un cadre plus large, tout en apportant sur le sujet le regard transverse d'un informaticien converti aux problèmes biologiques.

1.1 La vie cellulaire

Tous les phénomènes que nous allons décrire et étudier ont pour théâtre la cellule, qui est l'entité élémentaire de construction des êtres vivants. Il existe deux grandes familles d'organismes vivants, selon que leurs cellules comportent un noyau (les **eucaryotes**, FIG. 1.1) ou non (les **procaryotes**). Les mammifères (l'Homme en particulier) sont des eucaryotes. Les bactéries sont des procaryotes. L'information génétique est stockée de manière identique dans chaque cellule sous la forme de polymères connus sous le nom d'acides nucléiques. Les *acides nucléiques* existent sous deux formes, *l'acide désoxyribonucléique* ou **ADN**, et *l'acide ribonucléique* ou **ARN**.

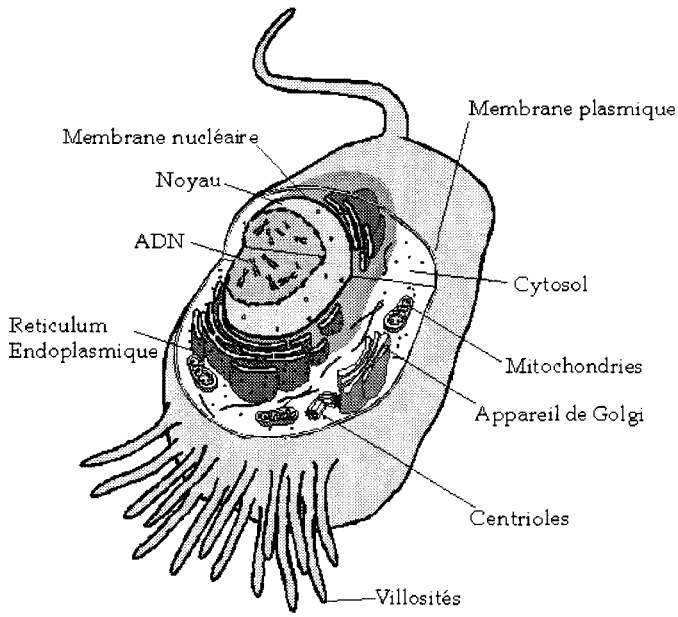


FIG. 1.1 – *Dessin schématique d'une cellule eucaryote*[2].

Le fait que l'ADN soit le détenteur de l'information génétique a été mis en évidence dans les années 40 par Oswald Avery. Le **dogme central** (FIG. 1.2), finalement proposé par Francis Crick à la fin des années 50, formalise la transformation de l'information génétique en molécules fonctionnelles. Cette information transite majoritairement de l'ADN, dépositaire du code génétique, aux protéines, qui sont les constituants élémentaires servant au fonctionnement de la cellule et de l'organisme entier. L'ARN joue dans ce schéma le rôle d'une molécule transitoire assurant la migration du code génétique vers une machine de traduction en protéines.

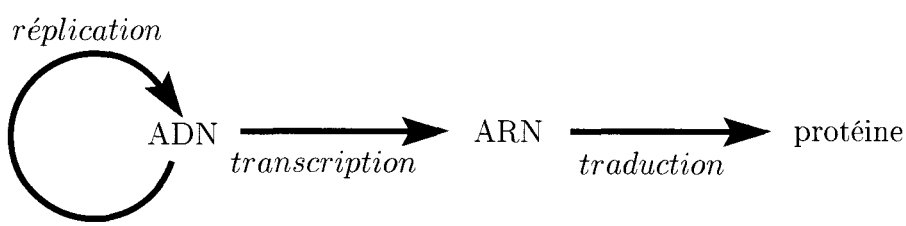


FIG. 1.2 – *Le Dogme Central.*

Dans la suite, nous parlons d'abord brièvement de la constitution des acides nucléiques, et donc en particulier de l'ADN, puis des protéines. Enfin nous examinons quel est le statut précis de l'ARN dans cette machinerie. Nous verrons alors qu'il existe différentes familles d'ARN dont les mécanismes de fonctionnement ne sont pas les mêmes.

2. *source de l'image :* `http://perso.wanadoo.fr/fguillonneau/these/Image101b.gif`

1.1.1 Les acides nucléiques

Pour coder l'information, l'ADN utilise 4 types de bases : l'**adénine** et la **guanine** (bases *puriques*), la **thymine** et la **cytosine** (bases *pyrimidiques*). L'ARN contient de l'**uracile** au lieu de la thymine mais ces deux bases sont équivalentes au point de vue de l'information qu'elles contiennent.

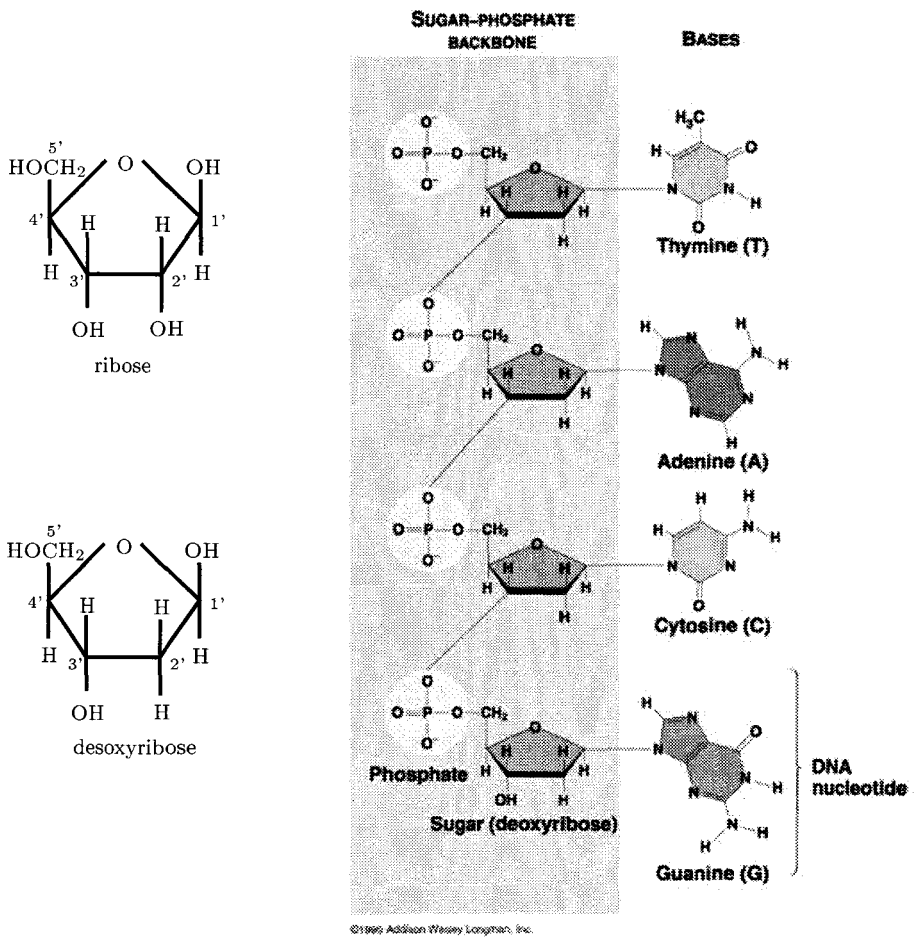


FIG. 1.3 – *Schéma du ribose et du désoxyribose, la composition des nucléotides et leur assemblage caténaire*[3].

La chaîne principale des deux types d'acides nucléiques est composée de sucres (FIG. 1.3) – les riboses pour l'ARN et les désoxyriboses pour l'ADN – liés entre eux par des phosphates. Au ribose et au désoxyribose, invariants dans la chaîne, sont attachés au carbone en position 1′ des bases puriques ou pyrimidiques, qui sont les éléments porteurs d'information. Le sucre, la base et le groupement phosphate constituent ensemble un **nucléotide**, la brique élémentaire qui sert à la construction des acides nucléiques. L'assemblage caténaire des nucléotides est identique chez tous les êtres vivants, c'est leur succession qui varie. Pour caractériser un

3. *source de l'image :* `http://fajerpc.magnet.fsu.edu/Education/2010/Lectures/24_dna.htm`

morceau d'ADN ou d'ARN, il suffit ainsi de donner la séquence des lettres A, C, G, T (U pour l'ARN). D'un point de vue purement informatique, cela revient à considérer l'objet comme un *mot*, ou un ensemble de mots sur l'alphabet à quatre lettres {A, C, G, T/U}.

Les phosphates qui lient entre eux les riboses de l'ARN, ou les désoxyriboses de l'ADN, relient le carbone 3′ d'un sucre au carbone 5′ du suivant, ce qui donne une polarité à la chaîne, et force un sens de lecture à la machinerie chargée de décoder l'information.

Pour assurer à l'ADN une capacité d'auto-reproduction et d'auto-correction, deux chaînes *complémentaires* sont associées. Les bases puriques et pyrimidiques s'assemblent en effet de façon complémentaire et exclusive de la manière suivante (FIG. 1.4) :

$$\boxed{A = T} \text{ ou } \boxed{T = A} \qquad \boxed{G \equiv C} \text{ ou } \boxed{C \equiv G}$$

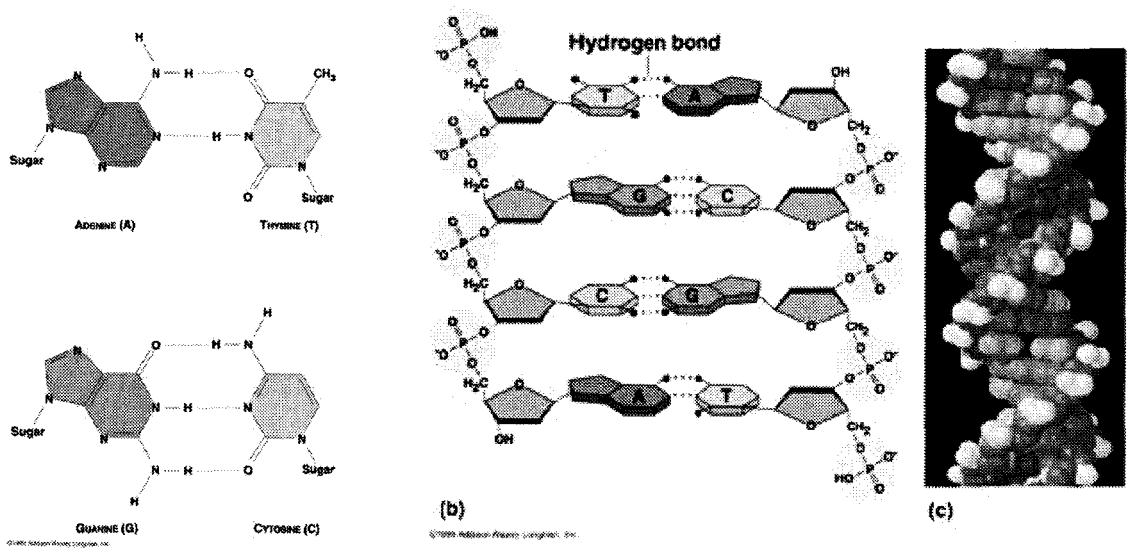


FIG. 1.4 – *Structure en double hélice de l'ADN, appariements de type Watson-Crick*[4].

Ces appariements, mis en évidence par Watson et Crick portent naturellement le nom de **Watson-Crick**. Il ne s'agit pas d'une liaison covalente (liaison atomique forte), mais d'une interaction faible (deux ponts Hydrogène entre l'adénine et la thymine, trois ponts entre la cytosine et la guanine). Cette faiblesse est toute relative car les appariements se forment les uns à la suite des autres, ce qui renforce leur cohérence tout en autorisant leur dégrafage. L'assemblage se vrille pour former une hélice, qu'on appelle communément la **double hélice**, en référence aux deux chaînes complémentaires de l'ADN. Les contraintes géométriques font que l'appariement de deux chaînes de nucléotides entre elles ne peut se faire que dans une position antiparallèle, où les polarités des deux chaînes sont inverses l'une de l'autre, le code de l'une étant le complément inverse de celle de l'autre :

5′ GCGAATCAGAGACACCCACAGCCGGGTGAACAA 3′
3′ CGCTTAGTCTCTGTGGGTGTCGGCCCACTTGTT 5′

4. *source des images :* `http://fajerpc.magnet.fsu.edu/Education/2010/Lectures/24_dna.htm`

Lu dans la direction habituelle $5' \rightarrow 3'$, le brin d'ADN inférieur aurait la séquence

TTGTTCACCCGGCTGTGGGTGTCTCTGATTCGC.

Il existe trois types de conformations pour la double hélice de l'ADN, dénommées A, B, et Z, la configuration habituelle étant la B. La double hélice n'est pas parfaite (elle présente des « courbures et des faux-plis », selon l'expression de Watson) puisqu'elle est perturbée par l'encombrement inégal des bases, qui ont par conséquent une influence sur la flexibilité et la régularité de l'hélice.

La structure de l'ADN intervient évidemment dans les mécanismes de la transcription et de la réplication, néanmoins elle est assez mal connue. Pour simplifier on pourrait dire qu'il s'agit d'une structure compactée de façon hiérarchique qui constitue un moyen efficace de stocker de manière dense les informations encodées, sachant qu'il s'agira de déplier l'empaquetage pour pouvoir y accéder.

Le patrimoine génétique d'une cellule – le **génome** – peut être composé de plusieurs empaquetages de molécules d'ADN (les chromosomes). La taille des génomes se compte usuellement en millions ou en milliards de bases (celle du génome humain est d'environ trois milliards de bases).

C'est au début des années 80 que les premières banques de séquences sont apparues, mais les techniques de séquençage étant de plus en plus performantes, les banques ont grandi de façon exponentielle et contiennent à l'heure actuelle plusieurs dizaines de milliards de bases. Les trois plus grosses banques – l'EMBL hébergée à Heidelberg puis Cambridge au sein de l'EBI (European Bioinformatics Institute), Genbank au NIH (National Institute of Health, Etats-Unis) diffusée par le NCBI (National Center for Biotechnology Information) et la DDBJ (DNA Data Bank, Japon) – ont finalement donné naissance en 1990 à un format unique pour la description et l'annotation des séquences.

1.1.2 Les protéines

Les protéines sont des polymères d'*acides aminés*. Ce sont les composants actifs des systèmes biologiques, assurant les fonctions les plus diverses dans les organismes vivants : constitution des tissus, transport transmembranaire des solutés, régulation de la transcription de l'ADN, enzymes, reconnaissance de molécules étrangères, etc.

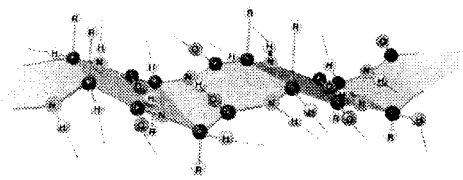
Comme la constitution des protéines n'utilise que vingt types différents d'acides aminés, une protéine peut être symbolisée par un mot sur cet alphabet à vingt lettres. La longueur d'une protéine varie habituellement de 50 à 1000 acides aminés.

L'ARN transcrit de l'ADN a pour fonction principale de spécifier la protéine à synthétiser. Le code à déchiffrer comporte quatre lettres, correspondant aux nucléotides A, G, C, U, alors que sa traduction en compte vingt, les acides aminés qui composent toutes les protéines. Le nombre minimum de nucléotides nécessaires pour spécifier un acide aminé est donc de trois (deux lettres tirées d'un alphabet de quatre ne peuvent coder que seize combinaisons différentes). Avec trois nucléotides, le nombre est de soixante quatre, ce qui implique une redondance. Plusieurs triplets de nucléotides (ou **codons**) spécifient ainsi le même acide aminé ce qui permet d'incorporer aussi une ponctuation (START, STOP), nécessaire à la bonne lecture du code.

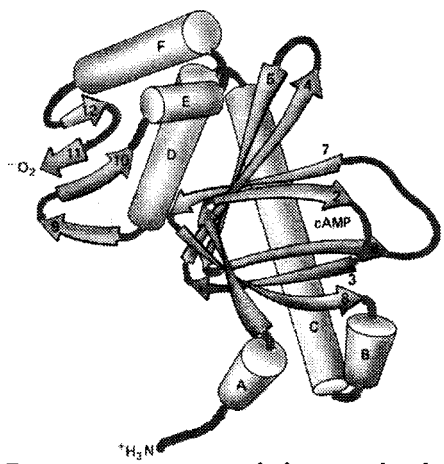
Selon les conditions de pH et de température du solvant, une protéine peut se trouver dans son état *dénaturé* lorsque la chaîne est dépliée, ou dans son état *natif*, lorsque la chaîne est repliée en globule. C'est dans son état natif que la protéine est fonctionnelle. Sa fonction au sein du système dépend de la configuration spatiale qu'elle adopte.

```
SIRNNPVQNK LMNGEDPINN NHAQTAGGLG
GEVEILGFKM PTEERMRKRK ESNRESARRS
ENSCLLRRIA ALNQKYNDAN VDNRVLRADM
SSSVPLSMPI SAPTPSSDVP VQVPVPPPIR
GFLRLQAQEP ASMVVGATLS ATEMNR
```

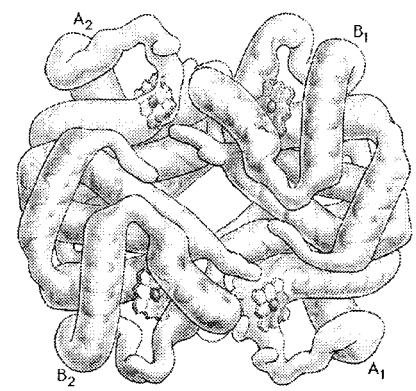
La structure primaire est la séquence chimique des acides aminés.



La structure secondaire correspond grosso modo à une description modulaire de différents assemblages *classiques* d'acides aminés (hélices, feuillets, boucles) Ici un feuillet β.



La structure tertiaire est la donnée du graphe des interactions entre les modules secondaires.



La structure quaternaire est l'éventuel agencement de plusieurs chaînes polypeptidiques.

La structure tridimentionnelle ou **stérique** est la forme caractéristique de la protéine dans l'espace 3D.

FIG. 1.5 – *Gradation de la structure des protéines en différents niveaux de description*[5].

Le *problème du repliement des protéines* est celui de la prédiction de leur structure spatiale à partir de leur séquence d'acides aminés. Différentes forces guident ce repliement : attraction ionique, ponts Hydrogène, ponts disulfures, forces hydrophobes. A chaque étape, les forces en jeu ont un rôle et une importance différentes. Les ponts disulfure sont des liaisons covalentes,

5. *source des images :* `http://lectures.molgen.mpg.de/ProteinStructure/Levels/betasheet.gif`, `http://academic.brooklyn.cuny.edu/biology/bio4fv/page/tertie.gif` *et* `http://www.agen.ufl.edu/~chyn/age2062/lect/lect_02/3_30.gif`

ils ont la même force de cohésion que les interactions de la chaîne peptidique. Les forces hydrophobes et les liaisons H sont des liaisons beaucoup plus faibles, mais ce sont elles qui guident et stabilisent le repliement. Les acides aminés qui ont des propriétés hydrophobes auront tendance à se concentrer au cœur de la molécule afin de minimiser les interactions avec le solvant. Tel résidu sera finalement accessible, tel autre sera enfoui, tel assemblage de résidus concoctera une forme spatiale avec des potentiels d'attraction (forces électriques, hydrophiles, hydrophobes, etc.) qui lui permettront de reconnaître une cible éventuelle, ou d'avoir des propriétés catalytiques.

Afin de modéliser ce repliement, on distingue différents niveaux d'organisation dans la structure des protéines (FIG. 1.5) :

- la structure primaire est la séquence,
- la structure secondaire est une description modulaire d'assemblages d'acides aminés,
- la structure tertiaire décrit les interactions entre les éléments de structure secondaire,
- la structure quaternaire est l'éventuel agencement de plusieurs chaînes polypeptidiques,
- la structure 3D est la forme de la protéine dans l'espace.

La prédiction de structure des protéines est un des domaines de recherche les plus anciens en bioinformatique. Dès les années 60, les premières structures de protéines obtenues par cristallographie étaient étudiées en détail pour comprendre les mécanismes de leur formation. Le domaine a toujours été très actif, d'autant plus à l'heure actuelle, où les séquences sont aisément disponibles.

1.1.3 L'ARN

Le Dogme Central fait apparaître l'ARN comme une molécule intermédiaire dans la traduction de l'ADN vers les protéines. La molécule d'ARN est constituée d'une seule chaîne, parfois associée à des protéines pour participer à de grands assemblages, comme le ribosome par exemple. En réalité, comme on le voit sur la FIG. 1.6, trois grandes familles distinctes d'ARN participent aux mécanismes de la traduction en tenant chacune une fonction précise : les ARN messagers (**ARNm**) sont la transcription d'un morceau d'ADN, les ARN ribosomiques (**ARNr**) des constituants du ribosome, et les ARN de transfert (**ARNt**) ont pour rôle d'amener l'acide aminé spécifique à chaque codon.

Lors de la traduction, se met en place sur l'ARNm une machine qui constitue une sorte de tête de lecture coulissante : le *ribosome*. Les ARNt assurent la traduction en amenant à chaque fois l'acide aminé correspondant au codon (voir page 9) qui le spécifie, et la chaîne polypeptidique s'allonge au fur et à mesure que le ribosome progresse sur l'ARNm.

L'ARN messager, copie fidèle du génome à sa synthèse, subit une série de modifications avant de devenir une molécule efficiente. Chez les procaryotes, ces modifications sont relativement mineures. Chez les eucaryotes par contre, pratiquement tous les types d'ARN sont modifiés après leur transcription. Il subissent une **maturation** dans le noyau qui dure environ 30 minutes et se fait en plusieurs étapes : adjonction de nucléotides et épissage. Lorsqu'on parlera par la suite d'ARNm, il s'agira toujours d'ARNm **mature**, sauf mention contraire.

L'adjonction d'une « coiffe » (un nucléotide modifié) en 5′ est une modification qui permet aux mécanismes de transport et de traduction de reconnaître l'ARNm des autres formes

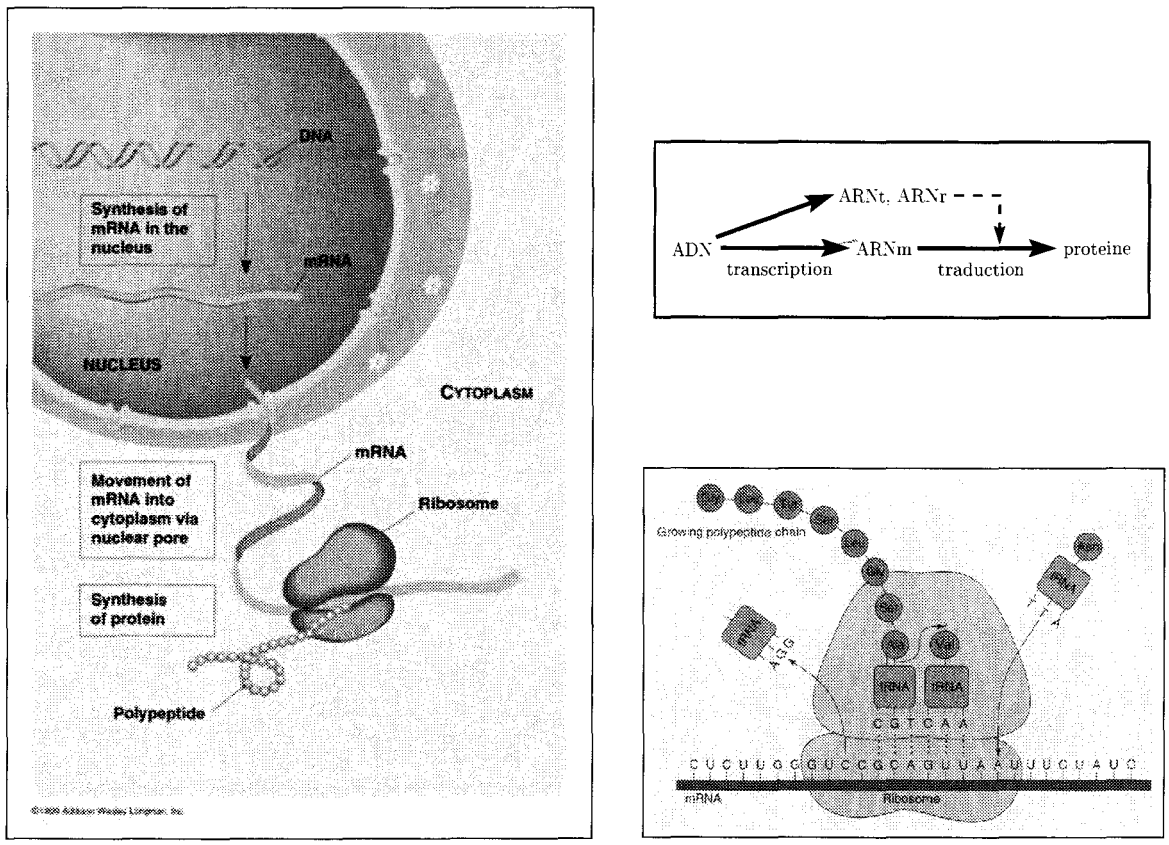


FIG. 1.6 – *L'ARN apparaissant comme molécule intermédiaire dans la traduction de l'ADN vers les protéines. Chez les eucaryotes, l'épissage et les adjonctions de nucléotides en 3′ et 5′ ont lieu dans le noyau, puis l'ARNm est expulsé dans le cytoplasme pour y être traduit*[6].

d'ARN. Il est particulièrement important pour la liaison des ribosomes assurant le processus de traduction.

La polyadénylation (ajout en 3′ d'un polymère d'adénosine d'environ 250 nucléotides) marque de façon précise l'extrémité 3′. La place où la « queue poly"A" » doit être ajoutée est marquée par la séquence spécifique AAUAAA. Il semble aussi que la dégradation progressive du poly"A" dans le cytoplasme permette de déterminer l'âge d'un ARNm et de le détruire en temps voulu.

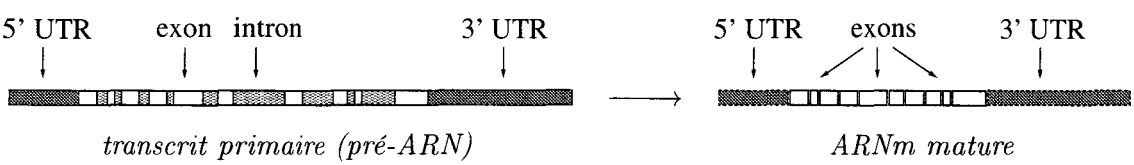


FIG. 1.7 – *Epissage du transcrit primaire pour former un ARN mature.*

6. *source des images :* `library.tedankara.k12.tr/chemistry/vol1/biochem/trans100.ht` *et* `http://fajerpc.magnet.fsu.edu/Education/2010/Lectures/26_DNA_Transcription_files/image006.jpg`

La modification la plus importante chez les eucaryotes est l'**épissage** (FIG. 1.7) qui élimine une grande partie de l'ARN initialement transcrit et reconstitue sa partie codante à partir de séquences discontinues. L'épissage est réalisé par le complexe ribonucléoprotéique (cxRNP) splicéosome. L'ARNm **mature** est formé par la concaténation de sous-segments du transcrit primaire (les **exons**) qui forment l'**ORF** (Open Reading Frame : cadre ouvert de lecture). Il reste aussi en 5′ et en 3′ des parties non traduites (les **UTR**, UnTranslated Regions). Les pièces entre les exons (les **introns**) sont dégradées.

Le nombre et la taille des introns augmentent avec la complexité de l'espèce et sa position dans l'arbre de l'évolution. Le partitionnement des gènes en exons séparés semble être un élément important dans l'évolution, dans la mesure où il permet la formation de nouveaux gènes à partir d'exons issus de gènes différents.

La partie 5′ est transcrite en premier. Dès qu'elle est débarassée des volumineux splicéosomes, elle peut commencer à quitter le noyau vers le cytoplasme afin d'être traduite. Les ARNm, une fois dans le cytoplasme, sont relativement stables (leur demi-vie est de l'ordre d'une dizaine d'heures) sauf quelques ARNm codant pour des protéines régulatrices.

Les ARN de transfert sont chacun spécifiques à un **codon**. Le mécanisme de décodage implique, pour chaque codon se trouvant sur l'ARNm, un appariement complémentaire avec l'**anticodon** se trouvant à une extrémité de l'ARNt. Cet appariement de trois paires de bases est guidé par le ribosome. A l'autre extrémité de l'ARNt (5′-terminale) se trouve un acide aminé attaché de façon covalente. Une réaction de transestérification attache l'acide aminé amené par le dernier ARNt à celui amené par l'avant-dernier, allongeant ainsi la chaîne protéique d'une unité.

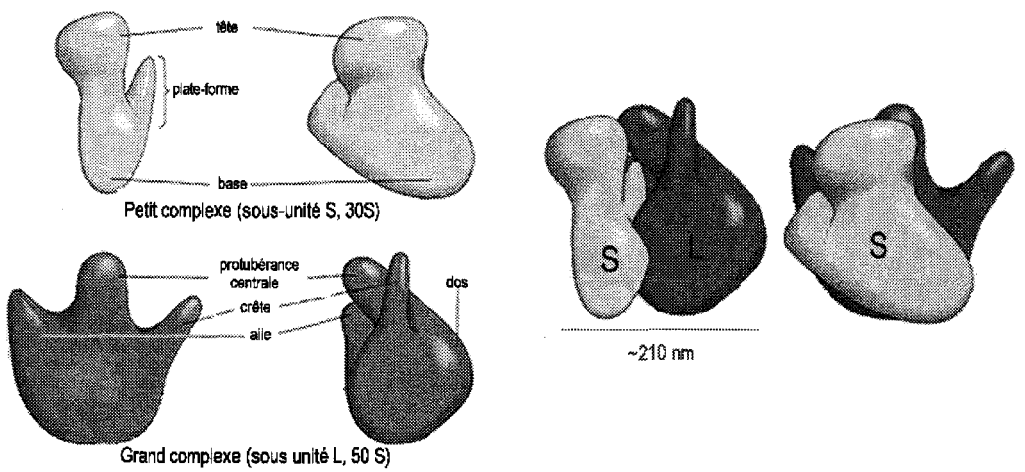


FIG. 1.8 – *Allure et assemblage des deux sous-unités ribosomiques*[7].

Le ribosome est une structure extrêmement complexe. Nous ne connaissons son architecture précise que depuis quelques années [BNH+00, WBC+00]. Auparavant, seule sa forme générale et l'identité de ses composants étaient connus. Plusieurs ARN différents, dont les noms correspondent à leur constante de sédimentation (5S, 5.8S, 16S, 18S, 23S, 28S), en

7. *source de l'image :* `http://ntri.tamuk.edu/cell/ribosomes.html`

forment l'échafaudage et jouent un rôle essentiel dans les propriétés du ribosome. Le gros de la masse est formé de protéines, qui avec les ARN correspondants forment deux sous-unités, séparées au moment de l'initiation de la traduction, mais toutes deux nécessaires à sa progression (FIG. 1.8). Une vue exacte de la forme tridimensionnelle des ribosomes déduite par analyse d'images en microscopie électronique montre plusieurs protubérances associées avec la région où coulisse l'ARNm pendant la traduction et un « tunnel » à travers lequel passe la chaîne protéique nouvellement synthétisée. Il existe aussi deux sites de liaison de l'ARNt, occupés l'un après l'autre pendant la traduction.

À l'origine, un monde fait d'ARN?... Les acides nucléiques présentent des capacités de stockage ; les protéines des propriétés enzymatiques. Depuis la découverte en 1981 par Cech et Altman du premier ARN autocatalytique, on sait que l'ARN cumule ces deux fonctionalités. Il s'agissait d'un intron d'un ARN ribosomique : la coupure (excision), suivie d'une ligature (épissage) est réalisée par l'ARN lui-même, d'où sa dénomination de «ribozyme», qui évoque à la fois son origine ribosomique et ses potentialités enzymatiques. De là a germé l'idée d'un monde primordial constitué d'ARN uniquement, capable de stocker l'information génétique, mais aussi de réaliser les réactions enzymatiques nécessaires à la vie. Au cours de l'évolution, ces fonctions seraient devenues bien distinctes. L'ADN est en effet beaucoup plus stable que l'ARN, et sa structure en double hélice lui confère des pouvoirs correcteurs d'erreurs, car l'information y est doublement codée. Les protéines, quant à elles, offrent des possibilités beaucoup plus larges en terme de structures et d'activités catalytiques.

1.2 La structure de l'ARN

Les protéines sont par excellence les molécules *fonctionnelles* de l'organisme. Mais certains ARN sont eux aussi directement fonctionnels, les ARN de transfert et ribosomique dont nous avons parlé, ainsi que de nombreuses autres familles dont nous parlerons ensuite. Leur fonction, comme pour les protéines, est liée à leur structure, qui peut elle-aussi être décrite de façon hiérarchique : structure primaire, secondaire, tertiaire, et tridimensionelle. Nous examinerons plus particulièrement la structure secondaire de l'ARN car c'est elle qui nous concernera par la suite.

1.2.1 Définition hiérarchique de la structure

La modélisation et l'analyse de la structure de l'ARN peut être hiérarchisée en plusieurs niveaux d'acuité croissante : **la structure primaire** est la structure de plus bas niveau, l'ARN vu comme un mot sur l'alphabet {A, U, C, G}. Les niveaux de structure de degrés supérieurs vont s'attacher à décrire les interactions qui se forment entre les bases.

La force de cohésion principale dans le repliement de l'ARN est la liaison Hydrogène. Elle met en jeu une énergie de l'ordre de 20 à 25 kJ/mol, soit environ dix à quinze fois moins qu'une liaison covalente (mais du même ordre de grandeur que les autres liaisons non covalentes). Cependant elle se distingue par son caractère directionnel très fort, qui permet les couplages spécifiques de **Watson-Crick** : les bases A et U peuvent former un appariement avec deux ponts Hydrogène $A = U$ et les bases C et G avec trois ponts Hydrogène $G \equiv C$.

L'appariement **wobble** ("bancal") $G = U$ est assez commun pour être considéré au même rang qu'un appariement de type Watson-Crick. Ces trois types d'appariements sont généralement appelés **canoniques**. Comme pour les protéines, ce sont les forces hydrophobes qui dirigent le repliement de l'ARN [SS99].

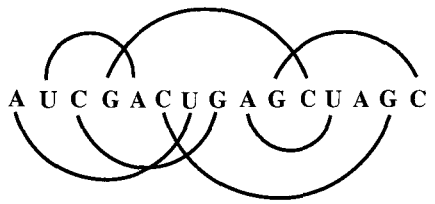


FIG. 1.9 – *Le graphe des appariements de type Watson-Crick et wobble constitue la première étape dans la description de la structure tridimensionelle de l'ARN.*

Le schéma de la FIG. 1.9 montre un graphe d'appariements canoniques qui pourraient se former sur une courte séquence d'ARN. Dans la réalité, les appariements ne se forment pas de façon *débridée* mais ils ont tendance à s'empiler pour renforcer la cohésion de l'ensemble. Ces empilements de paires de bases forment ce qu'on appelle des **tiges** (FIG. 1.10). Ces tiges se vrillent dans l'espace en une hélice de type A (11 paires par tours, comparée à la conformation habituelle de l'ADN en hélice de type B, qui comporte 10 paires par tour).

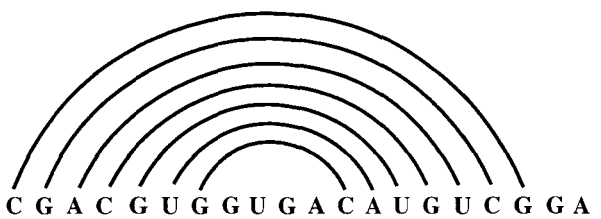


FIG. 1.10 – *Empilement des appariements pour former des tiges.*

La structure secondaire d'une molécule d'ARN est l'ensemble des tiges qui se forment, pourvu qu'elles respectent les contraintes suivantes : on impose qu'elles soient *emboîtées* ou *juxtaposées*, comme sur la FIG. 1.11. Dans la formation des tiges il peut malgré tout survenir des croisements (FIG. 1.12). De tels croisements sont appelés des **pseudonœuds** car jusqu'à présent jamais aucun nœud n'a été observé sur des molécules d'ARN. Par ailleurs le diagramme ne permet pas de distinguer si le squelette de la molécule est noué ou non. A cause de sa structure en hélice de type A (11 paires par tour), l'ARN ne peut pas former des pseudonœuds dont les tiges excèdent 9 à 10 paires de bases [DK02]. Cette contrainte restreint considérablement leurs occurences.

L'empilement des appariements pour former des tiges donne à la structure secondaire un caractère modulaire simple qui permet de la décrire comme un assemblage de différentes structures élémentaires.

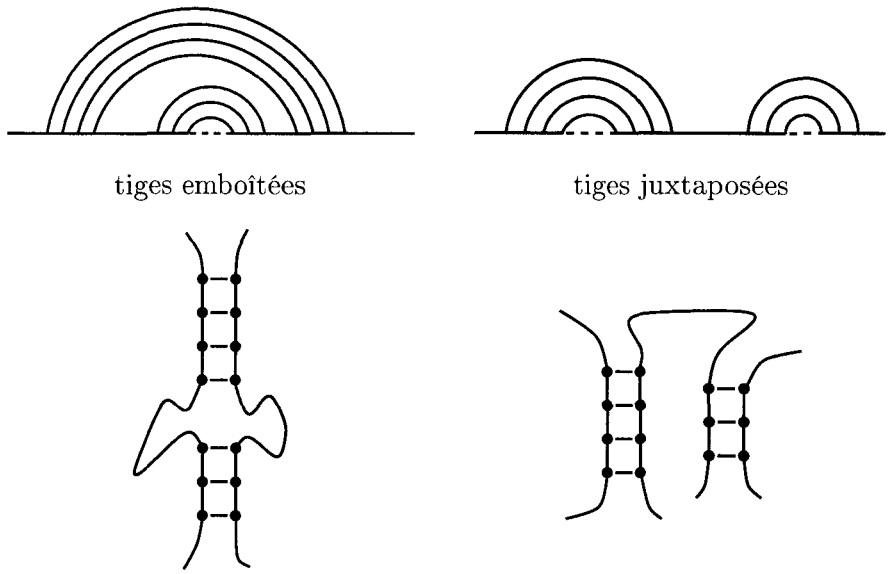


FIG. 1.11 – *La structure secondaire est constituée de tiges emboîtées ou juxtaposées.*

La structure tertiaire est le graphe de tous les appariements réalisés entre les bases. Lors de l'apparition d'un pseudonœud, seule une des deux tiges (a priori la tige la plus stable) appartient à la structure secondaire. Les pseudonœuds constituent ainsi des motifs intermédiaires entre la structure secondaire et la structure tertiaire. Les appariements énumérés par la structure tertiaire contiennent donc d'abord tous les appariements de la structure secondaire, puis les tiges en conflit avec la définition de structure secondaire (par exemple la seconde tige

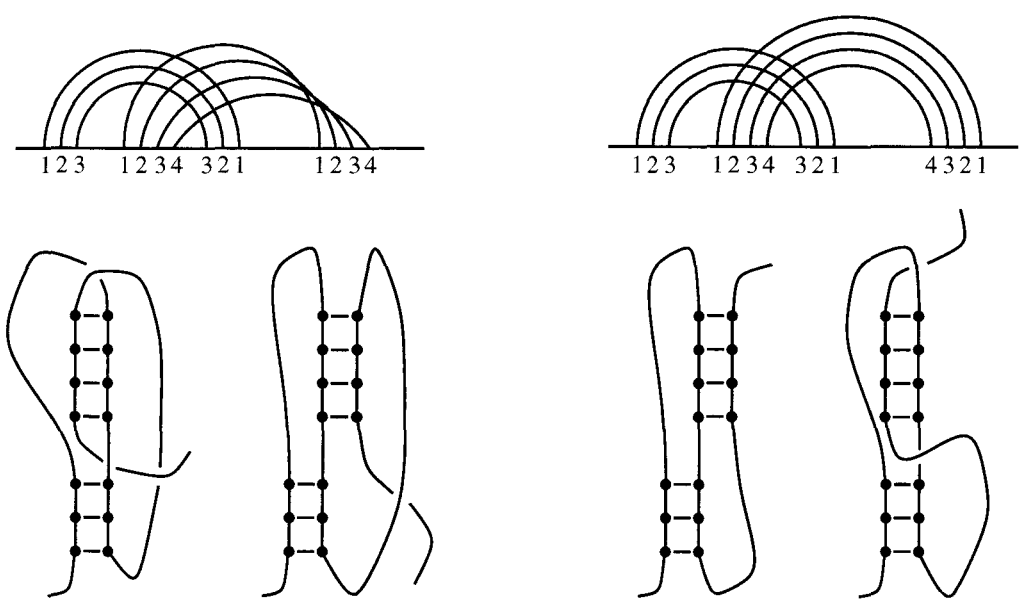


FIG. 1.12 – *Pseudonœuds.*

d'un pseudonœud) ainsi que d'autres types d'appariements : des appariements non canoniques ou isolés (ne formant pas une tige), des triplets de bases, des quartets.

La structure stérique, ou 3D est finalement la forme de la molécule dans l'espace, c'est-à-dire la donnée des positions relatives des atomes. La prédiction informatique *ab initio* de la structure 3D est à l'heure actuelle totalement hors de portée pour autre chose que des oligo-ARN (ARN de quelques bases).

1.2.2 Interactions non canoniques et motifs

Bien que les définitions de structures secondaire et tertiaire spécifient l'intégralité des liaisons faibles, elles ne sont pas parfaitement adaptées pour décrire toutes les subtilités de ces interactions. Le graphe secondaire ou tertiaire ne contient en effet que l'information de l'appariement, or les interactions, surtout celles qui sont non canoniques, peuvent revêtir des configurations assez variées dont l'étude exhaustive et la modélisation n'est pas encore achevée. Il n'existe pas encore de classification qui permette d'intégrer avec bénéfice ces informations à la définition hiérarchique de la structure qu'on vient de décrire.

Il est probable que lorsqu'elle sera assez conséquente, une telle classification amènera à repenser différemment le caractère hiérarchique de la structure de l'ARN tel qu'on vient de le décrire, au profit d'un caractère plus modulaire. La notion et la définition de **motif** se précisent à mesure qu'on en découvre, et des programmes de recherche de motifs sont développés en ce sens [MEG+01].

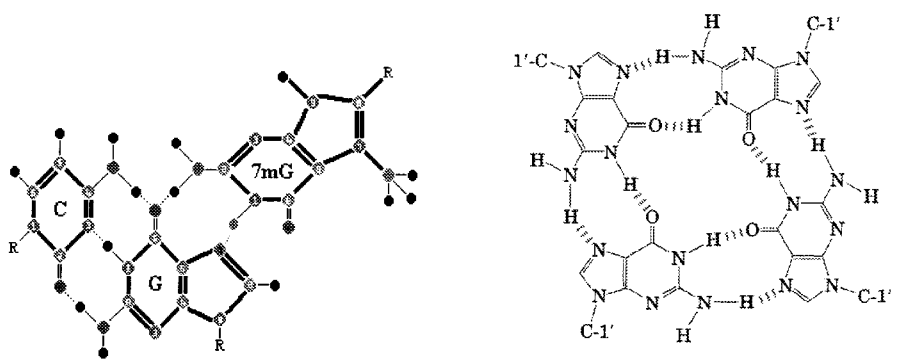


FIG. 1.13 – *Triplet (faisant intervenir des appariements de type Hoogsteen) et G-quartet*[8].

On trouve cependant des motifs assez courants pour avoir été bien étudiés [Moo99]. Les triplets, par exemple, sont des liaisons entre trois nucléotides qui font souvent intervenir des appariements de type Hoogsteen ou Reverse-Hoogsteen (FIG. 1.13) ; on observe des triplets dans les ARNt ou les introns du groupe II, et leur empilement est fréquent dans les séquences homopurines (les mots sur {A, G}), créant ainsi une *triple hélice*. Par définition, la structure tertiaire prend en charge une partie de cette description en affirmant que les trois bases

8. *source des images :* `http://www.biochimie.univ-montp2.fr/licence/interact_adn/arn/page3_img2p3.gif` *et* `http://www.imsb.au.dk/~raybrown/gtet.gif`

s'apparient, mais ne donne aucune information sur la façon dont elles s'apparient (Hoogsteen) ni sur la façon dont elles s'empilent (la structure de la triple hélice).

On trouve aussi d'autres formes d'interactions atypiques : des AA-plateformes, des G-quartets (dans les ARN constituant le télomérase par exemple), ou assez fréquemment des interactions tetraboucle-récepteur. On observe ainsi que certaines tetraboucles (GNRA et UNCG surtout[9]) apparaissent plus fréquemment, et qu'elles interagissent avec des parties libres de la molécule.

1.2.3 Détermination expérimentale de la structure

Il existe bien sûr des méthodes physico-chimiques pour observer ou déduire la structure spatiale des molécules. Ces méthodes s'appliquent en particulier à l'ARN mais sont longues et difficiles à mettre en œuvre, et souvent très coûteuses. Toutes les informations intermédiaires qu'on peut obtenir sur l'appariement ou non de tels nucléotides – c'est-à-dire principalement les informations apportées par la connaissance de la structure secondaire et tertiaire – sont d'une aide précieuse pour déterminer la structure atomique de la molécule.

Méthodes physiques – celle qui donne la plus haute résolution est la *diffraction aux rayons X*. Elle permet d'obtenir jusqu'à la structure stérique, mais il est nécessaire de cristalliser la molécule auparavant, ce qui est ardu. La molécule n'est pas inerte et rigide et de nombreuses liaisons peuvent se former, disparaître ou s'échanger lors de la cristallisation. L'ARN peut parfois adopter plusieurs conformations spatiales stables et *vibrer* autour de son état natif, en allant jusqu'à présenter des structures alternatives, concurrentes. La *RMN (Résonnance Magnétique Nucléaire)* donne des détails sur la conformation locale, mais ne peut s'appliquer qu'à des molécules de taille très réduite : on crée in vitro des oligoARN qu'on analyse par RMN et on extrapole en supposant que le repliement local est identique dans les macromolécules. On observe aussi au *microscope électronique* des ARN partiellement dénaturés mais le choix des conditions de dénaturation est crucial, et la résolution du microscope est limitée.

Méthodes chimiques – la structure de l'ARN peut être inférée en soumettant la molécule à des *attaques enzymatiques* : certaines RNases clivent préférentiellement l'ARN soit dans des zones libres, soit dans des zones hélicoïdales. L'examen, par les techniques classiques d'électrophorèse, de la taille des fragments disloqués par la sonde enzymatique donne des informations sur l'accessibilité des nucléotides et donc sur la forme spatiale de la molécule : les bases les plus enfouies ne seront pas atteintes par l'enzyme. Comme les RNases sont assez volumineuses, elles ne peuvent pas toujours atteindre leur cible, par conséquent on utilise aussi des sondes chimiques plus petites (ions métalliques, agent alkylants). On peut aussi forcer les liaisons Hydrogène à devenir covalentes en *irradiant la molécule aux UV* ; la détection des liaisons formées se fait par les techniques habituelles de séquençage.

Analyse mutationnelle – si on connait les fonctions de l'ARN qui nous concerne, on sait qu'il va s'unir avec telle protéine qui le reconnaitra de manière exclusive en se liant à certains

9. Selon la norme IUPAC, par convention N désigne un nucléotide quelconque, R une purine, et Y une pyrimidine.

sites à cause de leur forme spatiale. Le fait de provoquer intentionellement certaines mutations peut modifier la structure et empêcher l'ARN d'être reconnu. En itérant les expériences, on peut obtenir des informations par déduction sur la structure de la molécule [Rij95].

1.3 Des nouveaux jeux de séquences

La molécule d'ARN est sortie de l'ombre depuis une vingtaine d'années. A l'époque l'ARN messager était un intermédiaire passif, l'ARN de transfert un petit adaptateur, l'ARN ribosomique une simple armature. Mais la découverte des ribozymes au début des années 80 et de nouveaux ARN non codants depuis, a fait germer un regain d'intérêt pour l'ARN, tout en amenant de nouveaux problèmes.

1.3.1 Familles d'ARN non-codants

Le séquençage massif de nombreux génomes et les progrès dans notre compréhension des mécanismes cellulaires amènent à la découverte d'un certain nombre de nouvelles familles d'ARN non traduits, regroupés sous le nom générique d'ARN *non-codants* ou **ARNnc**.

famille	ARN	taille	description
ARNi (introns)	groupe I/II	variable	beaucoup de pseudonœuds propriétés d'épissage
ARNsc (small cytoplasmic)	7S, 4.5S	90–330 nt	cxRNP SRP (Signal Recognition Particle)
ARNsn (small nuclear) *ou* ARNu (uridin-rich)	U1, U2, U4, U5, U6, U11, U12	60–220 nt	cxRNP splicéosome excision/épissage des ARNm
ARNsno (small nucleolar)	U3, U14	60–220 nt	guides pour la méthylation de nucléotides sur les ARNr, ARNt et ARNnc
ARNg (guide)	–	50–90 nt	édition des ARNm : insertion/délétion de l'uridine
RNases P	–	350–410 nt	maturation des pré-ARNt et de certains ARNr par clivage en 5′
ARNtm (transfert–messager)	ARN 10S/10Sa gène bactérien ssrA	350 nt	propriétés combinées du ARNt et du ARNm
ARN Télomérase	–	170 / 440 nt	cxRNP Télomérase

TAB. 1.1 – *Quelques grandes familles d'ARN non-codants, déjà assez connues et étudiées.*

La fonction des ARNnc n'est pas toujours très bien connue, et la terminologie des familles n'est pas tout à fait stabilisée : les ARNnc sont parfois désignés comme *ARNmi (micro : 20 à 25 nucl.)* et *ARNs (small : 100 à 200 nucl.)*, ou comme *RNA genes*. Ces ARN participent souvent

à la régulation post-transcriptionnelle et ont des modes de fonctionnement assez diverses dans la cellule. La TABLE 1.1 donne un aperçu de certaines familles d'ARN non-codants, indiquant brièvement leur taille et leur fonction. Les ARNsno possèdent une séquence complémentaire à une séquence cible (régulation de type antisens), certains ARN ont des fonctions catalytiques, voire auto-catalytiques (introns du groupe I et II, RNases P), certains *simulent* des ARN fonctionnels en imitant leur structure (ARN 6S, ARNtm), d'autres participent à de gros complexes ribonucléoprotéique (ribosome, SRP, télomérase), etc. Leur point commun est de ne pas être exprimés en protéines. Les mécanismes en jeu font donc très souvent intervenir la structure de l'ARNnc.

Le caractère fonctionnel et les mécanismes associés sont généralement découverts par des méthodes biochimiques. On trouve ensuite des séquences homologues en effectuant des recherches dans les génomes ou les banques de séquences. Ces recherches peuvent se faire en utilisant des outils standards de balayages de génomes (BLAST [AGML90] en particulier), qui recherchent des ARN homologues par similitude de séquence. Mais les ARN non codants peuvent avoir assez peu d'homologie en séquence et être par conséquent difficiles à identifier par ces voies classiques. Si l'on connait des informations sur leur structure, il devient possible de développer des programmes de balayage spécifiques qui recherchent un motif particulier. Par exemple le motif en feuilles de trèfle des ARN de transfert [LE97] ou le motif plus complexe des ARNsno (ARN small-nucleolar, voir FIG. 1.14).

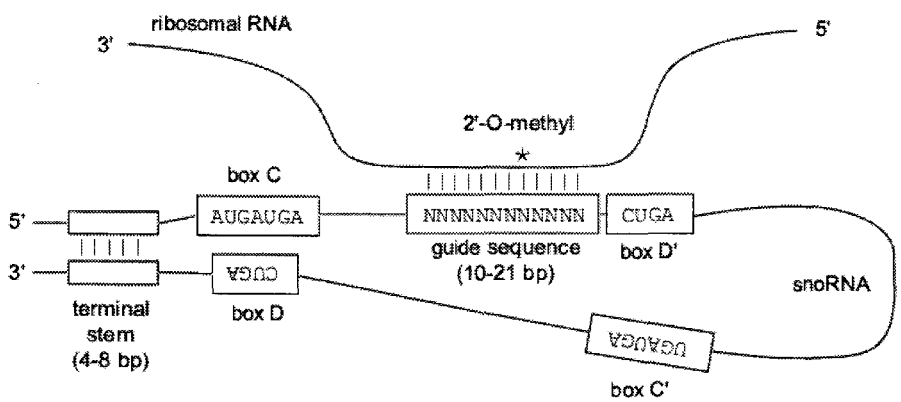


FIG. 1.14 – *La connaissance du motif structural des ARNsno permet de construire un programme de recherche spécifique à ce motif en utilisant les techniques de reconnaissance de langages [LE99]. Le motif recherché est la combinaison d'une expression régulière et d'éléments structuraux qui ne peuvent pas être décrits par une expression régulière, ce qui oblige à combiner des méthodes de type Hiden Markov Model et Stochastic Context Free Grammar. Le programme, une fois entrainé, permet une recherche sur ce motif dans les banques de génomes pour trouver de nouvelles séquences homologues*[10].

On découvre aussi des motifs structurés dans les régions non traduites (5′ UTR et 3′ UTR) des ARN messagers, jouant un rôle dans la transcription et la régulation de l'expression des

10. *source de l'image :* `http://lowelab.ucsc.edu/~lowe/thesis/img1.png`

gènes. Leur fonction n'est pas encore bien cernée ni les éléments de structure qu'on s'attend à y trouver. Chez les procaryotes, par exemple, une tige terminale suivie d'une séquence oligo U sur le transcrit primaire stoppe la progression de l'ARN polymérase (FIG. 1.15). Jusqu'à présent rien par contre n'a permis de mettre en évidence un rôle particulier que pourrait jouer la structure des ARNm matures à l'interieur de l'ORF.

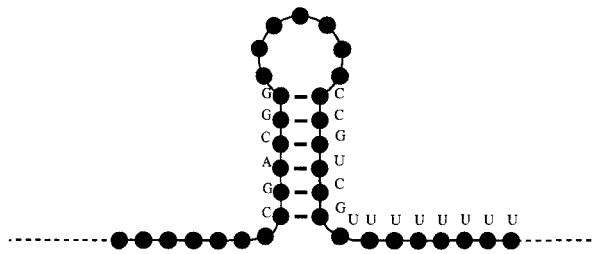


FIG. 1.15 – *préARNm : tige terminale qui libère la polymérase.*

La récente base de données RFAM (**RNA fam**ilies) [GJBM+03] regroupe les séquences d'anciennes bases et s'enrichit de semaine en semaine. Pour chaque famille, on trouve un descriptif de la fonction de la molécule, une image de la structure secondaire d'un des membres de la famille, et un alignement selon la structure. Les méthodes pour trouver la structure et l'alignement ne sont pas toutes les mêmes, mais les sources sont indiquées en bibliographie.

1.3.2 ARN "externes"

D'autres ARN peuvent être structurés, qui ne sont pas produits par l'organisme "hôte" : les virus, les molécules de synthèse.

Il existe plusieurs types de **virus** à ARN (simple brin, double brin, etc.), dont les modes de fonctionnement, comme pour les ARN non-codants, sont assez variés. Les rétrovirus, par exemple, s'intègrent dans l'organisme par *transcription inverse*. Au lieu d'ADN, leur génome est constitué d'ARN, protégé dans une capside (une coque de protéines), entourée d'une enveloppe lipidique. Après infection, c'est-à-dire lorsque l'ARN viral a pénétré dans le cytoplasme de son hôte, cet ARN est transcrit en ADN et va s'intégrer dans le génome de l'hôte, d'où il est à nouveau transcrit pour former de l'ARN et des protéines virales. Le VIH fonctionne de cette façon. Pour tous les virus à ARN, de nombreuses structures jouent un rôle majeur dans leur multiplication. Des motifs en tiges-boucles ou en pseudonœuds sont impliqués dans les mécanismes de changement de cadre de lecture, de dimérisation de l'ARN rétroviral, ou d'encapsidation (association entre un acide nucléique viral et les protéines de la capside). L'**IRES** (Internal Ribosome Entry Site) par exemple est une région du génome des virus à ARN qui permet l'association des sous-unités ribosomiques et la traduction du génome viral. Cette région est évidemment structurée.

De très petites molécules d'ARN, ou **oligo-ARN** sont construites et utilisées à des fins thérapeutiques. La structure de ces molécule est fondamentale pour leurs propriétés. Le problème ici est particulier puisqu'on va souvent chercher à produire une molécule qui adopte une structure donnée. Pour moduler l'expression de gènes viraux, des oligo-ARN sont conçus

artificiellement ou issus de procédés de sélection in vitro (aptamères). Ces séquences sont dirigées vers des motifs ARN qui jouent un rôle clé dans la multiplication virale (la structure IRES par exemple).

1.4 La structure secondaire de l'ARN

1.4.1 Représentations de la structure secondaire

La structure secondaire admet plusieurs modes de représentation équivalents. On en présentera trois et l'on se permettra par la suite d'utiliser l'un ou l'autre en fonction de nos besoins.

Le plus intuitif est le mode de représentation visuel ou **arborescent**, celui où l'on dessine les appariements par de petits segments de longueur très courte, tout en autorisant la flexibilité du squelette de la molécule. La structure de la molécule peut aussi être schématisée par un **diagramme**, où le "squelette" de la molécule est dessiné de manière rectiligne, et les appariements comme des arcs courbes. Par définition, la structure secondaire n'autorise que les emboîtements et les juxtapositions d'appariements : on voit aisément que cela correspond à une **expression bien parenthésée** : une parenthèse ouvrante pour la base appariée en 5′, et une fermante pour sa partenaire en 3′.

La FIG. 1.16 illustre ces trois modes de représentation sur la structure "en feuilles de trèfle" d'un ARN de transfert. Dans la représentation comme expression parenthésée, on utilisera indifféremment des parenthèses, crochets ou '<'. Habituellement les crochets sont utilisés en alternance avec les parenthèses pour indiquer la présence d'un pseudonœud, car le parsing de l'expression est alors ambigu. Dans notre cas le parsing n'est pas ambigu, mais la lecture est facilitée lorsqu'on alterne les symboles.

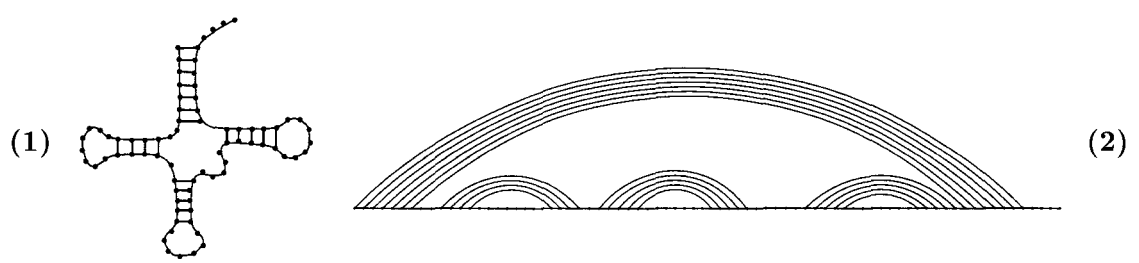


FIG. 1.16 – *Trois représentations équivalentes de la structure secondaire d'un ARN de transfert : la représentation arborescente (1), en diagramme (2), et en expression parenthésée (3).*

La FIG. 1.17 montre la représentation arborescente d'un ARN ribosomique. Tous les appariements sont dessinés de la même longueur, et le dessin de la molécule se pliant à cette

contrainte fait apparaître des boucles et des tiges, ce qui rend légitime le nom donné aux divers éléments structuraux.

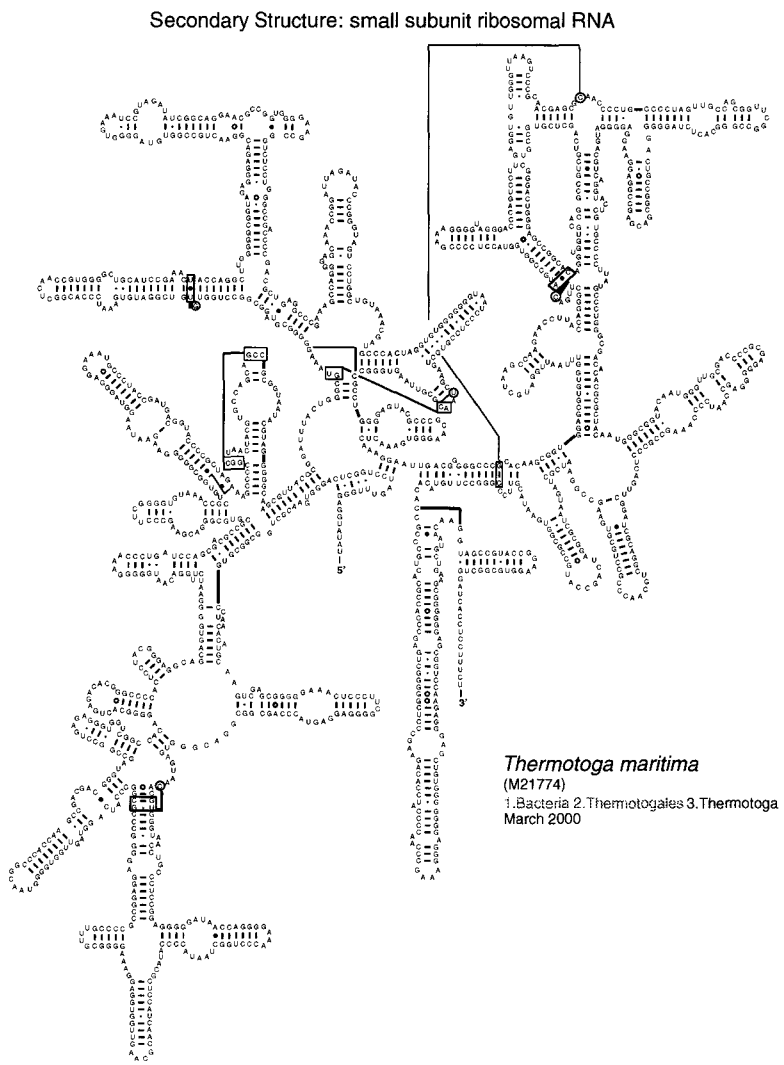


FIG. 1.17 – *Structure secondaire d'un ARNr ssu, ainsi que les interactions tertiaires (pseudonœuds et triplets)*[11].

Dans la représentation de la molécule comme une expansion arborescente de tiges et de boucles, l'adjectif "arborescente" n'est pas anodin, car la structure secondaire est directement modélisable par un **arbre**. Plus précisément il existe une correspondance biunivoque entre les structures secondaires et les arbres orientés enracinés. La construction de l'arbre correspondant à une structure secondaire donnée est réalisée très simplement de la façon suivante (FIG. 1.18) : on pose la racine de l'arbre à l'extérieur de tous les arcs puis on parcourt la molécule de 5′ en 3′ et on ajoute une feuille pour chaque base non-appariée accessible, et un nœud interne

11. *source de l'image :* `http://www.rna.icmb.utexas.edu/`

pour chaque appariement accessible. On procède ainsi récursivement pour chaque sous arbre. La construction réciproque ne pose aucun problème non plus.

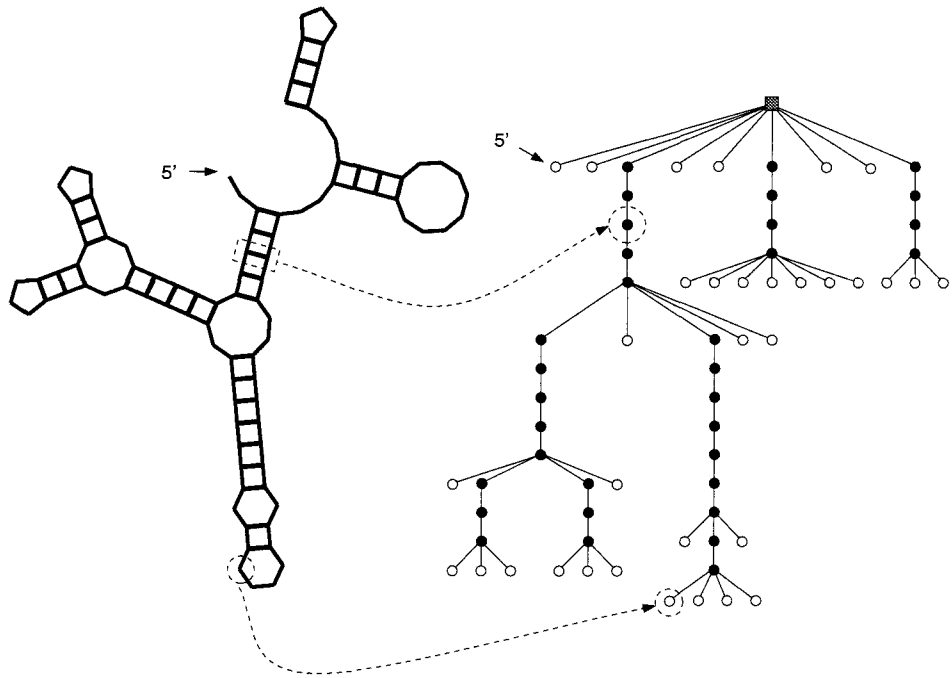


FIG. 1.18 – *Correspondance entre arbre et représentation arborescente de la structure secondaire. Les figures sont ici un peu simplifiées : les nœuds apparaissent non étiquetés, mais ils peuvent contenir l'information de la séquence, voire plus. Un nœud externe (i.e. une feuille de l'arbre) correspondra toujours à une base libre de la molécule, et un nœud interne à un appariement*[12].

Si on observe la molécule à un niveau de granularité plus élevé, au niveau des tiges, les différents modes de représentation peuvent être facilement adaptés en "factorisant" les appariements. Au niveau de l'arbre, celà revient à grouper en un seul nœud les lignées de fils uniques pour les nœuds internes, et les frères mitoyens pour les feuilles. On obtient ainsi une représentation arborescente compacte de la molécule.

La représentation graphique de la structure secondaire est un problème en soi, dont les difficultés sont analogues à celles concernant la visualisation des arbres où il s'agit d'avoir une bonne répartition spatiale tout en évitant les chevauchements [BH88, EG99].

Si l'on autorise les interactions tertiaires, la représentation graphique comporte des arcs qui se coupent et la structure n'est plus modélisable par un arbre. Sur les FIG. 1.17 et FIG. 1.19, on peut remarquer quelques interactions tertiaires (pseudonœuds et triplets) et vérifier par la même occasion qu'elles sont peu nombreuses relativement aux interactions qui constituent la structure secondaire[13].

12. *source de l'image :* `http://www.tbi.univie.ac.at/papers/Abstracts/93-07-08.ps.gz`

13. On trouve des pseudonœuds de manière plus abondante dans les introns du groupe I ou les virus, par exemple.

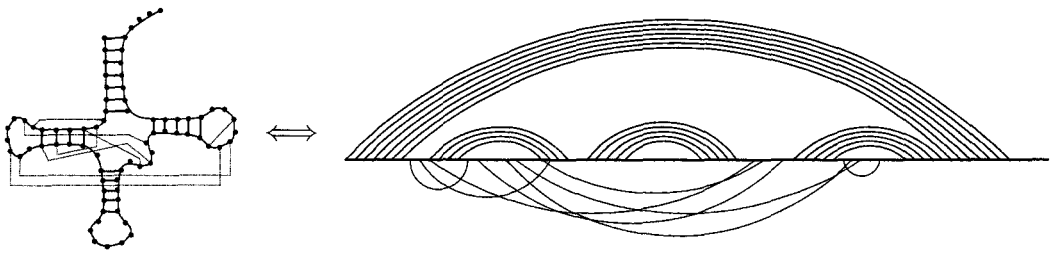


FIG. 1.19 – *Interactions secondaires et tertiaires d'un ARN de transfert.*

1.4.2 Discussion sur le palier intermédiaire qu'est la structure secondaire

La division de la structure de l'ARN en plusieurs niveaux hiérarchiques bien définis trouve un écho en algorithmique : la structure secondaire étant modélisable par un arbre, cela permet de la manipuler aisément et d'utiliser des outils et résultats concernant les arbres. Les algorithmes associés ont souvent une complexité algorithmique qui les rend utilisables, ce qui n'est pas le cas si l'on considère le graphe complet des interactions faibles. On peut en particulier réaliser certains dénombrements de structures [SW94, Wat95], définir et calculer rapidement une distance entre structures secondaires [NS83, DT03b, Tou03, DT03a, ZWM99] ou prédire la structure secondaire à partir de la structure primaire.

Cette hiérarchie n'est pas dénuée de sens biologique, car les niveaux de structure secondaire et tertiaire peuvent être séparés expérimentalement, ce qui donne une autre légitimité à une telle gradation [WT98, TB99, WA00]. Une fois la structure secondaire formée, les tiges s'agencent dans l'espace pour former la structure tridimensionnelle. Des pseudonœuds se forment dans l'eau en présence de guanosine et d'ions métalliques divalents (Mg^{2+}) ou d'une grande quantité d'ions monovalents (une partie des ions réduit la répulsion électrostatique des ions phosphate en se fixant sur des sites récepteurs, favorisant ainsi le rapprochement de certaines structures, une autre partie des ions assure la liaison). L'absence de ces ions permet d'observer *in vivo* la structure secondaire.

Il peut y avoir un réarrangement des appariements secondaires lors de la formation des pseudonœuds [WT98]. Au cours de ce réarrangement, les boucles sont déformées, mais l'expansion de la molécule (les tiges et leur agencement) reste identique dans la plupart des cas.

Chapitre 2

La prédiction de structure

Les méthodes physico-chimiques pour déterminer la structure des ARN sont coûteuses en énergie, en temps et en argent. Parallèlement, les séquences deviennent via l'internet de plus en plus facilement disponibles. La prédiction algorithmique de la structure de l'ARN à partir de sa seule séquence offre donc un intérêt évident. La prédiction structurelle au niveau atomique est hors de portée actuellement pour des molécules qui dépassent quelques dizaines de nucléotides, c'est pourquoi les recherches se sont naturellement portées sur les niveaux de structure intermédiaires, et plusieurs lignées de méthodes ont été développées pour prédire les interactions qui se forment dans l'ARN à partir de la seule information contenue dans la structure primaire. Si l'on observe bien ces différentes méthodes, on s'aperçoit qu'elles posent chacune le problème différemment, et surtout qu'elles ne se placent pas toutes dans les mêmes conditions.

Il existe principalement deux grands types d'approches – l'approche thermodynamique et l'analyse comparative. Ces deux approches ont donné lieu à des algorithmes qui se placent dans des contextes presque opposés : le premier type d'algorithme examine une séquence, le second un ensemble assez volumineux de séquences homologues. Ces méthodes prédisent toutes deux la structure secondaire. De nouvelles méthodes plus récentes, qui sont très souvent des méthodes *hybrides* dans le sens où elles procèdent simultanément des deux approches, agissent dans un contexte intermédiaire, en considérant un *petit* ensemble de séquences.

2.1 Prédiction de structure : un tour d'horizon

2.1.1 Prédire la structure, un problème à plusieurs facettes

Selon la manière dont on appréhende la prédiction de structure, on cherche en réalité à répondre à des problèmes différents, et l'on se place ainsi implicitement dans des *contextes* différents.

Si l'on aborde le problème en portant son regard sur la **séquence**, celui-ci est ordinairement scindé en deux selon qu'on considère une séquence ou plusieurs : lorsqu'on dispose d'une séquence, on ne peut tirer des informations que de sa structure primaire, et d'éventuels

indices en faveur de certaines zones appariées ou non-appariées ; si l'on a par contre plusieurs séquences homologues, on peut examiner l'information *commune* aux séquences.

Les programmes de prédiction qui utilisent plusieurs séquences ont parfois certains prérequis sur ces séquences : on peut demander qu'elles soient *alignées*, on peut aussi soi-même disposer d'un alignement muni d'une structure consensus et prédire la structure d'une séquence en l'alignant contre cette structure consensus.

Si l'on porte maintenant son œil sur ce qu'on veut obtenir, c'est-à-dire la **structure**, on peut objecter que parler de *la* structure est sans doute un peu fallacieux, car elle n'est pas nécessairement unique et figée. La molécule in vivo est sans cesse en mouvement et la structure inerte et rigide qu'on nous présente ressemble plus à celle d'une molécule cristalisée.

Avec une famille de séquences, parler de *la* structure devient plus légitime, car il s'agit d'une structure qui doit être partagée (bien qu'on observe parfois des conformations *alternatives*).

Les informations en sortie des algorithmes nous renseignent souvent sur la question implicitement posée. Les difficultés du problème amènent en effet – plus ou moins explicitement – à *choisir* où va "pêcher" le programme. On peut ainsi ne chercher qu'un ou plusieurs éléments, précis et localisés, de la structure commune (un motif, une tige), ou bien rechercher la structure entière, mais en effectuant certaines approximations sur la modélisation de la structure (le modèle énergétique, par exemple) ou en imposant des conditions sur la qualité des entrées (beaucoup de séquences, séquences alignées). Le programme peut aussi chercher à *détecter* la présence ou l'absence d'une structure commune tout en la prédisant.

Ainsi ce qu'on décrit de manière un peu péremptoire comme *la prédiction de structure* n'est pas *un* problème, mais ressemble plutôt à un ensemble de questions de natures différentes.

L'algorithme que nous présenterons au chapitre suivant se place dans un contexte moderne, celui où l'on dispose de peu de séquences homologues. Nous ne prétendrons pas produire l'intégralité de la structure, mais nous cherchons plutôt à obtenir un haut degré de certitude sur les éléments trouvés.

2.1.2 Pourquoi la structure secondaire

Dans la recherche de structure, le stade ultime est la prédiction de la forme tridimensionnelle de la molécule. Une prédiction à un tel niveau de précision, sans aucune autre information sur la molécule que sa structure primaire (prédiction *ab initio*) est largement hors de portée pour les molécules d'ARN de taille réelle. Le programme MC-Sym conçu par Major *et al.* [MTG+91, GMC93] prédit la structure 3D pour des ARN de quelques nucléotides seulement. C'est pourquoi on recherche des informations structurales à des niveaux d'acuité inférieurs (structures secondaire et tertiaire). Ces informations peuvent servir de contraintes et réduire considérablement l'espace des conformations à explorer par un algorithme de prédiction spatiale[1].

Lyngsø et Pedersen ont montré que le problème de déterminer si une structure secondaire avec pseudonœuds possédait une énergie libre inférieure à une valeur donnée E était NP-complet [LP00]. Pour ce faire, ils se placent dans un modèle thermodynamique qui est une

1. Même avec ces contraintes sur l'espace des configurations à explorer, le problème de la prédiction de la structure spatiale pour des molécules de taille réelle n'est pas encore résolu.

extension *raisonnable* de celui dont nous parlerons au paragraphe 2.3, utilisé pour prédire la structure secondaire. Eddy et Rivas ont proposé un algorithme polynomial capable de trouver la structure tertiaire par programmation dynamique, en se restreignant à certains types de pseudonœuds [RE99]. La classe de pseudonœuds considérée semble contenir la plupart des pseudonœuds qu'on peut rencontrer, mais cet algorithme est $O(n^4)$ en espace et $O(n^6)$ en temps. Il constitue un beau tour de force théorique, et une approche de type linguistique intéressante (grammaires d'ARN), mais s'applique très difficilement dans la pratique.

Les méthodes qui s'attaquent à la prédiction de structure à des niveaux supérieurs à celui de la structure secondaire sont toujours grandement freinées par la complexité algorithmique inhérente au problème. Par ailleurs, la structure secondaire est une étape intermédiaire qui est *réellement* franchie par la molécule au cours de son repliement (*Cf.* page 25), il est donc raisonnable de commencer par porter son attention sur la structure secondaire.

Comme nous l'avons dit au paragraphe 1.2.1, le repliement de l'ARN ne se forme pas de manière fantaisiste, mais les appariemments ont une tendance à s'empiler pour former des tiges. Les premières heuristiques pour la prédiction de structure [PM75, SRCS78, Mar84] consistaient à rechercher les tiges potentielles comme étant des empilement maximaux d'appariements canoniques (de type Watson-Crick ou wobble), puis à les incorporer de manière gloutonne, en supposant que les tiges les plus longues devaient nécessairement se former (et se former en priorité). Les algorithmes effectuaient une recherche exploratoire plus ou moins exhaustive, avec une approche de type branch & bound ou gloutonne, l'intervention manuelle restant très présente. La structure était décrite comme un assemblage de tiges, sans qu'il soit fait de distinction entre structure secondaire et tertiaire. Ensuite sont apparues des méthodes beaucoup plus fines, tirant parti de la gradation intermédiaire de *structure secondaire*.

Les méthodes de prédiction de structure secondaire se déploient principalement en deux grandes familles que nous détaillerons par la suite : l'**approche thermodynamique** et l'**analyse comparative**.

L'approche thermodynamique donne lieu à un algorithme qui cherche à minimiser l'énergie libre de la molécule, en utilisant une table de paramètres d'énergie générée expérimentalement. Les implémentations les plus couramment utilisées sont celle de Zuker (MFOLD [Zuk03, MSZT99]) et celle du paquetage de Vienne (Vienna RNA package [HFS+94]).

L'approche par analyse comparative consiste à utiliser un certain nombre de molécules homologues pour tirer parti des mutations compensatoires (les *covariations*). Cette méthode est souvent mise en œuvre manuellement ; son implémentation la plus connue utilise des grammaires stochastiques hors-contexte [ED94, DEKM98].

Les algorithmes de minimisation d'énergie libre s'appliquent d'avantage dans le cas d'une unique séquence, et l'analyse comparative ne fonctionne correctement que si l'on dispose d'un nombre conséquent de séquences assez bien conservées pour s'aligner correctement. Pour de grandes familles d'ARN bien connues (ARN de transfert et ARN ribosomique par exemple), le nombre de séquences disponibles est largement suffisant, et ces séquences s'alignent relativement bien (il y a peu d'insertions / délétions). Mais le séquençage intégral de nouveaux génomes fournit de nouvelles familles de séquences partageant une fonction commune, et donc probablement une structure commune. Or ces familles sont souvent trop petites pour les appréhender selon la deuxième approche. Des méthodes plus récentes, que nous nommons **hybrides** tentent de répondre à la question dans ce contexte.

2.2 Explorer l'espace des configurations

L'exploration de l'espace des configurations potentielles est souvent mise en œuvre dans des algorithmes stochastiques. Historiquement les premières méthodes (Martinez [Mar84]) sont gloutonnes. On recherche d'abord toutes les tiges possibles, soit comme empilements de paires maximaux, soit plus finement en calculant leur énergie libre. Puis on construit la structure secondaire en les incorporant successivement. Expérimentalement, on pourrait concevoir que cela revient à placer la molécule dans un contexte où aucun appariemment n'est stable (milieu aqueux, haute température, faible concentration ionique), puis à favoriser progressivement la formation des liaisons hydrogène. Les structures qui apparaîssent en premier lieu sont a priori les plus stables [TB99, TSB+96].

Lorsque cet ensemble de configurations est explicitement l'espace *temporel* des repliements successifs de la molécule, il revêt alors une réelle signification biologique, et l'on parle de **repliement cinétique**. Les algorithmes de repliement cinétique (Gultyaev [Gul91], Bouthinon *et al.* [BS99], Isambert *et al.* [IS00], Kim *et al.* [KCP96], Chen *et al.* [CLM00]) prennent en compte les contraintes temporelles en faisant intervenir les différents stades intermédiaires dans la formation du repliement final, c'est-à-dire le *chemin* du repliement au travers de l'énorme espace des conformations intermédiaires possibles. Cette approche invite assez naturellement à utiliser des méthodes stochastiques (Kim *et al.* utilisent le recuit simulé, Chen *et al.* un algorithme génétique) et permet aussi de mettre en évidence d'éventuelles *structures alternatives* de la molécule (pARNass [GHR99]). C'est une approche difficile car elle présuppose une bonne connaissance des interactions moléculaires, et fait par conséquent beaucoup de suppositions quant au modèle temporel et physique (interactions atomiques) du repliement, qu'on connait mal. Le fait même que la cinétique du repliement intervienne fondamentalement dans l'élection de la structure la plus stable est une question ouverte.

Tous ces algorithmes sont en général trop coûteux en temps pour s'appliquer systématiquement à de grosses molécules. Concernant les méthodes stochastiques (algorithmes génétiques, recuit simulé, monte-carlo...), il est difficile de caractériser la solution trouvée. Ces méthodes souffrent des problèmes d'optima locaux et le comportement des algorithmes stochastiques sur des molécules de structure inconnue est par définition difficile à prévoir. Même s'il existe des études théoriques faisant intervenir la notion de paysage par exemple [FKSS93, FSB+93], ou des outils statistiques [WJ98], de telles méthodes sont plutôt évaluées empiriquement par leurs résultats concrets.

2.3 L'approche thermodynamique

D'après les principes de thermodynamique, l'ARN replié possède une énergie libre minimale. L'approche thermodynamique consiste à obtenir le repliement d'un ARN en cherchant à minimiser l'énergie libre qui lui est associée. C'est sans doute la méthode la plus utilisée par les biologistes lorsqu'ils ont une ou quelques séquences à replier. Le programme le plus fréquemment utilisé est MFOLD, conçu par Zuker *et al.* [Zuk03, MSZT99]. Le modèle en lui-même repose sur des bases thermodynamiques très solides et relativement complexes. De prime abord, ses résultats très fins sur certains ARN de structure connue sont encourageants.

Nous exposons d'abord les récurrences de Nussinov et Jacobson, sous-jacentes à l'algorithme de Zuker, puis l'algorithme de minimisation d'énergie libre et ses optimisations.

2.3.1 Les récurrences de Nussinov et Jacobson

L'algorithme de Nussinov et Jacobson [NJ80] est le *noyau algorithmique* de la méthode de minimisation d'énergie libre. En retour il peut être vu comme un algorithme de minimisation d'énergie libre utilisant un modèle d'énergie très simple : celui où l'on aurait assigné une énergie unitaire à n'importe quel appariement.

L'algorithme procède par programmation dynamique en deux étapes : *emplissage* d'une matrice d'énergies, et *rebroussement* (tracing back) afin de proposer une structure optimale.

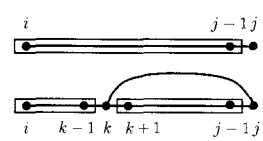
On se donne une séquence $s = s[1..n]$, c'est-à-dire un *mot* de longueur n sur l'alphabet $\{a,u,c,g\}$ et on construit une matrice carrée M de taille n^2. Pour chaque facteur $s[i..j]$ $(i \leq j)$ du mot s, on calcule dynamiquement le nombre maximum d'appariements réalisables sans croisement sur ce segment, ce score est inscrit dans la cellule $M(i,j)$ de la matrice M.

Emplissage de la matrice

Pour le segment $s[i..j]$, deux options sont possibles :

- soit la base $s[j]$ est libre, dans ce cas $M(i,j) = M(i,j-1)$,
- soit la base $s[j]$ forme un appariement avec une autre base $s[k]$ $(i \leq k < j)$.

$$M(i,j) = max \begin{cases} M(i,j-1) & \\ M(i,k-1) + M(k+1,j-1) + 1 & (i \leq k < j) \end{cases}$$



Rebroussement

Une fois la matrice M remplie, le score maximal se trouve naturellement dans la cellule en haut à droite. On piste alors à l'envers pour retrouver la structure maximisant le nombre d'appariements : celle qui a permis d'obtenir ce score. La FIG. 2.1 montre l'emplissage de la matrice ainsi que deux rebroussements possibles sur la petite séquence arbitraire *gcuaguacguacucugcuagcu*. On observe que la structure optimale, loin d'être unique, dépend de l'ordre dans lequel sont effectués les tests. La structure qu'on obtient est ainsi plus ou moins ramifiée.

L'algorithme de Nussinov et Jacobson peut subir quelques raffinements sans dépense supplémentaire en temps ou en espace. On peut limiter la taille des boucle $(3 < |j-i| < M)$, ou imposer l'appariement ou le non-appariement d'un motif.

La complexité de l'algorithme de Nussinov et Jacobson

Il y a $\binom{n}{2} = \frac{n(n-1)}{2}$ facteurs de s, donc la complexité spatiale est $O(n^2)$. Le score de ces $\frac{n(n-1)}{2}$ facteurs est stocké dans la partie supérieure droite de la matrice M. Chaque cellule

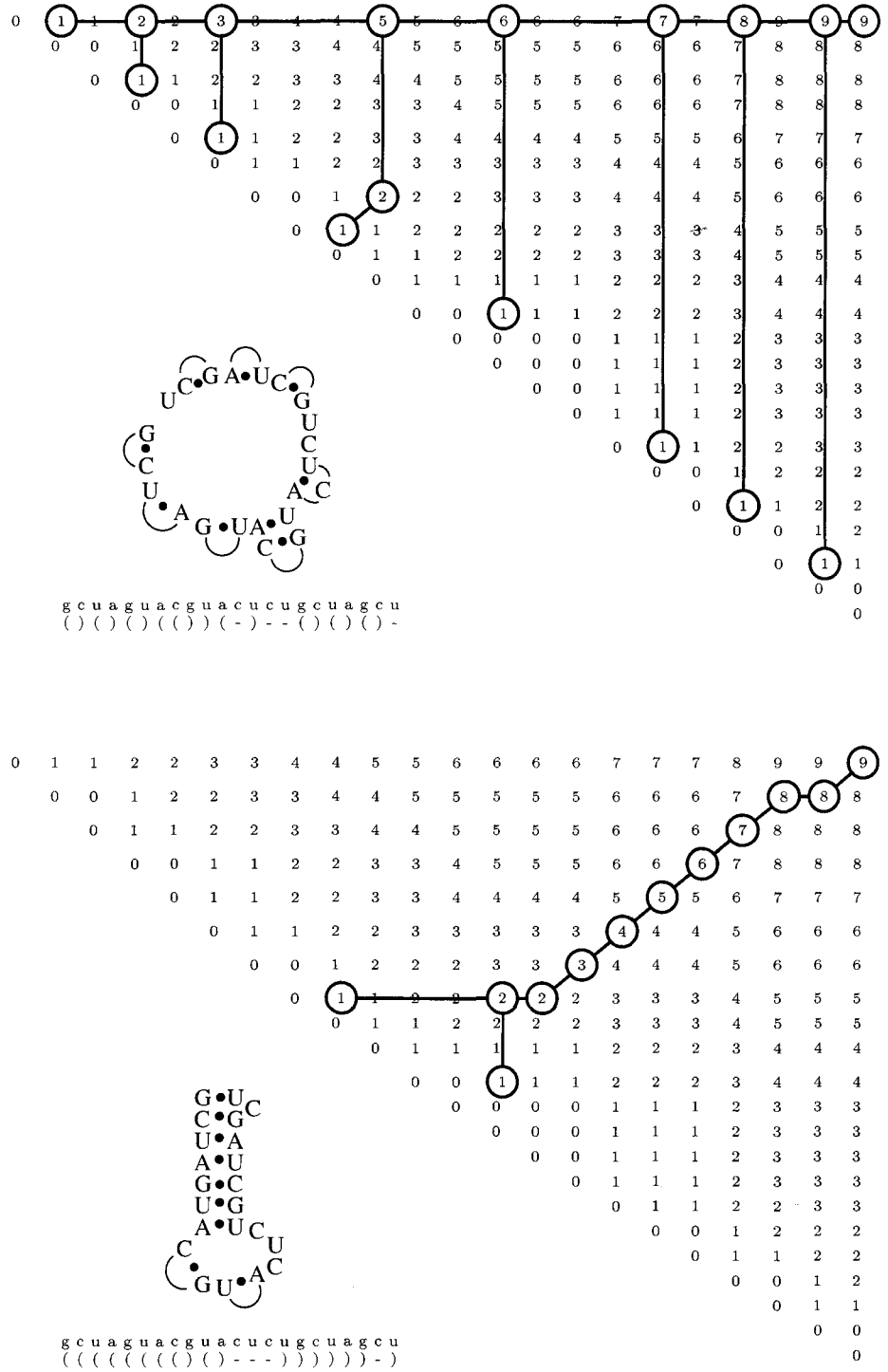


FIG. 2.1 – *Deux rebroussements parmi tous les rebroussements possibles dans la matrice de Nussinov de la séquence gcuaguacguacucugcuagcu. Le premier donnant une structure fort ramifiée, le second fort linéaire.*

de la matrice peut être calculée en temps linéaire, ce qui donne une complexité temporelle en $O(n^3)$. Plus précisément, un calcul exact de la complexité donne :

$$\sum_{j=1}^{n}\sum_{i=1}^{j-1}(j-i-1) = O(n^3)$$

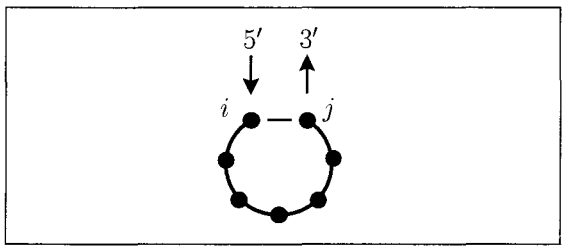
Waterman a utilisé, pour compter le nombre $S(n,k)$ de structures, la correspondance bijective entre l'espace des structures secondaires de longueur n comportant exactement k appariements et l'espace des arbres orientés enracinés de n nœuds, dont k exactement ne sont pas des feuilles. Il obtient la formule exacte $S(n,k)=\frac{1}{k}\binom{n-k}{k+1}\binom{n-k-1}{k-1}$. Cela conduit à une estimation asymptotique du nombre de structures secondaires de longueur n [Wat95, SW94]. Le nombre de structures secondaires de longueur n sans paire isolée, dont les boucles contiennent au moins 3 bases croît asymptotiquement en 1.86^n. Cette estimation monte à 2.35^n si on autorise les pseudonœuds les plus communs [SH99]. Ces comptages rendent d'emblée impossible toute tentative d'exploration exhaustive, et donc toute approche *naïve* pour la détermination de la structure secondaire la plus probable pour une séquence donnée. L'algorithme de Nussinov et Jacobson, dans ce sens, apparaît donc comme une nette amélioration par rapport à une méthode de recherche exhaustive. Il peut être considéré comme l'initiateur du genre, car les méthodes précédentes (Tinoco, Salser...) étaient coûteuses et ne garantissaient pas un résultat optimal au sens mathématique du terme.

La signification biologique de l'algorithme est bien sûr très limitée de part la simplicité du modèle utilisé, toutefois le cœur de cet algorithme constitue la "charpente" de l'algorithme plus sophistiqué de minimisation d'énergie libre.

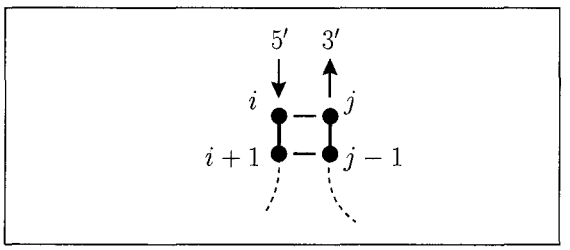
2.3.2 L'algorithme de Zuker

La méthode de minimisation de l'énergie libre de la molécule cherche le repliement optimal en terme d'énergie. Il ne s'agit plus de maximiser le nombre d'appariements, mais d'optimiser l'énergie en intégrant dans le modèle des connaissances biologiques. On tient compte de l'influence constructive ou destructive des éléments structuraux : une tige, par exemple, a un effet stabilisateur, une boucle un effet destructurant. Dans le *nearest neighbour energy model*, la molécule, au lieu d'être considérée comme un empilement d'appariements, est vue de manière duale comme un empilement de **modules**. Chaque module possède une énergie libre (négative), et l'énergie de la molécule est la *somme* de ces énergies.

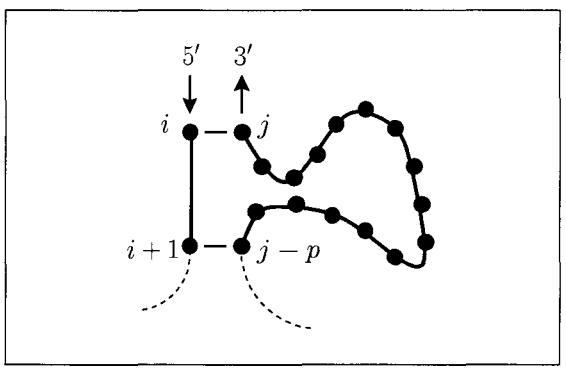
Un module est toujours fermé par un appariement. Son *degré* est le nombre de paires accessibles à partir de l'appariement fermant, et sa *taille* le nombre de bases libres accessibles. La FIG. 2.2 montre comment on peut classifier simplement ces modules élémentaires en fonction de leur degré et de leur taille.



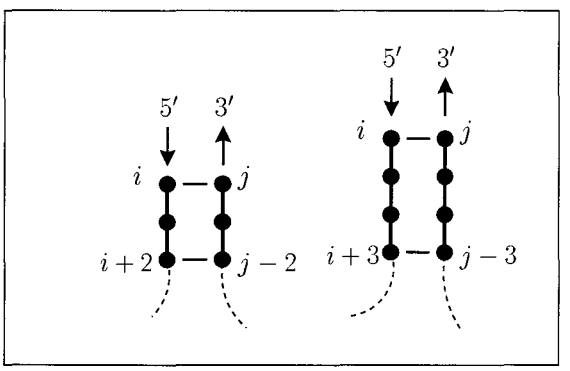
$(d = 1)$: boucle terminale



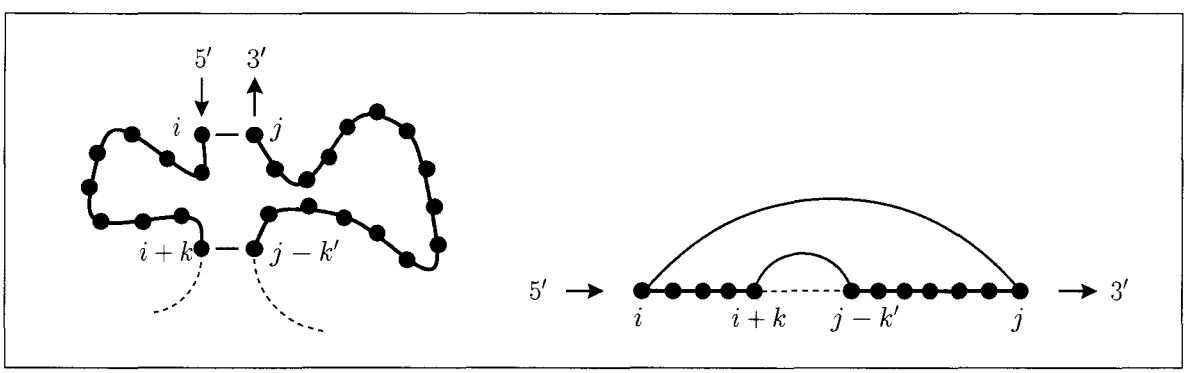
$(d = 2)$: empilement



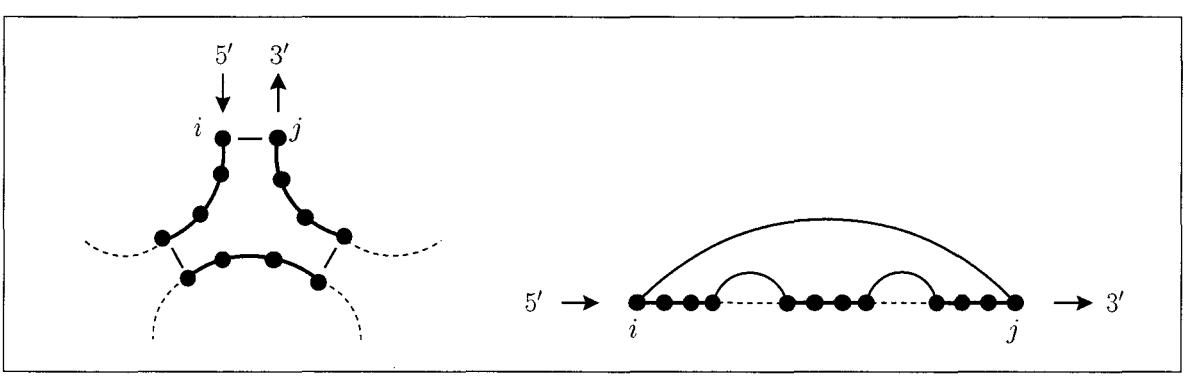
$(d = 2)$: renflement



$(d = 2)$: mésappariement (1 ou 2)



$(d = 2)$: boucle interne



$(d \geq 3)$: embranchement

FIG. 2.2 – *Classification des modules clos par la paire $s[i] \cdot s[j]$ en fonction de leur degré. Les empilements, renflements et mésappariements sont des cas particuliers de boucles internes.*

Les récurrences

Comme pour l'algorithme de Nussinov et Jacobson, le problème de la minimisation de l'énergie libre se résoud par programmation dynamique. On construit une matrice carrée d'énergies E de taille n^2, où n est la longueur de la séquence.

$E(i,j)$ contient l'énergie libre du fragment de molécule $s[i..j]$. Sa valeur est calculée dynamiquement par les récurrences suivantes, où l'on se sert d'une matrice auxilmiaire $\overline{E}\ (i,j)$:

$$E(i,j) = min \begin{cases} \overline{E}\ (i,j) & O(1) \\ E(i,j-1) & O(1) \\ E(i+1,j) & O(1) \end{cases} \tag{2.1}$$

$\overline{E}\ (i,j)$ contient l'énergie libre du fragment de molécule $s[i..j]$ *sachant que l'appariement* $s[i] \cdot s[j]$ *est réalisé.* De manière évidente, pour tout segment $s[i..j]$, $E(i,j) \leq \overline{E}\ (i,j)$.

Le score $\overline{E}\ (i,j)$ est calculé en lisant des valeurs dans une table de paramètres énergétiques :

$$\overline{E}\ (i,j) = min \begin{cases} \textbf{hairpin}\ (i,j) & & O(1) \\ \textbf{stack}\ (i,j,i',j') + \overline{E}(i',j') & & O(1) \\ \qquad (i+1 \leq i' \leq i+3) & (j-1 \leq j' \leq j-3) & \\ \textbf{loop}\ (i,j) & & O(2^n) \end{cases} \tag{2.2}$$

- degré 1 – boucle terminale :
  hairpin (i,j)
- degré 2 – empilement (0,1,2 mésappariements) :
  stack $(i,j,i+1,j-1)$ (taille 0 + 0)
  stack $(i,j,i+2,j-2)$ (taille 1 + 1)
  stack $(i,j,i+3,j-3)$ (taille 2 + 2)
- degré 2 – petits renflements asymétriques :
  stack $(i,j,i+2,j-3)$ (taille 1 + 2)
  stack $(i,j,i+3,j-2)$ (taille 2 + 1)
- degré ≥ 2 – boucles internes et embranchements :
  loop (i,j)

FIG. 2.3 – *Les paramètres d'énergie pour un module clos par la paire* $s[i] \cdot s[j]$. *Ces paramètres sont stockés dans des tables, et dépendent évidemment de la séquence de nucléotides.*

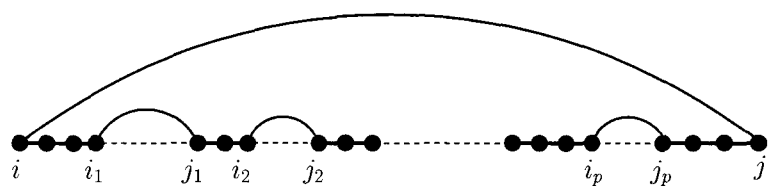
hairpin , **stack** et **loop** sont des valeurs d'énergie lue dans une table qui a été précalculée une fois pour toute (FIG. 2.3). Nous voyons que tous ces calculs se font en temps constant, sauf l'appel de **loop**, qui demande un temps exponentiel si l'on s'autorise tous les types de boucles. Nous allons voir comment certaines simplifications sur le modèle permettent de retrouver une complexité en $O(n)$ pour ce calcul, et donc en $O(n^3)$ pour l'algorithme.

Optimisations algorithmiques

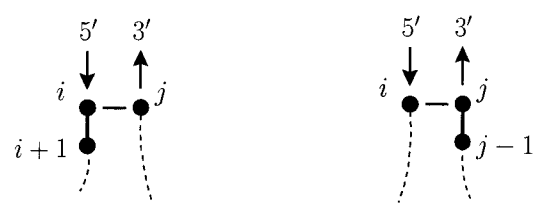
Les embranchements

Pour des fonctions d'énergie **loop** (i,j) quelconques, le temps de calcul est exponentiel. Même si l'on ne s'autorise que des boucles de degré borné, le calcul reste d'une complexité ploynomiale élevée :

$$\textbf{loop}\ (i,j) = \min_{i<i_1<j_1<\ldots<i_p<j_p<j} \textbf{loop}\ (i,\ i_1,j_1,\ldots,i_p,j_p,\ j) + \sum_{x=1}^{p} \overline{E}(i_x,j_x) \qquad O(n^{2p}) \quad (2.3)$$



Pour rendre l'algorithme praticable, **loop** (i,j) est décomposée en contributions linéaires du degré d (nombre de branches) et de la taille p (nombre de bases libres dans la boucle). Seules les bases libres *au bord* des paires qui clôturent la boucle sont prises spécifiquement en compte par les paramètres d'énergie de balancement (*dangling*) :



dangle $(i,j,i+1)$ **dangle** $(i,j,j-1)$

$$\textbf{loop}\ (i,j) = \textbf{loop}_{\ d,p}\ (i,j) = \alpha d + \beta p + \gamma$$

En utilisant le caractère additif du modèle d'énergie, on peut calculer **loop** (i,j) de proche en proche (selon une méthode classique déjà utilisée dans l'alignement de séquences [Got82]). La complexité globale peut ainsi être rendue cubique à nouveau. C'est ce modèle qui est utilisé dans l'algorithme de Zuker MFOLD [Zuk03, MSZT99].

Les équations de récurrence (2.1) incorporent le coefficient β et les valeurs de **loop** sont calculées en temps linéaire, à l'aide de cellules de la matrice E qui ont déjà été calculées, et des paramètres de balancement **dangle** :

$$E(i,j) = min \begin{cases} \overline{E}\ (i,j) & O(1) \\ \beta + E(i,j-1) & O(1) \\ \beta + E(i+1,j) & O(1) \end{cases} \quad (2.1)$$

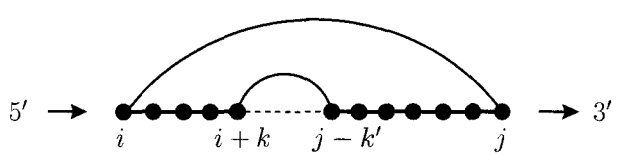
$$\textbf{loop}\ (i,j) = \alpha + min \begin{cases} E(i+1,k) + E(k+1,j-1) & O(n) \\ \qquad (i+1 \leq k < j-1) & \\ \textbf{dangle}\ (i,j,i+1) + E(i+2,k) + E(k+1,j-1) & O(n) \\ \qquad (i+2 \leq k < j-1) & \\ \textbf{dangle}\ (i,j,j-1) + E(i+1,k) + E(k+1,j-2) & O(n) \\ \qquad (i+1 \leq k < j-2) & \end{cases} \tag{2.4}$$

Les boucles internes

Les toutes petites boucles internes, dont la taille n'excède pas quelques nucléotides (en particulier les empilements et mésappariements), sont prises en compte de façon très précise dans le modèle d'énergie par les paramètres **stack**. Dans le cas général, pour les boucles internes (c'est-à-dire les embranchements de degré 2) un calcul brutal par l'équation (2.3) se ferait en $O(n^2)$, ce qui amènerait la complexité globale de l'algorithme à $O(n^4)$. Pour ces boucles, on met donc en œuvre une autre optimisation, plus fine, qui permet d'utiliser une fonction de pénalité plus générale que dans l'optimisation précédente. Bien sûr la fonction de pénalité f doit toujours être sous-additive, c'est-à-dire qu'elle doit pénaliser moins que si l'on additionne les pénalités élémentaires, mais les expériences montrent de façon précise que la fonction destabilisatrice croit plutôt de manière logarithmique en fonction de la taille de la boucle [SM97].

Le calcul de **loop** tiendra compte de la taille de la boucle et de l'empilement des nucléotides à ses deux extrémités. Par la suite on notera **end_stack** $(i,j) :=$ **stack** $(i,j,i+1,j-1)$ pour simplifier l'écriture.

$$\textbf{loop}\ (i,j) = \textbf{end\_stack}\ (i,j) + \min_{1 \leq k+k' \leq j-i-2} f(k+k') + \overline{E}(i+k+1,j-k'-1) \qquad O(n^2) \tag{2.4}$$



L'algorithme de Zuker utilise une optimisation, proposée par Waterman et Smith [WS86], pour laquelle le temps de calcul global reste cubique, tandis que l'espace, qui était quadratique, devient cubique. Au lieu de recalculer à chaque fois **loop** (i,j), on stocke pour chaque cellule de la matrice un vecteur **loop** $(i,j)[p]$, où $p = k + k'$ est la taille de la boucle interne.

$$\textbf{loop}\ (i,j)[p] = \textbf{end\_stack}\ (i,j) + f(p) + \min_{k+k'=p} \overline{E}(i+k+1,j-k'-1) \qquad (1 \leq p \leq j-i-2) \tag{2.5}$$

$$\textbf{loop}\ (i,j) = \min_{3 \le p \le j-i-2} \textbf{loop}\ (i,j)[p] \tag{2.6}$$

loop $(i,j)[p]$, au lieu d'être calculé en temps linéaire pour chaque taille p, peut être obtenu en temps constant à l'aide des vecteurs précedemment stockés. La FIG. 2.4 illustre les trois cas de figure: boucle interne, renflement à droite ou à gauche.

$$\textbf{loop}\ (i,j)[p] = min \begin{cases} \textbf{loop}\ (i+1,j-1)[p-2] + f(p) - f(p-2) & \\ \qquad + \textbf{end\_stack}\ (i,j) - \textbf{end\_stack}\ (i+1,j-1) & O(1) \\ \textbf{end\_stack}\ (i,j) + f(p) + \overline{E}(i+1,j-p-1) & O(1) \\ \textbf{end\_stack}\ (i,j) + f(p) + \overline{E}(i+p+1,j-1) & O(1) \end{cases}$$

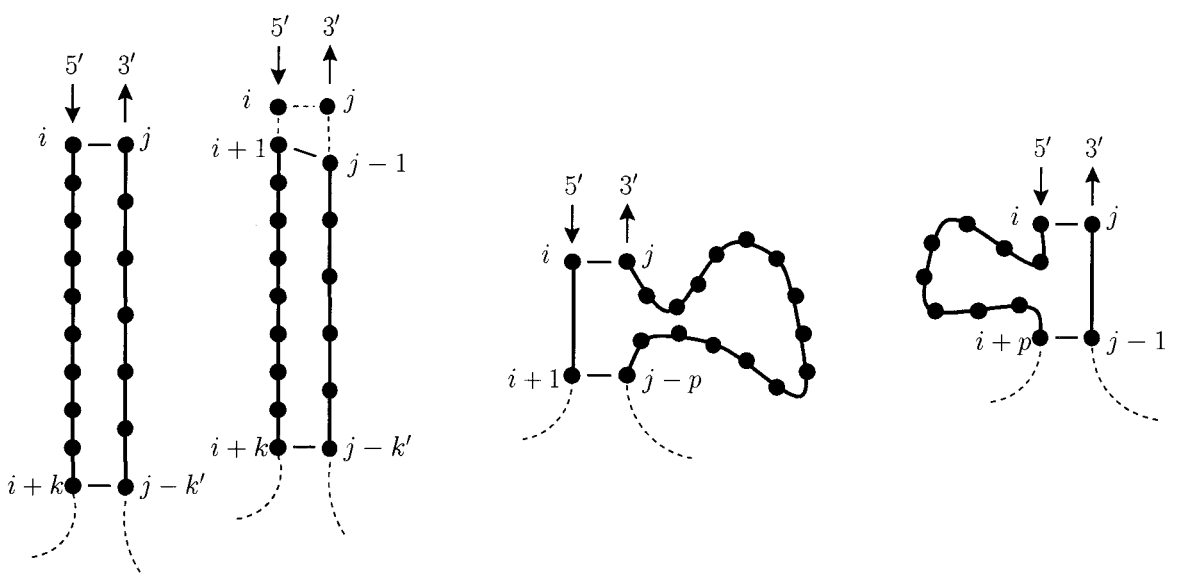


FIG. 2.4 – *Les trois cas de figure pour le calcul de* **loop** $(i,j)[p]$.

L'exposé est ici un peu simplifié, mais on peut aussi prendre en compte le caractère asymétrique de la boucle sans augmenter la complexité (pourvu que la fonction de pénalité vérifie certaines propriétés). L'espace, qui est devenu cubique, peut aussi être restauré quadratique: en calculant les vecteurs dans un ordre spécifique, on peut éviter de les stocker [LZP99].

Par ailleurs, pour des fonctions de pénalité convexes ou concaves, Eppstein *et al.* [EGG88] ont montré comment réduire le temps de calcul des boucles internes à $O(n^2 log^2 n)$. D'autres optimisations reposant sur des techniques de recherche matricielle ont été proposées [EGG88, EGGI91]. Ceci dit, ces optimisations restent théoriques, et sans doute peu efficaces à l'implémentation, le facteur constant pouvant être très grand. Quoiqu'il en soit, à cause des embranchements, l'algorithme reste globalement en $O(n^3)$.

Le problème de savoir si le calcul des boucles internes peut être réalisé en $O(n^2)$ d'une part, et si le coup global de l'algorithme peut être réduit à une complexité inférieure à $O(n^3)$ d'autre part restent ouverts.

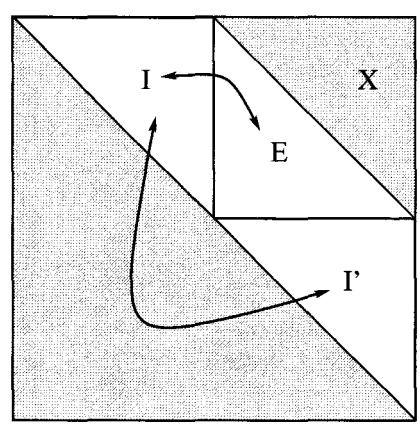
2.3.3 La version sous-optimale

L'algorithme de minimisation d'énergie libre opère des simplifications implicites. Les premières approximations sont inhérentes au *nearest neighbor energy model* : ce modèle ne tient compte de l'influence constructive ou destructive des appariements que pour les bases mitoyennes[2], l'énergie libre y est supposée strictement additive, certains paramètres ne sont pas connus car difficilement mesurables, par exemple concernant les régions libres, ou les pseudonœuds. Le problème est qu'il existe très peu de données expérimentales pour connaitre l'influence énergétique des pseudonœuds et celle des interactions avec des bases à longue distance. De tels paramètres d'énergie sont évidemment assez difficiles à mesurer. Enfin les pseudonœuds ne sont pas pris en compte, ni pour le repliement, ni *a posteriori* pour le calcul numérique de l'énergie libre de la molécule au final.

Le modèle opère un autre type d'approximation, qui n'est pas propre au modèle thermodynamique, mais concerne le fait que l'ARN ne soit pas envisagé dans son environnement : les interactions avec d'autres molécules ne sont pas prises en compte (rappelons que le ribosome, par exemple, est un complexe ribonucléoprotéique, assemblage de trois gros ARN, et de 50 à 80 protéines...). Si on voulait intégrer ces données au modèle lui-même, là encore de tels paramètres énergétiques seraient très difficiles à mesurer. Le chemin de repliement n'intervient pas non plus, or il peut être fortement dépendant de l'environnement de l'ARN et le "coincer" dans un minimum local.

Ces approximations nécessaires conduisent à ce que la molécule, au lieu d'être dans son état d'énergie minimale, soit dans un état proche du minimum. L'algorithme a donc été étendu afin de donner les structures sous-optimales en terme d'énergie [Zuk89]. L'astuce utilisée dans la version *sous-optimale* de l'algorithme de Zuker est similaire en substance à celle utilisée dans dans la version sous-optimale de l'alignement de séquences.

On applique l'algorithme à la séquence dupliquée : $s[1..2n] \leftarrow s[1..n].s[1..n]$. La taille de la matrice est donc multipliée par quatre. En réalité il s'agit juste d'un modèle théorique, car la partie supérieure droite de la matrice (l'autre est vide) peut être divisée en quatre triangles (inclus I, exclus E, interne dupliqué I', et une région X). Les régions I et I' sont identiques, et il existe une correspondance bijective entre les régions I et E, la dernière région X étant inutilisée.



2. Les trois types de forces en jeu dans le repliement de l'ARN – liaisons H entre les bases (appariements), forces de Van der Waals entre les plateaux de bases (empilement) et forces hydrophobes (guidage du repliement) – ont des effets de courte distance.

Pour $1 \leq i \leq j \leq n$, $\overline{E}(i,j)$ est l'énergie du meilleur repliement du segment $s[i..j]$ délimité par l'appariement $s[i] \cdot s[j]$, tandis que $\overline{E}(j,n+i)$ est l'énergie du meilleur repliement du segment $s[j..n+i]$ délimité par l'appariement $s[j] \cdot s[n+i]$.

En considérant la molécule comme circulaire, $\overline{E}(j,n+i)$ représente aussi l'énergie du meilleur repliement des deux fragments *extérieurs* $s[1..i] \cup s[j..n]$, délimités par le même appariement. Modulo quelques petites corrections (la molécule n'est pas circulaire ...), la somme $\overline{E}(i,j) + \overline{E}(j,n+i)$ est l'énergie du repliement minimal contraint par la formation de l'appariement $s[i] \cdot s[j]$.

Si E_{min} est l'énergie du meilleur repliement, et si δE représente une petite variation d'énergie, un rapide coup d'œil aux $n(n-1)/2$ appariements $s[i] \cdot s[j]$ possibles et à l'énergie du meilleur repliement contraint par cet appariement révèle lesquels peuvent être repliés sans que l'énergie s'éloigne de plus de δE de celle du repliement initial.

2.3.4 Les paramètres énergétiques

Malgré les approximations qu'on vient d'évoquer, la méthode de minimisation d'énergie libre s'appuie sur un modèle thermodynamique très poussé. Certaines énergies ont été déterminées par des méthodes physiques, puis la table entière a été extrapolée par des méthodes physiques et algorithmiques (algorithme génétique) [MSZT99].

Lors des premières tentatives, les paramètres d'énergie de Salser (1977) considéraient l'empilement des appariements, et l'algorithme de prédiction de structure secondaire était une application directe de l'algorithme de Nussinov, mais avec des poids plus fins. Les paramètres de Freier (1986) [FKJ+86] considèrent l'empilement des modules. Ces paramètres ont été récemment enrichis par la prise en compte de nouveaux effets (balancement, empilement coaxial) [WTK+94].

L'adaptation de l'algorithme de Nussinov à sa version thermodynamique (poids des modules empilés vs. poids des appariements empilés) ressemble fort, si l'on compare à ce qui se fait au niveau de la structure primaire avec des modèles de Markov, au passage d'un modèle de Bernoulli (*i.e.* Markov d'ordre 0) à un modèle de Markov d'ordre 1. Dans les deux cas, la complexité algorithmique ne change pas.

Le choix de ces paramètres a beaucoup plus d'influence que le choix de l'algorithme: l'algorithme utilisé avec les paramètres de Freier donne des résultats beaucoup plus proches de ceux du repliement cinétique qu'avec les paramètres de Salser [TSB+96]. Cependant, il est possible aussi que cet effet soit un peu tautologique, dans la mesure où les paramètres d'énergie sont déterminés expérimentalement par des méthodes qui ressemblent fort à la démarche algorithmique du repliement cinétique.

2.3.5 L'approche thermodynamique : un contexte minimaliste

L'approche thermodynamique offre une réponse au problème de la prédiction de la structure secondaire d'une séquence *ab initio*, dans le sens où très peu d'informations sont disponibles sur la séquence qu'on cherche à replier. Certaines informations peuvent néanmoins être incorporées de façon à guider le repliement : appariement ou non-appariement d'une ou plusieurs bases, son partenaire étant connu ou non.

La méthode donne des résultats assez impressionnants avec si peu d'informations sur la séquence et la correction des résultats valide en quelque sorte le modèle thermodynamique. La méthode souffre néanmoins de plusieurs inconvénients.

Tout d'abord elle ne tire pas parti de toutes les informations dont on pourrait disposer. Dans les faits, il arrive souvent qu'on dispose de plusieurs séquences, provenant d'organismes différents ou d'endroits différents dans le même organisme, dont on peut supposer qu'elles desservent la même fonction et par conséquent partagent une structure en commun. La méthode thermodynamique telle qu'elle est implémentée ne permet de prendre en compte aucune information *transverse* de ce type.

Ensuite, la molécule n'est pas nécessairement dans son état d'énergie minimal. Pour pallier cet inconvénient, les algorithmes qui sont proposés (MFOLD, Paquetage de Vienne) fournissent une collection de solutions sous-optimales en terme d'énergie. Parmi ces structures apparait souvent la bonne. Mais d'une part ce n'est pas toujours le cas, d'autre part il n'existe aucun moyen de déceler parmi ces structures sous-optimales laquelle est la bonne car aucun indice de *fiabilité* n'est donné sur les éléments de structure trouvés. Ainsi certaines tiges peuvent être correctes, tandis qu'il est impossible de savoir lesquelles.

L'algorithme de Zuker reclasse les structures sous-optimales après avoir recalculé plus finement l'énergie libre. Ce reclassement doit a priori augmenter la probabilité que la bonne structure soit trouvée comme étant optimale, mais le calcul qui est fait ne corrige aucune des approximations qui ont été mentionnées page 39.

Quoiqu'il en soit, le caractère convivial et l'efficacité du serveur MFOLD contribuent à sa popularité.

2.4 L'analyse comparative

2.4.1 Le principe

La fonction d'une molécule est liée à sa structure. Si l'évolution conserve la fonction, elle doit donc globalement conserver la structure, en particulier les tiges dans le cas de l'ARN. Par conséquent, lorsqu'une mutation a lieu dans une tige, il risque d'y avoir une pression sélective pour qu'il y ait en face une mutation compensatoire afin de conserver l'appariement. Un tel couple de mutations est ce qu'on appelle une **covariation**, et l'analyse comparative se propose de prédire la structure à l'aide de ces seules informations.

Les familles d'ARN considérées peuvent provenir de la même espèce ou d'espèces différentes, mais on fait l'hypothèse qu'ils remplissent une fonction commune et on s'attend à ce que cette fonction s'exprime au moins en partie au travers de leur structure, ce qui implique qu'elle soit conservée. Cette hypothèse est légitime lorsque ces molécules présentent de fortes similitudes au niveau de la séquence, par exemple, ou bien lorsqu'elles proviennent du même endroit : elles ont par exemple été extraites en amont d'un gène. Dans la suite, nous nous permettrons souvent de nommer ces séquences **homologues** dans un sens large, lorsqu'on supposera simplement qu'elles partagent une structure commune.

Pour appliquer l'analyse comparative, il faut bien sûr avoir assez de covariations, et donc disposer d'une famille assez conséquente de séquences homologues. Il faut au départ un bon

alignement, dont on examine tous les couples de colonnes. On dira que deux colonnes **covarient** si l'on met en évidence des mutations compensatoires entre ces deux colonnes. On ne fait aucune hypothèse sur le caractère canonique des appariements, mais on observe exclusivement la corrélation des mutations, ce qui permet de déceler les appariements qui seraient devenus non canoniques au cours de l'évolution.

Au vu du problème, deux directions peuvent alors être explorées :

- trouver une bonne mesure de la corrélation des colonnes ;
- exploiter correctement cette mesure.

2.4.2 La corrélation des colonnes

Il existe au moins deux mesures pour la corrélation des colonnes : la première utilise des outils probabilistes (mesure du χ^2), la seconde des outils liés à la théorie de l'information (mesure M de Shannon).

On désigne par $\mathcal{A} = \{a,u,c,g\}$ l'alphabet de l'ARN. On considère un alignement de p séquences, comportant N colonnes. Soient i et j les indices respectifs de deux colonnes, x et y deux bases quelconques.

On note X_i et X_j les variables aléatoires associées respectivement aux colonnes d'indice i et j, ces variables prenant leurs valeurs dans l'alphabet $\mathcal{A}$.

On désigne par $no_i(x) = |\{X_i = x\}|$ et $no_j(y) = |\{X_j = y\}|$ le nombre d'occurences observées de la base x dans la colonne i et de la base y dans la colonne j.

On note $no_{ij}(x,y) = |\{X_i = x \text{ et } X_j = y\}|$ le nombre d'observations du couple (x,y) à la position (i,j), et $ne_{ij}(x,y) = p.\frac{no_i(x)}{p}.\frac{no_j(y)}{p}$ le nombre attendu de couples (x,y) à la position (i,j).

La mesure du χ^2 des deux colonnes i et j est donnée par :

$$\chi^2_{ij} = \sum_{x,y \in \mathcal{A}} \frac{[no_{ij}(x,y) - ne_{ij}(x,y)]^2}{ne_{ij}(x,y)}$$

L'autre mesure de corrélation est tirée de la théorie de l'information de Shannon : elle mesure l'information mutuelle des deux colonnes.

$f_i(x) = \frac{no_i(x)}{p}$ et $f_j(y) = \frac{no_j(y)}{p}$ désignent les fréquences d'occurences de la base x dans la colonne i et de la base y dans la colonne j. On note aussi $f_{ij}(x,y) = \frac{no_{ij}(x,y)}{p}$ la fréquence d'occurences du couple de bases (x,y) dans les colonnes i et j. L'information mutuelle des deux colonnes i et j est alors définie par :

$$M_{ij} = \sum_{x,y \in \mathcal{A}} f_{ij}(x,y)\ log_2\left(\frac{f_{ij}(x,y)}{f_i(x)f_j(y)}\right) = \sum_{x,y \in \mathcal{A}} \frac{no_{ij}(x,y)}{p}\ log_2\left(\frac{no_{ij}(x,y)}{ne_{ij}(x,y)}\right)$$

L'indice M_{ij} varie entre 0 et 2 bits et mesure le degré de corrélation des deux colonnes. Il est maximal lorsque les colonnes i et j sont parfaitement corrélées (préservation totale

ou absence totale d'un appariement) et qu'elles apparaissent pourtant comme totalement aléatoires lorsqu'elles sont prises individuellement (c'est-à-dire que la fréquence de chacune des 4 bases vaut 1/4). Il est nul lorsque les colonnes ne présentent individuellement aucune mutation, ou bien lorsque les colonnes varient de façon indépendante, c'est-à-dire lorsque $f_{ij}(x,y) = f_i(x) \times f_j(y)$.

Cette mesure est affinée différemment selon les auteurs pour prendre en compte le taux de mutants par colonnes (influence positive ou négative selon les cas) et l'arbre phylogénétique [KH99, GHH+94].

```
<<<<<<<  [[[[         ]]]] [[[[[       ]]]]]     [[[[[       ]]]]]>>>>>>>
GTCCGAAtaGCTCagctggataGAGCaATAGCcttctaaGCTATcggtcGGGGGttcgaatCCCTCTTCGGACgcca
GCGCCCAtaGATCaattggataGATCgTTTGActacggaTCAAAaggttGAGGGttcgattCCTTCTGGGCGCgcca
GCACTCGtaGCTTaac-ggataAAGCaTCTGActacggaTCAGAaggttGCAGGttcgaatCCTGCCGAGTGCa---
GTCCACGtaGCTCagcaggataGAGCaCAGGAttcctaaTCCTGgggttGGAGGttcgaatCCTCTCGTGGACacca
GCGCCCGtaGCTCaattggataGAGCgTTTGActacggaTCAAGaggttATGGGttcgactCCTATCGGGCGCg---
GCACCCAtaGCGCaactggataGAGTgTCTGActacgaaTCAGAaggttGTAGGttcgagtCCTACTGGGTGCa---
GCGCCCGtaGCTCaattggataGAGCgTTTGActacggaTCAAAaggttAGGGGttcgactCCTCTCGGGCGCgcca
GCGCCCGtaGCTCagctggataGAGCgCTGCCctccggaGGCAGaggtcTCAGGttcgaatCCTGTCGGGCGCgcca
GCGCCCTtaGCTCagttggataGAGCaACGACcttctaaGTCGTgggccGCAGGttcgaatCCTGCAGGGCGCgcca
              ^             ^             ^
              i             j             k
```

	ij	ik	jk
χ^2	1.44	1.44	18
M	0.15	0.15	1.35

FIG. 2.5 – *Alignement de 9 séquences d'ARN de transfert, où l'on a isolé arbitrairement trois colonnes (d'indices i, j, k). Le tableau donne les valeurs de χ^2 et M pour chaque couple de colonnes. On a indiqué par des '[' et des '<' la structure secondaire en feuilles de trèfle.*

La FIG. 2.5 montre l'exemple simple d'un ARNt. On y a isolé arbitrairement trois colonnes i, j et k. Les deux dernières correspondent à un appariement de la structure consensus en feuilles de trèfle. On observe que les deux mesures donnent un score élevé pour les colonnes j et k, indiquant la présence de l'appariement. On remarque aussi sur cet exemple que les séquences sont assez bien conservées et donc relativement facile à aligner. La contrepartie est qu'il y a peu de covariations, et que certaines colonnes parfaitement conservées n'apportent aucune information en faveur ou non d'un appariement.

2.4.3 La prédiction du repliement commun

La prédiction du repliement commun à partir des corrélations de colonnes est longtemps restée manuelle. Les premières automatisations utilisent la mesure du χ^2 (Olsen 1983, Chiu *et al.* [CK91]), et l'incorporation des colonnes est gloutonne. Puis Eddy *et al.* ont proposé

un algorithme itératif basé sur la modélisation de la structure secondaire par une grammaire hors-contexte [ED94].

Une des grammaires les plus simples qui décrivent la structure secondaire de l'ARN peut être la suivante :

$$
\begin{array}{ll}
S \rightarrow Sa \mid Su \mid Sc \mid Sg & (\textit{base libre}) \\
S \rightarrow aSu \mid uSa \mid cSg \mid gSc \mid uSg \mid gSu & (\textit{appariement}) \\
S \rightarrow SS & (\textit{embranchement}) \\
S \rightarrow \varepsilon &
\end{array}
$$

En réalité on peut assigner des *poids* aux différentes règles de production, qui reflètent leur probabilité d'apparition. On a alors affaire à une **grammaire stochastique hors-contexte (SCFG)**. Bien sûr, si on veut prendre en compte des informations plus fines que le biais de fréquence des six types d'appariements canoniques, la grammaire doit être enrichie, et les règles de production deviennent beaucoup plus nombreuses.

La grammaire est hautement ambiguë, il existe donc pour une séquence donnée un nombre considérable de dérivations possibles : au moins autant que de structures. Selon ce modèle, chercher la structure d'une séquence revient à chercher quel est l'arbre de dérivation de poids maximal, son poids étant la somme des poids des règles appliquées.

Les grammaires stochastiques hors-contexte ne permettent pas de modéliser les pseudonœuds, mais des extensions ont aussi été proposées qui prennent en compte certaines classes de pseudonœuds à l'aide d'intersection de grammaires hors-contexte (Lefebvre : grammaires multi-bandes, Brown [BW95]) ou en définissant des grammaires plus larges (Eddy et Rivas [RE99]).

La recherche de l'arbre de dérivation de poids maximal fait appel classiquement à trois types d'algorithmes propres aux grammaires stochastiques, et qui sont ici appliqués à la recherche de structure de l'ARN.

- *l'algorithme inside* calcule le score d'une séquence donnée, relativement à un modèle SCFG paramétré, c'est-à-dire la probabilité qu'elle soit dérivée par le modèle.
- *l'algorithme CYK (Cocke-Younger-Kasami)* est une variante de l'algorithme précédent qui donne le meilleur alignement de la SCFG sur la séquence, c'est-à-dire l'arbre de dérivation de probabilité la plus élevée.
- *l'algorithme EM (expectation maximisation)* et *l'algorithme inside-outside* entrainent la grammaire sur un jeu de séquences par un processus itératif (programmation dynamique récursive).

L'algorithme CYK a les mêmes complexités en espace et en temps que celui de Nussinov et Jacobson : $O(n^2)$ et $O(n^3)$. Chercher la structure secondaire d'une unique séquence par l'algorithme de Nussinov et Jacobson revient à considérer qu'elle est engendrée par une grammaire dont les règles d'appariement ont un poids unitaire et toutes les autres un poids nul. Si l'on considère un alignement de séquences comme une unique séquence consensus, engendrée par une grammaire relativement simple (celle que nous avons présenté pour une séquence, avec comme poids l'information mutuelle calculée entre les différentes colonnes), la prédiction de la structure secondaire avec l'algorithme CYK revient à incorporer les colonnes en utilisant

grosso-modo l'algorithme de Zuker sur l'alignement, avec l'information mutuelle comme score à la place de l'énergie.

Le programme **COVE**, créé par Eddy et Durbin [ED94, DEKM98] est $O(n^2m)$ en espace et $O(n^3m^3)$ en temps, n désignant la longueur de l'alignement, m le nombre de séquences. En pratique COVE atteint ses limites avec des séquences de 140 bases environ. Des perfectionnements ont été proposés : un prétraitement utilisant l'algorithme d'échantillonnage de Gibbs [GHH+94], des heuristique de type Divide & Conquer [Gra95]. Le programme **PFOLD** [KH99] est une amélioration qui prend en compte l'arbre phylogénétique. Leur grammaire est entrainée sur un ensemble de séquences qui se veut le plus neutre possible pour ne pas tenir compte des biais qui pourraient être engendrés par une famille particulière d'ARN homologues. À titre indicatif, les auteurs ont constitué leur ensemble d'entrainement d'environ 2000 ARNt et 300 ARNr ssu.

2.4.4 L'approche comparative : un contexte sécurisé

L'approche comparative tire parti uniquement des informations *transverses* en considérant un ensemble de séquences semblables alignées sur leur structure primaire. Quand on peut l'appliquer, c'est la méthode qui donne les meilleurs résultats sur des séquences de toutes tailles.

Pour les familles d'ARN assez conséquentes (au moins plusieurs dizaines de séquences, voire plusieurs centaines ou milliers) la méthode donne des résultats très précis et très fiables. Les structures prédites ont souvent anticipé celles obtenues par des méthodes physiques (cristallographie et rayons X). Historiquement, avant même que les méthodes physiques permettent d'observer les structures, c'est cette méthode qui a permis de découvrir en 1969 la première structure secondaire d'ARN. M. Levitt *et al.* ont inféré *à la main* la structure en feuilles de trèfle de l'ARN de transfert à l'aide de 14 séquences homologues et de quelques indices en faveur de cette structure [Lev69].

Pour pouvoir déceler des covariations, l'alignement initial doit bien sûr être parfait à la base près. Le point sensible de cette méthode est le nombre de séquences dont on dispose, ceci pour plusieurs raisons.

Si les séquences sont bien conservées, on pourra obtenir un bon alignement, mais dans ce cas, il en faudra beaucoup pour pouvoir déceler des covariations. Si au contraire les séquences sont fort divergentes, des covariations apparaitront plus vite, mais les séquences seront en retour difficiles à aligner correctement. Là encore, le fait d'en avoir beaucoup constitue un avantage, car cela permet de construire l'alignement multiple selon un ordre qui tient compte de la phylogénie des espèces, et respecte mieux leurs divergences.

Pour y remédier, les alignements sont souvent corrigés à la main. Et les méthodes automatisées procédent par itérations convergentes où l'alignement est corrigé et affiné à chaque étape. Il en résulte nécessairement un algorithme très lourd à mettre en œuvre lorsqu'on connait la difficulté de l'alignement multiple classique (*i.e.* sur la structure primaire).

Dans le cas où l'on n'a que quelques séquences, dès qu'elles présentent assez de divergences pour qu'on ait du mal à les aligner, la méthode devient totalement inefficace, qu'elle soit mise en œuvre manuellement ou automatisée par un algorithme.

2.5 Les méthodes hybrides

Les deux grandes familles d'approches – thermodynamique et analyse comparative – donnent lieu à des algorithmes qui se placent, on l'a vu, dans des contextes extrêmes où l'on utilise tantôt uniquement l'information contenue dans les séquences, tantôt uniquement les informations mutationnelles d'une séquence à l'autre. De nouvelles méthodes sont apparues qui tirent parti des deux approches en utilisant à la fois l'énergie libre et l'information de conservation de la structure (très souvent vue au travers des covariations), c'est pourquoi nous les nommons **hybrides**. Dégager plus d'informations des séquences considérées permet de restreindre considérablement le nombre de séquences nécessaires. Ainsi ces méthodes se placent implicitement dans le contexte moderne où l'on dispose souvent d'un petit jeu de séquences homologues dont on ne connait pas toujours la fonction. Sans se vouloir exhaustive, cette section tente de montrer la diversité des approches du problème selon ce contexte moderne. Un petit tableau récapitulatif est proposé en dernière page (page 50).

2.5.1 Des pistes assez diverses

Les méthodes que nous présentons ont en commun leur contexte d'application : on leur fournit en entrée n séquences, alignées ou non selon les cas, et les algorithmes proposent en sortie une prédiction qui s'apparente à une structure commune pour ces n séquences. Cette structure peut prendre différentes formes, il peut s'agir d'un alignement *structural* comportant la structure consensus, ou bien d'une ou de plusieurs structures par séquence, mais quelque soit la forme de cette prédiction, le principe des algorithmes est de rechercher une structure *commune* en utilisant les deux types d'information, énergétique et mutationnelle.

X2S

X2S [JW99] est une méthode qui tire parti des deux approches, thermodynamique et analyse comparative. L'algorithme est structuré en plusieurs étapes :

étape 1 – Les séquences sont alignées par un programme d'alignement multiple classique (CLUSTALW par exemple).

étape 2 – Pour chaque séquence on calcule selon une légère variante de la méthode de Zuker une matrice de repliements E, contenant les scores énergétiques des différents appariements.

étape 3 – Les meilleurs appariements dans la matrice sont stockés dans une liste. Les auteurs gardent arbitrairement les 5000 meilleurs (indépendemment de la longueur de la séquence) mais précisent que les 2000 meilleurs suffisent habituellement. Pour chacun de ces appariements :

étape 3.1 – Un score de covariations C_{ij} est calculé pour chaque position d'appariement (i,j) à l'aide de l'alignement multiple.

étape 3.2 – Le score énergétique E_{ij} est corrigé pour défavoriser les appariements à longue portée.

étape 3.3 – Une fonction de coût L_{ij} prend en compte la taille de la boucle.

étape 3.4 – Le score final est une combinaison linéaire de ces trois scores : $score_{ij} = w_C C_{ij} + w_E E_{ij} + w_L L_{ij}$.

étape 4 – Les 5000 appariements potentiels sont reclassés selon ce score, puis les 1000 meilleurs sont conservés et incorporés de manière gloutonne, en gérant les chevauchements de tiges.

étape 5 – Une fois la structure secondaire construite, elle est complétée par un dernier balayage de la matrice énergétique, à l'aide d'un seuil concernant la taille de la tige et le pourcentage de mésappariements.

CIRCLE

Le **Maximum Weighted Matching** est une méthode qui permet d'obtenir, à partir d'un graphe pondéré, un sous-graphe vérifiant certaines contraintes. Elle a été appliquée à l'alignement de séquences par Tabaska *et al.* [TCGS98]. L'alignement des séquences est vu comme un graphe dont le poids des arcs est construit en plusieurs étapes.

Le score d'un arc est une combinaison d'information énergétique (les *energy dot-plots* tels ceux générés par MFOLD ou des 'helix-plots' dont la construction est détaillée ci-dessous) et d'information mutuelle (les covariations). Les auteurs mentionnent que l'information énergétique utilisée seule ne donne pas de bons résultats et sera donc toujours combinée avec l'information mutationnelle. Ils indiquent aussi que l'information mutuelle seule conduit l'algorithme à manquer beaucoup d'appariements évidents à cause de l'absence ou de l'insuffisance des covariations.

Voici le détail de la construction d'un *helix plot* (dot plot pour les hélices) :

étape 1 – On crée l'alignement multiple des séquences.

étape 2 – On construit pour chaque séquence une matrice carrée contenant dans chaque cellule un score qui indique dans quelle mesure les deux bases considérées peuvent s'apparier ou non : $Score = \alpha$ (petit nombre positif) si la paire est de type Watson-Crick ou de type G=U, $Score = \beta$ (grand nombre négatif) sinon. La paire se voit aussi attribuer un gros malus si elle participe à une délétion asymétrique dans une hélice.

étape 3 – Pour chaque séquence, on balaie la matrice précedente à la recherche d'hélices potentielles. Durant le balayage, le score des paires contenues dans des hélices dont la taille est inférieure à un seuil préfixé sont convertis en mauvais scores, et les scores des paires participant à la formation d'hélices suffisamment longues se voient incrémentés d'un facteur proportionnel à la taille de cette hélice (ce bonus peut être comparé en quelque sorte à l'énergie d'empilement du modèle thermodynamique).

étape 4 – On somme les matrices individuelles selon l'alignement multiple préalable.

Les scores peuvent aisément incorporer des contraintes classiques (appariement ou non-appariement d'une paire, le partenaire éventuel étant connu ou inconnu).

L'article de Tabaska *et al.* [TCGS98] présente les résultats du programme sur des ARNt. La pondération des arcs est engendrée à partir d'un alignement de 556 ARN de transfert. L'algorithme fournit une structure en feuilles de trèfle avec aussi des interactions tertiaires, mais dans la structure obtenue toutes les bases sont appariées ce qui oblige à appliquer un filtre en sortie pour ne garder qu'une partie de ces interactions. Le programme présente aussi de bons résultats à partir d'un alignement de 33 ARN SRP et de 2849 ARNr (petite sous-unité 16S).

RNAGA

RNAGA [CLM00] est un algorithme génétique. Il opère en trois étapes :

étape 1 – Pour chaque séquence les structures potentielles sont calculées et optimisées par un algorithme génétique, le critère d'adaptation étant l'énergie libre de la structure.

étape 2 – Pour chaque structure, il est défini une mesure de conservation par rapport aux structures trouvées dans les autres séquences. Là encore, c'est un algorithme génétique, utilisant cette mesure comme nouveau critère d'adaptation, qui optimise l'ensemble des structures en cherchant à améliorer la similitude structurelle entre les séquences.

étape 3 – L'algorithme fournit au final les 10 meilleures structures, selon certains critères de stabilité et de similitude.

Les auteurs valident leur méthode sur différents ensembles de séquences : un jeu de 20 ARNt (env. 80 nucl.), un jeu de 25 ARNr 5S (env. 120 nucl.) et un jeu de 7 ARN RRE-HIV (env. 230 nucl.).

2.5.2 La piste de Sankoff

Les récurrences de Sankoff

Les formules de récurrence de **Sankoff** [San85] étendent à deux séquences celles de Nussinov et Jacobson [NJ80]. Donnons-nous deux séquences $s := s\ [1\ ..\ n]$ et $t := t\ [1\ ..\ m]$. On remplit une matrice S de taille $O(m^2 \times n^2)$: chaque cellule $S\ [i,\ j,\ k,\ l]$ de la matrice de dimension 4 contient l'énergie du meilleur repliement commun des segments $s\ [i\ ..\ j]$ et $t\ [k\ ..\ l]$. Ce repliement est calculé récursivement en interrogeant la dernière base de chacun des deux segments, comme dans l'algorithme de Nussinov : est-elle libre ? est-elle appariée avec une autre base du segment ? Si elle est libre, on regarde si on a affaire à une insertion / délétion, si elle est appariée, on envisage qu'elle le soit simultanément sur les deux séquences ou non. Pour initialiser la matrice, on calcule les matrices N de Nussinov pour chacune des deux séquences. Ensuite on retrouve la structure en pistant à l'envers dans la matrice d'une façon similaire à celle mise en œuvre dans l'algorithme de Nussinov et Jacobson (page 31).

Initialisation de la matrice : temps $O(n^3)$

- $S\ [i\ ..\ i,\ k\ ..\ l] := N\ [k\ ..\ l] \qquad (1 \leqslant i \leqslant n) \qquad (1 \leqslant k \leqslant l \leqslant m)$
- $S\ [i\ ..\ i-1,\ k\ ..\ l] := N\ [k\ ..\ l]$

- $S\ [i\ ..\ j,\ k\ ..\ k] := N\ [i\ ..\ j] \qquad (1 \leqslant i \leqslant j \leqslant n) \qquad (1 \leqslant k \leqslant m)$
- $S\ [i\ ..\ j,\ k\ ..\ k-1] := N\ [i\ ..\ j]$

Récurrence : temps $O(m^3 \times n^3)$

$$S\,[i\,..\,j,\,k\,..\,l] := min \qquad (1 \leqslant i < j \leqslant n) \qquad (1 \leqslant k < l \leqslant m)$$

$$\begin{cases}
\bullet\ S\,[i\,..\,j,\,k\,..\,l-1] + indel\,(-,\,t[l]) \\
\bullet\ S\,[i\,..\,j-1,\,k\,..\,l] + indel\,(s[j],\,-) \\
\bullet\ S\,[i\,..\,j-1,\,k\,..\,l-1] + align\,(s[j],\,t[l]) \\
\\
\bullet\ S\,[i\,..\,j,\,k\,..\,y-1] + S\,[0\,..\,0,\,y+1\,..\,l-1] + bind\,(-,\,-,\,t[y],\,t[l]), \\
\bullet\ S\,[0\,..\,0,\,k\,..\,y-1] + S\,[i\,..\,j,\,y+1\,..\,l-1] + bind\,(-,\,-,\,t[y],\,t[l]), \\
\bullet\ S\,[i\,..\,x-1,\,k\,..\,l] + S\,[x+1\,..\,j-1,\,0\,..\,0] + bind\,(s[x],\,s[j],\,-,\,-), \\
\bullet\ S\,[i\,..\,x-1,\,0\,..\,0] + S\,[x+1\,..\,j-1,\,k\,..\,l] + bind\,(s[x],\,s[j],\,-,\,-), \\
\bullet\ S\,[i\,..\,x-1,\,k\,..\,y-1] + S\,[x+1\,..\,j-1,\,y+1\,..\,l-1] + cobind\,(s[x],\,s[j],\,t[y],\,t[l]), \\
\qquad (i+1 \leqslant x \leqslant j-2) \qquad (k+1 \leqslant y \leqslant l-2)
\end{cases}$$

Appliquées directement, la complexité algorithmique des formules de récurrence ($O(n^4)$ en espace, et $O(n^6)$ en temps) les rendent impraticables sur des séquences de taille supérieure à 100, ce qui est le cas de la plupart des séquences dont on voudra chercher la structure. Dans les faits, on observe que c'est d'abord l'espace qui est un facteur limitant. Les méthodes qui suivent (RNALIGN, FOLDALIGN, DYNALIGN, et la méthode de Zhang et Wang) sont basées sur les formules de Sankoff. CARNAC, l'algorithme que nous présentons au chapitre suivant, utilise lui aussi une variante de ces récurrences.

RNALIGN

Basé sur les récurrences de Sankoff, le programme **RNALIGN** [CM94] se propose d'aligner une séquence contre une structure. L'algorithme utilise des points d'ancrage sur la structure primaire pour réduire la complexité en temps ($O(n^4)$ en espace, $O(n^5)$ en temps). Bafna *et al.* [BMR95] en proposent une amélioration qui baisse le temps à $O(n^4)$, mais reste impraticable sur la plupart des séquences.

FOLDALIGN

Le programme **FOLDALIGN** est basé sur les mêmes récurrences mais propose un point de vue un peu différent, qui est explicite dans le titre de l'article [GHS97] : "Trouver le motif commun le plus significatif d'un ensemble de séquences d'ARN". A partir d'un jeu de séquences homologues, l'algorithme replie les séquences deux à deux, en s'interdisant les ramifications, ce qui permet de réduire la complexité en temps à $O(n^4)$. La complexité spatiale reste en $O(n^4)$. Le programme fournit, pour chaque sous-ensemble de l'ensemble de départ, la tige commune la plus probable. Comme la complexité reste élevée, les séquences sont nécessairement courtes (100 nucléotides maximum) et le motif non-ramifié commun est très souvent une tige terminale. L'algorithme en fournit un alignement multiple précis.

DYNALIGN

DYNALIGN [MT02] est un enrichissement de l'algorithme de Sankoff selon deux directions. D'une part les auteurs ont intégré les paramètres d'énergie, de la même façon que l'algorithme thermodynamique provient de celui de Nussinov. D'autre part, ils autorisent un décalage borné entre les séquences, ce qui permet de réduire considérablement la complexité. Cependant le temps de calcul explose dès qu'on augmente un peu ce décalage autorisé.

La méthode heuristique de Zhang et Wang

Wang et Zhang [WZ99] ont proposé un algorithme de type Sankoff combiné avec une heuristique. La méthode repose sur la supposition que la structure d'une séquence d'ARN apparaît toujours parmi les structures sous-optimales dans l'algorithme thermodynamique MFOLD. A partir de trois séquences homologues repliées de manière indépendante par MFOLD, leur algorithme prend en entrée toutes les tiges des structures sous-optimales. Les tiges sont repliées par programmation dynamique, selon une version adaptée des récurrences de Sankoff:

- Le repliement se fait à plus haut niveau sur les tiges au lieu des nucléotides. Il faut adapter les récurrences afin de gérer les problèmes de chevauchements, qui n'apparaissent pas au niveau des bases.
- Les récurrences sont adaptées à trois séquences au lieu de deux dans l'algorithme initial de Sankoff.
- Le programme impose une borne sur le décalage autorisé entre les tiges.

La méthode est présentée de manière théorique et appliquée à un unique exemple (trois ARN viraux). Le programme n'est pas disponible.

2.5.3 Tableau synthétique des méthodes

Comme nous l'avons constaté, les méthodes qui prédisent la structure secondaire à partir d'un petit jeu de séquences homologues en usant de deux types d'informations, intrinsèques (énergétiques) et transverses (mutationnelles), explorent des pistes très diverses et se placent dans des contextes différents. Les prédictions qui en résultent sont aussi fort diverses dans leur forme puisqu'elles répondent à la question implicitement posée par le contexte.

Nous proposons cependant dans la TABLE 2.1 une vue synthétique des méthodes pour lesquelles le programme est disponible. Nous montrerons aux chapitres 3 et 4 où se place CARNAC, le programme que nous proposons, par rapport à ces méthodes.

méthode	séquences nb.	séquences ali.	algorithme	sortie
MFOLD	1		prog. dyn.	une structure + sous-optimales
DYNALIGN	2		prog. dyn.	une structure commune + un alignement
RNAGA	n		algo. génétique	10 structures possibles par séquence
X2S	n	×	algo. glouton	une structure consensus sur l'alignement
FOLDALIGN	n		prog. dyn.	la meilleure tige commune
PFOLD	n	×	SCFG	une structure consensus sur l'alignement
CIRCLE	n	×	MWM	une structure consensus sur l'alignement
CARNAC	n		prog. dyn.	une structure par séquence

TAB. 2.1 – *Vue sommaire des différentes méthodes. On indique le nombre de séquences à fournir en entrée, en précisant si elles doivent être alignées, le type d'algorithme, et ce que fournit le programme en sortie. Nous mentionnons CARNAC, qui fera l'objet des chapitres suivants.*

Chapitre 3

Le programme CARNAC

Les méthodes hybrides dont nous avons parlé précédemment répondent au problème de la prédiction de structure dans le cas où peu de séquences sont disponibles. Elles y répondent toutes en cherchant plus ou moins explicitement à prédire le maximum d'appariements possibles[1]. Par ailleurs, la qualité des prédictions n'est pas toujours constante et les programmes ne donnent pas de mesure de la justesse des appariements trouvés. Le détenteur des séquences d'ARN se retrouve ainsi avec une ou plusieurs structures dont la qualité peut être tantôt très bonne, tantôt très mauvaise sans qu'aucun indice explicite ne lui permette de le deviner.

Forts de cette constatation, nous avons tenté de réaliser un algorithme qui ne cherche pas nécessairement l'exhaustivité, mais plutôt la sûreté dans la prédiction, quitte à n'obtenir qu'une partie de la structure.

Le programme CARNAC (**C**omputer **A**lignment of **RNA** by **C**ofolding / **C**ouples d'**ARN** **A**lignés par **C**orepliement) se place dans le contexte des méthodes hybrides et suppose donc que plusieurs séquences homologues sont disponibles. Plusieurs signifie au moins deux. Nous présentons en premier lieu l'algorithme pour deux séquences : $CARNAC_2$. L'exposé est mené sur une famille de séquences particulières – des RNases P – afin de comprendre sur un exemple concret d'où viennent les difficultés auquelles sont confrontées toutes les méthodes de prédiction. Nous comparons $CARNAC_2$ aux méthodes qui peuvent être appliquées avec seulement deux séquences (MFOLD et DYNALIGN). Nous voyons ensuite comment la méthode, en s'étendant à n séquences, permet d'améliorer encore la qualité des résultats. La comparaison de CARNAC aux méthodes qui s'appliquent à n séquences sera présentée au chapitre suivant.

3.1 $CARNAC_2$: corepliement de deux séquences

La recherche de la structure secondaire propre à une séquence amène souvent plusieurs structures potentielles *concurrentes*. Selon le modèle thermodynamique, les structures sous-optimales sont des structures concurrentes, dans le sens où la bonne structure n'est pas nécessairement celle d'énergie minimale. Lorsqu'on considère deux séquences, le fait de savoir qu'elles partagent une structure commune est une information très utile. Logiquement, cette

1. FOLDALIGN est un cas particulier, car le programme cherche une unique tige conservée et produit un alignement structural local sur cette tige.

information permet en quelque sorte de se frayer un chemin parmi les structures potentielles de chacune d'elles pour trouver la structure qui leur convient à toutes les deux. Une approche naturelle consisterait à produire, pour chaque séquence un ensemble de structures concurrentes, de les comparer deux à deux, puis d'élire celle qui leur est commune. Nous allons voir sur un exemple pourquoi cette solution n'est pas satisfaisante. Une seconde approche consiste à replier les deux séquences simultanément en les considérant comme une seule. Nous verrons que cette solution ne fonctionne pas non plus. Enfin nous décrirons la méthode que nous proposons.

3.1.1 Une structure commune à deux séquences

L'exemple qui nous guidera tout au long de ce chapitre est une famille d'ARN fonctionnels présents dans tous les organismes : les RNases P, qui sont des endoribonucléases. Leur activité la mieux caractérisée dans la cellule est la maturation des ARNt par clivage de la partie 5′ de l'ARN précurseur. Ayant la même fonction, les séquences homologues partagent une structure commune, avec quand même une certaine variabilité (une ou deux tiges peuvent disparaître d'une sous-famille à l'autre). La structure consensus comporte environ une quinzaine de tiges, plus deux pseudonœuds. On observe aussi beaucoup de variation sur la structure primaire. Ces variabilités, au niveau de la séquence comme au niveau de la structure, sont bien sûr d'autant plus grandes que les séquences sont éloignées phylogénétiquement.

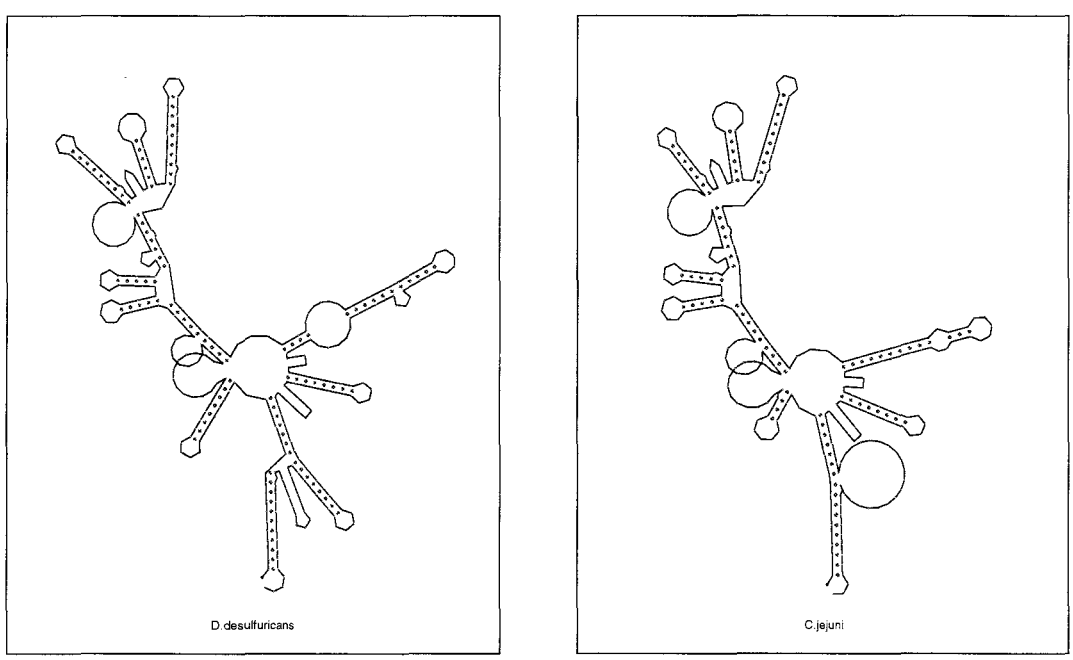


FIG. 3.1 – *D.desulfuricans et C.Jejuni – les structures proposées dans la base de données de J.Brown [Bro99] (sans les pseudonœuds).*

Nous avons cueilli deux ARN dans la banque de données de J.Brown [Bro99] : *Desulfovibrio desulfuricans*, qui est présentée comme la référence de sa sous-famille (*Delta Purple Bacteria*

RNase P) et *Campylobacter jejuni*, qui est la première séquence de la sous-famille suivante (*Epsilon Purple Bacteria RNase P*). En observant la structure qui y est proposée (FIG. 3.1) on remarque que les deux ARN partagent effectivement une structure commune, que la longueur de certaines tiges varie entre les deux séquences, et qu'une tige (en bas à droite) est absente dans la structure de la deuxième séquence.

Si l'on replie indépendemment les deux séquences avec MFOLD, on obtient un ensemble de structures sous-optimales. Utilisé avec les paramètres par défaut, MFOLD donne une collection de 18 structures sous-optimales pour *D.desulfuricans* et 6 pour *C.jejuni*. Comme on peut le constater sur les FIG. 3.2 et FIG. 3.3, les structures trouvées sont manifestement très diverses.

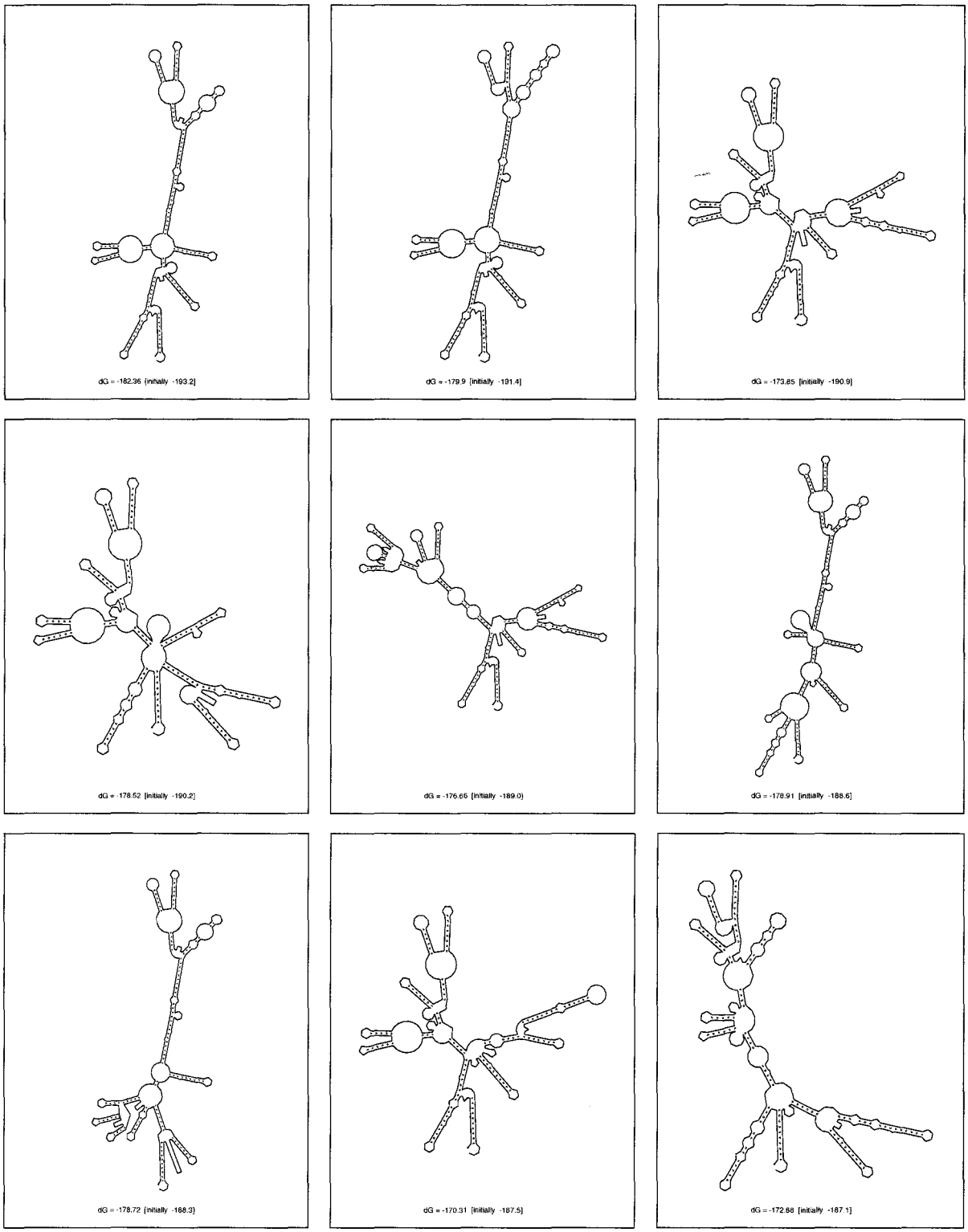
Sachant que les séquences partagent la même structure, on serait tenté de comparer les structures sous-optimales proposées pour chacune des deux séquences afin de voir si on n'en trouve pas une en commun. Mais si l'on examine attentivement les 18 structures obtenues pour *D.desulfuricans*, on s'aperçoit qu'aucune d'elles ne correspond à la vraie structure. Pour *C.jejuni* la première sous-optimale est celle qui est la plus proche de la vraie structure mais elle comporte des erreurs.

A vue d'œil, pour *D.desulfuricans*, les différents repliements proposés ne présentent pas de motifs récurrents. On peut juste remarquer la présence persévérante de deux tiges (tout en haut des structures) dans la plupart des repliements sous-optimaux, ce qui pourrait être un argument en leur faveur. Ceci invite à imaginer pour chaque tige un "indice de fiabilité" basé sur la fréquence d'apparition dans les structures sous-optimales. Une telle approche serait assez délicate à mettre en œuvre car il est impossible de donner toutes les structures sous-optimales, celles-ci devenant vite trop nombreuses lorsqu'on s'éloigne un peu de la valeur optimale de l'énergie. L'algorithme applique donc un filtre qui élimine les structures trop proches les unes des autres. Pour modifier les caractéristiques de ce filtre dans MFOLD, l'utilisateur peut jouer sur deux paramètres : `Percent suboptimality`[2] qui borne l'écart par rapport à la structure d'énergie la plus faible, et `Window`, qui définit la distance minimale entre deux structures pour être présentées comme différentes. Un indice basé sur la fréquence d'apparition dans les structures sous-optimales risque d'être fortement biaisé par ce filtrage. Ce filtrage nuit aussi à la recherche d'une structure commune *a posteriori*. Cela nous amène à chercher la structure commune directement au cours du repliement.

Si l'on considère à présent les deux séquences, non plus séparément, mais alignées, cet alignement constitue une sorte de séquence consensus qui peut être traité comme une unique séquence (pourvu qu'on adapte correctement les paramètres énergétiques en tenant compte des mutations et covariations).

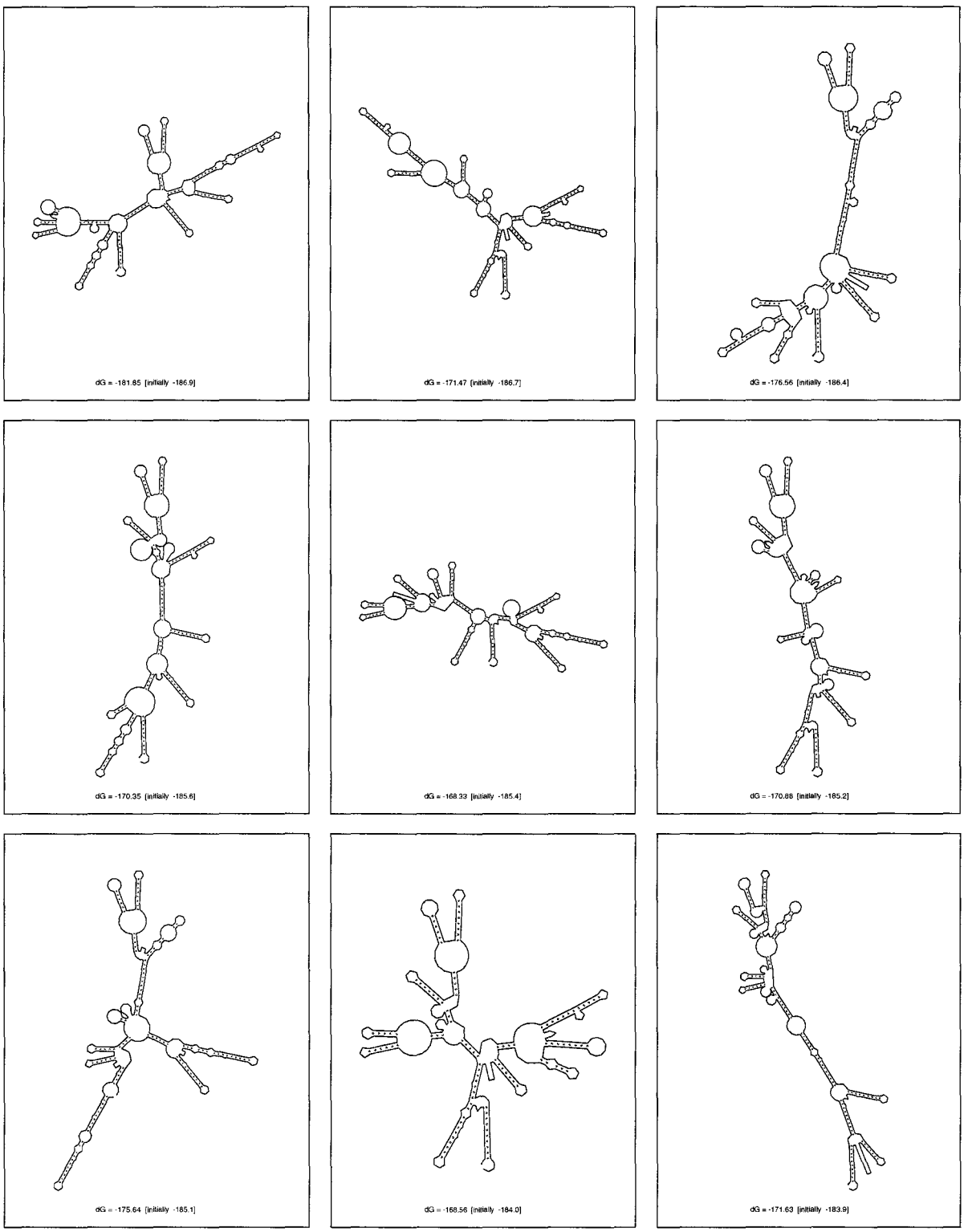
Malheureusement, construire un bon alignement, c'est-à-dire un alignement qui respecte la structure, est difficile lorsqu'on ne dispose que de deux séquences. Les séquences de *D. desulfuricans* et *C. jejuni* sont assez divergentes et la FIG. 3.4 montre que l'alignement classique par l'algorithme de Needleman & Wunsch est mauvais : les tiges de la vraie structure ne s'alignent pas correctement. On observe en effet très souvent que lorsque deux bases appariées sont alignées, leurs partenaires respectifs ne sont pas alignés. On observe aussi des bases appariées "fermantes" alignées avec des bases "ouvrantes".

2. Nous avons utilisé la valeur par défaut du pourcentage de sous-optimalité, c'est-à-dire 5%. En augmentant à 10% on obtient 19 structures pour *D.desulfuricans*. En poussant à 100% on obtient toujours les mêmes 19 structures. La plupart d'entre elles, dont les cinq premières, sont strictement identiques aux 18 qu'on obtient avec 5%. Les autres présentent des variations minimes.



(a) Sous-optimales 1 à 9

FIG. 3.2 – *Structures sous-optimales pour D.desulfuricans obtenues avec MFOLD – Paramètres par défaut.*



(b) Sous-optimales 10 à 18

FIG. 3.2 – *Structures sous-optimales pour D.desulfuricans obtenues avec MFOLD – Paramètres par défaut.*

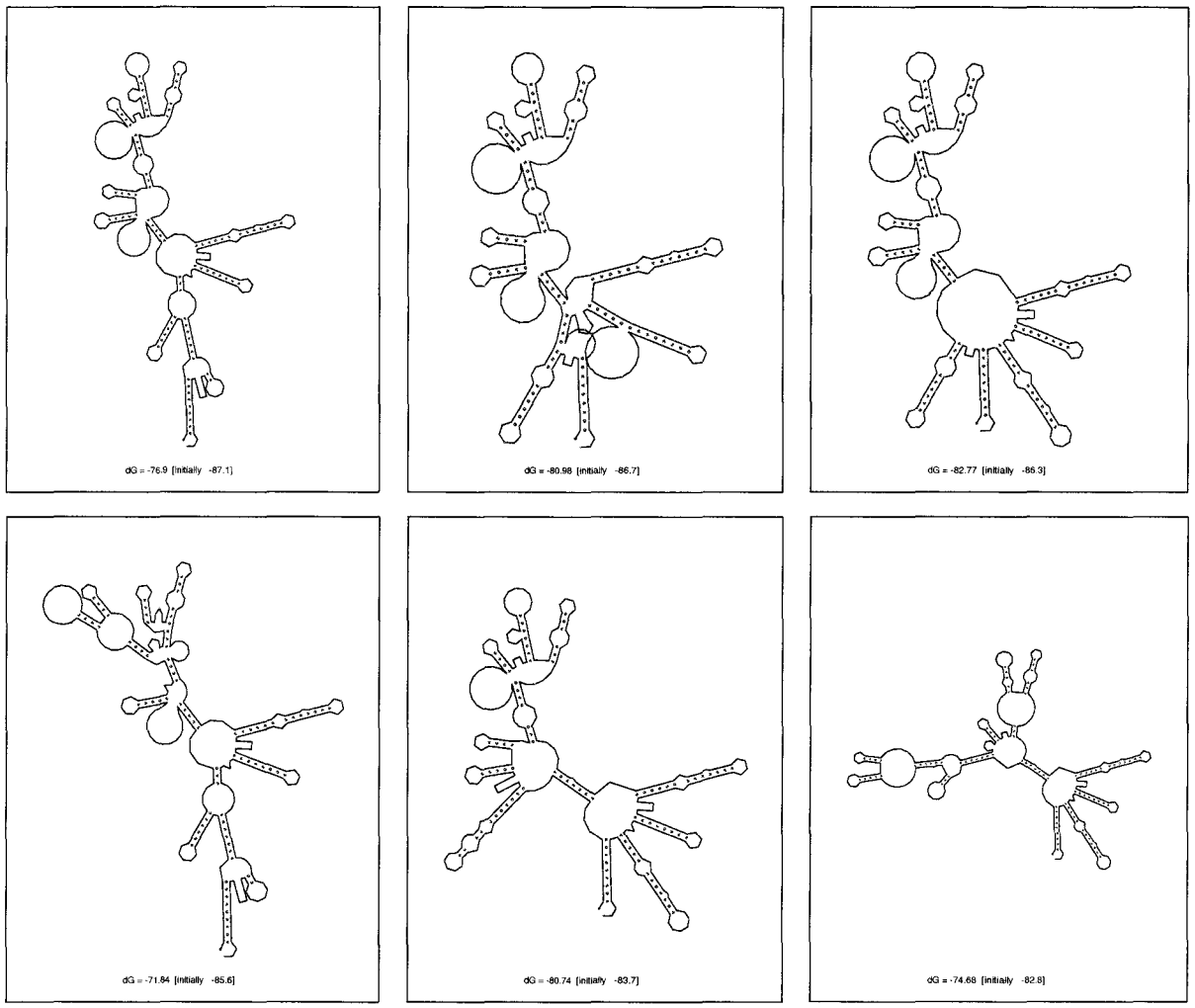


FIG. 3.3 – *Structures sous-optimales pour C.jejuni obtenues avec MFOLD – Paramètres par défaut.*

Pour replier les deux séquences comme une seule avec la méthode énergétique, il faut un alignement parfait, or pour former un alignement correct, il faudrait tenir compte de la structure. Devant cette réflexion tautologique, on se dit qu'une solution pourrait être de rechercher la structure et d'aligner simultanément. C'est précisément ce que réalise l'algorithme de Sankoff (présenté page 48). Tel quel, l'algorithme est impraticable à cause de sa complexité en espace et en temps. La méthode que nous présentons est bâtie autour des mêmes idées, mais utilise une heuristique et des optimisations qui la rendent praticable, voire même extrêmement rapide dans la plupart des cas.

3.1.2 L'algorithme CARNAC$_2$

CARNAC$_2$ est la première étape de CARNAC. Le programme permet d'inférer une structure commune à deux séquences. Il se scinde en plusieurs étapes chronologiques et modulaires que nous allons décrire successivement (la FIG. 3.5 en présente un résumé visuel). Nous of-

```
<<<<<<<<<<<[[[[[[[ << <<<<<    >>>>             >>             <<<<        <<<<<[[[[[
GGAGUCGGACGGAUCGUCGCCGCGGGGGCAACUCC.............GGGGAGGAAAGUCCGGGCUCCAAAGGGCAGAACGCUGGAUAA
        AAGCAUAGU............AAAUGCUCGCUUCUUUUUAGGAGAGGAAAGUCCGAGCUGCUAAAGACA.AAC.....AUUC
        <<<<<<<<<            <[[[[[[ <<<<<    >>>>>            <<<<          <<     <<<[

  ]]   ]]]  [[  [[[    ]]] ]]<<   <<              [     [[[[[[[[   ]]]]]]] ]]       [[[
CAUC...CAG..GG..AGGGCAACCU.CCGGACAGCGCCACAGAAAGCAAAC.....CGCCCGGCCUCGGCCGGGUAAGGGUGAAACGGU
CAUCUAACAGAUGGCUAGGGUAACCUAAGGGAUAGUGCAACAGAAAGAAAACUACCACGCAAG.........UGGAAAAGGUGAAACGGC
[[[[     ]]]]][[[[[    ]]]] ]<<   <<              [[ [[[[    ]         ]]]   ]]       [[

[[[      ]]]]]]  [[[[[[    [[    ]]    ]]]] ]]         >>>>>>>>>>>>>   <<<<      <<<<<<
GGUGUAAGAGACCACCAGAUGCCG....UGGUGACA....CGGC.AUGCUCGGCAUACCCCGUUCGGAGCAAGACCAAAUAGGGAAGG..
GGGGUAAAAGCCCACCAG....CGAUUUUGGUAACAAUUUCGGCUAU.....GUAAACCCAAUGUGCAGCAAGA......AGGGAUGGUU
[[[      ]]] ]]      [[[  [[[    ]]]  ]]]             >>>>>>>>>>>>>   <<      <<<<    <<

<<<<<     >>>>>            >>>>>>     >>>>      <<<<<<<<    >>>>>>>          ]]]]]]] <<
CGGCCGGCCCGGCCG..........AAGCCUUCCGGGUAGGUUGCUUGAGGGUGUGGGCAACCGCACUCCUAGAGGAAUGACGGUCACAC
AG..........CGUCUUUGUUUUAACCCUUC..........GCUUGAUUUUGUUUGCAAAAACAAAACUAGAUAAAUGA.........
<<                    >>>>>>>>>>              <<<<<<<<    >>>>>>>          ]]

<<<<    >>>>>>                 > >>>>>>>>>>
GCGGGCAACCGUGUGGACAGAACCCGGCUUACAGUCCGACUCCC...GCA
GCAUUCAA.......GACAGAACUCGGCUUA.........UCGCUAUGCUUUUU
]]]]                                  > >>>>>>>>>
```

FIG. 3.4 – *Alignement global des deux séquences D.desulfuricans et C.jejuni réalisé avec l'algorithme classique de Needleman & Wunsch. On a indiqué sur cet alignement les appariements secondaires de la vraie structure pour chacune des deux séquences sous la forme d'une expression parenthésée (voir* FIG. *1.16 au premier chapitre). L'alternance des symboles '<' et '[' permet juste de faciliter la lecture et l'identification des tiges.*

frons en entrée deux séquences *non alignées* à $CARNAC_2$, qui fournit finalement une structure commune sous la forme d'une liste de tiges pour chacune des deux séquences [PTD03]. Le programme $CARNAC_2$ est implémenté en C et disponible sur le web[3].

Etape 1 – $CARNAC_2$ génère les tiges potentielles maximales pour chaque séquence en utilisant les paramètres énergétiques du modèle thermodynamique.

Etape 2 – Puis l'algorithme détecte entre les deux séquences d'éventuelles zones exceptionnellement bien conservées. Ces zones constitueront des *points d'ancrage* pour filtrer les deux jeux de tiges, afin de réduire considérablement le nombre de tiges à replier.

Etape 3 – Les deux jeux de tiges sont filtrés en examinant d'une part s'il y a des covariations, d'autre part la cohérence des tiges par rapport aux points d'ancrage : tous les couples de tiges possibles sont formés et on attribue à chaque fois un booléen qui indique si ces deux tiges peuvent être pliées simultanément – on dira alors qu'elles sont **copliables**.

Etape 4 – Les deux jeux de tiges *filtrés* sont finalement confrontés pour en extraire par programmation dynamique (selon une adaptation des récurrences de Sankoff) le sous-ensemble commun d'énergie optimale constituant une structure secondaire.

3. $CARNAC_2$ accepte en entrée le format FASTA, et donne en sortie différents formats. Le format CONNECT, en particulier, peut être parsé et visualisé par NAVIEW. Les sources du programme sont disponibles à l'adresse suivante : http://www.lifl.fr/~perrique/rna/

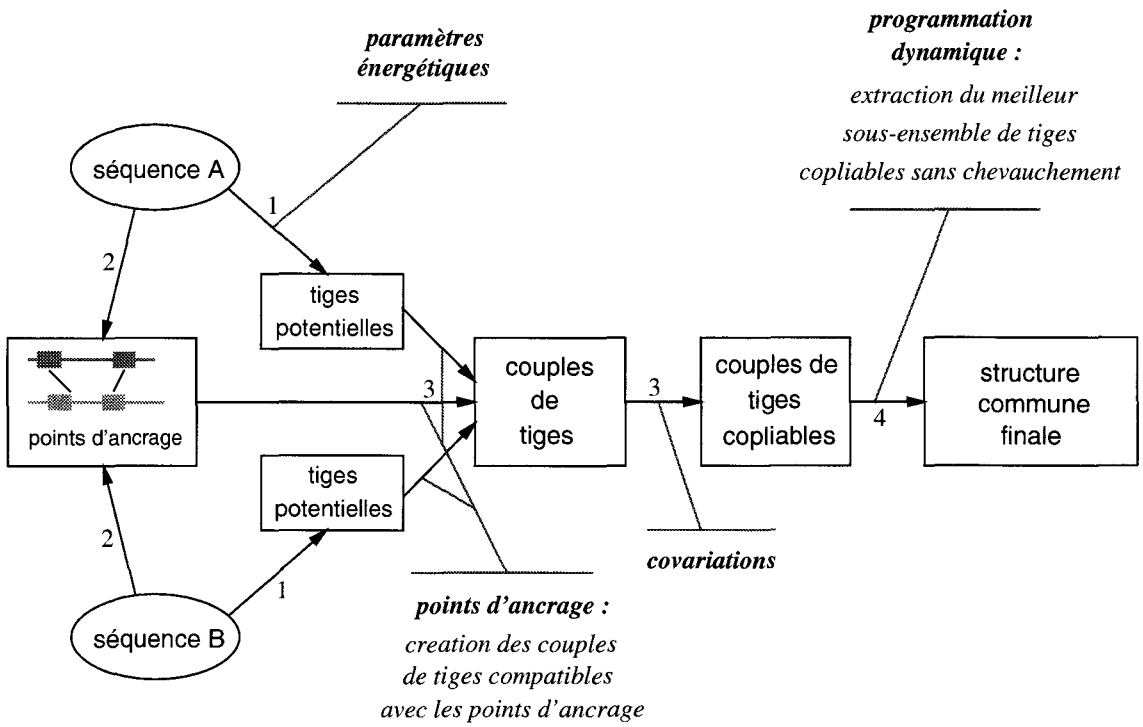


FIG. 3.5 – *Synopsis de CARNAC$_2$. Les nombres sur les flèches indiquent les quatre étapes.*

Etape 1 – La recherche des tiges

Cette étape est présentée ici comme un calcul interne propre à la méthode. Il s'agit en réalité d'une étape proposée *par défaut*. Le caractère modulaire de la méthode autorise en effet à utiliser en entrée n'importe quel jeu de tiges proposé par l'utilisateur, et l'implémentation du programme offre cette option. A la lumière de l'exemple présenté au début du chapitre, on pense en particulier aux jeux de tiges issus des repliements sous-optimaux calculés par MFOLD. Le programme CARNAC$_2$ se *brancherait* en quelque sorte sur la sortie de MFOLD.

Les tiges potentielles recherchées par CARNAC$_2$ sont des empilements *maximaux* de paires de bases. En d'autres termes, une tige peut contenir des mésappariements, mais doit toujours se terminer sur un appariement canonique, c'est-à-dire $A = U$, $G \equiv C$ ou $G = U$ (FIG. 3.6).

Les tiges sont recherchées simplement en utilisant une partie de la dernière version des paramètres thermodynamiques de Turner *et al.* [JTZ89]. Certaines adaptations ont dû être faites. Nos tiges sont précalculées indépendemment les unes des autres : elles ne délimitent donc pas de boucles internes, ainsi nous avons mis de coté tout ce qui concernait les boucles internes. Le traitement des tiges terminales dans ce modèle n'est pas le même que celui des tiges internes : leur énergie est souvent bonifiée. Or dans notre cas, on doit *décider* qu'une tige est terminale en fonction de la distance entre son ouverture et sa fermeture : lorsque la taille de la boucle qu'elle ferme n'excède pas huit nucléotides, nous lui appliquons le bonus. Enfin nous n'avons pas utilisé les paramètres concernant les mésappariements, car ils ne semblaient pas adaptés à la recherche de tiges seules. Certaines valeurs dans la table, au lieu d'être positives (c'est-à-dire destabilisatrices), sont presque nulles, mais *négatives* et apportent ainsi

une contribution bénéfique à la tige. L'utilisation de ces paramètres amène beaucoup trop de fausses tiges en entrée. Nous les avons donc écartés pour procéder de la façon suivante : une fois l'ensemble des tiges maximales calculé, nous examinons si certaines tiges peuvent s'empiler, créant ainsi une unique tige, plus longue, comportant un ou deux mésappariements. Ces empilements reçoivent un léger malus, et sont ajoutés à notre jeu de tiges.

Aucune optimisation algorithmique n'est faite sur ce calcul préliminaire car ce n'est pas la partie coûteuse de l'algorithme. Une table de programmation dynamique en $O(n^2)$ est calculée, où n est la longueur de la séquence, puis les tiges sont extraites de la matrice si elles sont supérieures à un certain seuil.

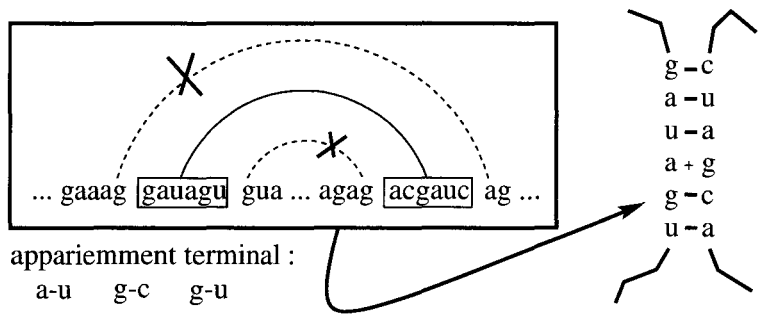
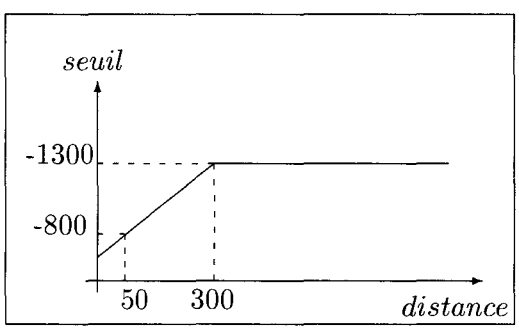


FIG. 3.6 – *Les tiges potentielles cherchées par $CARNAC_2$ sont des tiges maximales : elles se terminent sur un appariement canonique, juste avant un mésappariement.*



Le seuil de sélection des tiges est une fonction affine par morceaux, qui dépend de la distance entre l'ouverture et la fermeture de la tige. Les valeurs sont empiriques, elles ont été optimisées manuellement sur un vaste ensemble de séquences de diverses longueurs, de manière à donner au seuil un caractère aussi universel que possible.

Une énergie de -800[4] correspond habituellement à une tige de longueur 3 ou 4, et -1300 à une tige de longueur 6 ou 7. Le seuil peut éventuellement être modifié par l'utilisateur pour prendre en compte le biais de contenu en GC : $seuil = seuil \times (1 + (50 - GC\%)/50)$.

La FIG. 3.7 illustre la prolifération des tiges potentielles, et les chevauchements incessants qui se produisent entre les tiges. Elle donne une idée visuelle de la difficulté d'extraire parmi le jeu de tiges le sous-ensemble qui correspond à la structure de la molécule. On pourrait dire en quelque sorte que la séquence est couverte de tiges potentielles. Toute la suite de l'algorithme consiste à utiliser d'autres informations pour extraire de cet ensemble de tiges le jeu de tiges correspondant à la structure de la molécule.

4. Les énergies données par les tables sont en kcal/mol. Nous les utilisons par la suite simplement comme un score, nous les avons donc systématiquement multipliées par 100 afin de travailler en entiers. Par abus de langage, nous parlons toujours d'énergie.

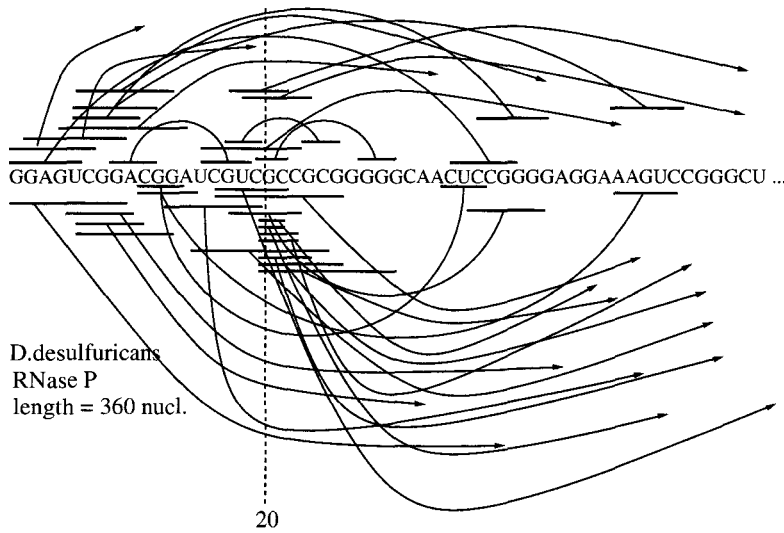


FIG. 3.7 – *Les tiges potentielles produites par CARNAC$_2$ pour une RNase P, de longueur 360. Nous avons représenté ici les tiges dont la position d'ouverture est inférieure à 20.*

Etape 2 – Les points d'ancrage

CARNAC$_2$ recherche les régions hautement conservées entre les deux séquences afin de s'en servir comme points d'ancrage. Ces région sont détectées simplement en comparant les deux séquences base à base puisqu'on interdit les insertion / délétion. On score 1 pour une identité, et -2 pour une substitution. La sélection des régions parmi les meilleures diagonales de la matrice ainsi construite est réalisée de façon gloutonne, à l'aide d'un seuil basé sur une mesure de probabilité.

Si les séquences étaient engendrées de manière aléatoire, selon un modèle de Bernoulli, où $p_A = p_U = p_C = p_G$, la probabilité que deux séquences de même longueur l partagent au moins $k \leq l$ bases en commun suit une loi binomiale

$$B(k, l) = \sum_{i \leq k} \binom{l}{i} p^i (1-p)^{l-i},$$

où $p = \frac{1}{4}$ d'après l'hypothèse d'équirépartition des nucléotides.

Donc, si l'on se donne un facteur de longueur l, l'espérance de trouver le même facteur dans une séquence de longueur $l' \geq l$ avec au plus m erreurs (*i.e.* substitutions) peut être approximée par :

$$E(l, l', m) = (l' - l) \times B(l - m, l).$$

Nous considérons qu'une région est conservée si $E(l, l + rel\_dist, m) \leq 10^{-8}$, où rel_dist est le décalage relatif entre les deux morceaux de séquence (l'origine de réference étant la position de la fin de la dernière région conservée). En pratique, le seuil est suffisamment élevé pour négliger le biais de contenu en nucléotides. Les rares conflits qui se produisent entre deux régions sont levés en éliminant purement et simplement ces points d'ancrages récalcitrants.

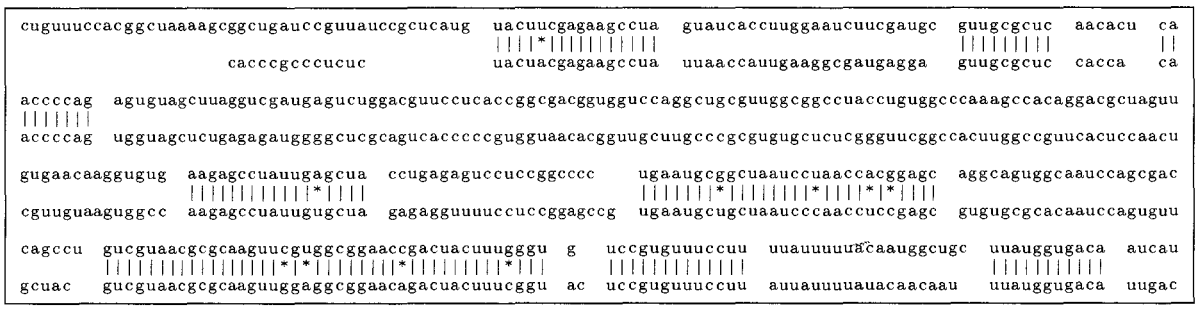


FIG. 3.8 – *Points d'ancrage trouvés par $CARNAC_2$ sur deux RNases P. Les parties non alignées restent "flottantes".*

Etape 3 – Le filtrage des tiges

Une fois qu'on a obtenu les points d'ancrage, les deux jeux de tiges sont confrontés pour former des couples de tiges copliables. Lors de cette confrontation, deux types de contrôle sont réalisés :

- on impose au moins une covariation d'une part,
- on vérifie la compatiblité avec les points d'ancrage d'autre part.

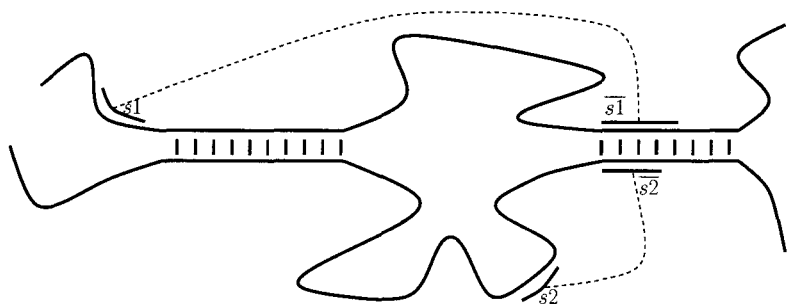
Comme on ne considère que deux séquences, notre définition de covariation est légèrement différente de celle donnée page 42. L'analyse comparative mesure la corrélation des colonnes sans se préoccuper du caractère canonique ou non de l'appariement. Avec deux séquences, évidemment cela ne fonctionne pas. Si l'on se donne une tige, alignée avec une autre, et que l'on considère un appariement dans cette tige, nous dirons simplement qu'il y a covariation si les deux bases de l'appariement ont muté d'une tige à l'autre tout en préservant l'appariement. Les tiges sont alignées deux à deux selon leur structure primaire, puis on examine s'il y a des covariations. Sur cet exemple on observe des mutations, mais aucune covariation :

		1	2	3	4	5				5	4	3	2	1		
C	C	A	G	C	C	U		...		A	G	G	U	U	G	G
		G	G	U	C	U	G	...	C	A	A	G	U	U		

On examine ensuite pour chaque couple de tige si celles-ci sont compatibles avec les points d'ancrage, c'est-à-dire si le fait de les replier simultanément ne contredit pas l'alignement local des séquences selon ces points d'ancrage. Pour plus de clarté, nous décrivons les cas où deux tiges *ne sont pas* compatibles.

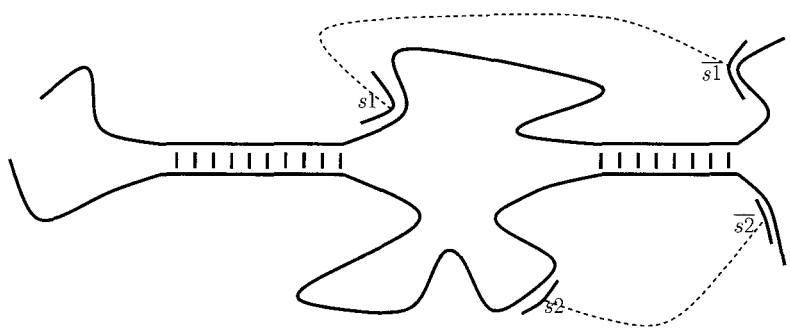
cas 1 – Violation d'un point d'ancrage :

Si (s_1,$\overline{s_1}$) et (s_2,$\overline{s_2}$) étaient repliés simultanément, alors l'alignement de s_1 et s_2 contredirait l'alignement local au niveau du point d'ancrage.



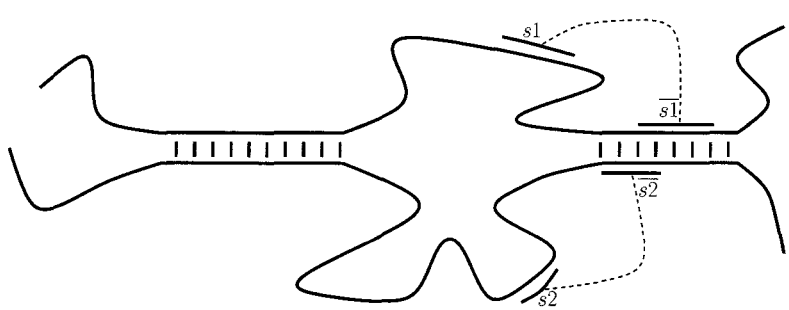
cas 2 – Décalage trop large à l'extérieur d'un point d'ancrage :

Lorsque l'ouverture ou la fermeture d'une tige tombe entre deux points d'ancrage, un décalage borné est autorisé. Ce décalage est variable selon les zones. Il dépend de la différence de longueur des fragments de séquences entre les points d'ancrage. Sur cet exemple, le décalage entre s_1 et s_2 est trop large.



cas 3 – Décalage à l'intérieur d'un point d'ancrage :

Lorsque l'ouverture ou la fermeture d'une tige tombe à l'intérieur d'un point d'ancrage, aucun décalage n'est autorisé. Sur cet exemple, il y a un léger décalage entre $\overline{s_1}$ et $\overline{s_2}$.



Deux tiges – une dans chaque séquence – qui passent brillamment ce contrôle technique sont dites **copliables**. Un graphe bipartite est finalement créé entre les deux jeux de tiges où l'on pose une arête lorsque deux tiges sont copliables. A l'issue du filtrage, les tiges **célibataires**, c'est-à-dire les singletons du graphe, qui n'ont aucun partenaire potentiel, sont supprimées.

Comme l'algorithme de repliement décrit dans le paragraphe suivant permet le repliement d'une tige dans une seule séquence, on conserve cependant certaines tiges célibataires si elles vérifient les trois conditions suivantes :

- il s'agit d'une tige terminale : par défaut, si la taille de la boucle est inférieure à huit nucléotides, on dira que la tige est terminale ;
- elle est sitée dans une région d'insertion potentielle : une telle région est détectée par une grande différence de longueur entre deux morceaux de séquences qui joignent des points d'ancrage consécutifs ;
- son énergie est inférieure à -1500 (ce seuil fixe est largement supérieur au seuil affine de sélection initial).

Etape 4 – Le corepliement

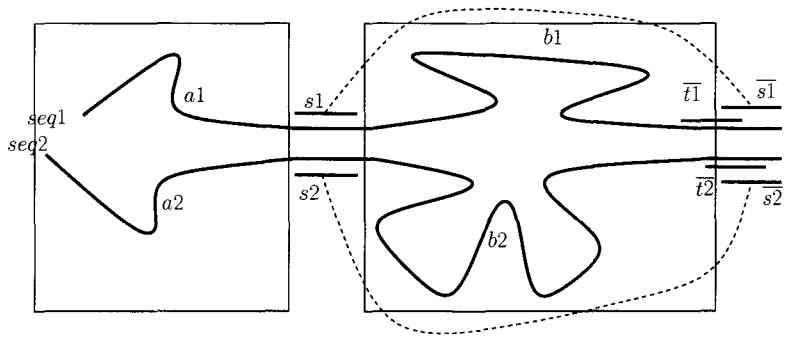
La partie *corepliement* de l'algorithme est une adaptation des récurrences de Sankoff (voir page 48). Le principe et la complexité théorique sont les mêmes, sauf que nous procédons selon l'heuristique qui consiste non pas à replier une paire de nucléotides, mais une tige entière. La taille des entrées n'est donc plus la longueur des séquences, mais le nombre de tiges. Par ailleurs, nous ne considérons que les couples de tiges copliables, ce qui amène deux bénéfices : d'une part nous réduisons encore considérablement la taille des entrées, d'autre part cela nous permet de mettre en œuvre une optimisation supplémentaire sur la taille de la mémoire à allouer. Nous discutons de cette optimisaiton au paragraphe 3.1.4.

Nous présentons d'abord brièvement la manière dont les récurrences fonctionnent sur les tiges, puis très précisément les formules de récurrence telles qu'elles ont été implémentées. La deuxième présentation est plus formelle, la première plus intuitive.

Dans la suite, α désigne l'énergie du repliement commun, c'est-à-dire la somme des énergies des tiges constituant ce repliement. A ce stade, on ne prend plus en compte toutes les finesses du modèle thermodynamique, et l'énergie des tiges joue le rôle d'un score.

Repliement commun de 2 tiges copliables

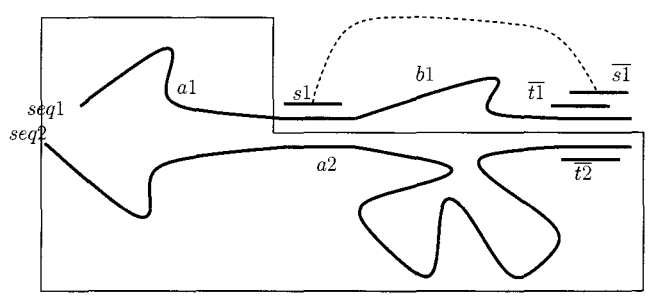
Si $(s_1,\overline{s_1})$ et $(s_2,\overline{s_2})$ sont deux tiges copliables, on peut les coplier (1) ou non (2). $\overline{t_1}$ et $\overline{t_2}$ désignent les fermetures des prochaines tiges potentielles copliables qu'on rencontre lorsqu'on procède par position de fermeture décroissante.



$$\alpha(seq_1,seq_2) = min \begin{cases} \alpha(a_1,a_2) + \alpha(b_1,b_2) + énergie(s_1,\overline{s_1}) + énergie(s_2,\overline{s_2}) & (1) \\ \alpha(a_1 s_1 b_1 \overline{t_1},\ a_2 s_2 b_2 \overline{t_2}) & (2) \end{cases}$$

Repliement d'une tige dans une unique séquence

Si $(s_1,\overline{s_1})$ est une tige terminale potentielle (la distance entre s_1 et $\overline{s_1}$ doit être inférieure à 8 bases), on autorise qu'elle se replie seule. Nous divisons son énergie par dix afin qu'il soit toujours plus pénalisant de replier deux tiges séparément plutôt qu'ensemble[5]. $\overline{t_1}$ désigne la fermeture de la prochaine tige potentielle qu'on rencontre lorsqu'on procède par position de fermeture décroissante.



$$\alpha(seq_1,seq_2) = min \begin{cases} \alpha(a_1,a_2) + 0.1 \times énergie(s_1,\overline{s_1}) & (1) \\ \alpha(a_1 s_1 b_1 \overline{t_1},\ a_2) & (2) \end{cases}$$

5. Nous avons essayé différents types de pénalités, mais elles semblent empiriquement toutes équivalentes.

Dans les récurrences pour la construction du repliement commun à deux séquences, il faut gérer les chevauchements de tiges, qui n'apparaissent pas lorsqu'on se place au niveau des nucléotides. L'implémentation nécessite donc quelques subtilités additionnelles par rapport à celle des récurrences de Sankoff.

Une tige potentielle **p** étant repérée par sa position d'ouverture **p.open** et sa position de fermeture **p.close**, nous construisons les ensembles $\mathcal{P}_{\rightarrow}$, $\mathcal{P}_{\leftarrow}$, $\mathcal{Q}_{\rightarrow}$, $\mathcal{Q}_{\leftarrow}$, et les applications **next** et **last** de la manière suivante :

$\mathcal{P}_{\rightarrow} := (p_1,p_2,p_3,...,p_n)$ désigne la liste des tiges potentielles, rangées par ordre croissant selon la position d'ouverture : $p_i \leqslant p_j$ si $p_i.open \leqslant p_j.open$. L'application **next** : $(\mathcal{P}_{\rightarrow}) \longrightarrow [1 .. n]$ donne l'indice de la prochaine tige non conflictuelle : $next\ (p_i) := min\{k \in [i+1 .. n] : p_k \cap p_i = \varnothing\}$

$\mathcal{P}_{\leftarrow} := (p'_1,p'_2,p'_3,...,p'_n) := (p_{\sigma 1},p_{\sigma 2},p_{\sigma 3},...,p_{\sigma n})$ désigne la liste des tiges réordonnées selon la position de fermeture : $p'_i \leqslant p'_j$ si $p'_i.close \leqslant p'_j.close$. Et l'application **last** : $(\mathcal{P}_{\leftarrow}) \longrightarrow [1 .. n]$ donne l'indice de la dernière tige non conflictuelle : $last\ (p'_j) := max\{k \in [1 .. j-1] : p'_k \cap p'_j = \varnothing\}$

On définit de la même façon les ensembles $\mathcal{Q}_{\leftarrow}$ et $\mathcal{Q}_{\rightarrow}$, ainsi que les applications **next** et **last** pour la deuxième séquence (on pose $m := |\mathcal{Q}_{\rightarrow}| = |\mathcal{Q}_{\leftarrow}|$).

N désigne la matrice de Nussinov et Jacobson sur les tiges, aisément calculée pour les deux séquences (on identifie dans les formules les deux matrices, qui sont en réalité différentes, bien sûr). S désigne la matrice de Sankoff sur les tiges.

Initialisation : temps $O(n^3)$

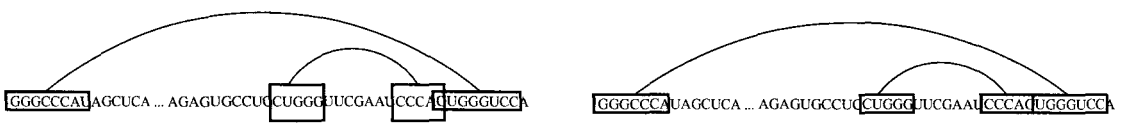
- $S\ [i .. 0, k .. l] := N\ [k .. l] \qquad (0 \leqslant i \leqslant n) \qquad (1 \leqslant k, l \leqslant m)$
- $S\ [0 .. j, k .. l] := N\ [k .. l] \qquad (0 \leqslant j \leqslant n) \qquad (1 \leqslant k, l \leqslant m)$
- $S\ [i .. j, k .. 0] := N\ [i .. j] \qquad (0 \leqslant k \leqslant m) \qquad (1 \leqslant i, j \leqslant n)$
- $S\ [i .. j, 0 .. l] := N\ [i .. j] \qquad (0 \leqslant l \leqslant m) \qquad (1 \leqslant i, j \leqslant n)$

Récurrence : temps $O(m^3 \times n^3)$

$S\ [i .. j, k .. l] := min \qquad (1 \leqslant i < j \leqslant n) \qquad (1 \leqslant k < l \leqslant m)$

$$\left\{\begin{array}{l}
\bullet\ S\ [i .. j, k .. l-1] \\
\bullet\ S\ [i .. j-1, k .. l] \\
\bullet\ S\ [i .. j, k .. last(y)] + S\ [0 .. 0, next(y) .. last(l)] + bind\ (-, q_y = q'_l), \\
\bullet\ S\ [0 .. 0, k .. last(y)] + S\ [i .. j, next(y) .. last(l)] + bind\ (-, q_y = q'_l), \\
\bullet\ S\ [i .. last(x), k .. l] + S\ [next(x) .. last(j), 0 .. 0] + bind\ (p_x = p'_j, -), \\
\bullet\ S\ [i .. last(x), 0 .. 0] + S\ [next(x) .. last(j), k .. l] + bind\ (p_x = p'_j, -), \\
\bullet\ S\ [i .. last(x), k .. last(y)] + S\ [next(x) .. last(j), next(y) .. last(l)] + \\
\quad cobind\ (p_x = p'_j, q_y = q'_l), \qquad (1 \leqslant x \leqslant n) \qquad (1 \leqslant y \leqslant m)
\end{array}\right.$$

L'algorithme tel qu'il est conçu ne permet pas aux tiges de se chevaucher. Le problème peut se poser lorsque les tiges de la vraie structure ne sont pas maximales sous peine de se chevaucher (FIG. 3.9). S'il arrive de tels conflits, seule l'une des deux tiges pourra être prédite, mais cela n'amènera pas de faux positif.



les tiges maximales se chevauchent

les tiges de la vraie structure ne sont pas maximales

FIG. 3.9 – *Un problème de chevauchement de tiges peut survenir lorsqu'on considère des tiges maximales.*

3.1.3 Les résultats de CARNAC$_2$

Revenons à présent sur les deux séquences de RNases P qui nous ont guidé tout au long du paragraphe 3.1.1 : *D.desulfuricans* et *C.jejuni*. Nous présentons tout d'abord les résultats de CARNAC$_2$ sur cet exemple, puis nous vérifions la robustesse des prédictions face au choix des séquences à coreplier en envisageant tous les corepliements possibles avec les autres séquences de la même famille (*Delta/Epsilon Purple Bacteria RNase P RNA*). Nous comparons finalement les prédictions de CARNAC$_2$ à celles des méthodes qui peuvent s'appliquer avec seulement deux séquences : MFOLD [Zuk03, MSZT99] et DYNALIGN [MT02].

La FIG. 3.10 met en relation certaines caractéristiques des séquences avec le "flux" de tiges conservées par CARNAC$_2$ au cours des différentes étapes. On observe, comme on peut s'y attendre, que le nombre de tiges potentielles est au départ fortement dépendant de la longueur des séquences et du biais de contenu en GC. Le filtrage de l'étape 3 élimine une bonne partie de ces tiges et l'espace mémoire requis est finalement inférieur à 0.2 Mo. La structure commune est obtenue en moins d'une seconde.

	longueur	GC %	flux de tiges
D.desulfuricans	360	64 %	357 $\longrightarrow$ 105 $\longrightarrow$ 12
C.jejuni	318	41 %	92 $\longrightarrow$ 41 $\longrightarrow$ 10

FIG. 3.10 – *Corepliement sous CARNAC$_2$ des RNases P D.desulfuricans et C.jejuni. Le flux de tiges correspond au nombre de tiges à chaque étape : le nombre de tiges potentielles maximales précalculées à l'étape 1, puis le nombre de tiges à replier à l'issue du filtrage de l'étape 3, et finalement le nombre de tiges obtenues en sortie, c'est-à-dire la structure secondaire prédite.*

Les FIG. 3.11 et FIG. 3.12 présentent la structure prédite par CARNAC$_2$ en comparaison avec la vraie structure. Lorsqu'on envisage la molécule au niveau des tiges, on remarque l'ab-

sence de certaines tiges, mais on observe par contre que toutes les tiges prédites sont correctes, dans le sens où chaque tige prédite *correspond* à une tige de la vraie structure. Quelques erreurs de prédiction apparaissent au niveau des appariements à l'extrémité de certaines tiges, puisque $CARNAC_2$ utilise en entrée des tiges *maximales*.

La FIG. 3.12 montre aussi les résultats de $CARNAC_2$ lorsqu'on replie les séquences en utilisant en entrée les tiges fournies par les repliements sous-optimaux de MFOLD, ce qui permettrait en principe de s'affranchir du postulat de maximalité des tiges. Nous avons joué sur le pourcentage de sous-optimalité et le paramètre `window`: nous avons d'abord utilisé les paramètres par défaut, puis augmenté le pourcentage de sous optimalité et ajusté au minimum le paramètre de différenciation `window`. Dans les deux cas, sur cet exemple, il n'est pas avantageux d'utiliser les tiges fournies par MFOLD. Le fait est que plusieurs tiges correctes sont absentes des repliements sous-optimaux donnés par MFOLD.

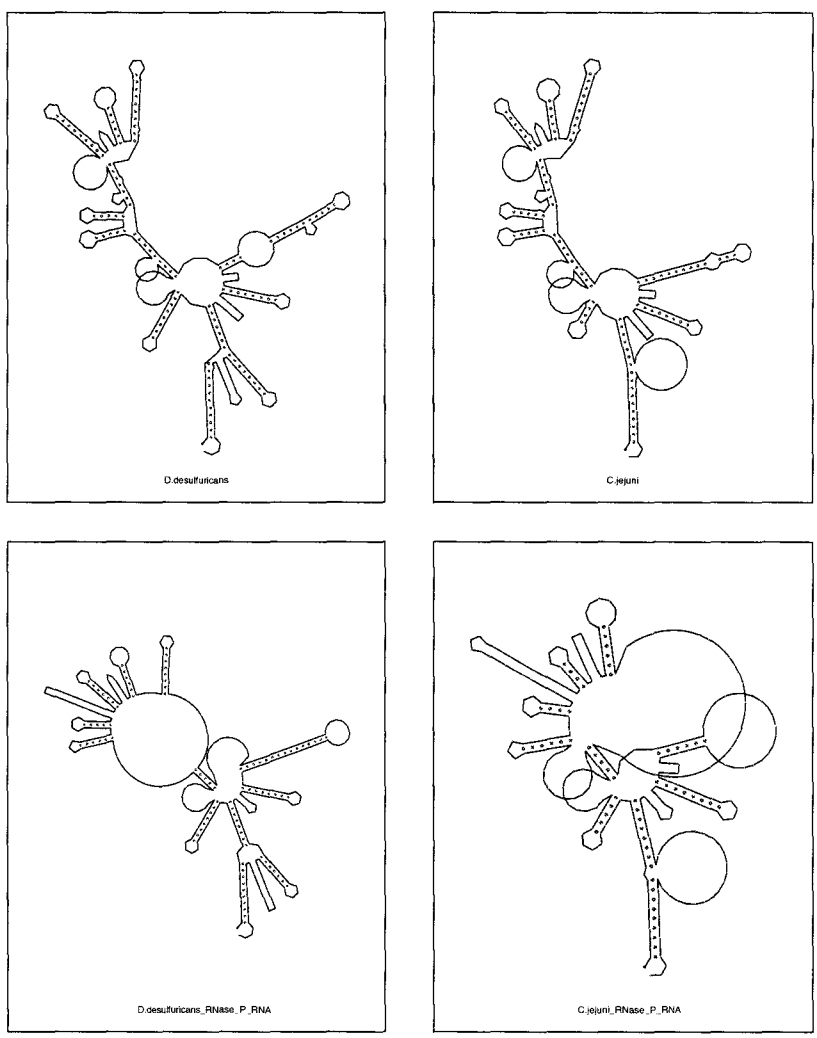


FIG. 3.11 – *D.desulfuricans et C.Jejuni - les structures prédites par $CARNAC_2$ (en bas) comparées aux vraies structures (en haut). Toutes les tiges sont correctes, mêmes si la forme de la molécule parait un peu déformée par le logiciel de visualisation.*

Nous considérons maintenant l'ensemble des séquences des deux sous-familles *Delta/Epsilon Purple Bacteria RNase P*) qui sont disponibles dans la banque de J.Brown [Bro99], dont nous avons éliminé les séquences partielles ou redondantes, ce qui nous laisse un jeu de cinq séquences. Nous présentons dans la TABLE 3.1 les résultats numériques pour tous les repliements deux à deux. Pour mesurer la qualité des prédictions, nous nous sommes placés au niveau des nucléotides. En moyenne, CARNAC$_2$ prédit à chaque fois plus de la moitié de la structure avec une correction de 85 % sur les appariements. Lorsqu'on coreplie sous CARNAC$_2$ la séquence référence *D.desulfuricans* avec n'importe laquelle des quatre autres séquences de la famille, on obtient – en termes de tiges – toujours la bonne structure, malgré quelques appariements prédits à tort apparaissant principalement aux bords des tiges. La TABLE 3.2 offre une comparaison avec les structures produites par MFOLD et DYNALIGN, qui s'appliquent dans le même contexte : celui où l'on ne considère que deux séquences.

	D.desulfuricans	**D.vulgaris**	**G.sulfurreducens**	**C.jejuni**	**H.pylori**
D.desulfuricans	–	59 93% 49%	87 85% 33%	61 88% 41%	51 92% 45%
D.vulgaris	65 93% 44%	–	93 81% 31%	52 78% 55%	54 70% 55%
G.sulfurreducens	88 81% 34%	81 86% 35%	–	63 84% 42%	48 89% 49%
C.jejuni	93 80% 31%	65 89% 46%	84 80% 39%	–	42 88% 56%
H.pylori	77 94% 33%	65 73% 56%	69 85% 47%	35 94% 64%	–

TAB. 3.1 – *Les résultats de CARNAC$_2$ pour la famille Delta/Epsilon Purple Bacteria RNase P RNA toute entière. La table est à lire de la façon suivante : l'organisme de la colonne (en gras) est la séquence courante, corepliée avec l'organisme de la ligne. Pour chaque corepliement, le premier nombre indique le nombre total d'appariements prédits, le second le pourcentage de vrais positifs, et le troisième le pourcentage de faux négatifs.*

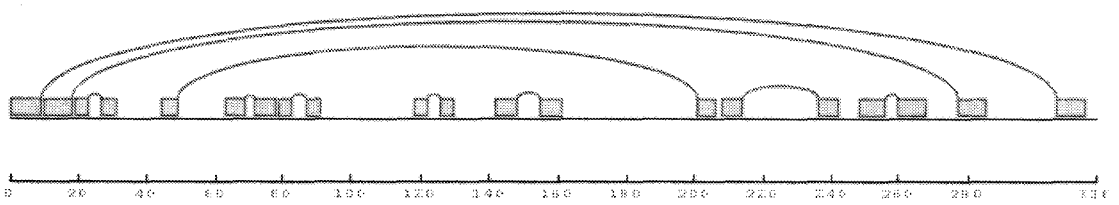
MFOLD [Zuk03, MSZT99] (voir page 33) prédit plus de vrais appariements en quantité, mais moins en proportion. Le programme a aussi tendance à surprédire. Le problème de déceler la meilleure structure parmi les sous-optimales reste entier.

DYNALIGN [MT02] (voir page 49) autorise entre les deux séquences un décalage borné ajustable. Nous avons lancé le programme avec différentes valeurs pour ce paramètre *max separation* (max sep). La qualité des prédictions va croissante lorsque *max sep.* devient grand, mais le temps de calcul aussi ! Avec *max sep*=3, le repliement dure environ 15 minutes et avec *max sep*=10, le programme doit tourner plus de 8 heures tandis que CARNAC$_2$ donne ses résultats en moins d'une seconde. Les mauvais résultats de DYNALIGN pour *C.jejuni* et pour *H.pylori* sont expliqués par les variations relativement à la structure du partenaire : une tige de *D.desulfuricans* est absente dans la structure de *C.jejuni* et *H.pylori*, ce qui induit un décalage important au niveau de la structure primaire, décalage qui ne pourrait être couvert que par une grande valeur de *max sep.* CARNAC$_2$, autorisant un décalage variable, est plus robuste dans ce cas.

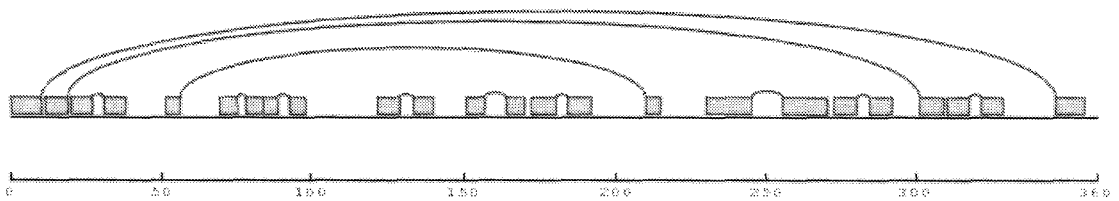
3.1.4 Discussion sur la complexité de CARNAC$_2$

Le repliement des deux séquences *D.desulfuricans* et *C.jejuni* est executé en moins d'une seconde. Il est bon de remarquer que la complexité spatiale en $O(n^4)$ de la méthode de San-

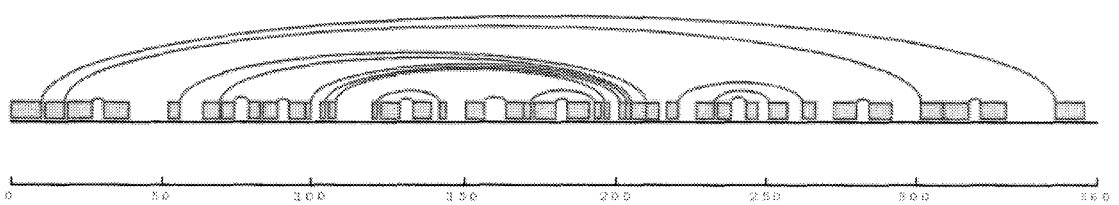
CARNAC$_2$: *C.jejuni*



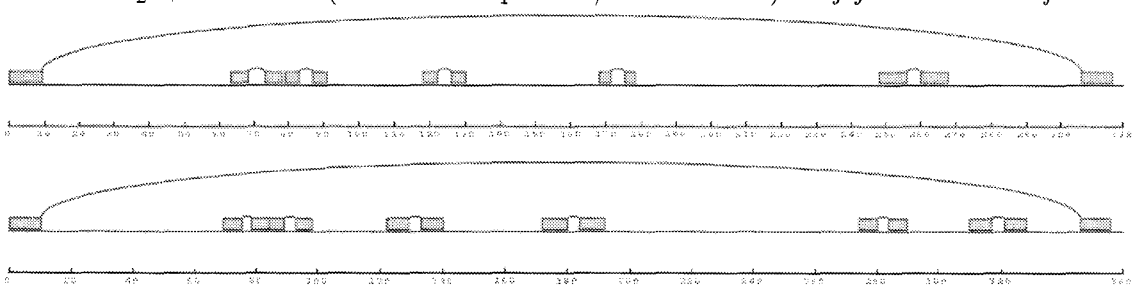
CARNAC$_2$: *D.desulfuricans*



la vraie structure : *D.desulfuricans*



CARNAC$_2$ + MFOLD (defaut : subopt. 5% / window = 7) : *C.jejuni* et *D.desulfuricans*



CARNAC$_2$ + MFOLD (subopt. 10% / window = 0) : *C.jejuni* et *D.desulfuricans*

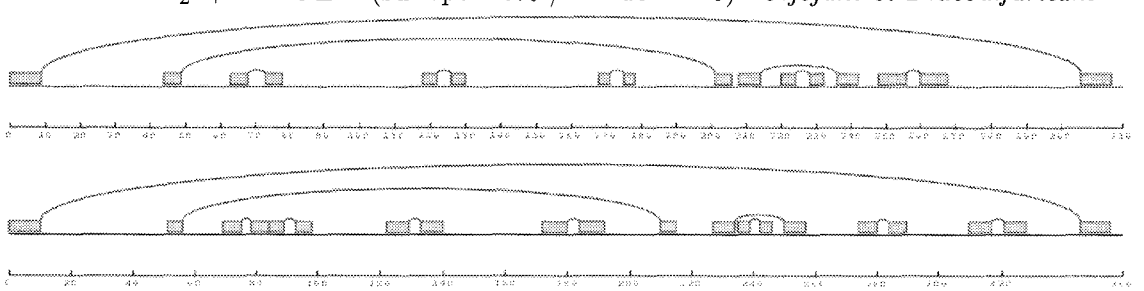


FIG. 3.12 – *Les résultats de $CARNAC_2$ en copliant les deux séquences homologues D.desulfuricans et C.jejuni (nous donnons aussi la vraie structure pour D.desulfuricans, à titre de comparaison) et les résultats avec en entrée les tiges fournies par les structures sous-optimales de MFOLD.*

CARNAC$_2$

nom	appariements	vrais positifs	faux négatifs
D. vulgaris	59	55 (93%)	49%
G. sulfurreducens	87	74 (85%)	33%
C. jejuni	61	54 (88%)	41%
H. pylori	51	47 (92%)	45%

MFOLD

nom	sous-opt.	rang	appariements	vrais positifs	faux négatifs
D.desulf.	18	4	113	75 (66%)	31%
D.vulgaris	22	2	112	80 (71%)	26%
G.sulf.	15	2	112	90 (80%)	19%
C.jejuni	6	1	86	59 (68%)	36%
H.pylori	9	4	95	56 (58%)	34%

DYNALIGN

max sep = 3, 5 et 10

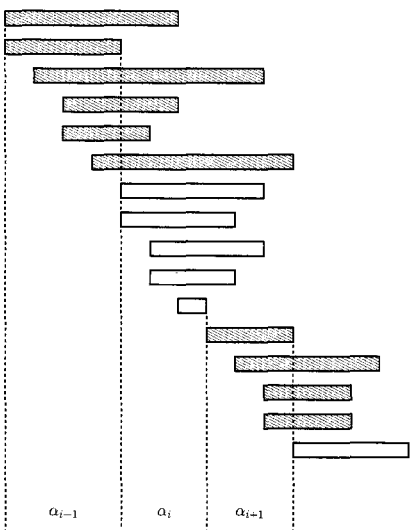
	max sep	appariements	vrais positifs	faux négatifs
D. vulgaris	3	101	39 (38%)	64%
	5	101	53 (52%)	51%
	10	108	85 (78%)	21%
G. sulfurreducens	3	96	81 (84%)	27%
	5	105	91 (86%)	18%
	10	105	94 (89%)	15%
C. jejuni	3	47	0 (0%)	100%
	5	67	0 (0%)	100%
	10	64	37 (57%)	60%
H. pylori	3	53	3 (5%)	97%
	5	41	14 (34%)	84%
	10	58	31 (53%)	64%

TAB. 3.2 – *Comparaison des résultats de CARNAC$_2$ face aux autres méthodes de prédictions se plaçant dans le même contexte. Pour DYNALIGN nous étudions pour chaque séquence son corepliement avec la séquence de l'organisme de référence D.desulfuricans, pour MFOLD les séquences ont été repliées séparément. A chaque repliement nous annonçons le nombre d'*appariements *prédits, le nombre et le pourcentage de* vrais positifs, *le pourcentage de* faux négatifs. *Pour MFOLD nous donnons le nombre de structures sous-optimales, parmi lesquelles nous avons sélectionné la meilleure dont nous indiquons le* rang *au sein des sous-optimales.*

koff imposerait sur cet exemple l'allocation d'une matrice de l'ordre de 10 Go, ce qui rend la méthode absolument inopérante. Son temps de calcul en $O(n^6)$ est lui aussi totalement rédhibitoire pour n de l'ordre de 350 nucléotides. Comme c'est d'abord l'espace mémoire qui est contraignant, $CARNAC_2$ utilise une optimisation pour réduire sa taille. Nous présentons deux heuristiques qui ont été utilisées successivement sur l'algorithme. Bien qu'elles soient difficilement compatibles entre elles, les deux optimisations ont leur intérêt propre. La version actuelle utilise la deuxième heuristique, qui est nettement plus efficace en pratique.

Regroupement en classes d'incompatibilité

Deux tiges de la même séquence sont dites **incompatibles** si elles ne sont ni emboîtées, ni juxtaposées. C'est notamment le cas lorsqu'il y a un chevauchement, deux tiges qui se chevauchent ne peuvent pas se replier simultanément sans contredire la définition de structure secondaire. En se basant sur cette constatation, on peut regrouper ensemble des tiges incompatibles en remarquant qu'au plus l'une d'entre elles sera repliée.



Les tiges étant rangées par positions croissantes selon leur ouverture, on ajoute une tige à la classe en cours de création tant qu'elle est incompatible avec les tiges déjà dans la classe. C'est en particulier le cas si la position de fin de l'ouverture est strictement inférieure au minimum des positions de fin d'ouverture des autres tiges de la classe.

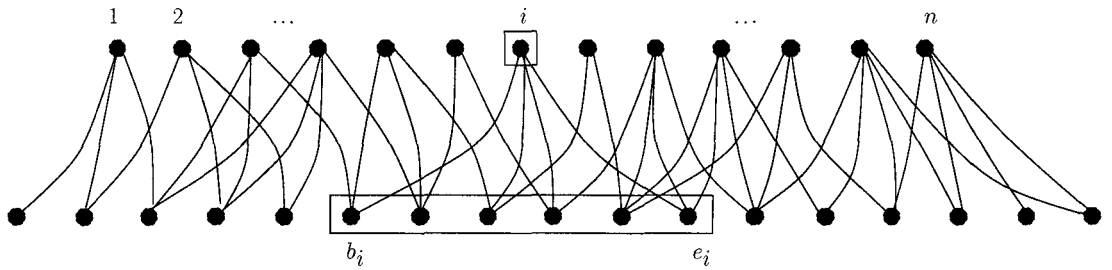
Le regroupement qui est construit n'est pas canonique, mais il peut être calculé *à la volée* séparément pour chacun des deux jeux de tiges, en temps linéaire. La matrice est construite sur les classes de tiges, plutôt que sur les tiges, ce qui réduit la taille mémoire. Empiriquement, on constate que les classes contiennent en moyenne 2 ou 3 tiges, ce qui donne un gain d'un facteur 40 environ sur la taille de la matrice.

Allocation de l'hyperdiagonale

Cette deuxième optimisation – celle qui est utilisée dans $CARNAC_2$ – tient compte du fait que deux tiges dont les positions sont trop éloignées ne pourront pas être copliables. Elle ne peut être évaluée qu'empiriquement, car elle dépend de données difficilement quantifiables : covariations entre les tiges potentielles, filtrage des tiges avec les points d'ancrage.

En reprenant les notations des récurrences de CARNAC (voir page 65), on considère $\mathcal{P}_{\rightarrow} := (p_1,p_2,p_3,...,p_n)$ et $\mathcal{Q}_{\rightarrow} := (q_1,q_2,q_3,...,q_m)$ les ensembles de tiges des deux séquences, rangées par ordre croissant de position d'ouverture, $S\ [i\ ..\ j,\ k\ ..\ l]$ l'énergie minimale pour les sous-segments $\mathcal{P}_{ij\rightarrow} := (p_i,...,p_j)$ et $\mathcal{Q}_{kl\rightarrow} := (q_k,...,q_l)$. On note aussi $\overline{S}_i\ [i\ ..\ j,\ k\ ..\ l]$ l'énergie minimale pour les sous-segments $\mathcal{P}_{ij\rightarrow} := (p_i,...,p_j)$ et $\mathcal{Q}_{kl\rightarrow} := (q_k,...,q_l)$ où l'on suppose que la tige p_i est repliée.

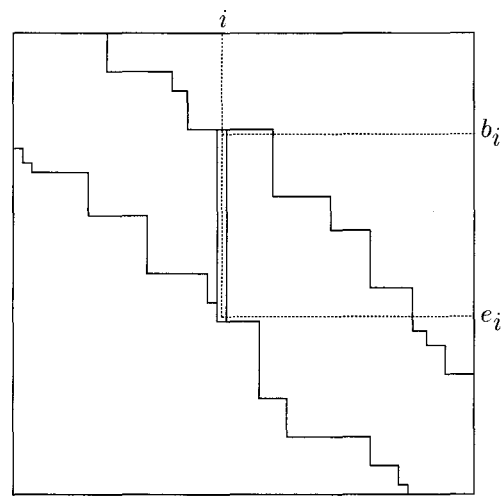
Pour la tige de rang i dans la première séquence, on calcule b_i et e_i qui sont respectivement, dans l'autre séquence, le rang de la première tige qui lui est copliable et le rang de la dernière tige qui lui est copliable.



On fait de même avec les ensembles de tiges réordonnées par position croissante de fermeture ($\mathcal{P}_{\leftarrow} := (p'_1,p'_2,p'_3,...,p'_n)$ et $\mathcal{Q}_{\leftarrow} := (q'_1,q'_2,q'_3,...,q'_m)$) : pour chaque tige d'indice j on calcule b'_j et e'_j.

Comme les tiges dont l'indice est inférieur à b_i ou supérieur à e_i ne peuvent pas être copliées avec la tige d'indice i, on a $\overline{S}_i\ [i\ ..\ j,\ k\ ..\ l] = \overline{S}_i\ [i\ ..\ j,\ k_0\ ..\ l]$ où :

$k_0 = b_i$ si $k < b_i$,
$k_0 = e_i$ si $k > e_i$,
$k_0 = k$ sinon.



Si dans le repliement donnant l'énergie optimale $S\ [i\ ..\ j,\ k\ ..\ l]$ la tige p_i n'est pas repliée, alors $S\ [i\ ..\ j,\ k\ ..\ l] = S\ [i+1\ ..\ j,\ k\ ..\ l]$. Quitte à considérer $S\ [i+t\ ..\ j,\ k\ ..\ l]$, où p_{i+t} est la première tige repliée, on peut supposer qu'on est bien dans le cas où p_i est repliée, donc que $S\ [i\ ..\ j,\ k\ ..\ l] = \overline{S}_i\ [i\ ..\ j,\ k\ ..\ l]$. Dans ce cas, $S\ [i\ ..\ j,\ k\ ..\ l] = S\ [i\ ..\ j,\ k_0\ ..\ l]$, et il est inutile d'allouer la matrice pour les valeurs de k inférieures à b_i ou supérieures à e_i. En raisonnant de la même manière pour j, on voit qu'il suffit d'allouer la matrice de la façon suivante :

$[\,1\,..\,i\,..\,n\,]$ $[\,1\,..\,j\,..\,m\,]$ $[\,b_i\,..\,k\,..\,e_i\,]$ $[\,b'_j\,..\,l\,..\,e'_j\,]$

On n'alloue donc que la partie proche de l'hyperdiagonale. Si K est la largeur moyenne de l'intervalle $[\ b_i\ ..\ e_i\ ]$ et L la largeur moyenne de l'intervalle $[\ b'_j\ ..\ e'_j\ ]$, la complexité spatiale devient $O(mnKL)$.

Si on observe les relations de récurrence (page 65), on voit qu'en ayant alloué la matrice comme une série de vecteurs de tailles différentes, lors des appels récursifs, on peut tomber hors de la zone allouée. Il faut donc aller rechercher dans la zone allouée la donnée numérique correspondante. Cette petite routine de recherche de valeur dans la matrice ralentit un peu l'algorithme, surtout lorsqu'on autorise les tiges célibataires.

Dans les faits, on observe un gain considérable en espace, qui peut aller jusqu'à un facteur de plusieurs centaines (voire plusieurs milliers sur les grosses séquences) sur la matrice, ce qui rend l'algorithme praticable.

Quelques remarques sur le comportement de CARNAC$_2$

CARNAC s'applique à des jeux de séquences dont les caractéristiques sont variées, mais lorsque les séquences d'un même jeu de données ont des tailles très différentes ou ne possèdent aucun point d'ancrage, l'optimisation sur la taille de la mémoire à allouer ne fonctionne plus et l'espace à allouer devient trop grand. Si CARNAC$_2$ ne trouve pas d'assez bons points d'ancrage, on se voit ainsi obligé d'élever le seuil de sélection des tiges pour réduire la taille des entrées. Pour faire face à de tels cas, plusieurs pistes de recherche pourraient être explorées par la suite.

On pourrait se concentrer spécifiquement sur le jeu de tiges potentielles présentées à l'entrée de CARNAC afin d'avoir d'emblée une plus faible proportion de fausses tiges. CARNAC peut en tirer profit, de même que les méthodes qui prennent en entrée des séquences alignées bénéficient des progrès des algorithmes d'alignement multiple. Trouver un ensemble de tiges potentielles avec un faible taux de faux positifs est un problème à part entière, qu'on pourrait considérer, dans une certaine mesure, comme un problème de prédiction de structure. MFOLD, en proposant un ensemble de structures sous-optimales répond en quelque sorte à cette question. Deux pistes relativement simples ont déjà été explorées avec CARNAC : la génération par défaut de tiges maximales et l'utilisation des tiges produites par MFOLD.

Au niveau du filtrage, on pourrait concevoir une détection plus fine de points d'ancrages en utilisant des informations provenant des n séquences au lieu de les considérer par deux ou en incluant des contraintes relatives à certains motifs. Concernant les motifs, les boucles GNRA ou UNCG sont déjà prises en compte par les paramètres énergétiques (elles reçoivent une bonification). Nous avons essayé d'inclure d'autres motifs récurrents comme les AA-plateformes par exemple, mais sans succès, car les occurences de faux positifs étaient trop nombreuses.

3.2 CARNAC pour n séquences : combiner les corepliements

Le cœur de l'algorithme CARNAC est conçu autour de deux séquences. Avec plus de séquences, on peut bien sûr former tous les corepliements deux à deux possibles. n séquences nous donneraient ainsi $n-1$ structures pour chacune, étant donné que chaque séquence aura été comparée de manière indépendante aux $n-1$ autres. On aimerait de ces $n-1$ structures tirer une unique structure, dont la qualité et la fiabilité soit encore meilleure que chacune des $n-1$ structures prises individuellement. Pour cela nous construisons un *graphe* avec les tiges issues des corepliements, et nous examinons ses propriétés de connectivité. Nous voyons qu'à l'aide de ce graphe, il est aussi possible de *détecter* si les n séquences partagent en commun une structure.

3.2.1 Passer de 2 à n séquences

Avec n séquences homologues au lieu de deux, on peut cette fois tirer parti des informations *transverses* aux corepliements en examinant, pour une séquence donnée, si c'est bien la même structure qui a été prédite au cours des $n-1$ corepliements.

Le terme "homologue" doit ici être pris dans un sens large : il signifie que les séquences appartiennent à une même famille fonctionnelle, ou tout au moins qu'on le suppose. C'est le cas si les séquences proviennent du même *endroit*[6] dans des organismes différents ou au sein d'un même organisme.

Etant donnée une séquence, on peut abstraitement lui associer l'ensemble de ses structures potentielles de forte énergie, selon un modèle énergétique qu'on s'est donné, sans nécessairement chercher à construire cet ensemble. Avec deux séquences, la recherche d'une structure commune revient conceptuellement à déterminer l'intersection de ces deux ensembles.

Si l'on prend deux séquences d'ARN, partageant ou non la même fonction, on remarque empiriquement qu'elles possèdent très souvent des structures communes de très faible énergie libre. Pour s'en convaincre intuitivement, on peut se souvenir de l'explosion de tiges potentielles dès que la séquence dépasse une certaine longueur (FIG. 3.7). On peut aussi rapprocher cette constatation des résultats théoriques obtenus par l'équipe de Vienne en utilisant les notions de paysage et réseaux neutres [FHMSZ01, FSB+93, SFSH94]. En comparant les tailles des espaces de séquences et des espaces de structures, ils observent que le second est beaucoup plus petit que le premier. Ils s'intéressent aussi aux distributions comparées des objets dans chacun des deux espaces. Un de leurs résultats non intuitif précise que si on se donne deux structures totalement différentes, on peut construire une séquence pour laquelle ces structures sont *très proches* de celle d'énergie optimale (selon un modèle d'énergie classique, proche de celui de Turner) [GHR99]. Ces deux structures apparaitront donc naturellement parmi les structures sous-optimales si on applique à cette séquence un algorithme thermodynamique classique.

Ainsi le fait de trouver une structure commune à deux séquences ne permet pas d'affirmer qu'elles partagent effectivement cette structure commune.

Le *et al.* ont défini une mesure statistique qui permet d'évaluer l'unicité du repliement d'une molécule d'ARN en la comparant à des molécules artificielles produites en mélangeant ses nucléotides. Selon leur mesure, les ARN *fonctionnels* admettraient une unique structure [LZM02]. L'argument sous-jacent à cette constatation est que la pression sélective a favorisé au cours de l'évolution des structures thermodynamiquement plus stables, *donc* plus éloignées de celles qu'on obtient sur des séquences aléatoires puisqu'elles n'ont pas subi cette pression sélective. De telles considérations statistiques sont toujours discutables car elles sont fortement dépendantes du type de mélanges effectués sur les nucléotides [WK99] et du modèle thermodynamique sous-jacent. Néanmoins, on connait assez peu d'ARN fonctionnels possédant des structures alternatives et de tels basculements de structure concernent de petites molécules ou bien gardent un caractère très local sur de grosses molécules (on imagine assez bien en effet qu'il faille d'abord complétement déplier une grosse molécule – ce qui nécessiterait un très gros apport énergétique – afin de la replier dans une conformation entièrement nouvelle).

6. Par exemple des séquences prélevées en amont ou en aval de parties codantes, dont on envisage qu'elles puissent être structurées, sans qu'on en soit certain.

Selon cette hypothèse, si l'on sait au départ que les deux séquences homologues partagent effectivement une fonction – et donc une structure – communes, et si l'on observe que l'intersection de leur ensemble de structures sous-optimales est réduite à un singleton, cette intersection doit logiquement être la vraie structure. Mais il s'agit là d'une considération théorique, qui élude deux difficultés : d'une part il s'agit de trouver cette intersection, d'autre part il faut que l'intersection soit un singleton.

Caractériser l'intersection est le problème qui nous occupe de manière générale lorsqu'on cherche à prédire la structure commune à deux séquences. Nous en avons discuté tout au long du chapitre. C'est donc de la seconde difficulté dont nous allons maintenant parler.

Supposons qu'au lieu de deux séquences, nous disposions de n séquences homologues. Nous avons vu que les intersection deux à deux de leurs ensembles de structures de faible énergie ne sont pas nécessairement des singletons. Mais on conçoit aisément que l'intersection des n ensembles a de fortes chances d'être vide si les séquences ne partagent pas de structure commune, et réduite à un singleton dans le cas contraire. C'est sur cette hypothèse que nous nous baserons pour étendre l'algorithme à n séquences.

Première tentative : les fréquences de repliement

L'idée la plus immédiate est de calculer, pour chaque tige, sa *fréquence* de repliement, *i.e.* le nombre de corepliements dans lesquels elle est apparue. Si les séquences partagent effectivement une structure en commun, on s'attend à trouver des fréquences élevées pour les tiges constituant cette structure.

A titre d'expérience, nous avons formé deux ensembles de séquences ayant des caractéristiques similaires : d'une part un jeu de 15 RNases P appartenant à la même sous-famille (*Gamma Purple Bacteria*) ; d'autre part un jeu de 15 ARNm matures codant pour le cytochrome. Ces séquences ont toutes environ la même longueur (350 bases), la même similitude (donnant lieu à des points d'ancrage de qualité homogène sous CARNAC_2) et à peu près le même biais de contenu en GC. Le premier ensemble est structuré, dans le sens où les séquences partagent une structure commune, le second ne l'est pas.

Nous avons exécuté avec CARNAC_2 les $\binom{15}{2} = 105$ corepliements possibles et calculé pour chaque tige prédite sa fréquence de repliement, comprise entre 1 et 14. Nous avons ensuite réparti les tiges dans leur classe de fréquence respective et dessiné l'histogramme des classes de fréquences. Sur la FIG. 3.13, on voit clairement que la distribution est plutôt plate pour les RNases P, comparée à celle des ARNm où l'on observe un pic vers les basses fréquences. Plus précisément, on observe que 63% des tiges apparaissent dans plus de la moitié des corepliements pour les RNases, et moins de 1% pour les ARN messagers. Dès maintenant, on est tenté de dire que les RNases P partagent une structure commune, contrairement aux ARNm. Mais cette information sur les fréquences est-elle suffisante pour distinguer les bonnes tiges des mauvaises, au cas par cas? Nous allons voir que certaines tiges de fréquence élevée ne sont pas toujours de vraies positives. La FIG. 3.14 indique les tiges trouvées pour la première séquence de RNases (*A.ferrooxidans*) par ordre de fréquences décroissantes. Comme on s'y attend, les tiges de haute fréquence sont globalement de vraies tiges, bien sûr. On remarque cependant que la tige n°8, qui apparait 9 fois sur 14 lors des corepliements, est incorrecte. Pour chercher à éliminer ce genre d'imposteur, nous inspectons l'allure du *graphe* des repliements de tiges.

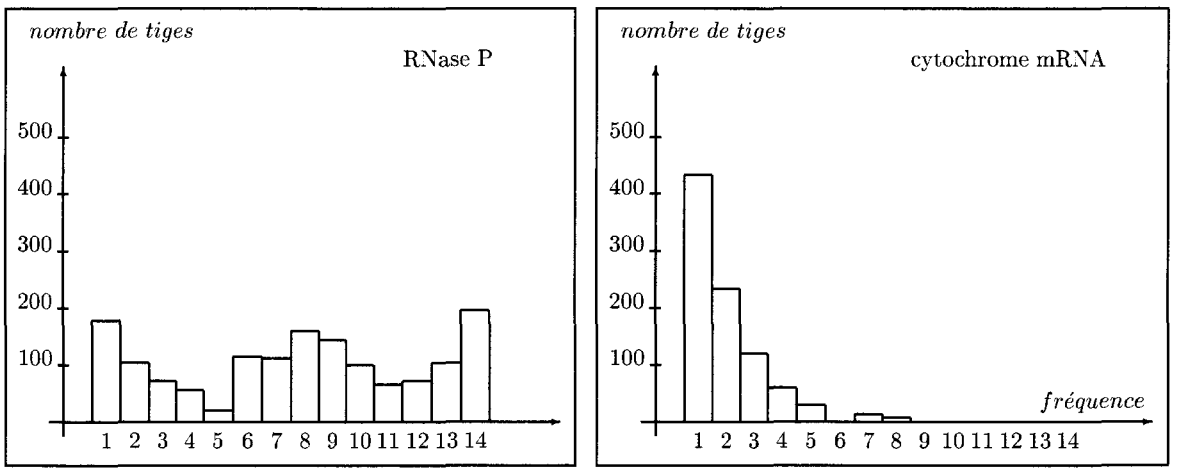
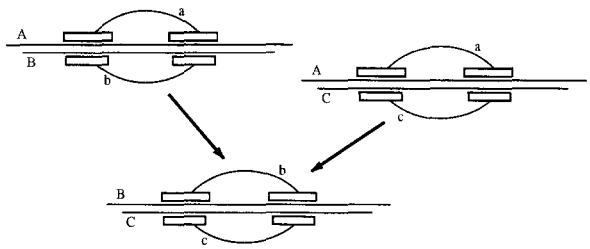


FIG. 3.13 – *Histogrammes des tiges réparties dans leur classe de fréquence pour chacun des deux ensembles de 15 séquences (RNases et ARNm).*

	position	fréquence	tige correcte	séquence	
1	**273 : 280 / 285 : 292**	**14**	✓	**gccugugc**	... **gugcaggc**
2	**1 : 10 / 331 : 340**	**14**	✓	**ggagugggcc**	... **ggcccacucc**
3	**173 : 180 / 185 : 192**	**13**	✓	**cauuuccg**	... **cggaaaug**
4	**68 : 73 / 77 : 82**	**12**	✓	**ccgguu**	... **ggccgg**
5	**12 : 19 / 302 : 309**	**11**	✓	**ggcgaccg**	... **uggucguc**
6	**20 : 26 / 31 : 37**	**11**	✓	**ccgcgga**	... **uccgggg**
7	**127 : 130 / 135 : 138**	**9**	✓	**gcgc**	... **gcgc**
8	152 : 160 / 202 : 210	9		aggugcggu	... accccgccu
9	**218 : 221 / 263 : 266**	**8**	✓	**gacc**	... **gguu**
10	**228 : 234 / 251 : 257**	**8**	✓	**gcgugcg**	... **ugcacgc**
11	**84 : 87 / 92 : 95**	**7**	✓	**gggc**	... **gccu**
12	**236 : 242 / 246 : 252**	**5**	✓	**uaccgug**	... **cgcggug**
13	**63 : 67 / 206 : 210**	**2**	✓	**aggcg**	... **cgccu**
14	**23 : 26 / 31 : 34**	**2**	✓	**cgga**	... **uccg**
15	63 : 66 / 92 : 95	2		aggc	... gccu
16	243 : 249 / 255 : 261	1		gcccgcg	... cgcgggu
17	152 : 160 / 169 : 177	1		aggugcggu	... accgcauuu
18	19 : 24 / 136 : 141	1		gccgcg	... cgcggu
19	76 : 84 / 203 : 211	1		cggccgggg	... ccccgccug
20	118 : 123 / 156 : 161	1		uaccgc	... gcggua
21	67 : 71 / 178 : 182	1		gccgg	... ccggu
22	135 : 138 / 228 : 231	1		gcgc	... gcgu
23	75 : 80 / 84 : 89	1		acggcc	... gggcgu

FIG. 3.14 – *Les tiges trouvées pour la première séquence de RNases (A.ferrooxidans) à l'issue des 14 corepliements. Le tableau indique pour chaque tige sa position et sa fréquence d'apparition. Une colonne indique aussi si la tige est correcte, c'est-à-dire si elle appartient à la structure proposée dans la banque de données.*

Deuxième tentative : les graphes de repliement



Supposons qu'une tige a de la séquence A se replie avec une tige b de la séquence B lors du corepliement A vs. B, et qu'elle se replie avec une tige c de la séquence C lors du corepliement A vs. C. Notre hypothèse est que ces repliements doivent être **transitifs** : si a est une vraie tige de la séquence A alors les tiges b et c doivent être correctes elles aussi, et se replier ensemble lors du corepliement B vs. C.

La FIG. 3.15 nous montre que la tige intruse est ce qu'on pourrait appeler une tige *attractive*, c'est-à-dire une tige potentielle d'énergie très forte (ici -1620), qui aura tendance à se replier avec n'importe quelle autre tige potentielle copliable des autres séquences, car elle apporte une contribution importante à l'énergie globale du repliement. Il existe en effet très souvent dans les autres séquences de fausses petites tiges potentielles (généralement d'énergie assez faible), susceptibles d'être copliées avec la tige intruse. Le fait que ces petites tiges ne se plient pas entre elles lors des corepliements, c'est-à-dire le défaut de connectivité du graphe, donne l'alerte sur cette tige.

Tige n^o8 attractive (intruse) vs. **Graphe fortement connecté**

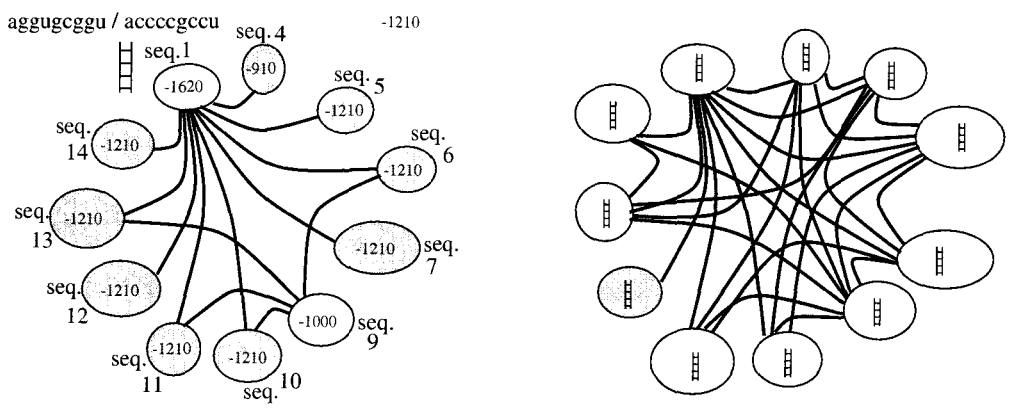


FIG. 3.15 – *Graphe d'une tige attractive, comparé à un "bon" graphe. Nous avons grisé les nœuds d'arité inférieure ou égale à 2.*

Nous allons systématiquement étudier la connectivité en générant, à l'issue des corepliements, le graphe de correspondance des tiges repliées. L'examen des composantes connexes de ce graphe nous permettra de diminuer le bruit de fond causé par les tiges rarement prédites, d'éliminer les intrus de haute fréquence, et finalement de *détecter* la présence ou l'absence d'une structure partagée par les séquences étudiées.

3.2.2 L'algorithme pour n séquences

La version finale de CARNAC prend en entrée n séquences d'ARN et produit pour chaque séquence un repliement consistant en une liste de tiges emboîtées ou juxtaposées (c'est-à-dire formant une structure secondaire).

Pour cela l'algorithme forme d'abord tous les repliements communs des séquences prises deux à deux avec CARNAC_2, puis ces corepliements sont combinés pour obtenir une unique structure par séquence. Ce travail est réalisé en quatre étapes, que nous allons détailler dans la suite :

Etape 0 – Formation de tous les repliements deux à deux avec CARNAC_2

Etape 1 – Construction du *graphe des tiges*

Etape 2 – Modifications sur le graphe :

- regroupement de nœuds (tiges empilées)
- ajout d'arcs (tiges non covariées)

Etape 3 – Examen de la connectivité dans les composantes connexes par un indice numérique qui mesure :

- la qualité des nœuds
- la densité des arcs
- la quantité de covariations

Etape 4 – Incorporation gloutonne des tiges séquence par séquence (suivant l'indice de leur composante)

Etape 1 – Construction du graphe des tiges

A l'issue des repliements deux à deux formés par CARNAC_2, un graphe est construit dont les nœuds sont l'ensemble des tiges, un arc entre deux tiges signifiant qu'elles ont été corepliées. Nous appelons ce graphe le *graphe des tiges.*

La FIG. 3.16 illustre sur l'exemple simple de trois séquences la construction du graphe des tiges. De par sa définition, ce graphe revêt certaines propriétés immédiates. C'est d'abord un graphe n-partite (où n est le nombre de séquences), deux tiges de la même séquence ne pouvant pas avoir été corepliées. C'est généralement un graphe qui aura plusieurs composantes connexes car deux tiges appartenant à des séquences différentes, mais dont les positions sont fort éloignées auront peu de chances d'avoir été corepliées. Idéalement, ces composantes devraient correspondre aux différentes tiges du repliement consensus. Enfin l'information de la fréquence de repliement est lisible sur le graphe puisqu'elle correspond à l'arité du nœud.

La construction du graphe ne présente aucune difficulté : on prend une tige non visitée et on forme la clôture transitive de ses relations de corepliement, et ainsi de suite jusqu'à avoir épuisé l'ensemble des tiges prédites. De cette manière on construit une à une les composantes connexes.

séquence 1 vs. séquence 2 : (A,E) (B,F) (D,G)
séquence 1 vs. séquence 3 : (A,H) (B,I) (C,J)
séquence 2 vs. séquence 3 : (E,H) (F,J)

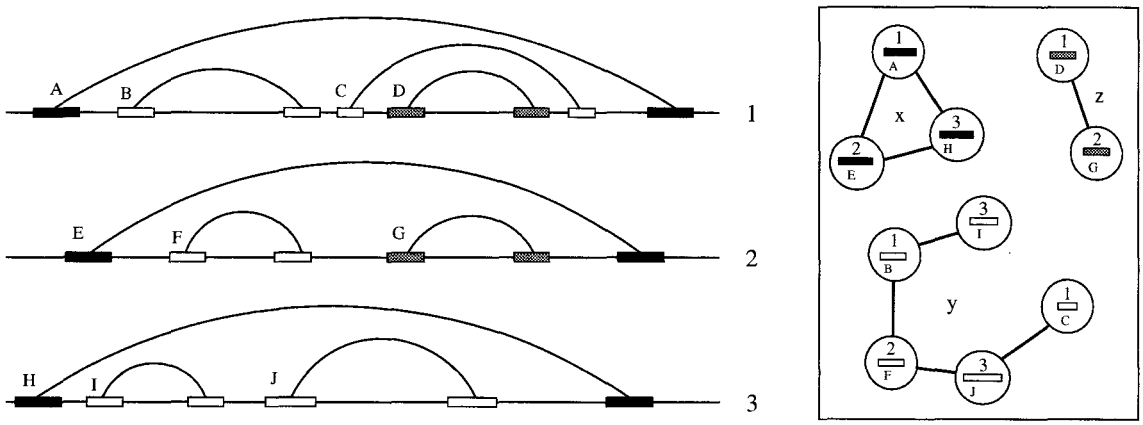


FIG. 3.16 – *Construction du graphe des tiges sur un exemple simple de trois séquences. Après avoir replié les séquences deux à deux avec CARNAC$_2$, on a obtenu pour chacune l'ensemble des tiges représentées sur le schéma de gauche. Nous avons indiqué les couples de tiges pour chaque corepliement (1 vs. 2, 1 vs. 3 et 2 vs. 3). Le graphe des tiges (représenté sur le schéma de droite) est alors construit de la manière suivante : ses nœuds sont l'ensemble des tiges obtenues, un arc entre deux nœuds signifie que les deux tiges ont été repliées ensemble lors du repliement des deux séquences correspondantes. La tige F par exemple a été repliée avec la tige J lors du repliement des séquences 2 et 3. Le graphe comporte ici trois composantes connexes. Les couleurs sur le schéma de gauche ne sont là que pour identifier aisément les tiges des composantes connexes. La composante x inspire confiance (c'est une clique), la composante z un peu moins (toutes les séquences n'y sont pas représentées), la composante y est franchement mauvaise (une même séquence est parfois représentée par plusieurs tiges).*

Etape 2 – Modifications sur le graphe

Nous réalisons deux modifications significatives dans la définition et la construction du graphe de manière à prendre en compte les empilements de tiges d'une part, et l'absence de covariation d'autre part.

empilements

Par défaut les tiges de CARNAC se terminent sur un appariement canonique. Une tige un peu trop bruitée par des appariements non-canoniques ou comportant une boucle interne asymétrique apparaitra comme l'empilement de deux tiges[7]. A l'issue du repliement commun de ces tiges, il arrive ainsi qu'une partie de la tige seulement ait été prédite au détriment de l'autre. Or dans les différents corepliement ce n'est pas toujours la même partie qui aura été prédite. Au niveau du graphe, on est donc amené à amender un peu la construction qui vient

7. CARNAC$_2$ prend déjà en compte avant corepliement une partie de ces tiges : sous certaines conditions, lorsqu'un empilement est détecté, on ajoute au jeu de tiges potentielles la tige formée par l'empilement des deux. Mais pour ne pas faire exploser le nombre de tiges en entrée, seuls les cas évidents sont pris en compte (très petite boucle interne, empilement de deux tiges seulement, voir page 58).

d'être décrite en regroupant les tiges empilées (FIG. 3.17). Les nœuds du graphe deviennent ainsi des classes d'équivalence de tiges et les arcs sont factorisés.

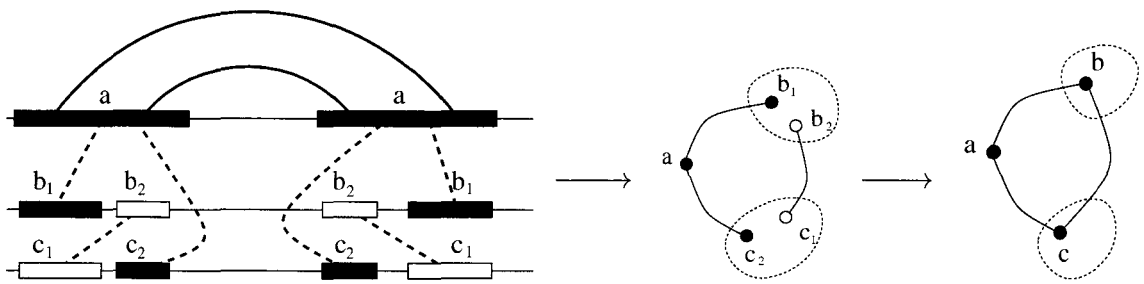


FIG. 3.17 – *Regroupement de tiges lorsqu'elles sont empilées. Sur le schéma de gauche, les tiges b et c sont fractionnées respectivement en b_1 et b_2, et c_1 et c_2. Les traits pointillés indiquent les tiges qui se sont repliées ensemble. Dans le graphe, nous "factorisons" les tiges empilées en les considérant comme une unique tige. Les arcs correspondants sont aussi regroupés.*

absence de covariation

Les arcs du graphe correspondent aux tiges qui ont été copliées, nous nommons ces arcs des arcs de type **'coplié'** et nous ajoutons d'autres arcs – de type **'identité'** – entre les tiges ne présentant aucune covariation. Deux tiges reliées par un arc de type 'identité' ne peuvent pas avoir été repliées ensemble, puisqu'elles ne sont pas copliables. Par conséquent, elles ne sont pas informatives, mais il ne faut pas non plus que l'absence d'arc pénalise la connectivité du graphe. L'indice de qualité que nous définissons à l'étape suivante pour chaque composante connexe tient compte des deux types d'arc 'coplié' et 'identité'.

Etape 3 – Examen des composantes connexes et incorporation des tiges

La composante connexe idéale est une clique où chaque séquence est représentée par un unique nœud. Nous définissons pour chaque composante un indice qui exprime dans quelle mesure celle-ci se rapproche du cas idéal. Cet indice est le produit de deux autres indices (*node_index* et *edge_index*) qui mesurent la qualité de la composante respectivement au niveau des nœuds et au niveau des arcs.

Ces indices sont calculés à l'aide des caractéristiques de la composante ***N***, ***Ns***, ***sq***, ***co***, ***id***, ***me***, dont nous donnons ci-dessous la définition.

- ***N*** est le nombre de nœuds de la composante connexe.
- ***Ns*** désigne le nombre de séquences différentes présentes dans la composante.
- Certains corepliements ne sont pas réalisés (lorsqu'aucun point d'ancrage n'est trouvé ou que les ensembles de tiges copliables sont vides). On examine la clôture transitive des corepliements au niveau des séquences et on désigne par ***sq*** son cardinal.
- ***co*** est le nombre d'arcs de type 'coplié'.
- ***id*** le nombre d'arcs de type 'identité'.
- ***me*** $= \frac{N(N-1)}{2}$ le nombre maximal possible d'arcs.

$$node\_index := \left[\frac{Ns - (N - Ns)}{sq}\right]^2 \qquad edge\_index := \frac{co}{me - id}$$

L'indice de la composante est le produit des deux indices précédents :

$$index := node\_index \ . \ edge\_index$$

Les deux indices $node\_index$ et $edge\_index$ sont compris entre 0 et 1. On a en effet de manière évidente $Ns \leq N$ et $Ns \leq sq$. Et on suppose aussi que $N \leq 2Ns$, sinon la composante est d'emblée considérée comme mauvaise car cela signifierait qu'il existe au moins une séquence présente trois fois dans le graphe.

sq est généralement égal au nombre n de séquences fournies en entrée. Il arrive que $sq \neq n$ lorsqu'au lieu d'avoir une structure commune globale, l'ensemble de séquences se partitionne en plusieurs sous-ensembles partageant chacun une structure commune.

$index = 1$ (*i.e.* maximal) lorsque $node\_index = edge\_index = 1$, c'est-à-dire lorsque $Ns = N = sq$ (chaque séquence est présente une et une seule fois dans la composante) et $me = co + id$ (la composante est une clique).

Etape 4 – Incorporation des tiges séquence par séquence

Chaque tige se voit attribuer l'indice de sa composante. Puis pour chaque séquence les tiges sont incorporées de manière gloutonne par indice décroissant jusqu'à un seuil donné. Nous avons essayé une approche plus fine, de type Nussinov et Jacobson avec les indices des composantes comme poids pour les tiges. Cela n'apporte rien ici, l'incorporation gloutonne est parfaitement satisfaisante. Il est par contre indispensable de fixer un seuil car un très mauvais graphe aura par définition un très mauvais indice, mais cet indice ne sera pas nul. On examine quand même à chaque incorporation si la tige n'est pas en conflit avec les éléments de structure déjà formés, bien que de tels conflits soient rares à cette étape de l'algorithme. A ce stade, on tolère que les tiges se chevauchent légèrement quitte à raccourcir l'une d'entre elles ou bien les deux, ce qui permet de récupérer *a posteriori* des tiges maximales qui auraient subi lors des corepliements le problème de chevauchement évoqué page 65.

Le chapitre suivant présente les résultats expérimentaux de la version n séquences de CARNAC que nous venons de décrire.

Chapitre 4

Résultats expérimentaux

La version n séquences de CARNAC, en combinant les corepliements, améliore la fiabilité des prédictions. Nous présentons les résultats de CARNAC sur des jeux de séquences dont les caractéristiques sont assez disparates : leur taille s'échelonne de quelques dizaines de nucléotides à quelques milliers, leur volume (le nombre de séquences) de deux à quelques dizaines, leur taux de similitude peut être particulièrement bas. Nous comparons, au fil des exemples, les prédictions de CARNAC avec celles des autres méthodes dont nous avons parlé au paragraphe 2.5.

Nous revenons tout d'abord sur le jeu de RNases P qui nous a guidé dans le chapitre précédent et nous montrons sur cet exemple la robustesse de CARNAC face aux variations de structure et le gain de CARNAC sur CARNAC_2. Nous comparons les résultats obtenus à ceux de RNAGA [CLM00] et FOLDALIGN [GHS97], méthodes qui prennent elles-aussi n séquences non alignées.

Nous montrons ensuite sur une famille de séquences qui s'aligne particulièrement mal (Telomerase-cil) que les prédictions de CARNAC sont en accord avec celles proposées dans la banque RFAM [GJBM$^+$03]. A cette occasion nous montrons que des méthodes qui prennent en entrée des séquences alignées, telles que PFOLD [KH99] ou X2S [JW99] ne fonctionnent pas. Nous comparons aussi nos résultats à ceux de MFOLD [Zuk03, MSZT99] et RNAGA [CLM00].

Puis nous avons choisi des séquences pour lesquelles la structure est connue de longue date (ARNt, ARNr). Nous montrons sur ces jeux la rapidité de CARNAC et son caractère universel, ces séquences ayant des longueurs très différentes : environ 80 nucléotides pour les ARNt, jusqu'à plusieurs milliers pour les ARNr. Avec les ARNt nous discutons de l'obligation de covariation qui est faite sur les tiges dans l'algorithme CARNAC_2. Pour les ARNr, nous comparons les prédictions de CARNAC à celles de MFOLD [Zuk03, MSZT99].

Nous voyons enfin sur deux exemples (ARNm Cytochrome, 5′ UTR enterovirus) que CARNAC permet aussi de *détecter* la présence ou l'absence d'une structure commune, c'est-à-dire tout simplement la présence ou non d'une structure, si l'on fait d'emblée l'hypothèse que les séquences sont fonctionnellement homologues.

4.1 RNases P

En considérant à nouveau le jeu de séquences du chapitre précédent (la famille *Delta/Epsilon Purple Bacteria RNase P RNA*), nous montrons les spécificités de CARNAC en comparaison à deux autres méthodes (RNAGA [CLM00] et FOLDALIGN [GHS97, GSS01]) qui se placent sensiblement dans le même contexte. Nous constatons en particulier la robustesse de CARNAC face aux variations au niveau de la structure (insertion ou délétion d'une tige) et au niveau de la séquence (séquences divergentes dont la structure est conservée). Nous cherchons aussi à mettre en lumière la philosophie de CARNAC, qui consiste à obtenir une prédiction de haute fiabilité, quitte à ne prédire qu'une partie de la structure.

Nous rappelons sur la TABLE 4.1 les résultats de $CARNAC_2$ que nous avions obtenus au chapitre précédent (voir la TABLE 3.1) afin d'illustrer le gain obtenu par CARNAC lorsqu'on dispose de 5 séquences au lieu de 2. Quand on compare à la moyenne des prédictions pour les séquences prises deux à deux, on observe qu'on prédit à chaque fois plus d'appariements corrects en quantité et en proportion, et qu'on couvre une partie plus importante de la structure.

	D.desulfuricans	**D.vulgaris**	**G.sulfurreducens**	**C.jejuni**	**H.pylori**
D.desulfuricans	–	59 93% 49%	87 85% 33%	61 88% 41%	51 92% 45%
D.vulgaris	65 93% 44%	–	93 81% 31%	52 78% 55%	54 70% 55%
G.sulfurreducens	88 81% 34%	81 86% 35%	–	63 84% 42%	48 89% 49%
C.jejuni	93 80% 31%	65 89% 46%	84 80% 39%	–	42 88% 56%
H.pylori	77 94% 33%	65 73% 56%	69 85% 47%	35 94% 64%	–
moyenne $CARNAC_2$	81 87% 36%	68 85% 47%	83 83% 38%	53 86% 51%	49 85% 51%
CARNAC	84 94% 27%	74 93% 36%	80 85% 39%	61 88% 41%	57 89% 40%

TAB. 4.1 – *Les résultats de $CARNAC_2$ et CARNAC pour la famille Delta/Epsilon Purple Bacteria RNase P RNA. Pour chaque séquence, le premier nombre indique le nombre total d'appariements prédits, le second le pourcentage de vrais positifs, et le troisième le pourcentage de faux négatifs.*

Les méthodes RNAGA et FOLDALIGN se placent dans les mêmes conditions que CARNAC puisqu'elles prennent en entrée n séquences non alignées. Le principe et le fonctionnement des algorithmes ont été détaillés au paragraphe 2.5, qui comporte en page 50 un tableau synthétique.

La TABLE 4.2 présente les résultats comparés de CARNAC, RNAGA et FOLDALIGN.

RNAGA [CLM00] (voir page 48) construit la structure commune à partir d'un ensemble de séquences non-alignées. Par défaut RNAGA donne dix repliements plausibles pour chaque séquence. Nous observons que toutes les tiges prédites sont des tiges terminales, parfois empilées, mais sans structure arborescente.

FOLDALIGN [GHS97, GSS01](voir page 49) L'objet de FOLDALIGN est clairement différent de celui de CARNAC car le programme recherche des tiges localement conservées, sans structure arborescente. On obtient ainsi en sortie une unique tige terminale. Comme nous n'avons pas réussi à replier les séquences complètes, nous avons préparé un jeu de séquences

tronquées en excisant correctement (relativement à leur structure) la partie droite et la partie gauche des séquences de manière à avoir des ARN d'environ 250 bases. Sur ce jeu de séquences, la tige commune prédite par FOLDALIGN est correcte (en tant que tige). Quelques erreurs au niveau des nucléotides apparaissent aux extrémités de la tige.

CARNAC

nom	appariements	vrais positifs	faux négatifs
D. desulfuricans	84	79 (94%)	27%
D. vulgaris	74	69 (93%)	36%
G. sulfurreducens	80	68 (85%)	39%
C. jejuni	61	54 (88%)	41%
H. pylori	57	51 (89%)	40%

RNAGA

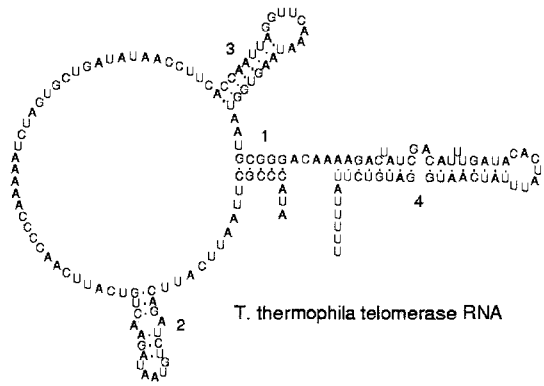
nom	rang	appariements	vrais positifs	faux négatifs
D.desulfuricans	1	90	64 (71%)	41%
D.vulgaris	2	91	53 (58%)	51%
G.sulfurreducens	4	102	67 (66%)	39%
C.jejuni	10	79	43 (55%)	52%
H.pylori	1	84	40 (48%)	52%

FOLDALIGN

nom	appariements	vrais positifs	faux négatifs
D. desulfuricans	12	5 (41%)	95%
D. vulgaris	12	5 (41%)	95%
G. sulfurreducens	12	5 (41%)	95%
C. jejuni	13	5 (38%)	94%
H. pylori	10	5 (50%)	94%

TAB. 4.2 – *Comparaison des résultats de CARNAC à ceux de RNAGA et FOLDALIGN. Pour chaque repliement nous annonçons le nombre d'*appariements *prédits, le nombre et le pourcentage de* vrais positifs, *et le pourcentage de* faux négatifs. *Pour RNAGA, nous avons sélectionné la meilleure structure parmi les dix proposées en indiquant son* rang.

4.2 Telomerase-Cil



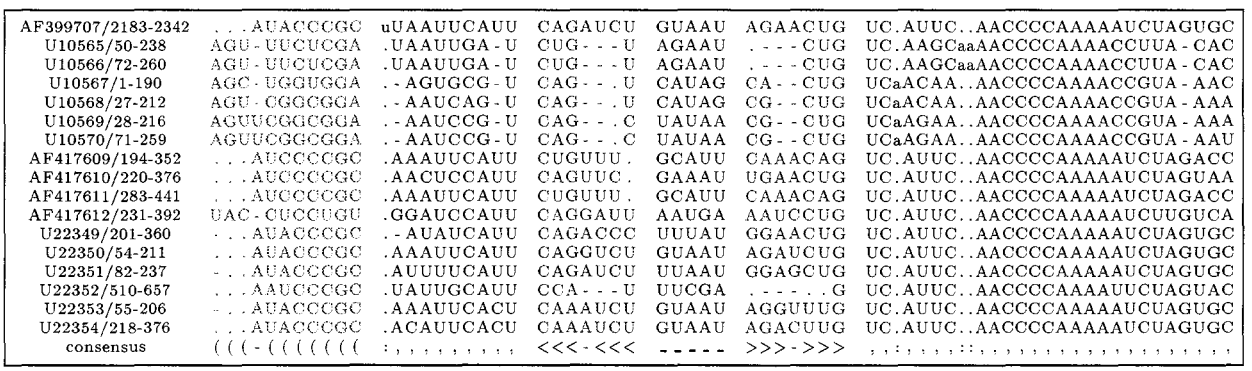
Le Télomérase est un complexe RiboNucléo-Proteique qui permet l'élongation et la conservation de la taille des télomères. Cette enzyme comprend une transcriptase inverse spécialisée renfermant une sous-unité catalytique (TERT : Telomerase Reverse Transcriptase) et une molécule d'ARN. Les ARN du Télomérase diffèrent beaucoup en séquence et en structure entre les différents domaines.

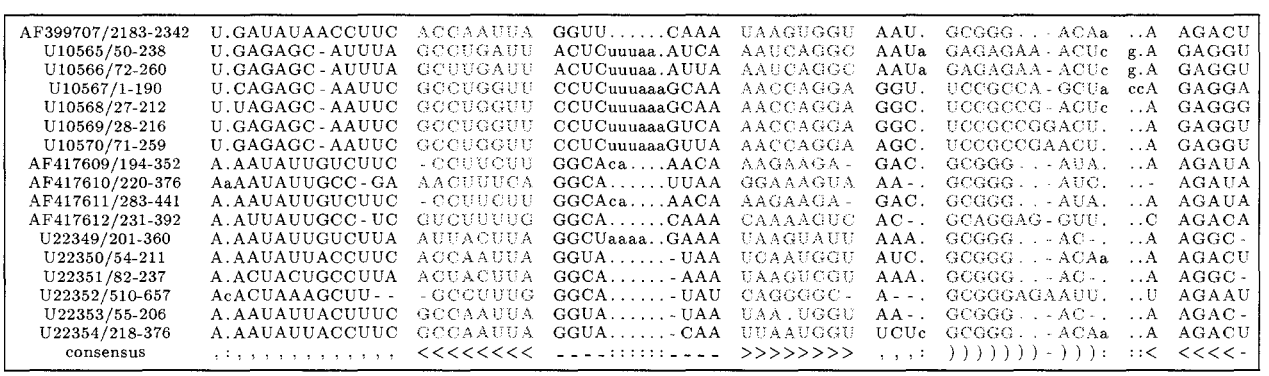
La FIG. 4.1 donne l'alignement et la structure consensus des 17 séquences de télomérases ciliés présentes dans la base RFAM [GJBM+03] `http://rfam.wustl.edu/`. Lorsqu'on tente de les aligner par des algorithmes classiques d'alignement multiple, ces séquences s'alignent très mal, que ce soit par un alignement global (FIG. 4.2) ou local (FIG. 4.3).

Nous avons appliqué CARNAC sur les trois séquences qui semblent présenter le plus de disparités (AF417611/283-441, U10565/50-238, AF417612/231-392). On vérifie visuellement sur la FIG. 4.6 que la structure obtenue est en accord avec celle de la FIG. 4.1 proposée dans la banque. La FIG. 4.7 présente les résultats comparés de CARNAC, RNAGA et MFOLD sur ces trois séquences. RNAGA ne trouve pas la structure. MFOLD, par contre, fonctionne très bien sur ces séquences, et prédit même un peu plus d'appariements que proposés dans la base de données. Il est probable que l'alignement des 17 séquences de la banque RFAM ne permette pas de conclure quant à ces tiges (pas assez de covariations, par exemple) et que les tiges supplémentaires prédites par MFOLD soient quand même correctes.

Lorsqu'on cherche la structure commune avec des méthodes qui partent d'un alignement, on n'arrive pas à trouver la bonne structure. Les FIG. 4.4 et FIG. 4.5 présentent respectivement les résultats des méthodes PFOLD [KH99] (voir page 45) et X2S [JW99] (voir page 46). Les résultats de PFOLD sont ici faux, sans surprise puisque l'alignement est faux et qu'il n'est pas corrigé par l'algorithme. Il existe par contre dans l'exécutable de X2S la possibilité de corriger l'alignement par itérations successives selon les éléments de structure secondaire déjà trouvés. Les auteurs n'en parlent pas dans leur publication, mais les commentaires qu'on peut lire dans le code source indiquent que ce réalignement est réalisé par recuit simulé selon l'algorithme de Metropolis. Malgré cette possibilité, l'alignement initial semble trop mauvais pour obtenir la convergence vers la bonne structure. Nous montrons sur la FIG. 4.5 les structures obtenues lors des cinq premières itérations.

Un tel jeu de séquences constitue typiquement le domaine d'application de CARNAC : les séquences s'alignent mal car elles ont fortement muté d'une espèce à l'autre et la structure (globalement conservée) a subi quelques variations. Les méthodes qui s'appuient sur un alignement, comme on vient de le voir, ne trouvent pas la structure.





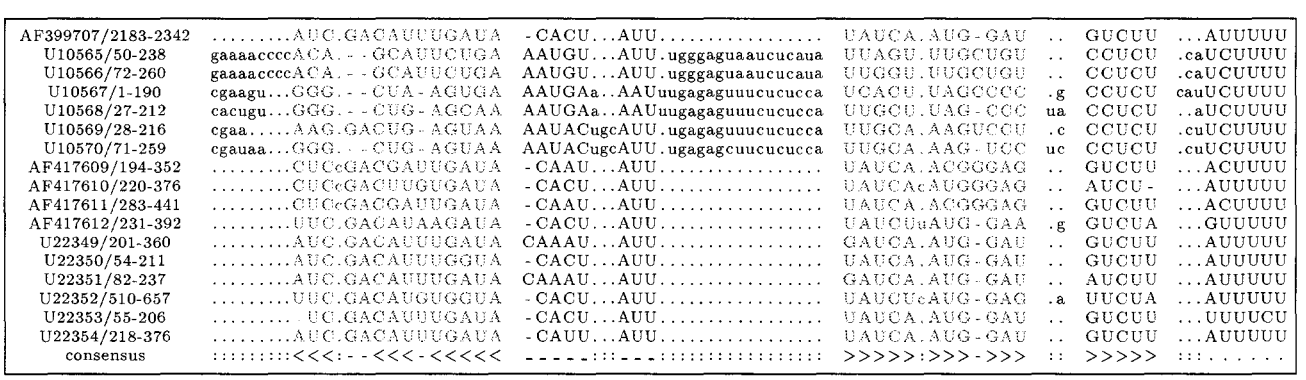


FIG. 4.1 – *Alignement des séquences de Telomerase-Cil (AF417611 / 283-441, U10567 / 1-190, U10570 / 71-259, U22350 / 54-211, U22351 / 82-237, AF417609 / 194-352, U10569 / 28-216, U22354 / 218-376, U22353 / 55-206, AF399707 / 2183-2342, U22352 / 510-657, U22349 / 201-360, U10568 / 27-212, U10566 / 72-260, AF417610 / 220-376, U10565 / 50-238, AF417612 / 231-392) selon la structure proposé dans la base RFAM.*

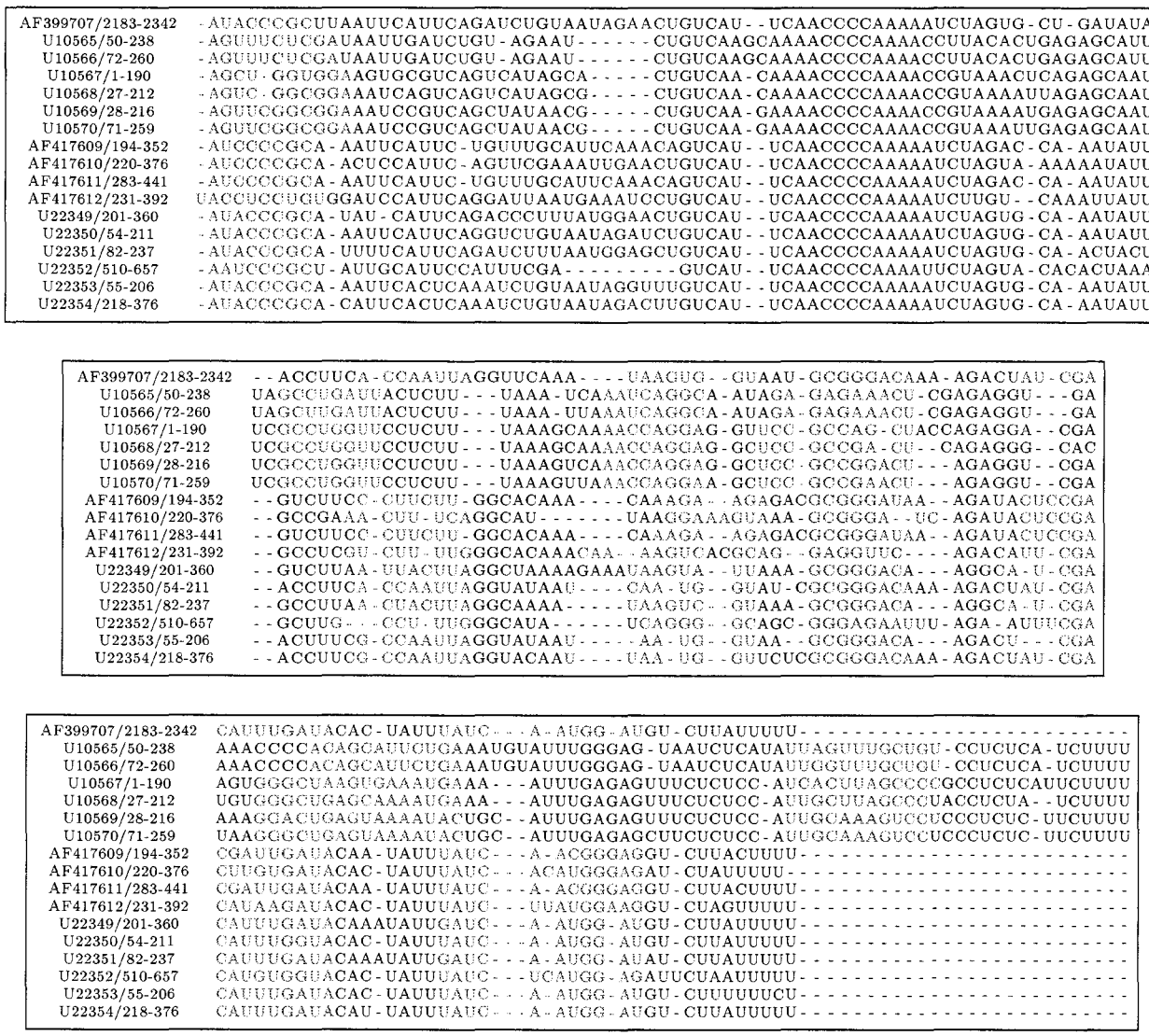


FIG. 4.2 – *Alignement multiple des séquences de Telomerase-Cil avec CLUSTALW [THG94].*

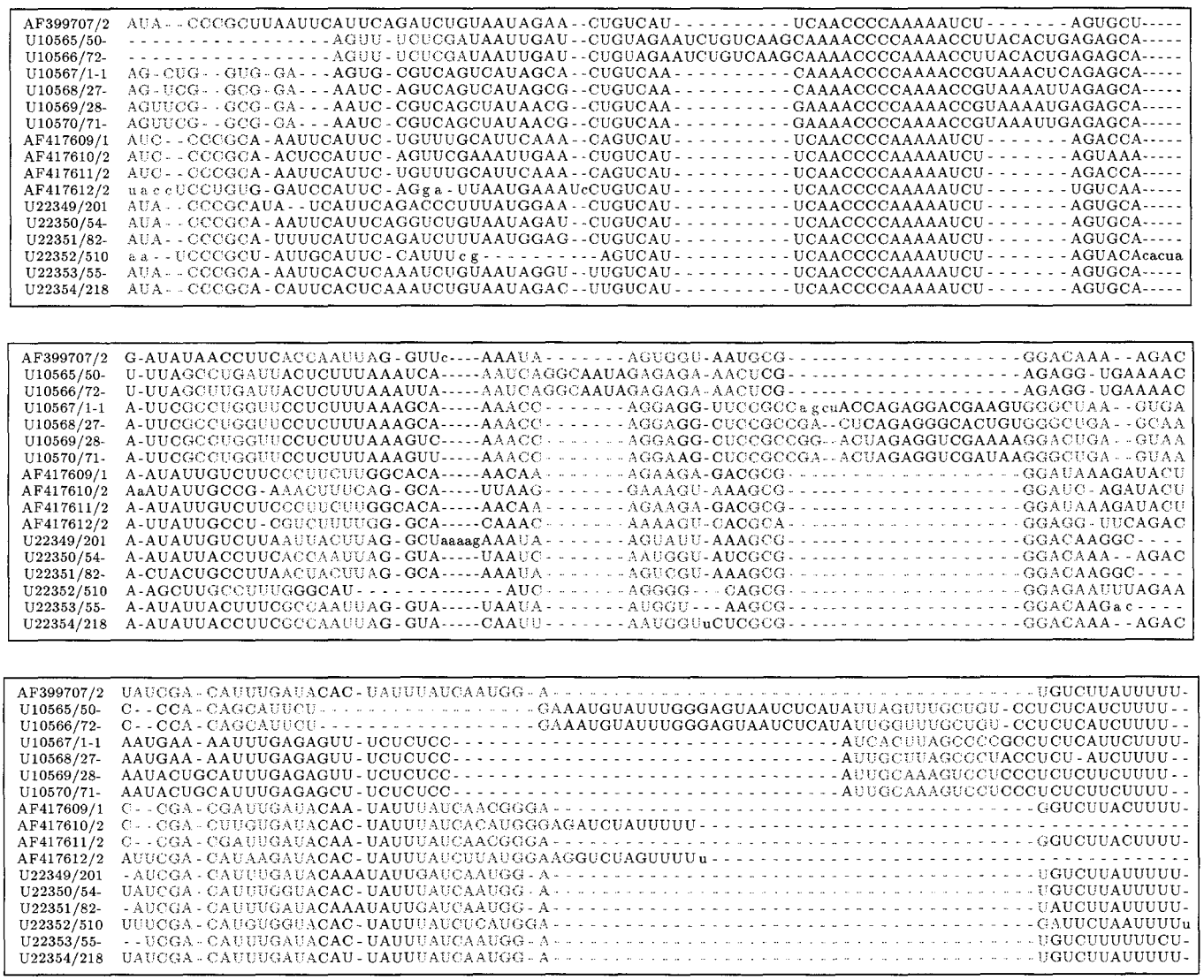


FIG. 4.3 – *Alignement multiple des séquences de Telomerase-Cil avec DIALIGN 2 [MDW96].*

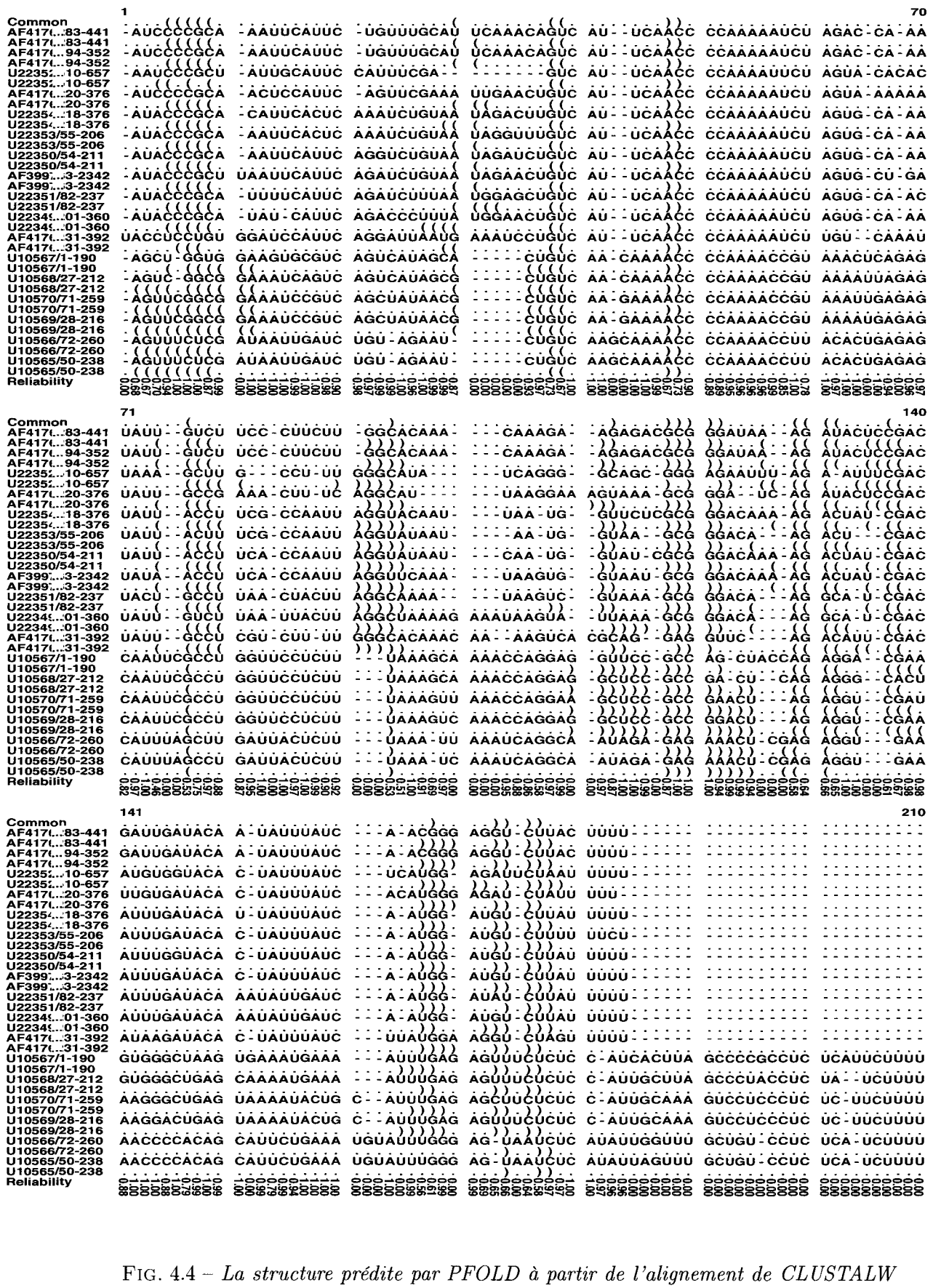
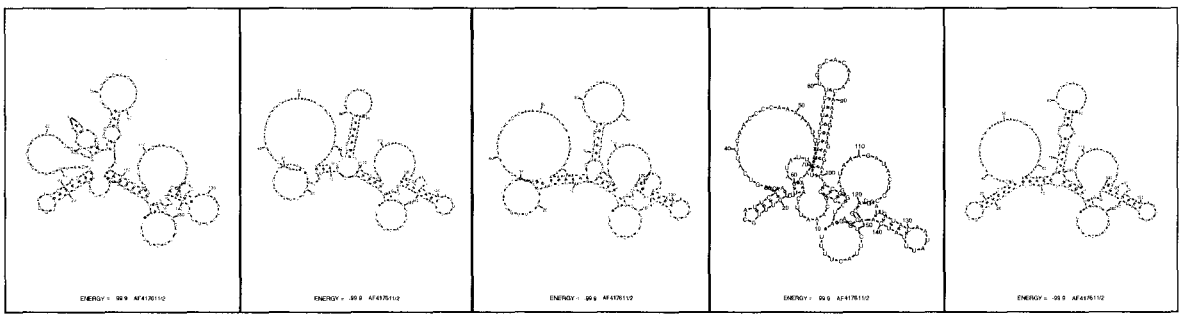
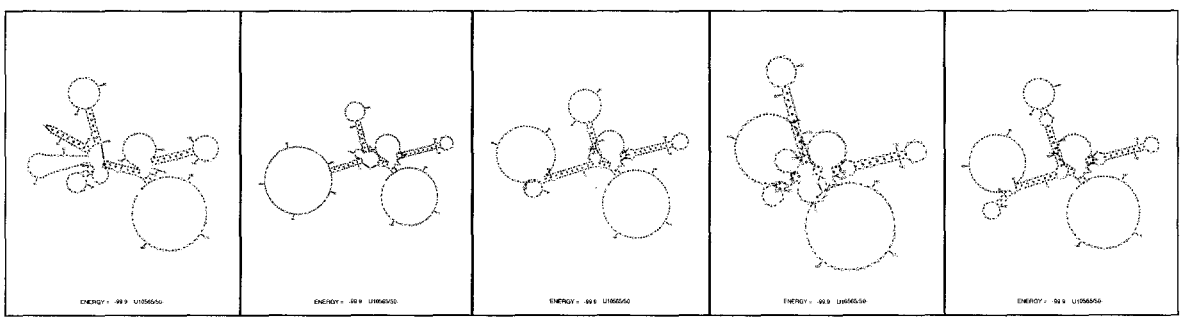


FIG. 4.4 – *La structure prédite par PFOLD à partir de l'alignement de CLUSTALW*

> AF417611/283-441 (itérations 1 à 5)



> U10565/50-238 (itérations 1 à 5)



> AF417612/231-392 (itérations 1 à 5)

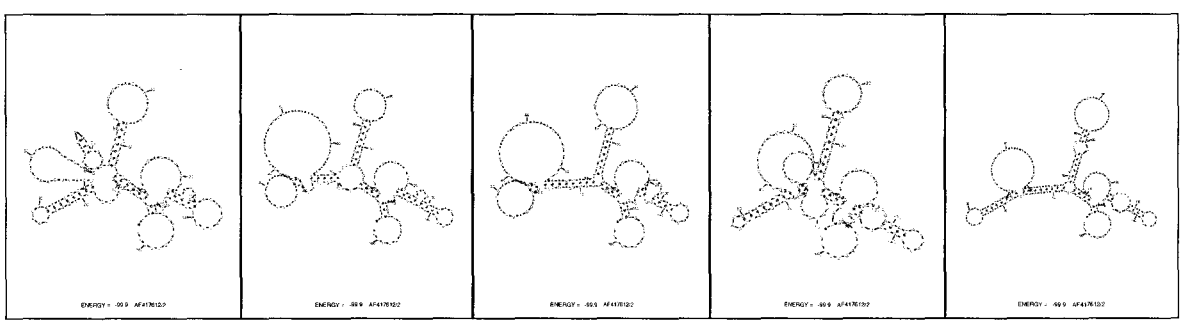


FIG. 4.5 – *Telomerase-Cil – Les structures pour les trois séquences AF417611/283-441, U10565/50-238 et AF417612/231-392, prédites par X2S à partir de l'alignement de CLUSTALW de l'ensemble des séquences : les cinq premières itérations.*

AF417611/283-441 U10565/50-238 AF417612/231-392

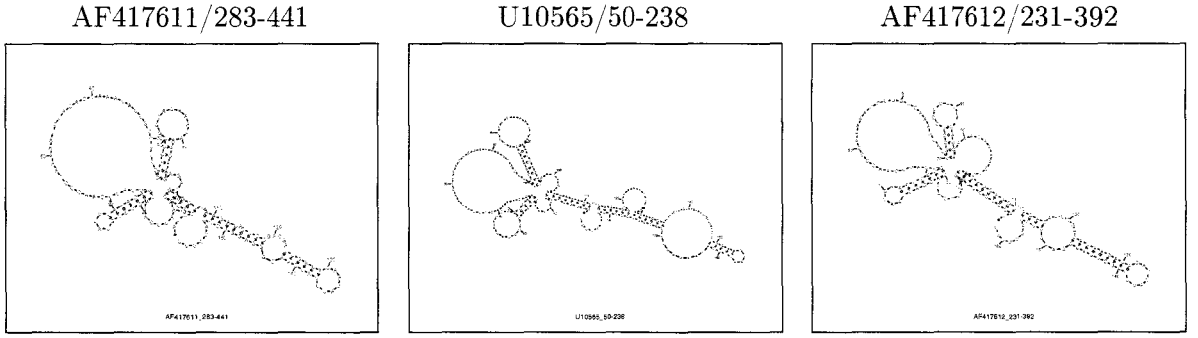


FIG. 4.6 – *Telomerase-Cil – Les structures prédites par CARNAC en utilisant les trois séquences AF417611/283-441, U10565/50-238 et AF417612/231-392.*

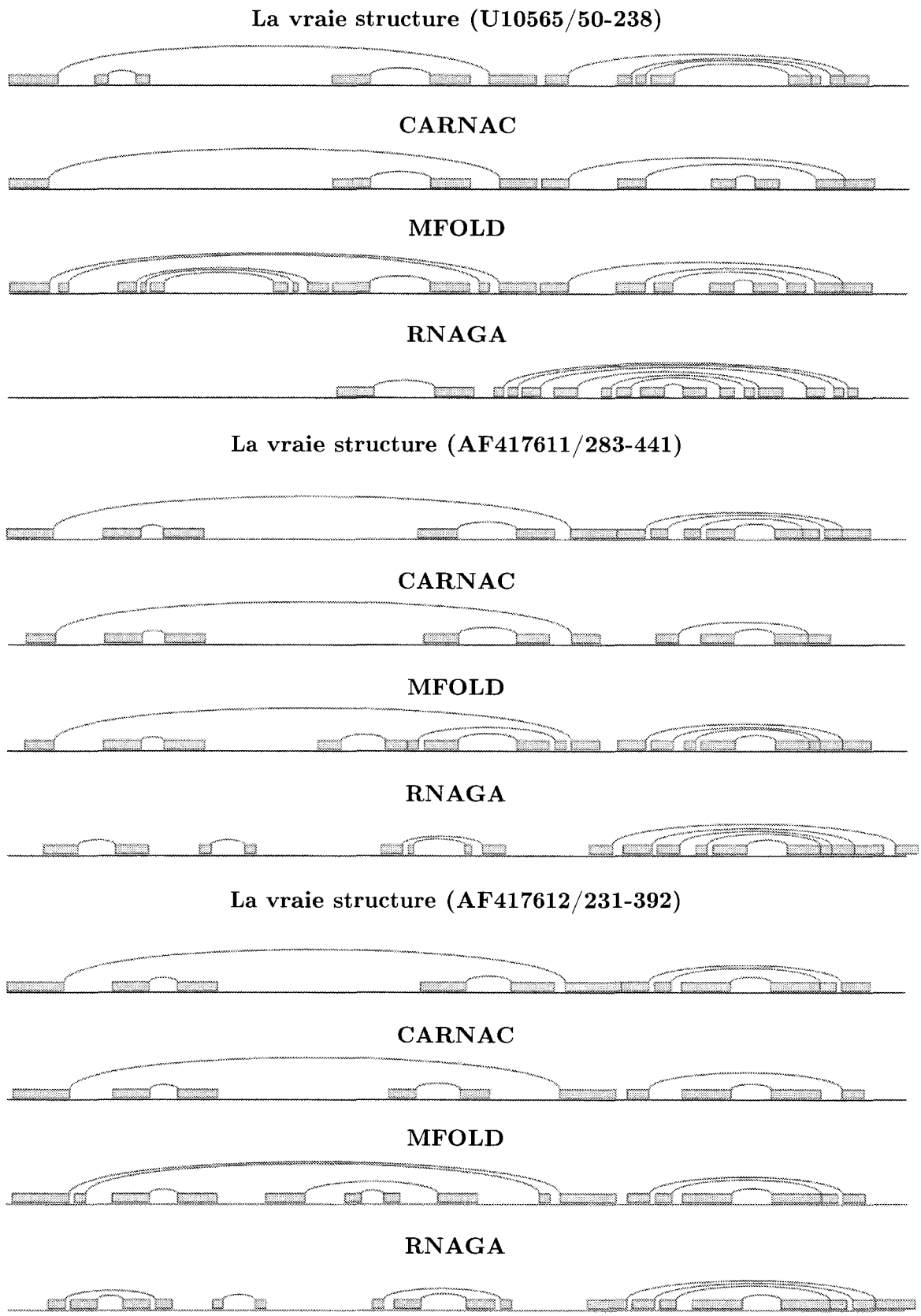


FIG. 4.7 – *Résultats comparés de CARNAC, RNAGA et MFOLD sur les trois séquences AF417611/283-441, U10565/50-238, AF417612/231-392.*

4.3 ARN de transfert

Les familles de séquences pour lesquelles on détient une grosse banque de données – ce qui est le cas des ARN de transfert et des ARN ribosomiques (dont nous parlerons dans la prochaine section) – ont en général été assez étudiées pour que leur structure consensus soit connue très précisément, admise et même vérifiée par des méthodes physiques.

Les ARN de transfert comportent entre 73 et 93 nucléotides. Leur séquence est assez variable, mais certains nucléotides occupent des positions spécifiques et impératives, l'extrémité 3′ par exemple se termine toujours par la séquence CCA. Les ARNt subissent beaucoup de modifications post-transcriptionnelles : méthylations et autres transformations.

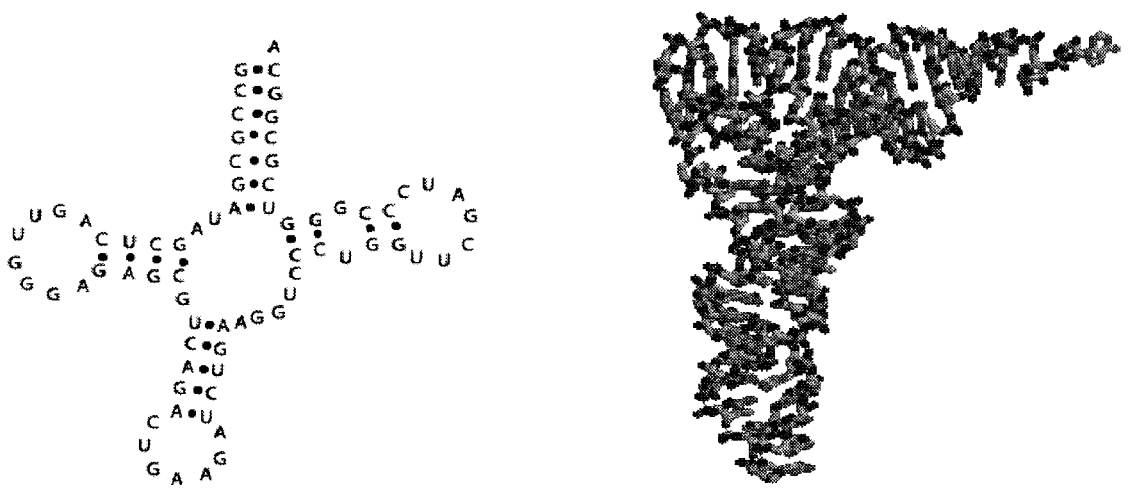
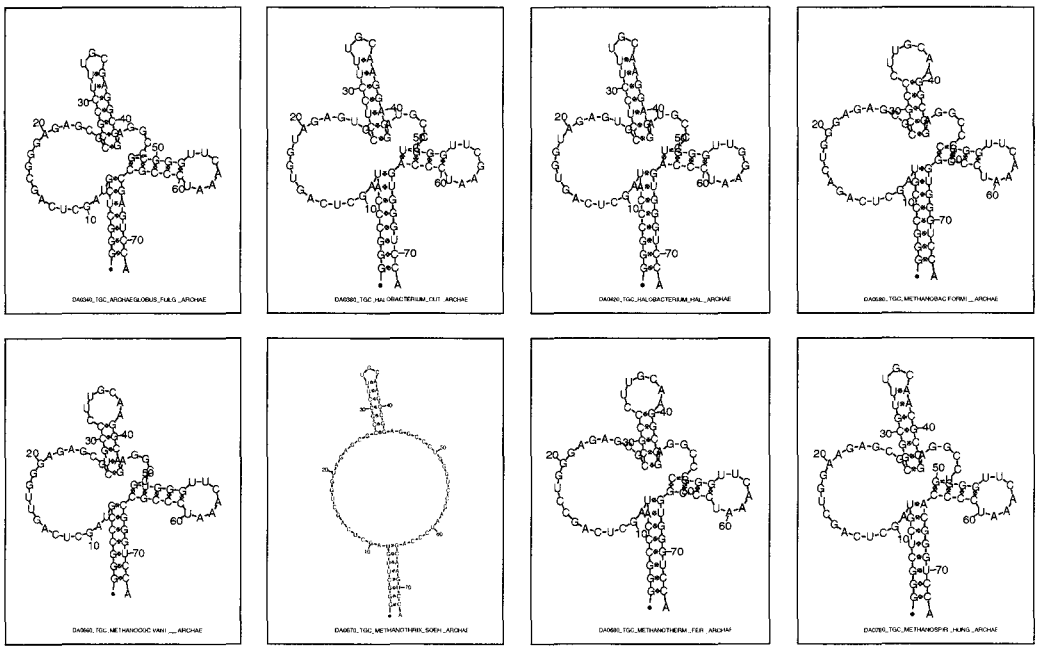


FIG. 4.8 – *La structure de l'ARN de transfert "en feuilles de trèfle" a été mise en évidence par M. Levitt en 1969 à l'aide de quatorze ARNt homologues et de beaucoup d'indices en faveur de cette structure (mutations compensatoires et informations physicochimiques). Les tiges se replient sur elles-mêmes pour former la structure spatiale en* $\boldsymbol{L}$[1].

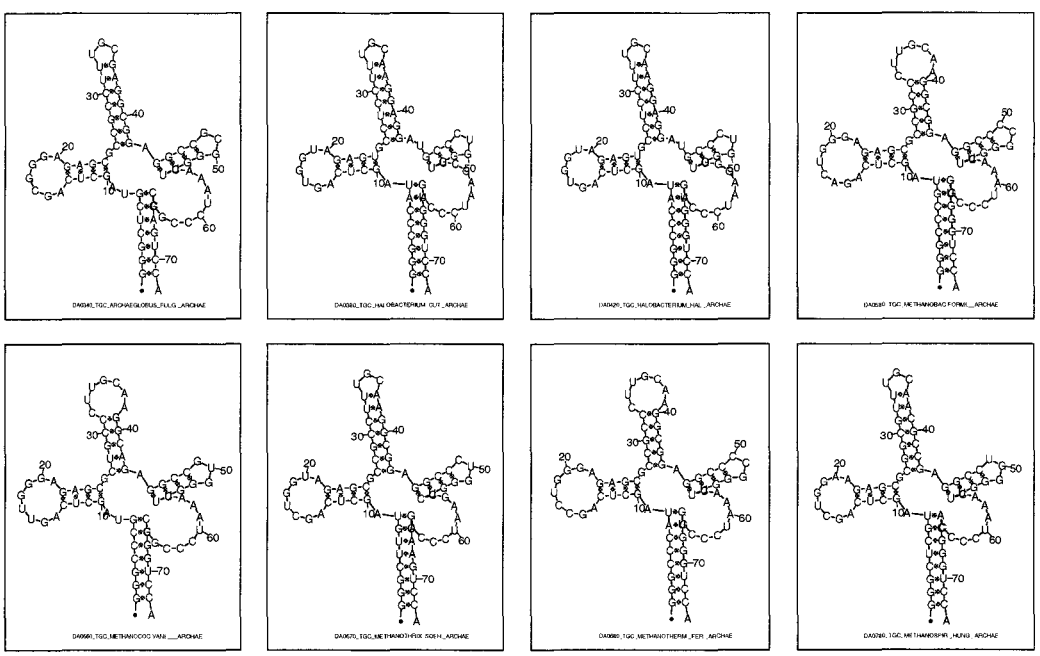
Nous avons replié 8 ARN de transfert (DA0340, DA0380, DA0420, DA0580, DA0660, DA0670, DA0680, DA0780, ponctionnés sur la base de donnée de l'EBI : `ftp://ftp.ebi.ac.uk/pub/databases/trna/seq.txt`) en choisissant arbitrairement des ARN qui présentaient des variations au niveau de la séquence (dans les grosses banques de données disponibles, on trouve souvent beaucoup de séquences très similaires). CARNAC trouve trois tiges sur les quatre, excepté pour une des huit séquences où l'algorithme n'en trouve que deux. Ceci s'explique assez simplement par le fait que dans la tige qui manque, on observe une absence de covariation d'une séquence à l'autre, la tige n'est donc jamais copliable.

1. *source de l'image* : `http://www.sc.fukuoka-u.ac.jp/~bc1/Biochem/NAfig/Phe-tRNA.gif`

Il arrive aussi, et c'est particulièrement vrai pour les ARN de transfert, que certaines tiges ne soient pas trouvées par CARNAC, dû au fait qu'elles se chevauchent (voir page 65).



Si l'on omet la condition de covariation lors du filtrage des tiges, CARNAC trouve la structure complète. Cependant dans la plupart des cas, il est dangereux de s'affranchir de cette hypothèse car elle amène souvent plus de faux positifs que de vrais positifs.



4.4 ARN ribosomiques

Les ARN ribosomiques se déclinent en trois types de séquences de longueurs différentes qui participent à la constitution du ribosome. Le ribosome est un assemblage de deux parties, la petite et la grande sous-unités, chacune étant un complexe RNP où interviennent les trois types d'ARNr, associés à plusieurs dizaines de protéines.

Chez les procaryotes, les trois types sont désignés par 23S (3000 nt), 5S (120 nt) et 16S (1500 nt). Chez les eucaryotes : 28S (5000 nt), 5.8S (150 nt) et 18S (1800 nt). Les deux premiers composent la grande sous-unité ribosomique (lsu rRNA), le dernier la petite sous-unité (ssu rRNA). Ces trois séquences sont issues d'un même préARNr (30S chez les bactéries, 45S chez les eucaryotes), ce qui assure leur production en quantités identiques. A titre indicatif, la transcription du gène 45S représente environ 50% de l'activité transcriptionnelle du noyau.

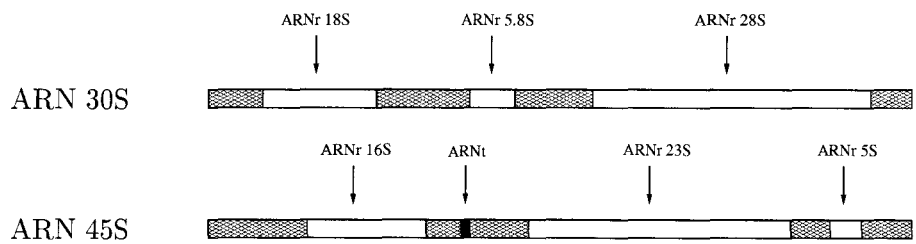


FIG. 4.9 – *Excision sans épissage du gène donnant les trois types d'ARNr*

L'ensemble des ARNr 16S de tous les domaines du vivant partagent une superstructure commune, mais le consensus est plus fort lorsque ces séquences appartiennent au même domaine. Nous avons choisi volontairement des espèces bien connues dans différents domaines de façon à tester la robustesse de CARNAC face à ces variations par rapport à la structure commune globale. Les quatre ARNr (M59126, X03235, J01695, K00637) proviennent de la banque de Woese *et al.* [WMG+80] (`http://www.rna.icmb.utexas.edu/`). Les séquences sont longues d'environ 1500 nucléotides et leur structure consensus comporte environ 80 tiges.

	MFOLD					CARNAC		
nom	sous-opt.	rang	app. pred.	vrais pos.	faux neg.	app. pred.	vrais pos.	faux neg.
M.jannaschii	14	9	494	294 (59%)	35%	325	257 (79%)	44%
S.solfataricus	16	7	495	287 (57%)	38%	355	300 (84%)	35%
E.coli	27	21	476	340 (71%)	28%	321	278 (86%)	41%
B.subtilis	17	5	489	271 (55%)	42%	324	268 (82%)	43%

TAB. 4.3 – *Résultats comparés de CARNAC et MFOLD (paramètres par défaut) sur un ensemble de 4 séquences d'ARNr 16S. Nous indiquons le nombre de structures sous-optimales données par MFOLD et, pour la meilleure d'entre elles, son rang parmi les sous-optimales ainsi que le nombre d'appariements prédits et les pourcentages de vrais positifs et faux négatifs.*

Parmi les méthodes hybrides dont nous avons parlé au chapitre précédent, X2S est la seule qui puisse s'appliquer aux ARNr 16S. Les autres font défaut, soit parce que les séquences sont

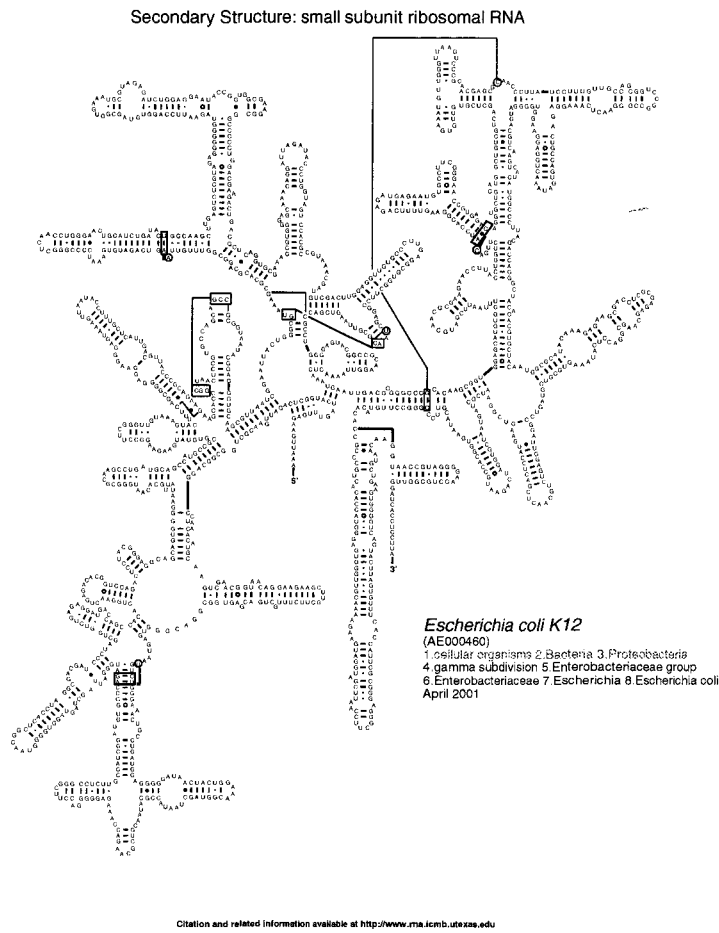


FIG. 4.10 – *ssu rRNA E.coli*

E.coli 16S ssu rRNA – la vraie structure

La structure prédite par X2S

La structure prédite par CARNAC

FIG. 4.11 – *E.coli 16S ssu rRNA – Structure produite par X2S à l'aide de 4 séquences d'ARNr alignées par CLUSTALW, en comparaison à la vraie structure et à celle produite par CARNAC en utilisant les 4 mêmes séquences non alignées.*

trop longues, soit parce qu'elles sont trop peu nombreuses (nous ne considérons qu'un jeu de quatre séquences). X2S trouve une structure qui est *visuellement* plus ou moins en accord, en terme de ramifications, avec celle qui est connue, mais l'algorithme semble comporter un bug dans la génération du fichier de structure au format CONNECT : on observe des décalages entre les appariements, ce qui nous empêche de quantifier la qualité des résultats en terme d'appariements. Nous avons donc comparé nos prédictions à celles données par MFOLD, mais nous donnons quand même dans la FIG. 4.11 un aperçu de la structure trouvée par X2S pour *E.coli* en comparaison à la vraie structure et à celle trouvée par CARNAC. Nous voyons que la qualité est sensiblement la même, bien qu'elle semble légèrement meilleure pour X2S, qui trouve un peu plus de tiges.

Sur la TABLE 4.3 on observe comme pour les RNases que MFOLD prédit, dans la meilleure de ses structures sous-optimales, souvent plus d'appariements corrects en quantité, mais nettement moins en proportion. Sur des molécules de cette taille la couverture de CARNAC tend à être aussi bonne que celle de MFOLD (peu de faux négatifs), tandis que sa spécificité (les vrais positifs) est nettement meilleure.

Nous avons aussi tenté d'améliorer la couverture de CARNAC en effectuant un "deuxième passage". On conserve les tiges déjà repliées, mais on tente d'en ajouter d'autres en faisant tourner CARNAC avec un jeu de tiges filtrées de manière plus *laxiste*. Pour ce filtrage, nous ne demandons plus que les tiges covarient. Par contre, nous ne sélectionnons que des tiges qui ne seraient pas en conflit avec les tiges déjà repliées. Sur ce jeu de séquences, contrairement aux ARNt, une telle approche ne fonctionne pas car elle amène plus de faux positifs que de bonnes tiges.

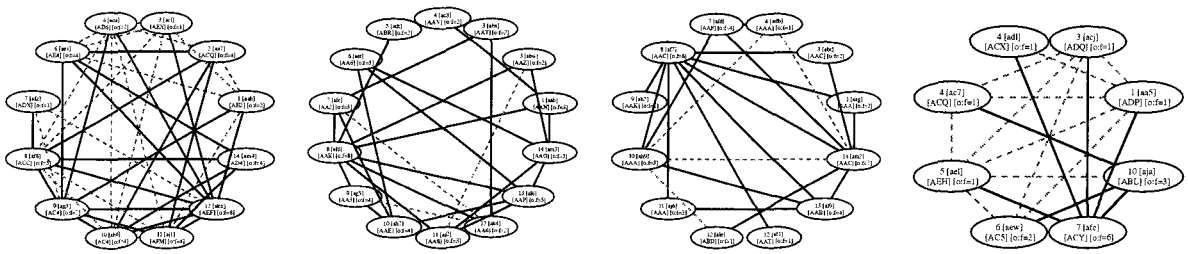
Concernant le temps de calcul, deux tiers des corepliements ont pris moins de 20s, un tiers moins d'une minute sur un Pentium III à 1GHz. Pour ces repliements nous avons utilisé moins de 5Mo de RAM. On observe que le temps de calcul et la mémoire utiles dépendent beaucoup du contenu en GC de la séquence car les séquences GC-riches produisent nettement plus de tiges potentielles. Il dépend aussi de la similitude des séquences : le nombre de tiges copliables est plus grand lorsque les séquences sont divergentes, à cause de la plus faible qualité des points d'ancrage.

4.5 ARNm Cytochrome

Dans des séquences d'ARN messagers codant pour une même protéine, on ne s'attend pas à ce que durant l'évolution, ces séquences aient subi une pression sélective afin qu'une structure commune soit conservée. En tous cas, aucune découverte en ce sens n'a été faite jusqu'à présent. L'application de n'importe quel algorithme de prédiction de structure à ces séquences prises individuellement donnera une, voire plusieurs structures. CARNAC ne trouve pas de structure, excepté une tige qui semble relativement bien conservée d'une séquence à l'autre.

Nous avions utilisé au paragraphe 3.2.1 un ensemble de 15 séquences d'ARN messagers codant pour le cytochrome pour montrer que la distribution des fréquences de repliement était déjà une première mesure du fait que les structures partagent ou non une structure commune. Nous avions vu qu'effectivement les tiges obtenues pour les ARNm avaient toutes une très basse fréquence, la meilleure fréquence étant 8 sur 14, si l'on observe bien l'histogramme. Nous

Arabidopsis thaliana	174: 180 / 193: 199	cgacgcc / gguaucg
Candida glabrata	148: 151 / 168: 171	gacg / cguc
Curvularia lunata		
Drosophila melanogaster	157: 162 / 175: 180	gaugcc / ggcguu
Emericella nidulans		
Fusarium oxysporum	156: 162 / 175: 181	cgaugcc / ggcauug
Fritillaria agrestis		
Gallus gallus	151: 156 / 169: 174	gaugcc / gguauc
Helianthus annuus	175: 178 / 194: 197	gcug / cagu
Mus musculus	151: 156 / 169: 174	gaugcc / ggcauc
Neurospora crassa	163: 168 / 181: 186	gaugcc / ggcauc
Rice (Oryza sativa)	174: 181 / 191: 198	uacggcca / uggcugug
Saccharomyces cerevisiae		
Tigriopus californicus	152: 156 / 169: 173	acgcc / ggugu



0.34 0.21 0.17 0.16

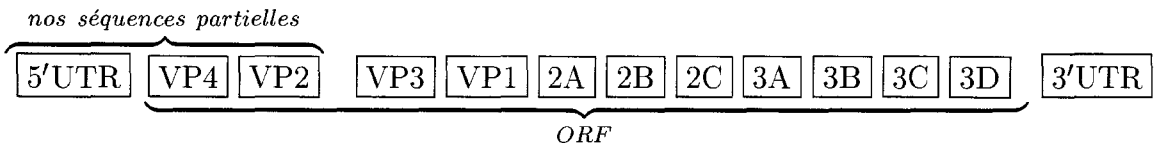
FIG. 4.12 – *ARNm cytochrome – Les quatre meilleurs composantes du graphe avec leur indice. Les traits pleins indiquent un arc de type 'coplié' et les traits pointillés un arc de type 'identité'. Seul le premier graphe passe le seuil de sélection. Le tableau donne l'emplacement dans chaque séquence de la tige commune correspondant au premier graphe.*

avons vu ensuite que l'étude de la connectivité dans les composantes connexes du graphe de corepliements était un critère de sélection plus fin (la fréquence, apparaissant comme l'arité des nœuds dans le graphe, est prise en compte lorsqu'on évalue la qualité des composantes connexes).

Lorsqu'on replie ce même jeu de séquences avec la version n séquences de CARNAC, on ne trouve pas de structure, excepté une tige commune à 10 séquences vers le milieu de la séquence. L'examen de la composante connexe de cette tige montre qu'elle n'est pas si mauvaise (FIG. 4.12). Cela explique que son indice passe le seuil de sélection.

4.6 5′ UTR Enterovirus

Certaines séquences sont partiellement structurées. CARNAC arrive aussi dans ce cas à déterminer quelles sont les parties structurées des séquences considérées. Nous avons constitué un groupe de 12 séquences d'ARNm entérovirus codant pour une polyprotéine. La partie 5′ UTR a été montrée structurée [LZ90] alors qu'on ne s'attend pas à trouver de structure à l'intérieur de l'ORF (partie codante de l'ARNm). Comme les séquences sont un peu longues et qu'il est inutile de les considérer en entier pour l'étude qui nous intéresse, nous les avons toutes sectionnées arbitrairement après le deuxième tronçon codant la protéine, ce qui nous laisse des séquences de longueur environ 1750 nucléotides, le codon START (début de l'ORF) apparaissant autour de la position 700 :



Lors des corepliements deux à deux, $CARNAC_2$ trouve des tiges communes sur toute la longueur des séquences. On observe que la structure est un peu plus éparpillée, et apparait comme moins fiable à l'interieur de l'ORF, sans qu'il soit possible de quantifier ce manque de fiabilité. Après filtrage par l'examen du graphe, CARNAC ne propose de la structure qu'à l'extérieur de l'ORF (FIG. 4.13) : Toutes les tiges prédites apparaissent *avant* le codon START.

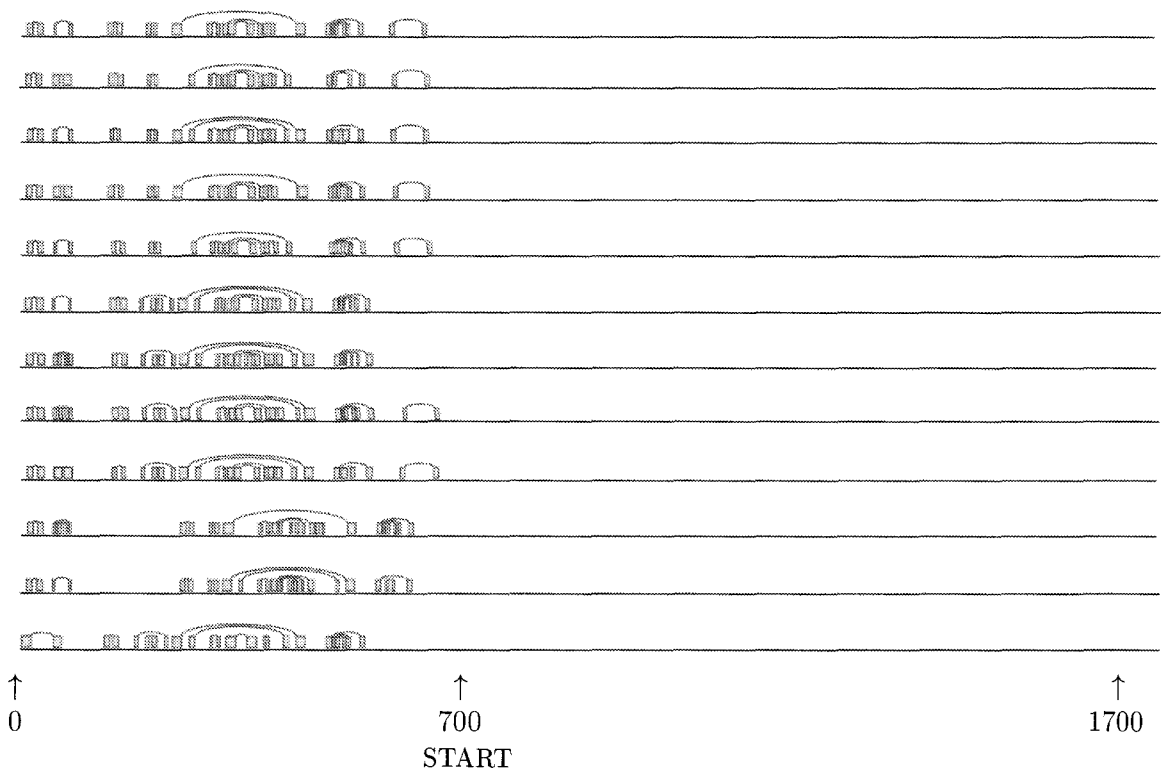


FIG. 4.13 – *ARNm enterovirus – la structure prédite par CARNAC est exclusivement située avant le codon START.*

Conclusion

Avec la découverte de nouvelles familles d'ARN non-codants, l'un des problèmes actuels majeurs est de comprendre les fonctions qu'occupent ces différentes familles au sein de la cellule. Or ces fonctions sont fortement liées à la structure. Les deux approches classiques de prédiction de structure secondaire – la méthode thermodynamique et l'analyse comparative – sont mal adaptées à ces familles souvent peu volumineuses de séquences homologues. Des méthodes hybrides ont donc vu le jour, qui procèdent simultanément des deux approches classiques.

Notre méthode – CARNAC – se place dans ce contexte, avec le parti pris de rechercher une prédiction de qualité, quitte à manquer une partie de la structure.

Nous avons le sentiment qu'un pas significatif a été franchi avec CARNAC dans plusieurs directions. Notre méthode s'applique à des jeux de séquences dont les caractéristiques sont assez variables : la taille des séquences à replier peut être de l'ordre de quelques dizaines de nucléotides jusqu'à quelques milliers, ce qui couvre la majeure partie des ARN. Les ARN peuvent être très divergents en séquence et la structure commune peut avoir subi quelques variations. La méthode ne demande aucun prétraitement sur les séquences : on évite en particulier les difficultés de l'alignement. La méthode permet finalement de détecter si une famille de séquences possède une structure entièrement ou partiellement partagée.

Nous estimons aussi qu'il ne s'agit que d'un *pas* dans ces directions, montrant à la fois les possiblités et les limites de cette approche. Certaines pistes méritent d'être encore approfondies, d'autres montrent les limites inhérentes à la méthode et sont donc intéressantes en ce sens, mais ne méritent pas d'être poussés plus avant selon nous. CARNAC est un algorithme de *compromis*. Sa rapidité et son efficacité proviennent en grande partie de l'approximation qui est faite de replier des tiges entières. En contrepartie, il n'a pas la finesse des méthodes qui se placent au niveau des nucléotides. Suite à ce constat, deux directions franchement différentes nous paraissent engageantes pour le futur. D'un coté, on peut choisir de pousser plus loin la méthode pour l'appliquer dans des contextes un peu différents (des jeux de séquences possédant des caractéristiques plus variées, des problématiques différentes : balayage de génomes par exemple). On peut aussi tenter d'améliorer la fiabilité du programme en munissant les prédictions d'une mesure statistique précise. D'un autre coté, on peut choisir de tirer parti des optimisations réalisées au fur et à mesure sur CARNAC pour revenir à un niveau de précision plus fin, celui des nucléotides.

Les directions futures que pourrait prendre la piste engagée par CARNAC

L'algorithme tel qu'il est implémenté recherche une structure commune *globale*, même s'il en résulte parfois au final des éléments de structure locaux. Si les séquences possèdent un petit motif structural conservé, mais dont la localisation sur la séquence varie fortement, CARNAC ne le détectera pas. En comparaison avec l'alignement classique, on pourrait dire qu'il fonctionne en quelque sorte comme un algorithme d'alignement global, par opposition à l'alignement local. Adapter l'algorithme dans le sens du *local* lui ouvrirait assez naturellement d'autres champs d'application : la balayage de génomes à la recherche d'un motif (connu ou inconnu) par exemple.

L'efficacité pourrait être améliorée en incorporant des informations de contraintes plus larges que celles données par les points d'ancrage sur la structure primaire. Au cours de l'élaboration de CARNAC, les essais pour incorporer des contraintes utilisant les motifs (AA-plateformes, triple hélices, tetra-boucles) ont été infructueux car nous n'en avions pas assez, mais il nous semblerait très interessant de l'envisager à nouveau avec une palette plus large de motifs connus.

La qualité et la fiabilité des prédiction pourraient être elles aussi encore améliorées en dotant les tiges produites par CARNAC d'un critère statistique qui en mesure précisément la pertinence. Si l'on trouve par exemple quelques éléments structuraux dans des séquences homologues, on aimerait qu'ils soient accompagnés d'un indice théorique qui mesure le caractère exceptionnel de leur occurence, l'équivalent d'une *P-value*. À l'heure actuelle, le programme se base sur un indice interne pour donner une prédiction de type *tout ou rien.*

Enfin, dans une autre direction, l'heuristique des tiges était indispensable au départ de nos réflexions pour pouvoir rendre l'algorithme praticable. Au bénéfice des différentes optimisations que nous avons élaborées en cours de route, il nous semblerait maintenant intéressant de revenir au niveau le plus bas, celui des nucléotides, et une récente approche telle que DYNALIGN, proposée par Mathews *et al.* [MT02], combinée avec les optimisations mises en œuvre dans CARNAC, pourrait donner des résultats fort concluants.

Annexes

1 Repères chronologiques

La chronologie que nous donnons ne se veut absolument pas exhaustive, mais elle permet de visualiser d'un coup d'œil la progression simultanée des différentes pistes de recherche concernant la prédiction de structure d'ARN. À l'image des *marqueurs* sur un brin d'ADN, certaines dates clefs concernant l'informatique ou la biologie sont indiquées à titre de repères temporels.

- 1944 : Avery *et al.* – l'ADN est l'agent chimique responsable de l'hérédité.
- 1953 : Watson, Crick – découverte de la structure en double hélice B de l'ADN.
- 1954 : G. Gamow – diamond code (triplets chevauchants).
- 1957 : Crick – idée du codon.
- 1963 : R. Holley – séquençage de l'ARNt^{Ala} de la levure.
- 1969 : M. Levitt [Lev69] – découverte de la structure en feuille de trèfle à l'aide de 14 ARNt (covariations + indices en faveur de la feuille de trèfle).
- 1970 : Needleman, Wunsch – algorithme pour l'alignement de deux séquences.
- 1971 : Tinoco *et al.* – paramètres d'énergie à partir de l'étude de petits ARN.
- 1974 : Borer *et al.* – *nearest neighbor energy model.*
- 1974 : Kim *et al.* – structure spatiale en L de l'ARNt^{Phe} de la levure (diffraction aux rayons X de la molécule cristallisée).
- 1975 : Pipas, McMahon [PM75] – algorithme de prédiction/manipulation de structures (branch & bound au niveau des tiges).
- 1979 : Ninio – affinage des paramètres d'énérgie par apprentissage.
- 1980 : Noller, Woese – structure de l'ARNr 16S par analyse comparative.
- 1980 : Nussinov, Jacobson [NJ80] – $O(n^2)$, $O(n^3)$.
- 1981 : Altman, Cech – découverte des ribozymes (prix nobel chimie en 1989).
- 1981 : Smith, Waterman [SW81] – alignement local de deux séquences.
- 1983 : G. Olsen – automatisation de l'analyse comparative (χ^2).
- 1983 : H. Martinez [Mar84] – algorithme génétique de type Monte Carlo (temps $O(n^2)$).
- 1985 : D. Sankoff [San85] – $O(n^4)$, $O(n^6)$.
- 1986 : Freier *et al.* – affinage des paramètres d'énergie à l'aide d'ARN synthétisés.
- 1989 : MFOLD [Zuk89] – structures sous-optimales.
- 1991 : MC-SYM [MTG$^+$91] – structure stérique par satisfaction de contraintes.

- 1997 : FOLDALIGN [GHS97] – motif commun à quelques séquences.
- 1999 : X2S [JW99] – méthode hybride.
- 2000 : Chen, Le, Maizel [CLM00] – algorithme génétique.

2 Liens web

2.1 Formulaires / Applet

- MFOLD – prédiction de structure secondaire
  http://www.bioinfo.rpi.edu/applications/mfold/
- X2S – prédiction de structure secondaire
  http://tyrant.ucsc.edu/X2s/
- RNA fold – prédiction de structure secondaire
  http://rna.tbi.univie.ac.at/cgi-bin/RNAfold.cgi
- RNA alifold – prédiction de structure commune (séquences alignées)
  http://rna.tbi.univie.ac.at/cgi-bin/alifold.cgi
- paRNAss – prédiction de structures alternatives
  http://bibiserv.techfak.uni-bielefeld.de/cgi-bin/parnass_submit
- SeqGen – Générateur de séquences aléatoires
  http://bioweb.pasteur.fr/seqanal/interfaces/seqgen.html
- PFOLD – prédiction de structure secondaire (SCFG)
  http://www.daimi.au.dk/~compbio/rnafold/
- RNAGA – prédiction de structure secondaire (algo génétique)
  http://www.abcc.ncifcrf.gov/app/htdocs/appdb/drawpage.php?appname=RNAGA
  http://bioweb.pasteur.fr/seqanal/interfaces/rnaga.html
- ReadSeq – conversions de formats
  http://www.infobiogen.fr/services/analyseq/cgi-bin/readseq_in.pl

2.2 Sources / Exécutables

- RnaViz – visualiser des structures secondaires
  http://rrna.uia.ac.be/rnaviz/
- RNAdraw – visualiser des structures secondaires
  http://rnadraw.base8.se/
- RNA Movies – visualiser un chemin cinétique de structures secondaires
  http://BiBiServ.TechFak.Uni-Bielefeld.DE/rnamovies/
- Circles – prédiction de structure (MWM)
  http://taxonomy.zoology.gla.ac.uk/rod/circles/
- Dynalign – prédiction de structure secondaire
  http://rna.chem.rochester.edu/dynalign.html
- Vienna RNA package – prédiction, alignement, etc.
  http://www.tbi.univie.ac.at/~ivo/RNA/
- CARNAC – prédiction de structure
  http://www.lifl.fr/~perrique/rna/

2.3 Banques de données

- ssu/lsu-rRNA – *http://rrna.uia.ac.be/*
- ssu/lsu-rRNA + introns – *http://www.rna.icmb.utexas.edu/*
- 12S ssu-rRNA – *http://taxonomy.zoology.gla.ac.uk/rod/data/rna/*
- tRNA – *http://www.staff.uni-bayreuth.de/~btc914/search/*
- tRNA – *http://medlib.med.utah.edu/RNAmods/trnabase/*
- tRNA – *ftp://ftp.ebi.ac.uk/pub/databases/trna/*
- tRNA – *ftp://ftp.cse.ucsc.edu/pub/rna/*
- RFAM (ncRNA) – *http://rfam.wustl.edu/*
- ncRNA – *http://biobases.ibch.poznan.pl/ncRNA/*
- tmRNA – *http://psyche.uthct.edu/dbs/tmRDB/tmRDB.html*
- uRNA – *http://psyche.uthct.edu/dbs/uRNADB/uRNADB.html*
- SRP – *http://psyche.uthct.edu/dbs/SRPDB/SRPDB.html*
- snoRNA – *http://rna.wustl.edu/snoRNAdb/*
- RNase P – *http://jwbrown.mbio.ncsu.edu/RNaseP/home.html*
- ISIS (introns) – *http://isis.bit.uq.edu.au/description_info.html*
- introns – *http://www.cse.ucsc.edu/~kent/intronerator/idb/cDNA*
- introns – *http://www-db.embl-heidelberg.de/jss/servlet/de.embl.bk.wwwTools.GroupLeftEMBL/ExternalInfo/seraphin/yidb.html*
- viroid – *http://132.210.163.235/subviral/home.cgi*
- picornavirus – *http://www.personal.uni-jena.de/~i6zero/database.html*
- picornavirus – *http://www.iah.bbsrc.ac.uk/virus/Picornaviridae/SequenceDatabase/*
- enterovirus – *http://www.personal.uni-jena.de/~i6zero/entero.html*
- IRES – *http://ifr31w3.toulouse.inserm.fr/IRESdatabase/*
- UTR – *ftp://bighost.area.ba.cnr.it/pub/Embnet/Database/UTR/data/*

Bibliographie

[AGML90] S.F. Altschul, W. Gish, W. Miller, and E.W. Myersand D.J. Lipman. Basic local alignment search tool. *JMB*, pages 215:403–410, 1990.

[BH88] R.E. Bruccoleri and G. Heinrich. An improved algorithm for nucleic acid secondary structure display. *Comput Appl Biosci*, 4:167–173, 1988.

[BMR95] V. Bafna, S. Muthukrishnan, and R. Ravi. Computing similarity between RNA strings. In Z. Galil and E. Ukkonen, editors, *Proceedings of the 6th Annual Symposium on Combinatorial Pattern Matching*, LNCS 937, pages 1–16, Espoo, Finland, 1995. Springer-Verlag, Berlin.

[BNH+00] N. Ban, P. Nissen, J. Hansen, P.B. Moore, and T.A. Steitz. The complete atomic structure of the large ribosomal subunit at 2.4 A resolution. *Science*, 289(5481):905–920, aug 2000.

[Bro99] J.W. Brown. The Ribonuclease P database. *NAR*, 27(314), 1999. `http://www.mbio.ncsu.edu/RNaseP/`.

[BS99] D. Bouthinon and H. Soldano. A new method to predict the consensus secondary structure of a set of unaligned RNA sequences. *Bioinformatics*, 15(10):785–798, 1999.

[BW95] M. Brown and C. Wilson. RNA pseudoknot modeling using intersections of stochastic context free grammars with applications to database search. In L. Hunter and T. Klein, editors, *Pacific Symposium on Biocomputing*, pages 109–125, 1995.

[CK91] D.K.Y. Chiu and T. Kolodziejczak. Inferring consensus structure from nucleic acid sequences. *CABIOS*, 7(3):347–352, 1991.

[CLM00] J.H. Chen, S.Y. Le, and J.V. Maizel. Prediction of common secondary structures of RNAs: a genetic algorithm approach. *NAR*, 28(4):991–999, 2000.

[CM94] F. Corpet and B. Michot. RNAlign program : alignment of RNA sequences using both primary and secondary structures. *CABIOS*, 10(4):389–399, 1994.

[DEKM98] R. Durbin, S. Eddy, A. Krogh, and G. Mitchison. *Biological sequence analysis*, chapter 10 : "RNA structure analysis". Cambridge University Press, 1998.

[DK02] F. Dardel and F. Képès. *Bioinformatique - Génomique et post-génomique.* Ellipses, 2002.

[DT03a] S. Dulucq and L. Tichit. Rna secondary structure comparison: exact analysis of the zhang-shasha tree edit algorithm. *Theoretical Computer Science*, à paraître, 2003.

[DT03b] S. Dulucq and H. Touzet. Analysis of tree edit distance algorithms. In *Combinatorial Pattern Matching: 14th Annual Symposium*, LNCS, pages 83–95. Springer-Verlag, Heidelberg, 2003.

[ED94] S.R. Eddy and R. Durbin. RNA sequence analysis using covariance models. *NAR*, 22:2079–2088, 1994.

[EG99] D. Evers and R. Giegerich. RNA movies: visualizing RNA secondary structure spaces. *Bioinformatics*, 15(1):32–37, 1999.

[EGG88] D. Eppstein, Z. Galil, and R. Giancarlo. Speeding up dynamic programming. In *Proceedings of the29th IEEE Annual Symposium on Foundations of Computer Science*, pages 488–496, White Plains, NY, 1988. IEEE Computer Society Press.

[EGGI91] D. Eppstein, Z. Galil, R. Giancarlo, and G.F. Italiano. Efficient algorithms for sequence analysis. In *SEQS: Sequences '91*, 1991.

[FHMSZ01] C. Flamm, I.L. Hofacker, S. Maurer-Stroh, and P.F. Stadlerand M. Zehl. Design of multi-stable RNA molecules. *RNA*, 7:254–265, 2001.

[FKJ+86] S.M. Freier, R. Kierzek, J.A. Jaeger, N. Sugimoto, M.H. Caruthers, T. Neilson, and D.H. Turner. Improved free-energy parameters for prediction of RNA duplex stability. *PNAS*, 83:9373–9377, 1986.

[FKSS93] W. Fontana, D.A.M. Konings, P.F. Stadler, and P. Schuster. Statistics of RNA secondary structure. *Biopolymers*, 33:1389–1404, 1993.

[FSB+93] W. Fontana, P.F. Stadler, E.G. Bornberg-Bauer, T. Griesmacher, I.L. Hofacker, M. Tacker, P. Tarazona, E.D. Weinberger, and P.Schuster. RNA folding and combinatory landscapes. *Phys. Rev. E*, 47:2083–2099, 1993.

[GHH+94] L. Grate, M. Herbster, R. Hughey, I.S. Mian, H. Noller, and D. Haussler. RNA modeling using Gibbs sampling and stochastic context free grammars. In *Proc. of Second Int. Conf. on Intelligent Systems for Molecular Biology*, pages 138–146, Menlo Park, CA, 1994. AAAI/MIT Press.

[GHR99] R. Giegerich, D. Haase, and M. Rehmsmeier. Prediction and visualization of structural switches in RNA. In *Pacific Symposium on Biocomputing*, pages 126–137, 1999.

[GHS97] J. Gorodkin, L.J. Heyer, and G.D. Stormo. Finding the most significant common sequence and structure motifs in a set of RNA sequences. *NAR*, 25:3724–3732, 1997.

[GJBM+03] S. Griffiths-Jones, A. Bateman, M. Marshall, A. Khanna, and S.R. Eddy. RFAM: an RNA family database. *NAR*, 31(1):439–441, 2003.

[GMC93] D. Gautheret, F. Major, and R. Cedergren. Modeling the three-dimensional structure of RNA using discrete nucleotide conformational sets. *JMB*, 229:1049–1064, 1993.

[Got82] O. Gotoh. An improved algorithm for matching biological sequences. *JMB*, 162:705–708, 1982.

[Gra95] L. Grate. Automatic RNA secondary structure determination with stochastic context-free grammars. In AAAI/MIT Press, editor, *ISMB*, pages 136–144, July 1995.

[GSS01] J. Gorodkin, S.L. Stricklin, and G.D. Stormo. Discovering common stem-loop motifs in unaligned RNA sequences. *NAR*, 29(10):2135–2144, 2001.

[Gul91] A.P. Gultyaev. The computer simulation of RNA folding involving pseudoknot formation. *NAR*, 19:2489–2494, 1991.

[HFS+94] I.L. Hofacker, W. Fontana, P.F. Stadler, L.S. Bonhoeffer, M. Tacker, and P. Schuster. Fast folding and comparison of RNA secondary structures (the vienna RNA package). *Monatshefte für Chemie*, 125:167–188, 1994.

[IS00] H. Isambert and E.D. Siggia. Modeling RNA folding paths with pseudoknots: Application to hepatitis delta virus ribozyme. *PNAS*, 97(12), 2000.

[JTZ89] J.A. Jaeger, D.H. Turner, and M. Zuker. Improved predictions of secondary structures for RNA. *PNAS*, 86:7706–7710, october 1989.

[JW99] V. Juan and C. Wilson. RNA secondary structure prediction based on free energy and phylogenetic analysis. *JMB*, 289:935–947, 1999.

[KCP96] J. Kim, J.R. Cole, and S. Pramanik. Alignment of possible secondary structures in multiple RNA sequences using simulated anneling. *CABIOS*, 12(4):259–267, 1996.

[KH99] B. Knudsen and J. Hein. RNA secondary structure prediction using stochastic context-free grammars and evolutionary history. *Bioinformatics*, 15(6):446–454, 1999.

[LE97] T. Lowe and S.R. Eddy. tRNAscan-SE: a program for improved detection of transfer RNA genes in genomic sequence. *NAR*, 25:955–964, 1997.

[LE99] T.M. Lowe and S.R. Eddy. A computational screen for methylation guide snoRNAs in yeast. *Science*, 238:1168–1171, 1999.

[Lev69] M. Levitt. Detailed molecular model for transfer ribonucleic acid. *Nature*, 224:759–763, 1969.

[LP00] R.B. Lyngsø and C.N. Pedersen. RNA pseudoknot prediction in energy-based models. *Journal of Computational Biology*, 7(3):409–427, Nov 2000.

[LZ90] S.Y. Le and M. Zuker. Common structures of the 5' non-coding RNA in enteroviruses and rhinoviruses. *JMB*, 216:729–741, 1990.

[LZM02] S.Y. Le, K. Zhang, and J.V. Maizel. RNA molecules with structure dependent functions are uniquely folded. *NAR*, 15(30(16)):3574–3582, 2002.

[LZP99] R.B. Lyngsø M. Zuker, and C.N.S. Pedersen. Fast evaluation of internal loops in RNA secondary structure prediction. *Bioinformatics*, 15(6):440–445, 1999.

[Mar84] H.M. Martinez. An RNA folding rule. *NAR*, 12(1):323–334, 1984.

[MDW96] B. Morgenstern, A. Dress, and T. Werner. Multiple dna and protein sequence alignment based on segment-to-segment comparison. *PNAS*, pages 12098–12103, 1996.

[MEG+01] T.J. Macke, D.J. Ecker, R.R. Gutell, D. Gautheret, D.A. Case, and R. Sampath. RNAMotif: an RNA secondary structure definition and search algorithm. *NAR*, 29:4724–4735, 2001.

[Moo99] P.B. Moore. Structural motifs in RNA. *Annu. Rev. Biochem.*, 68:287–300, 1999.

[MSZT99] D.H. Mathews, J. Sabina, M. Zuker, and D.H. Turner. Expanded sequence dependence of thermodynamic parameters improves prediction of RNA secondary structure. *JMB*, 288:911–940, 1999.

[MT02] D.H. Mathews and D.H. Turner. Dynalign: An algorithm for finding the secondary structure common to two RNA sequences. *JMB*, -:in press, 2002.

[MTG+91] F. Major, M. Turcotte, D. Gautheret, G. Lapalme, E. Fillion, and R. Cedergren. The combination of symbolic and numerical computation for three-dimensional modeling of RNA. *Science*, 253:1255–1260, september 1991.

[NJ80] R. Nussinov and A.B. Jacobson. Fast algorithm for predicting the secondary structure of single stranded RNA. *PNAS*, 77:6309–6313, 1980.

[NS83] A.S. Netzel and S.M. Selkow. *Time warps, string edits, and macromolecules: the theory and practice of sequence comparison*, chapter 8: "An analysis of the general tree-editing problem", pages 237–252. Addison-Wesley, 1983.

[PM75] J.M. Pipas and J. McMahon. Method for predicting RNA scondary structure. *PNAS*, 72(6):2017–2021, 1975.

[PTD03] O. Perriquet, H. Touzet, and M. Dauchet. Finding the common structure shared by two homologous RNAs. *Bioinformatics*, 19:108–116, 2003.

[RE99] E. Rivas and S.R. Eddy. A dynamic programming algorithm for RNA structure prediction including pseudoknots. *JMB*, 285:2053–2068, 1999.

[Rij95] P. De Rijk. *Optimisation of a database for ribosomal RNA structure and application in structural and evolutionary research.* PhD thesis, Universiteit Antwerpen, 1995.

[San85] D. Sankoff. Simultaneous solution of the RNA folding, alignment and protosequence problems. *SIAM J. Appl. Math.*, 45(5):810–825, 1985.

[SFSH94] P. Schuster, W. Fontana, P.F. Stadler, and I.L. Hofacker. From sequences to shapes and back: A case study in RNA secondary structures. *Proc.Roy.Soc.Lond.*, 255:279–284, 1994.

[SH99] P.F. Stadler and C. Haslinger. RNA structures with pseudo-knots: Graph-theoretical and combinatorial properties. *Bull. Math. Biol.*, 61:437–467, 1999.

[SM97] J. Setubal and J. Meidanis. *Introduction to computational biology*, chapter 8: "Molecular structure prediction". International Thomson Publishing Company, 1997.

[SRCS78] G. Studnicka, G. Rahn, I. Cummings, and W. Salser. Computer method for predicting the secondary structure of single-stranded RNA. *NAR*, 5(9):3365–3387, 1978.

[SS99] P. Schuster and P.F. Stadler. Discrete models of biopolymers. *TBI preprint*, december 1999.

[SW81] T.F. Smith and M.S. Waterman. Identification of common molecular subsequences. *JMB*, 147:195–197, 1981.

[SW94] W.R. Schmitt and M.S. Waterman. Linear trees and RNA secondary structure. *Discrete Applied Mathematics*, 51:317–323, 1994.

[TB99] I. Tinoco and C. Bustamante. How RNA folds. *JMB*, 293:271–281, 1999.

[TCGS98] J.E. Tabaska, R.B. Cary, H.N. Gabow, and G.D. Stormo. An RNA folding method capable of identifying pseudoknots and base triples. *Bioinformatics*, 14(8):691–699, 1998.

[THG94] J.D. Thompson, D.G. Higgins, and T.J. Gibson. CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, positions-specific gap penalties and weight matrix choice. *NAR*, 22:4673–4680, 1994.

[Tou03] H. Touzet. Tree edit distance with gaps. *Information Processing Letters*, 85(3):123–129, 2003.

[TSB+96] M. Tacker, P.F. Stadler, E.G. Bornberg-Bauer, I.L. Hofacker, and P. Schuster. Algorithm independent properties of RNA secondary structure predictions. *Eur. Biophy. J.*, 25:115–130, 1996.

[WA00] E. Westhof and P. Auffinger. *RNA tertiary structure, Encyclopedia of Analytical Chemistry*, pages 5222–5232. John Wiley & Sons Ltd, Chichester, 2000.

[Wat95] M.S. Waterman. *Introduction to Computational Biology*, chapter 13: "RNA secondary structure", pages 327–343. Chapman & Hall, 1995.

[WBC+00] B.T. Wimberly, D.E. Brodersen, W.M. Clemons, R.J. Morgan-Warren, A.P. Carter, C. Vonrhein, T. Hartsch, and V. Ramakrishnan. Structure of the 30S ribosomal subunit. *Nature*, 407(6802):327–339, sep 2000.

[WJ98] L. Wuju and W. JiaJin. Prediction of RNA secondary structure based on helical regions distribution. *Bioinformatics*, 14(8):700–706, 1998.

[WK99] C. Workman and A. Krogh. No evidence that mRNAs have lower folding free energies than random sequences with the same dinucleotide distribution. *NAR*, 27(24), 1999.

[WMG+80] C.R. Woese, L.J. Magrum, R. Gupta, R.B. Siegel, D.A. Stahl, J. Kop, N. Crawford, J. Brosius, R. Gutell, J.J. Hogan, and H.F. Noller. Secondary structure model for bacterial 16S ribosomal RNA: phylogenetic, enzymatic and chemical evidence. *NAR*, 24:8(10):2275–2293, 1980. `http://www.rna.icmb.utexas.edu/`.

[WS86] M.S. Waterman and T.F. Smith. Rapid dynamic programming methods for RNA secondary structure. *Advances in Applied Mathematics*, 7:455–464, 1986.

[WT98] M. Wu and I. Tinoco. RNA folding causes secondary structure rearrangement. *PNAS*, 95:11555–11560, 1998.

[WTK+94] A.E. Walter, D.H. Turner, J. Kim, M.H. Lyttle, P. Müller, D.H. Mathews, and M. Zuker. Coaxial stacking of helixes enhances binding of oligoribonucleotides and improves predictions of RNA folding. *PNAS*, 91:9218–9222, september 1994.

[WZ99] Z. Wang and K. Zhang. Finding common RNA secondary structures from RNA sequences. In *CPM'99 Lecture Notes in Computer Science*, volume 1645, pages 258–269, 1999.

[Zuk89] M. Zuker. On finding all suboptimal foldings of an RNA molecule. *Science*, 244:48–52, 1989.

[Zuk03] M. Zuker. MFOLD web server for nucleic acid folding and hybridization prediction. *NAR*, 31(13):1–10, 2003.

[ZWM99] K. Zhang, L. Wang, and B. Ma. Computing similarity between RNA structures. In *CPM*, volume 1645 of *Lecture Notes in Computer Science*, pages 281–293. Springer, 1999.

Index