Algorithmes multi-critères pour la prédiction de structures d'ARN

Louis Becquey

▶ **To cite this version:**

Louis Becquey. Algorithmes multi-critères pour la prédiction de structures d'ARN. Bio-informatique [q-bio.QM]. Université Paris-Saclay, 2021. Français. NNT : 2021UPASG065 . tel-03440181

HAL Id: tel-03440181
https://theses.hal.science/tel-03440181

Submitted on 22 Nov 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

université PARIS-SACLAY

Algorithmes multicritères pour la prédiction de structures d'ARN

Thèse de doctorat de l'université Paris-Saclay

École doctorale n° 580, Sciences et Technologies de l'Information et de la Communication (STIC)
Spécialité de doctorat : Informatique
Unité de recherche : Université Paris-Saclay, Univ Evry, IBISC, 91020, Évry-Courcouronnes, France
Référent : Université d'Évry Val d'Essonne

Thèse présentée et soutenue à Évry, le 6/10/2021, par

Louis Becquey

Composition du Jury

Alain DENISE Professeur, Université Paris-Saclay, LISN	Président
François MAJOR Professeur titulaire, Université de Montréal, IRIC	Rapporteur & Examinateur
Jérôme WALDISPÜHL Professeur associé, Université McGill	Rapporteur & Examinateur
Marie-Dominique DEVIGNES Chargée de recherches, CNRS, LORIA	Examinatrice
Samuela PASQUALI Professeur, Université Paris-Descartes, CiTCoM	Examinatrice
Fariza TAHI Professeur, Université Paris-Saclay, IBISC	Directrice de thèse
Eric ANGEL Professeur, Université Paris-Saclay, IBISC	Co-Directeur de thèse

Thèse de doctorat

NNT : 2021UPASG065

Table des matières

Abréviations et Anglicismes

AA	*All-Atom*	(Avec tous les atomes)
ADN	Acide Désoxy-ribo-Nucléique	
ARN	Acide Ribo-Nucléique	
BGSU	*Bowling-Green State University*	(Université d'état de Bowling-Green)
CASP	*Critical Assessment of Structure Predictions*	(Évaluation critique des prédictions de structure)
CG	*Coarse-Grained*	(Gros-grains, brut, peu détaillé)
CHIM	*Convex Hull of Individual Minima*	(Enveloppe convexe des minima individuels)
CM	*Covariance Model*	(Modèle des covariations)
CNN	*Convolutional Neural Network*	(Réseau de neurones à convolutions)
CPU	*Core Processing Unit*	(Processeur standard, par opposition au GPU)
DCA	*Direct Coupling Analysis*	(Analyse des couplages directs)
EM	*Electron Microscopy*	(Microscopie électronique)
GA	*Genetic Algorithm*	(Algorithme génétique)
GMM	*Gaussian Mixture Model*	(Modèle Mixte Gaussien)
GPU	*Graphics Processing Unit*	(Carte graphique, processeur fortement parallèle)
HL	*Hairpin Loop*	(Boucle en épingle à cheveux)
HMM	*Hidden Markov Model*	(Modèle de Markov caché)
IL	*Internal Loop*	(Boucle interne)
ILP	*Integer Linear Programming*	(Programmation linéaire en nombres entiers)
KB	*Knowledge-Based*	(Basé sur des connaissances [ou des données])
LSTM	*Long Short-Term Memory*	(Grande mémoire à court terme)
LSU	*Large Sub-Unit*	(Grande sous-unité [du ribosome])
MAE	*Mean Absolute Error*	(Moyenne des valeurs absolues des erreurs)
MC	Monte-Carlo ou méthode de Monte-Carlo	
MD	*Molecular Dynamics*	(Dynamique moléculaire)
Mg^{2+}	Cation Magnésium	
ML	*Multiple Loop* ou *Multi-Loop*	(Boucle à la jonction de multiples hélices)
MSE	*Mean Squared Error*	(Moyenne des erreurs au carré)
NaN	*Not a Number*	(Pas-un-vrai-nombre)
NBI	*Normal Boundary Intersection*	(Intersection de frontière normale)
NCM	*Nucleic Cyclic Motif*	(Motif cyclique d'acide nucléique)
NDB	*Nucleic Acid Database*	(Base de données d'acides nucléiques [31,65])
NERF	*Natural Extension Reference Frame*	(Extension naturelle du cadre de référence)
PDB	*Protein Data Bank*	(Banque de données de protéines [32])
PK	*Pseudo Knot*	(Pseudonoeud)
PSSM	*Position-Specific Scoring Matrix*	(Matrice de scores spécifiques des positions)
QM	*Quantum Mechanics*	(Mécanique Quantique)
REMD	*Replica-Exchange MD*	(Dynamique moléculaire avec échange de réplicas)
RIN	*Recurrent Interaction Network*	(Réseau d'interactions récurrent)
RGN	*Recurrent Geometric Network*	(Réseau géométrique récurrent)
RMN	Résonnance Magnétique Nucléaire	

RMSD	*Root of Mean Squared Deviation*	(Racine de la moyenne des *distances au carré*)
RNN	*Recurrent Neural Network*	(Réseau de neurones récurrent)
SCFG	*Stochastic Context-Free Grammar*	(Grammaire sans contexte stochastique)
SRP	*Signal Recognition Particle*	(ARN de reconnaissance du signal)
SSE	*Secondary Structure Element*	(Elément de structure secondaire)
SSU	*Small Sub-Unit*	(Petite sous-unité [du ribosome])
XRC	*X-Ray Cristallography*	(Cristallographie aux rayons X)

Les codes à 4 caractères donnés en référence des structures moléculaires 3D utilisées à des fins d'illustration ou de benchmark correspondent à leur identifiant dans la Protein Data Bank [32] (https://rcsb.org/).

Liste des figures

Liste des tables

CHAPITRE I : Introduction à la modélisation des structures d'ARN

Ce chapitre présente l'ARN au lecteur néophyte, et récapitule les modèles et outils développés pour modéliser les structures d'ARN dans les dernières décennies. Le lecteur confirmé pourra l'ignorer. La première partie présente la structure et les fonctions communes de l'ARN ainsi que les méthodes expérimentales d'étude de sa structure. La seconde présente les échelles de modélisation et les principaux modèles de motifs d'ARN. La suivante introduit la notion de paysage énergétique, les ensembles de Boltzmann, et les estimateurs de la structure secondaire. La dernière partie présente les grandes familles de méthode d'étude et de prédiction de la structure de l'ARN, et propose un historique commenté de l'état de l'art.

I.1 Qu'est-ce que l'ARN ?

Les Acides Ribo-Nucléiques ou ARN sont une classe de biomolécules de grandes tailles présentes chez toutes les espèces de l'arbre du vivant. Leur origine est très ancienne et considérée antérieure à l'ancêtre commun à toutes les espèces actuellement découvertes. Certains leur donnent même un rôle fondamental dans l'apparition de la vie, décrivant un "monde à ARN". Dans cette hypothèse, des molécules d'ARN autocatalytiques et capables de réplication associées à des membranes lipidiques isolant les compartiments intérieur et extérieur auraient pu être les premières formes de cellules, bien avant l'apparition des protéines. Cette hypothèse s'appuie sur les propriétés remarquables de l'ARN, capable à la fois d'être un support d'information (porteur d'une *séquence*) et un catalyseur chimique (capable d'accélérer des réactions chimiques spontanées).

I.1.1 Structure chimique de la molécule

Chimiquement, l'ARN est un polymère de 4 *résidus* majoritaires, l'adénosine (notée A), la cytidine (C), la guanosine (G) et l'uridine (U). Les trois premières sont également trouvées dans le polymère cousin, l'ADN (Acide Désoxy-ribo-Nucléique), où le U est remplacé par un T (thymidine). Il en existe d'autres, moins fréquents et souvent considérés comme des modifications des principaux : 5-méthylcytosine, diméthylguanosine, inosine, pseudo-uridine…

Chacun de ces résidus, appelés nucléotides, est composé de trois parties chimiques distinctes, illustrées sur la Figure 1.1A :

- une base azotée différente selon le nucléotide (adénine, cytosine, guanine, ou uracile)

composée d'un (C, U, les pyrimidines) ou deux (A, G les purines) cycles aromatiques (voir illustration sur la Figure 1.1B),

- un ribose (sucre à 5 carbones) sous sa forme cyclique furane,
- un ou plusieurs groupements phosphate (PO_4), chargés négativement. Une fois polymérisé, le nucléotide ne possède plus qu'un seul phosphate. Cependant, les formes à deux ou trois phosphates existent aussi dans les cellules et ont d'autres rôles (ADP, ATP).

Par convention, les atomes de la base sont numérotés d'abord, et ceux du ribose ensuite avec des nombres portant une prime (1', 2', ...). Ainsi, le ribose porte une fonction alcool sur ses carbones C2', C3' et C5' (celles des carbones 1' et 4' ayant réagi entre elles pour former le cycle furane). L'alcool du carbone 5' réagit avec le phosphate et le fixe. La base azotée est fixée sur le carbone C1' du ribose.

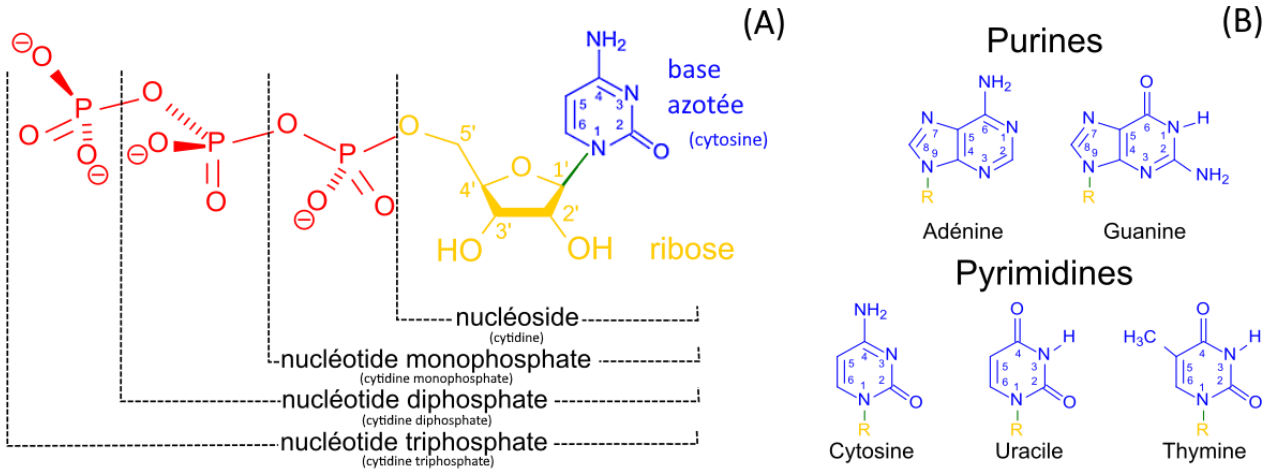


Figure 1.1 – **Plan du nucléotide.** *(A) Un nucléotide complet avec trois phosphates, ici, le CTP. (B) les formules chimiques des principales bases azotées rencontrées dans l'ADN et l'ARN. On remarque que la cytosine C et la guanine G possèdent 3 atomes accepteurs ou donneurs de liaisons hydrogène alignés sur le côté opposé au ribose, quand les autres n'en possèdent que 2.*
Figure adaptée d'après Wikipédia, domaine public (https://fr.wikipedia.org/wiki/Nucl%C3%A9otide)

Ensuite, les nucléotides peuvent être polymérisés en faisant réagir l'alcool du carbone 3' avec le phosphate d'un nucléotide voisin. La liaison ainsi obtenue entre deux nucléotides s'appelle liaison phosphodiester (équivalente de la liaison peptidique chez les protéines). La molécule obtenue devient donc une séquence de deux nucléotides, qu'on note en combinant les deux lettres, par exemple "AG". En répétant l'opération, on peut obtenir des séquences plus longues (ACGACGGCUU...)

Ainsi, les séquences de lettres utilisées pour écrire rapidement la formule d'une molécule d'ARN sont toujours orientées. La convention en biologie moléculaire veut qu'on note toujours les séquences dans le sens dit 5'vers 3', c'est-à-dire que le début correspond à l'extrémité 5'-phosphate, et la fin à un alcool 3'-OH, comme illustré sur la Figure 1.2.

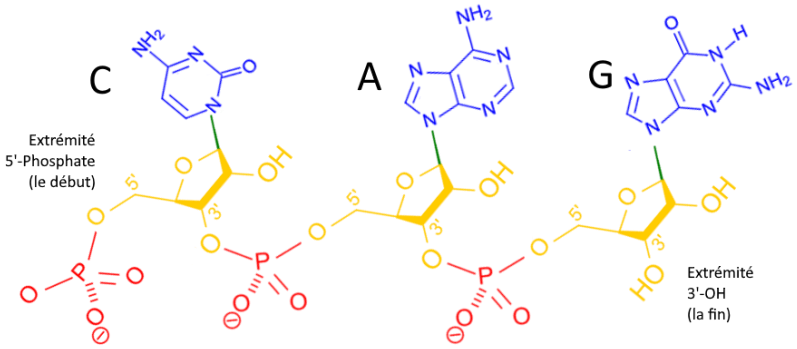


Figure 1.2 – **Polynucléotide.** *Détail moléculaire d'un fragment d'ARN de séquence "CAG" orienté dans le sens standard 5' → 3' (par opposition à "GAC").*

On note que l'alcool restant sur le carbone C2' différencie le ribonucléotide du désoxyribonucléotide de l'ADN, qui a perdu cette fonction alcool.

I.1.2 Empilements, appariements et complémentarité de bases

Les nucléotides sont reliés chimiquement par des liaisons covalentes à leurs voisins, ce qui forme le "squelette" de la molécule. Cependant, d'autres "liaisons", ou plutôt "interactions" existent entre nucléotides, qui vont permettre de structurer la molécule au-delà de sa forme linéaire.

La première source de stabilisation de la molécule est l'empilement (*stacking* en anglais) des nucléotides, consécutifs ou distants. Cette force stabilisante a tendance à maintenir deux cycles aromatiques parallèles et bien superposés à une distance de 3 à 4 Angströms (1 Angström = 10^{-10} mètre). Elle est due à la superposition des orbitales *p* des carbones du cycle aromatique avec celles du cycle voisin. Les différentes formes fréquentes sont illustrées sur la Figure 1.3. En chimie, on parle de *π-π stacking*. Un nucléotide peut donc être empilé avec ses deux voisins dans la séquence, ce qui les maintient bien parallèles entre eux.

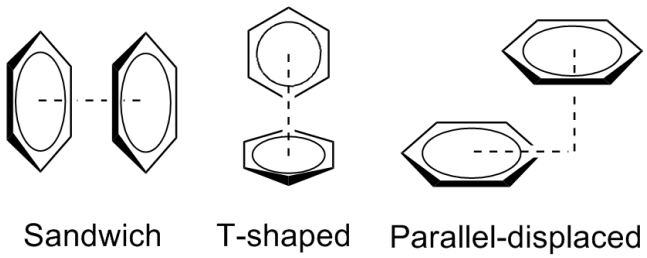


Figure 1.3 – **Empilements de cycles.** *Interactions possibles entre cycles aromatiques par π-π stacking. Du fait du reste des atomes de la base azotée qui prennent de la place autour du cycle aromatique, on observe peu de stacking en forme de T dans l'ARN.*
Source de la figure sous licence CC BY-SA : Wikipedia (https://wikipedia.org/wiki/Pi-Stacking_(chemistry))

La seconde source de stabilisation pour une structure d'ARN est l'interaction directe de nucléotides distants dans la séquence par des liaisons hydrogènes. On appelle cette interaction "appariement", et on dit qu'un nucléotide est apparié à un autre, ou qu'il forme un *basepair* en anglais. En particulier, la complémentarité géométrique de placement des atomes donneurs et accepteurs de liaisons hydrogènes entre les pyrimidines et purines permet de définir des appariements dits "canoniques" très stables. D'une part, A s'apparie avec U en formant deux liaisons hydrogènes, alignées dans le plan avec les deux bases azotées. Et d'autre part, G s'apparie avec C avec trois liaisons hydrogènes elles aussi dans le plan, l'interaction la plus robuste possible entre nucléotides. Il existe aussi un appariement possible entre G et U appelé *wobble* et considéré comme canonique (2 liaisons H). D'autres appariements sont possibles mais moins stables, les appariements « non canoniques ».

Lorsque la séquence le permet, les empilements et appariements organisent la molécule dans l'espace sous forme de double-hélice, similaire à celle observée pour l'ADN (voir Figure 1.4). Les hélices sont assez rigides, leur structure est très peu variable et la position des atomes dans l'espace est quasiment déterminée. Ceci est rendu possible par une propriété fondamentale des trois appariements canoniques : ils sont isostériques, c'est-à-dire que les trois occupent exactement le même volume dans l'espace. Ils sont donc interchangeables : la forme de l'hélice ne changera pas. Évolutivement, les séquences d'ARN présentent donc une différence fondamentale avec les protéines : un nucléotide peut souvent muter sans entraîner aucune conséquence sur la fonction de l'ARN si le nucléotide apparié à celui-ci est également muté pour maintenir l'appariement. Il est donc nécessaire, dans tout calcul d'homologie entre séquences d'ARN, de prendre en compte la localisation des appariements possibles entre nucléotides et pas seulement la séquence. Informatiquement, il est possible de générer les coordonnées 3D des atomes d'une hélice d'ARN de séquence donnée avec une bonne précision simplement à partir de la séquence. La difficulté de prédiction de la structure d'une molécule d'ARN réside surtout dans les régions qui ne forment pas d'hélices.

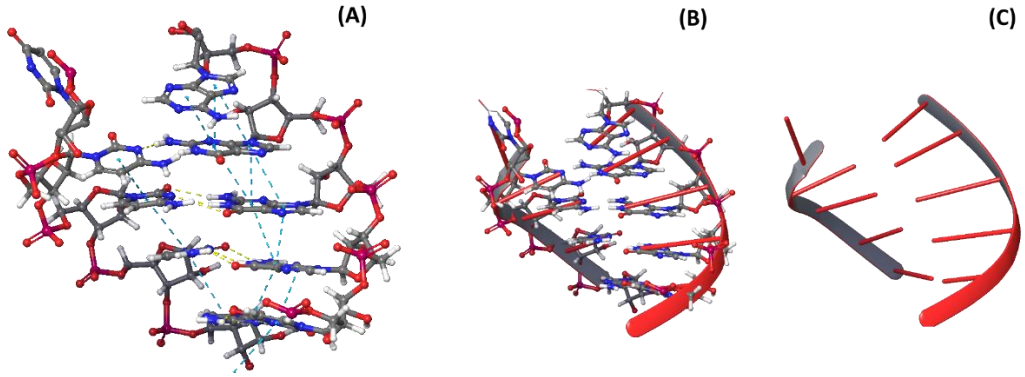


Figure 1.4 – **Le modèle en double-hélice**. ***(A)*** *Modèle atomique en 3D et forces stabilisant une double hélice d'ARN (ici une petite portion d'hélice extraite d'un intron du groupe I, ref. 4P8Z chaine A). Les forces d'empilement sont représentées en bleu et les forces des liaisons hydrogène des appariements en jaune.* ***(B)*** *La même portion, si on superpose aux atomes le modèle d'hélice.* ***(C)*** *L'hélice obtenue, seule.*

I.1.3 Rôles de l'ARN

Rôle de support et vecteur d'information : Comme on vient de le voir, pour un polymère de taille N nucléotides, chaque position peut prendre 4 valeurs différentes (A, C, G, U). Ceci permet d'encoder une grande quantité d'informations (4^N) dans un espace très réduit, quantité d'ailleurs supérieure aux valeurs possibles (2^N) que l'on peut encoder avec N bits informatiques (0 ou 1) tels qu'utilisés dans nos systèmes électroniques.
Ainsi, les eucaryotes utilisent par exemple des ARN pour transporter l'information du code génétique présente dans l'ADN de leur noyau en dehors de celui-ci. On appelle ces ARN des ARN messagers (ARNm). Ce sont eux que la machinerie cellulaire (le ribosome) va décoder pour fabriquer les protéines de la cellule selon le code fourni par le génome.

Rôle de composant structural : D'autres ARN, dits "non-codants", sont toutefois indispensables à la cellule. Ils entrent dans la composition structurale des machines moléculaires. C'est le cas du ribosome, composé de deux à quatre molécules d'ARN.

Rôle de transmetteur du signal (riboswitches, silencers) : D'autres ARN exploitent leur séquence pour reconnaitre d'autres ARN par complémentarité des bases. C'est le cas par exemple des ARN "silencers" (ARNsi) qui vont reconnaitre des ARNm présents dans la cellule et favoriser leur destruction. Leur présence inhibe donc l'expression du gène porté par l'ARNm en question.
D'autres ARN encore, par leur structure, sont sensibles à certains paramètres physico-chimiques (température, force ionique, présence d'un neurotransmetteur ou d'une hormone, etc). On les observe adopter une forme différente en présence et en l'absence de leur molécule cible, ils peuvent être considérés comme des "détecteurs".
Le lien entre la séquence d'un ARN et la forme qu'il prendra dans l'espace est la question clé au centre de cette thèse et de nombreux travaux de recherche de ces 50 dernières années. L'exemple des riboswitches montre que l'environnement est également fortement responsable de la forme adoptée par une molécule d'ARN.

Rôle de catalyseur (ribozyme) : Certains ARN ont la particularité de catalyser des réactions chimiques. Par exemple, la peptidyltransférase présente au sein du ribosome, qui catalyse la réaction d'élongation des chaines protéiques, en transférant un acide aminé d'un ARN de transfert à la chaine protéique en cours de synthèse. D'autres exemples d'activités enzymatiques portées par des ARN sont présentées par exemple dans cette étude de 2019 [356].

I.1.4 Intérêt pour l'industrie biotechnologique et pharmaceutique

De par ses propriétés multiples, l'ARN intéresse les chercheurs. Du point de vue de la séquence, la recherche en biologie de synthèse s'est plutôt concentrée sur la conception rationnelle de séquences d'ADN pour aboutir à des protéines personnalisées ou plus efficaces. Le choix de l'ADN est logique puisqu'il se transmet aux descendants de l'organisme génétiquement modifié. L'ARN n'est alors qu'un intermédiaire. Le principal exemple

d'ingénierie directement liée à la séquence de l'ARN est la récente mise sur le marché des vaccins anti-SarsCov2. La séquence injectée est codante pour une protéine antigénique du virus, ce qui permet de développer l'immunité du sujet sans avoir à réellement utiliser des protéines de virus.

L'objectif premier de la bio-ingénierie de l'ARN est plutôt de comprendre le lien entre séquence et structure, comme ce fut le cas pour les protéines. Si ce lien était maitrisé par les humains, l'ingénierie moléculaire permettrait de concevoir des machines moléculaires de synthèse ou des détecteurs moléculaires spécifiques activés par une espèce chimique précise. Le tout serait bien entendu entièrement biologique, biodégradable, renouvelable, et très simple à synthétiser (chimiquement) ou faire synthétiser (par une cellule vivante). Un problème largement étudié actuellement est le problème du design : *quelle séquence d'ARN dois-je synthétiser pour obtenir la forme X ?* De nombreux exemples de tentatives d'ingénierie de structures d'ARN ont déjà été répertoriées, parfois avec succès [369].

Enfin, l'ARN peut être vu comme une cible thérapeutique de médicaments lorsqu'il n'est pas possible d'atteindre une protéine d'intérêt. On peut alors décider de cibler l'ARNm codant pour cette protéine d'intérêt [71], ou encore développer un ARN synthétique capable de le neutraliser, lui ou sa protéine correspondante.

I.1.5 Méthodes expérimentales de détermination des structures d'ARN

Il existe de nombreux protocoles expérimentaux pour déterminer la structure d'un ARN. Ces méthodes sont considérées pour l'instant comme la référence, les outils numériques ne proposant que des "prédictions", considérées à priori comme moins fiables que les structures expérimentales. Cependant, la résolution d'une structure, même à l'aide de données expérimentales, utilise des outils de résolution *in-silico*. Tout modèle de structure est donc le produit d'une résolution par un programme d'un problème d'optimisation, pour obtenir une solution de structure fortement compatible avec la donnée expérimentale.

I.1.5.1 Cristallographie aux rayons X

La méthode la plus célèbre, aussi fortement utilisée pour résoudre d'autres structures moléculaires (protéines, ADN, sucres, matériaux synthétiques...), est la cristallographie aux rayons X. Elle permet d'obtenir la structure 3D d'un système avec une résolution approchant les 10^{-10}m, soit la largeur d'un atome d'hydrogène. En revanche, il faut identifier des conditions expérimentales (température, salinité, acidité...) qui permettent au système étudié de former un cristal. Cela demande aussi d'avoir pu produire le système à très grande concentration et très grande pureté, pour le faire cristalliser. Il n'y a jamais de garantie qu'il est possible de cristalliser un système donné, ni de méthode prédéfinie pour trouver les bonnes conditions de cristallisation. L'obtention d'un cristal de qualité est donc difficile et souvent longue et couteuse. Une fois le cristal obtenu, on le soumet à un faisceau de rayons X dans un synchrotron, et on obtient une figure de diffraction sur un écran. La figure de

diffraction permet la résolution mathématique de la carte de densité électronique du système. Si le cristal est très régulier, la carte de densité aura une bonne résolution, et il sera possible de replacer précisément les atomes dans l'espace. Enfin, la conformation 3D obtenue des molécules du système doit être considérée comme la conformation cristallisée : rien ne garantit que la molécule ait la même forme quand elle est seule, libre, dans la cellule, dans des conditions différentes. De plus, c'est une structure figée : on n'a pas d'information sur la rigidité, la flexibilité, ou la dynamique de la structure.

I.1.5.2 Résonnance magnétique nucléaire

Une autre méthode populaire est l'étude des structures moléculaires par Résonnance Magnétique Nucléaire (RMN en français, NMR en anglais). Ce protocole utilise la propriété qu'ont les noyaux de certains atomes à être influencés lorsqu'ils sont plongés dans un champ magnétique. Ces noyaux ont un moment magnétique, orienté au hasard à l'origine. L'application d'un champ magnétique extérieur va perturber ces moments magnétiques, qui vont se mettre à pivoter sur eux même jusqu'à s'aligner avec le champ magnétique externe (phénomène de précession de Larmor). La vitesse angulaire de cette précession du moment magnétique (fréquence de Larmor) est caractéristique de l'atome. Elle est cependant modifiée par les interactions intra et inter moléculaires, comme les liaisons covalentes ou simplement la distance dans l'espace. Ainsi, il est possible de repérer l'influence de la rotation d'un moment magnétique sur ses voisins, en mesurant une grandeur appelée déplacement chimique. L'interprétation des données obtenues permet de deviner la présence de noyaux d'atomes connus à certaines positions de l'espace à certaines distances les uns des autres et distanciés par un certain nombre de liaisons covalentes, ce qui permet la reconstruction de la structure générale en haute résolution. En pratique, les structures résolues par RMN ne dépassent pas quelques centaines d'atomes en taille. Au-delà, les signaux se superposent tellement qu'il devient impossible de les distinguer les uns des autres et de reconstituer la structure.

I.1.5.3 Cryo-microscopie électronique

Une méthode plus récente, qui a valu le prix Nobel de chimie en 2017 à ses inventeurs, est la cryo-microscopie électronique (ou cryo-électromicroscopie par anglicisme). Elle permet la détermination de très grands complexes de macromolécules de plusieurs dizaines à centaines de milliers d'atomes, à des résolutions historiquement plutôt mauvaises (environ une dizaine d'angströms = un nanomètre) mais s'améliorant de plus en plus et rivalisant maintenant avec la cristallographie aux rayons X (avec un record à 0.6 Angströms en 2019 [387]). Le principe est proche de la microscopie électronique à transmission, c'est-à-dire qu'on soumet l'échantillon à un faisceau d'électrons (à la place du faisceau de photons utilisé en microscopie optique). Cependant, ce faisceau d'électrons devrait détruire instantanément l'échantillon biologique. L'astuce consiste à fixer par une congélation brutale le système à étudier, qui se retrouve piégé dans sa conformation native au sein d'une fine couche de glace vitreuse. On peut ainsi étudier au microscope électronique un ribosome ou un virus entier sans les abimer, et dans leur conformation native. Cette méthode est

prometteuse pour la détermination de structures d'ARN en particulier [83], souvent très difficiles à cristalliser dans leur forme native. Depuis que la méthode a atteint une maturité suffisante, on s'attend à une augmentation de la qualité et de la quantité de structures 3D d'ARN disponibles publiquement.

I.1.5.4 Transfert d'énergie par résonnance de Fröster

La structure et la dynamique des ARN courts peuvent être étudiées par FRET (*Fröster Resonnance Energy Transfer*) [179,323]. Très grossièrement, la technique consiste à décorer deux nucléotides d'intérêt par des fluorophores dont l'un inhibe la fluorescence de l'autre. La mesure de la fluorescence obtenue communique donc une information sur la distance entre les deux nucléotides lorsque l'ARN est replié. En suivant l'intensité de fluorescence dans le temps, on peut étudier la stabilité de la distance en question, ou observer le repliement d'un ARN. La méthode reste cependant à très basse résolution, elle peut servir à déterminer des structures secondaires ou la topologie générale d'une chaine. Elle nécessite parfois de réaliser plusieurs expériences pour mesurer plusieurs distances, et elle néglige l'effet que peuvent avoir les fluorophores sur la conformation de l'ARN, qu'ils perturbent pourtant probablement.

I.1.5.5 Sondage chimique

Les protocoles de sondage chimique cherchent à étudier la réactivité individuelle de chaque nucléotide. La réactivité d'un nucléotide renseigne sur sa disponibilité au niveau du solvant, à l'extérieur et en surface de la molécule. Un nucléotide très interne ou pris dans un appariement sera très peu réactif. On met l'ARN au contact d'un réactif, dont l'agent actif va venir se fixer sur les nucléotides les plus externes, flexibles, accessibles. Cette fixation est permanente, souvent par liaison covalente. On lave ensuite l'ARN, puis on ajoute une transcriptase inverse (enzyme qui rétro-transcrit l'ARN en ADN). L'enzyme va buter sur les nucléotides modifiés par le réactif, qu'elle n'est pas capable de rétro-transcrire, et elle va se décrocher du brin d'ARN : la séquence d'ADN complémentaire (ADNc) nouvellement synthétisée sera donc tronquée et incomplète. En séquençant tous les ADNc obtenus dans un échantillon, on obtient un ensemble de séquences tronquées, qui renseigne sur la position des nucléotides « accessibles » (non appariés), qualitativement mais aussi quantitativement. L'interprétation des données brutes diffère selon les protocoles, c'est pourquoi il est souvent nécessaire de normaliser les données de sondage chimique [359]. Les données peuvent ensuite être utilisées comme contraintes, guides ou pseudo-énergies dans les modèles computationnels aboutissant à une prédiction de structure [38,90,205,319]. Parmi les protocoles, on connait le sondage au DMS (Di-Methyle Sulfate), qui ajoute un groupement méthyle (CH_3) aux bases A et C non appariées [338], les empêchant d'être reconnues par la transcriptase inverse qui se décroche. Une variante plus efficace (le protocole DMS-MaPSeq, pour *Mutational Profiling with Sequencing*) utilise une transcriptase particulière qui donne une erreur de transcription au lieu de se décrocher [393]. Chaque ADNc obtenu contient donc beaucoup plus d'informations, sur toute la longueur de l'ARN. Un autre protocole très connu (dont on utilise parfois abusivement le nom pour

désigner des données de sondage chimique en général) est le protocole SHAPE [231], *Selective 2'-Hydroxyl Acylation analyzed by Primer Extension*. Ce protocole utilise des réactifs comme 1M7 [243] et NMIA, qui réagissent avec le 2'-OH du ribose dans les régions où la chaine est flexible, ce qui leur permet d'ignorer complètement la séquence de l'ARN. En 2014, SHAPE a également été adapté pour engendrer des mutations au lieu de décrocher la transcriptase, ce qui augmente son efficacité et permet de l'appliquer à haut débit (protocole SHAPE-MaP [310], appliqué au génome du HIV1 en entier). Ces deux variantes là sont les plus utilisées, mais il en existe d'autres (Naz, CMC ou CMCT). Pour augmenter la sensibilité et la spécificité du modèle, il est recommandé de recouper les données de plusieurs protocoles utilisant des réactifs différents [178,289].

I.1.5.6 Diffusion à petit angle

Enfin, la diffusion à petit angle [326] (SAS pour *Small Angle Scattering*, ou plus précisément SAXS pour *Small-Angle Xray Scattering* et SANS pour *Small Angle Neutron Scattering*) est une autre technique optique, mais qui ne nécessite pas l'obtention d'un cristal. Des rayons X de longueur d'onde de l'ordre de l'Angström (10^{-10} m), monochromatiques, ou encore un faisceau de neutrons, sont envoyés sur l'échantillon (en solution, non cristallisé). Ils le traversent et atteignent un écran situé en face, une partie d'entre eux ayant été très légèrement diffusés d'un petit angle autour de la tache centrale. On mesure le profil d'intensité lumineuse reçue autour de la tache centrale. Cette courbe informe sur la position des nuages électroniques de la structure et permet de déterminer la forme d'une structure avec une faible résolution (structures de taille 1 nm et plus). On l'utilise donc surtout pour déterminer la taille et la forme globale d'un ARN ou d'un complexe avec d'autres macromolécules. Les modèles computationnels peuvent ensuite chercher à déterminer la structure à plus haute résolution de façon à respecter les contraintes de forme et de volume données par l'expérience [228].

En pratique, la résolution d'une structure 3D pour une application pharmaceutique ou biotechnologique se fait dans la majorité des cas grâce à une combinaison de méthodes expérimentales, de modèles computationnels, et souvent aussi de modèles par homologie à partir de structures à haute résolution similaires. L'outil intégré capable d'utiliser la plupart des types de données expérimentales et modèles computationnels existants pour produire de nouveaux modèles est Rosetta (dont nous présenterons les composantes et algorithmes plus loin en section I.4.2.5), développé par l'équipe de Rhiju Das à Stanford. Un très bel exemple pratique fut la modélisation rapide des structures d'ARN du SARS-Cov-2 par cette équipe [269,270].

I.2 Modélisation de la structure de la molécule d'ARN

Pour manipuler informatiquement les structures d'ARN, plusieurs niveaux de description sont utilisés. Le premier déjà décrit plus haut est la séquence, chaque lettre correspondant à un groupe d'atomes connus. Ces différents niveaux de modélisation, qu'on va présenter plus en détail, sont illustrés sur la Figure 1.5. Les trois mêmes nucléotides y sont encadrés en rouge et modélisés à différentes échelles.

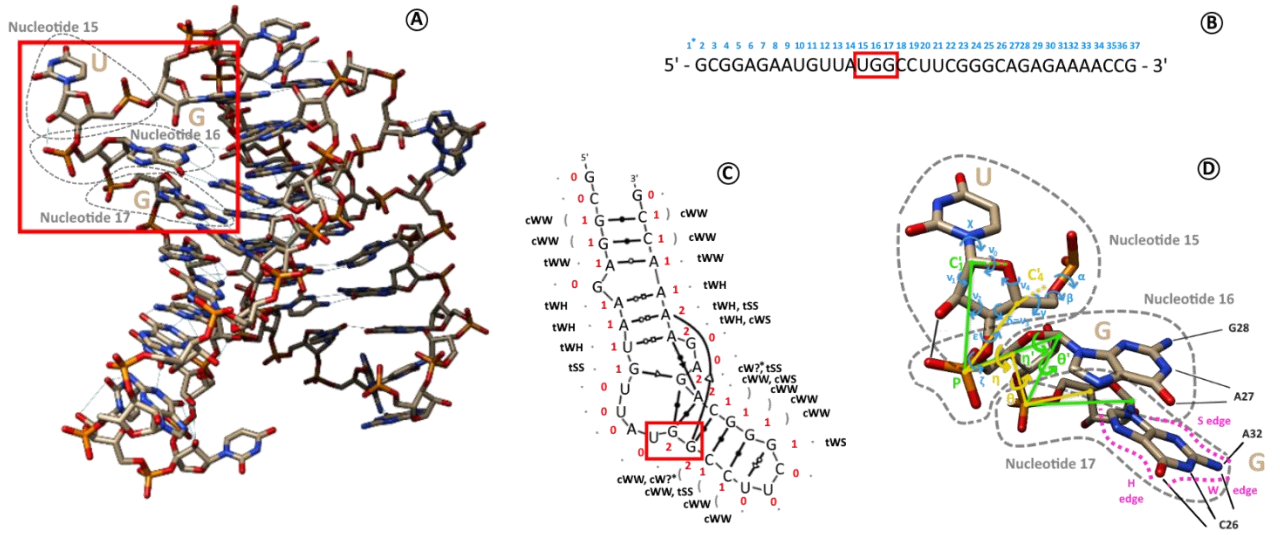


Figure 1.5 – **Modélisation de la structure d'ARN.** *Différentes façons d'encoder la structure d'un ARN.* ***(A)*** *Un modèle atomique 3D complet d'une molécule d'ARN (ref. 6SY6, chaine D).* ***(B)*** *La séquence de cette même chaine. (*) Le résidu numéro 1 n'est pas résolu dans la structure 3D et donc non listé ici.* ***(C)*** *Le graphe de structure secondaire de la chaine en (A). Le nombre d'appariements formés par chaque nucléotide est indiqué en rouge. Le symbole associé au nucléotide en notation point-parenthèses est indiqué en gris, et le(s) nom(s) de(s) l'appariement(s) dans la nomenclature de Leontis-Westhof* [197] *en noir. La première lettre c ou t indique une configuration cis ou trans, et les deux lettres suivantes H, S ou W indiquent quels côtés des bases impliquées sont en interaction, tels qu'annotés sur le nucléotide 17 en (D). Un point d'interrogation est utilisé lorsque l'appariement n'utilise pas l'un de ces cotés en entier.* ***(D)*** *Descripteurs géométriques détaillés des nucléotides 15, 16 et 17, encadrés en rouge en (A), (B) et (C). Le nucléotide 15 est annoté avec ses angles de torsion (en bleu). Le nucléotide 16 est annoté avec les systèmes de pseudo-liaisons P-C4′ en jaune (plus les angles dièdres associés θ et η) et P-C1′ en vert (et les angles θ′ et η′). Ces représentations, qui n'utilisent que deux atomes par nucléotide, permettent de décrire l'orientation du squelette de la molécule et sa forme globale, sans s'occuper de l'orientation des bases azotées. Le nucléotide 17 est annoté avec les noms donnés aux cotés de la base : le coté du sucre (S), le coté Watson-Crick (W) et le coté Hoogsteen (H).*

I.2.1 Le graphe et les éléments de la structure secondaire (2D)

On peut représenter une structure d'ARN comme un graphe, les nucléotides étant les nœuds, les liaisons phosphodiesters et les appariements étant les arêtes. Même en deux

dimensions, ceci donne déjà une bonne idée du repliement de la molécule. Les graphes de structures secondaires sont des objets mathématiques, on peut les agencer sous forme de structure arborescente, sous forme de cercle, ou encore sous forme de ligne, comme sur la Figure 1.6A.

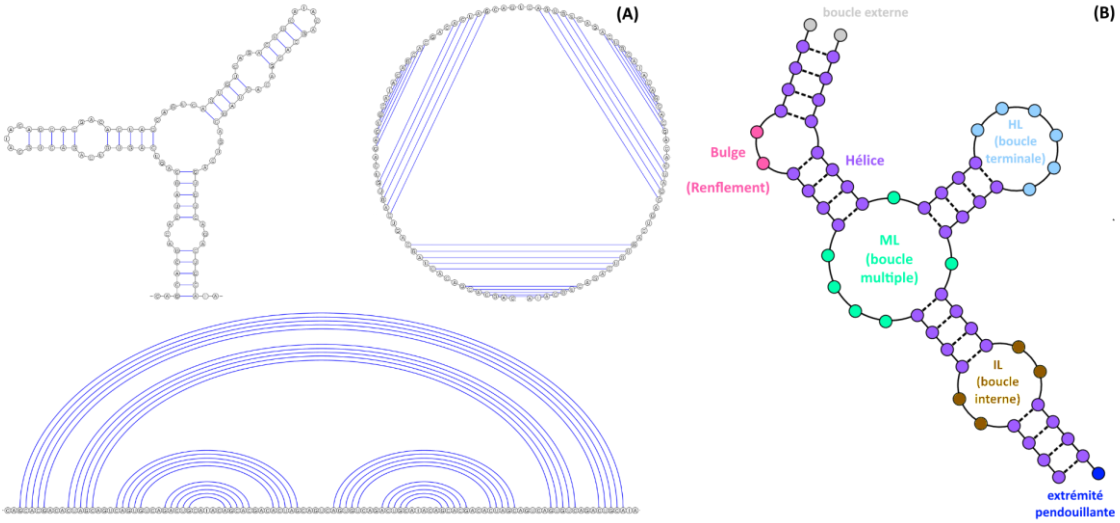


Figure 1.6 – **La structure secondaire.** ***(A)*** *Trois représentations de la même molécule d'ARN couramment utilisées, dessinées par le logiciel VARNA* [81]. ***(B)*** *Principaux éléments de structure secondaire.*
Figure (B) adaptée d'après le Gitlab wiki du logiciel RNAStructure (Mathews group) (https://rna.urmc.rochester.edu:81/mathews-lab/bootcamp/wikis/RNA-Secondary-Structure)

Cette représentation permet de décomposer une structure secondaire en une somme d'éléments (dénotés SSE pour *Secondary Structure Elements*) de même classe : les hélices, boucles, et jonctions. Une hélice est constituée par l'enchainement successif de plusieurs appariements contigus. Comme expliqué en 1.1.3, elles sont peu variables, assez rigides et leur structure est bien connue. Les boucles sont classifiées en fonction du nombre d'appariements "fermant" la boucle, qui est aussi le nombre d'hélices voisines de la boucle. Pour les boucles terminant une seule hélice, on parle de boucle terminale ou en épingle à cheveux (HL pour *Hairpin Loop*). Pour deux hélices, on parle de boucle interne (IL pour *Internal Loop*). Pour plus de deux hélices, on parle de boucle multiple (ML) ou de jonction multiple, bien que le terme "jonction" soit aussi employé pour désigner plus spécifiquement le segment de séquence reliant deux des hélices de la boucle multiple.
On peut encore lister les *bulges* parfois traduits "renflements" ou "hernies" en français, qui sont des cas particuliers d'IL où l'une des deux jonctions est de taille nulle. Le bulge peut aussi s'interpréter comme un ou plusieurs nucléotides pointant vers l'extérieur au milieu d'une hélice plus longue. Enfin, le terme de "boucle externe" désigne les extrémités en début et fin de séquence qui ne contribuent à aucune des autres classes de SSE et dont les contributions énergétiques sont modélisées comme celles d'un nouveau type de boucle.
Ces éléments sont illustrés sur la Figure 1.6B.

Enfin, on distingue un type d'élément de structure secondaire pouvant s'apparenter à une boucle multiple : l'empilement coaxial d'hélices, illustré sur la Figure 1.7. Il est plus stable que la boucle multiple, grâce aux empilements.

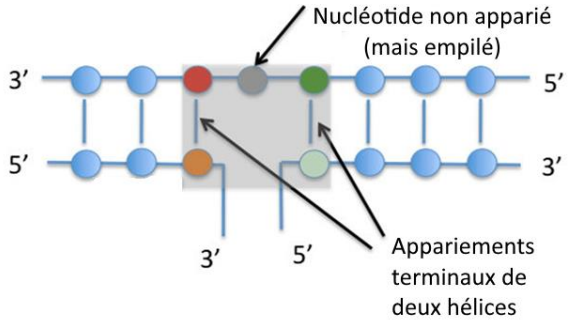


Figure 1.7 – **Les empilements coaxiaux d'hélices.** *La zone grisée correspond théoriquement à une boucle ML, mais elle est plus stable grâce à l'empilement possible du nucléotide gris avec ses voisins. Ceci maintient les deux hélices bleues à gauche et à droite alignées sur le même axe.*
Figure adaptée d'après la Figure 2 de la référence [305] sous licence Creative Commons CC-BY.

I.2.2 Les Pseudonoeuds

Sur la Figure 1.6A, l'exemple donné montre une structure simple, qu'on qualifie de "bien nichée". C'est-à-dire que si l'on prend un segment de la séquence d'ARN fermé par un appariement, tous les appariements qui concernent les nucléotides du segment se feront à l'intérieur du segment. On le voit notamment sur la représentation linéaire : les appariements sont "nichés" les uns dans les autres, sans qu'aucune arête du graphe n'en croise une autre.

Cependant, de nombreuses structures d'ARN ne respectent pas cette règle et contiennent ce qu'on appelle des pseudonoeuds (PK pour *Pseudo Knots*). Les Figures 1.8 et 1.9 présentent quelques classes de pseudonoeuds utilisées par la base de données Pseudobase++ [334]. Cependant, cette classification est incomplète et ne correspond pas à une réalité topologique. On parle de pseudonoeud dès que dans une structure, il existe deux appariements (i,j) et (k,l) se croisant, c'est-à-dire tels que i <k < j < l. Ceci apparait dans les graphes de structures secondaire, où des arêtes se croisent. Dans de nombreux cas, les pseudonoeuds apportent une contrainte structurale qui ne peut pas être modélisée énergétiquement comme la seule somme des contributions énergétiques des hélices formées.

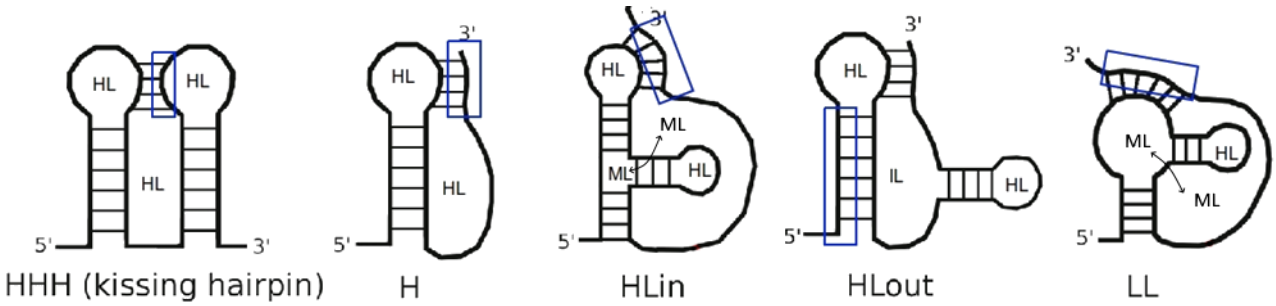


Figure 1.8 – **Classes de pseudonoeuds.** *Les classes de pseudonoeuds répertoriées dans la base de données Pseudobase++. Dans chaque cas, une partie de la molécule (encadrée en bleu) vient former des contacts supplémentaires avec une boucle déjà existante. Dans les pseudonoeuds de classe HLin et LL, les deux boucles multiples (ML) sont en fait une seule et même ML faisant la jonction entre plusieurs hélices.*

L'une des difficultés rendant difficile la modélisation des pseudonoeuds à partir d'une structure secondaire est l'impossibilité de dire, à priori, quels appariements correspondent à des hélices (dont la structure est bien décrite) ou à des contacts "longue distance" formés après la structure secondaire. Ces contacts, bien qu'ils soient des appariements canoniques, ne respectent pas la forme des hélices en 3D.
Certains outils et bases de données [273] séparent donc dans leur modélisation ces "contacts tertiaires" ou "contacts longue distance" pour les différencier des interactions canoniques formant des hélices. Il n'est pas possible a priori, en regardant le graphe de structure secondaire, de savoir dans quel cas un appariement se situe.

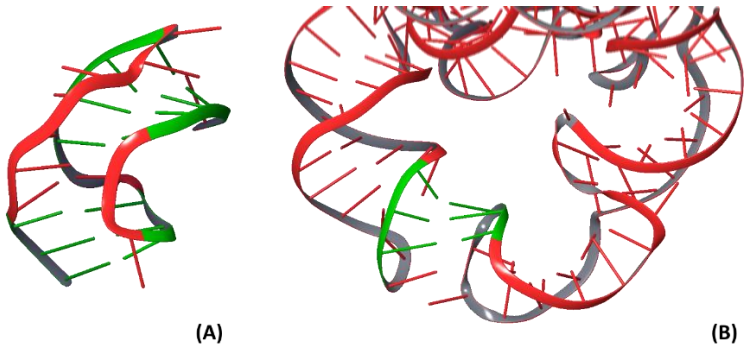


Figure 1.9 – **Exemples de pseudonoeuds en 3D.** *Les hélices sont mises en valeur en vert.* ***(A)*** *pseudonoeud de type H (ref. 437D chaine A).* ***(B)*** *Pseudonoeud de type HHH ou "kissing hairpins" (ref. 4P8Z chaine A).*

L'autre difficulté qu'imposent les pseudonoeuds concerne la complexité algorithmique des méthodes de prédiction. De nombreux outils (dont certains présentés en section I.4.2.4) s'appuient sur l'énumération exhaustive des structures secondaires possibles pour une séquence. Cette tâche est aisée pour des structures "bien nichées" et des schémas de programmation dynamique ont été proposés pour le faire [229,396]. Même si certains schémas ont été proposés pour les classes de pseudonoeuds simples (comme la classe H) [99,279], la complexité de l'énumération n'est pas polynomiale si on supporte les pseudonoeuds de complexité arbitraire, et donc la tâche de détermination de la meilleure structure est NP-difficile [6,212,308]. De ce fait, la plupart des outils supportant les pseudonoeuds soit se limitent à une classe simple de pseudonoeuds, soit utilisent des algorithmes approchés pour énumérer les structures possibles [165,391].

I.2.3 Les interactions non-canoniques et la structure secondaire étendue

Les appariements canoniques ne sont pas les seules interactions possibles pour un nucléotide. Il existe d'autres façons d'interagir via des liaisons hydrogène, moins énergétiques que les canoniques, et pas toujours isostériques entre elles [196]. On peut distinguer les interactions base-squelette (liaison avec un ribose ou un phosphate), et base-base en général. Dans un article fondateur de 2001, Leontis et Westhof décrivent une nomenclature pour un ensemble d'appariements "non-canoniques" plus étendu [197]. Les appariements non canoniques permettent notamment à chaque côté de la base (coté sucre, coté Watson-Crick, et coté Hoogsteen, voir Figure 1.10A) de former un appariement

indépendamment des autres. Ce modèle rend possible les appariements à 3 ou 4 nucléotides ensemble (voir Figure 1.10D), et permet de modéliser les structures contenant des hélices triples ou des structures particulières comme les G-quadruplex (voir figures 1.11B et 1.11C).

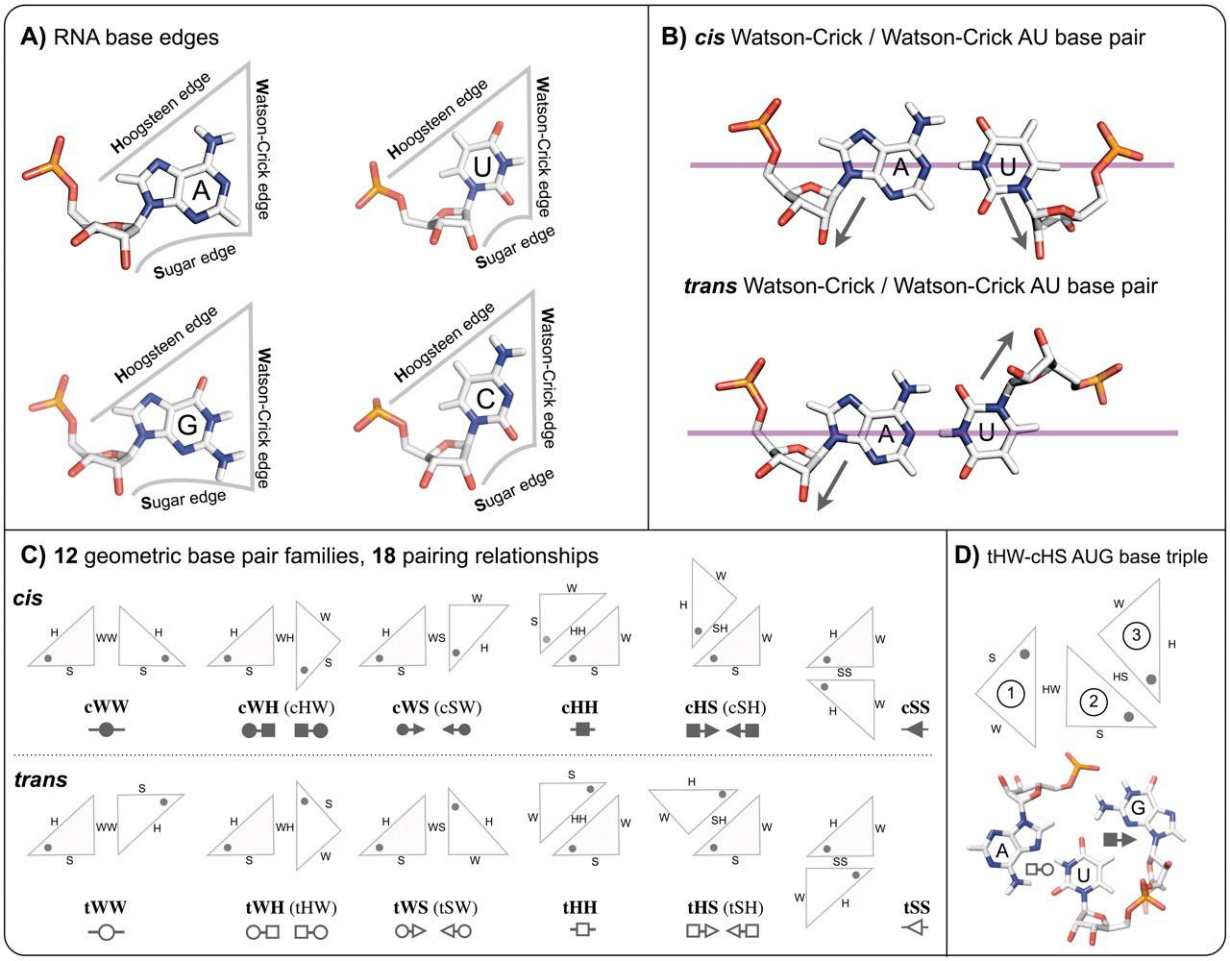


Figure 1.10 – **Nomenclature de Leontis-Westhof.** *Résumé de la nomenclature de Leontis-Westhof.* ***(A)*** *Les bases des nucléotides ont 3 cotés, S, W et H.* ***(B)*** *Dans un appariement, si les sucres et phosphates sont du même côté, ont dit que l'appariement est cis. Sinon, il est dit trans.* ***(C)*** *Les 18 appariements non canoniques proposés, ainsi que leurs codes à 3 lettres, et le symbole utilisé pour annoter les graphes de structure secondaire étendue.* ***(D)*** *Un exemple d'appariement à trois bases. Le U forme en fait deux appariements distincts sur ses cotés W et H.*

Figure réutilisée avec la permission d'Oxford University Press : référence [2] (https://academic.oup.com/nar/article/40/4/1407/2411246)

Les appariements non-canoniques sont calculés à partir d'une structure 3D, c'est-à-dire des coordonnées cartésiennes des atomes de la molécule. On appelle cela "annoter" la structure. Plusieurs logiciels d'annotation existent, capables de dériver la structure secondaire, les interactions non canoniques, les empilements, les angles, les distances, les conformations des sucres, etc... On citera brièvement MC-Annotate [132], FR3D [295], RNAview [367], ClaRNA [344] ou enfin DSSR [208] qui sera utilisé dans les présents travaux. La précision est importante, car tous n'aboutissent pas toujours à la même structure secondaire à partir du même fichier 3D donné par exemple. On citera aussi le serveur web RNApdbee [20,390] qui permet d'automatiser certaines étapes de re-modélisation en plus de l'annotation par un des précédents outils.

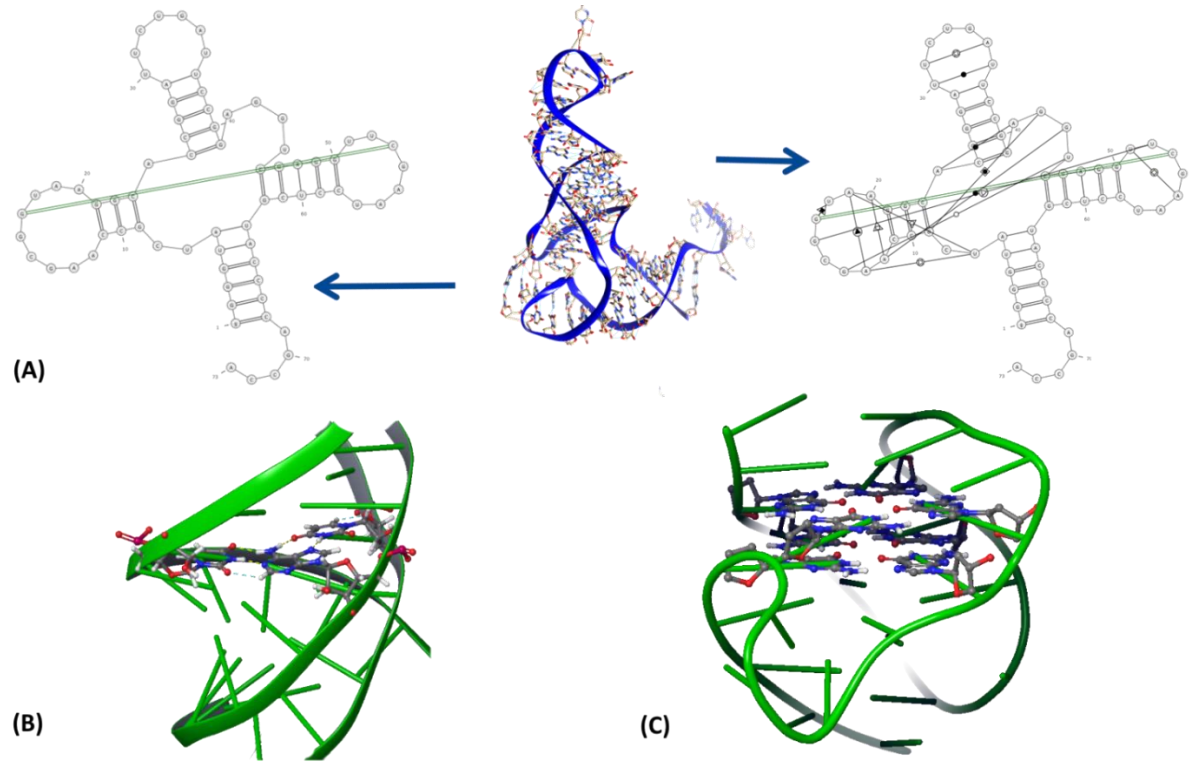


Figure 1.11 – **Les appariements non canoniques.** ***(A)*** *La structure 3D d'un ARNt (au centre) peut être modélisée par la structure secondaire "classique" (à gauche) ou par la structure secondaire étendue prenant en compte les interactions non-canoniques (à droite). On note le pseudonoeud apparaissant en vert. Les symboles utilisés sur les arêtes permettent de connaitre le type d'interaction (voir Figure 10C). La structure secondaire étendue permet de mieux visualiser le repliement global de la chaine d'ARN.* ***(B)*** *Les appariements triples permettent d'expliquer les structures d'ARN triplex (hélice à 3 brins, ref. 6SVS chaine A).* ***(C)*** *Une structure en G-quadruplex (ref. 2hy9). Quatre guanines s'apparient par leurs côtés Hoogsteen. Plusieurs plateformes de quatre guanines peuvent s'empiler. La structure finale est très stable, d'autant plus si elle est stabilisée par un cation entre deux empilements.*

I.2.4 Autres graphes simples pour modéliser une structure d'ARN

Au-delà de la structure secondaire simple ou étendue, d'autres façons de modéliser l'ARN avec des graphes existent. On citera le graphe de SSE (où chaque SSE est un nœud), et les modèles de Tamar Schlick et son équipe [128,298]. En 2D, la modélisation de Schlick consiste à prendre un graphe proche du graphe dual de celui de la structure secondaire. Les boucles sont représentées par des nœuds, et les hélices par des arêtes. Ainsi, une boucle de type IL sera par exemple modélisée par un nœud relié à deux autres par deux arêtes. Ce modèle permet de visualiser facilement les classes de pseudonoeuds. En 3D, ce même « graphe dual » positionne les nœuds au centre des boucles, et les longueurs des arêtes sont bien déterminées, pour correspondre à l'axe de l'hélice modélisée. Le graphe 3D décrit donc très bien la topologie générale de la chaine. Cette approche de graphe reliant les SSE est utilisée dans certaines méthodes de prédiction, par exemple ERNWIN [175], GARN [44,45] ou RAGTOP [161].

I.2.5 Modèles et outils d'analyse des motifs 3D

L'utilisation d'interactions non canoniques permet de décrire des portions de structures plus complexes que les éléments de structures secondaires simples. Cependant, la classification est moins consensuelle que pour les SSE. L'idée principale est de préciser les différents types de boucles en prenant en compte les interactions non-canoniques. Certains auteurs associent au nœuds du graphe des fréquences nucléotidiques (des statistiques sur quel nucléotide trouver à quel endroit), on parle alors de "motif 3D" ou de "module" pour éviter la confusion avec le terme "motif" qui s'emploie souvent en référence à un motif de séquence seulement (comme "TATAA"). D'autres ignorent la séquence et se concentrent sur le graphe d'interactions, on parle alors de réseau d'interactions. On distingue également les modèles reposant sur l'isostéricité et la ressemblance géométrique de ceux purement basés sur une analyse symbolique du graphe d'interactions non-canoniques, en prenant ou non en compte la séquence. Enfin, on distingue deux types de méthodes :

- celles permettant de faire la recherche d'une sous structure (le motif 3D) dans une collection d'autres structures [21,107,143,201,295,368,384],
- et celles, plus complètes, permettant de découvrir dans un jeu de données de structures tous les motifs 3D potentiels sans les connaitre à l'avance [343,194,101,255,273].

Parfois, l'usage systématique des méthodes du premier type pour rechercher systématiquement tous les fragments de toutes les structures dans toutes les autres structures permet d'aboutir à une matrice de proximité ou matrice d'équivalence entre fragments, et l'identification de clusters ou cliques dans cette matrice conduit à une liste de motifs récurrents [101,255,105,386,130]. Mais ceci n'a pas toujours été implémenté par les auteurs. Les modèles existants à ma connaissance sont présentés ci-dessous.

- Dès 2003, Duarte & Pyle [107] proposent d'identifier des « motifs 3D » sur la base des angles de torsion du squelette. Leur programme PRIMOS calcule un *RNA worm* pour chaque structure donnée, ce qui correspond à une valeur du couple d'angles de torsion θ et η (voir Figure 1.5D) pour chaque nucléotide. Il est ensuite possible de rechercher les occurrences d'un motif particulier dans d'autres structures. L'année suivante, Wadley & Pyle proposent COMPADRES [343], qui automatise la recherche de nouveaux motifs dans un large jeu de données.

- La même année 2003, le programme NASSAM [143] est proposé. Leur modèle décrit les différents types d'appariements sur la base de mesures de distances entre 3 ou 4 atomes de chaque base azotée de l'appariement. La structure 3D est approximée par un graphe en 3D reliant ces 3 ou 4 atomes pour chaque nucléotide, en prenant en compte les distances sur les arêtes du graphe. Puis, un algorithme de recherche de sous-graphes isomorphes permet de retrouver différentes occurrences d'un motif proposé comme requête en entrée.

- En 2005, Dror, Nussinov et Wolfson proposent ARTS [105], un algorithme pour

superposer deux structures d'ARN en 3D sur la base du plus grand sous-ensemble possible d'atomes de P. L'algorithme utilisé résout en fait un problème de LCP (Largest Common Point set). Le calcul de toutes les comparaisons possibles entre structures d'un jeu de données leur permet ensuite d'identifier de nouveaux motifs 3D. Cependant, une bonne partie de ces "motifs 3D" correspondent surtout aux formes conservées au sein d'une famille d'ARN et dépassent la définition que nous nous donnons d'un module.

- En 2006, Lemieux et Major [194] proposent d'utiliser les plus petits cycles observés dans les graphes de structure secondaire étendue comme motifs élémentaires de structure (les NCM, *Nucleic Cyclic Motifs*). Ainsi, les boucles IL et HL standard sont des NCM, mais également celles qui seraient fermées par des appariements non canoniques. Deux appariements (canoniques ou non) empilés forment aussi un NCM. Une collection de NCM a été extraite de la structure d'une LSU et a été utilisée par leurs outils de modélisation plus tard.

- En 2008, Djelloul et Denise [101] ont fait la recherche des plus grands sous-graphes communs entre les graphes des SSE lorsqu'on leur restore leurs interactions non-canoniques. La comparaison est faite deux à deux sur les SSE d'un ensemble de structures 3D d'ARN connues. Pour chaque paire, l'identification du plus grand sous-graphe non canonique permet de déduire en une mesure de similarité. À partir des similarités, les SSE peuvent être organisés en clusters. Le sous-graphe commun aux membres d'un même cluster peut être identifié et catalogué comme un « module » d'ARN. Des statistiques sur la nature des nucléotides occupant réellement chaque nœud dans le cluster peuvent accompagner le modèle. Si la méthodologie est intéressante car purement "symbolique" (aucune mesure géométrique n'est utilisée dans ce modèle), on déplore le fait que la collection de sous-graphes résultante n'a jamais été mise à jour depuis, et la non-disponibilité du code n'aide pas à la reproduction. Ce jeu de données est connu sous le nom « Rna3dMotifs ».

- La même année 2008, une méthode reposant sur des *shape histograms* est proposée [21]. Un shape histogram représente la distribution statistique mesurée des distances entre chaque atome d'un fragment traversant une région délimitée de l'espace (en 3D) et un point de référence donné de cet espace, comme le centroïde des atomes du fragment. La comparaison de deux sous-structures d'ARN peut alors se réduire à la comparaison de leurs shape histograms. Une méthode de recherche de motifs est donc proposée sur ce principe. La même idée est réutilisée par des auteurs différents en 2010 pour étudier la distribution des positions des motifs le long des chaines d'ARN [292].

- En 2009, un groupe Taïwanais propose FASTR3D [188], un « moteur de recherche » pour trouver des structures d'ARN contenant une séquence, structure secondaire ou structure 3D souhaitée. La recherche de structure 3D se fait à partir des similitudes d'angles (θ, η). En 2011, ce même groupe adapte l'algorithme de recherche de

séquences BLAST [13] en R3D-BLAST [201] pour faire des recherches de sous-structures directement sur la conformation du squelette de la molécule. La séquence 3D de la sous structure est construite en remplaçant chaque nucléotide par une lettre d'un alphabet de 23 symboles, chaque symbole correspondant à une valeur discrète du couple d'angles (θ, η). L'intégralité des structures 3D disponibles sont ensuite converties en une base de données de séquences dans cet alphabet, et BLAST peut être appliqué pour chercher n'importe quelle séquence 3D dans cette base. On peut donc retrouver des boucles de formes similaire, mais les appariements et conformations des bases ne sont pas pris en compte du tout. Une mise à jour propose R3D-BLAST2 [368] avec un alphabet structural plus récent en 2017.

- En 2010 sort RNAMotifScan [384]. C'est un autre outil permettant de faire de la recherche des occurrences d'une sous-structure dans un ensemble de structures, en prenant en compte l'isostéricité et les contraintes symboliques des appariements non canoniques. La démarche ressemble à celle de Djelloul & Denise, mais au lieu de considérer uniquement l'étiquette non-canonique d'une arête du graphe de structure secondaire pour déterminer l'isomorphisme entre deux sous-graphes, RNAMotifScan autorise des variations si les appariements sont isostériques. RNAMSC [386] rassemble des motifs récurrents identifiés par RNAMotifScan et montre qu'on y retrouve les modules connus habituels. Une mise à jour en 2015 propose RNAMotifScanX [385], qui rajoute la prise en compte des empilements (une nouveauté) et retourne aussi des solutions partielles. Plus tard en 2017, l'algorithme de clustering est mis à jour pour aboutir à la base de données RNAMotifClusters [130] : toutes les boucles HL, IL et ML sont comparées entre elles dans leurs classes respectives par RNAMotifScanX, puis un graphe de proximité est construit à partir des scores obtenus, organisant les boucles en clusters. Enfin, l'algorithme de clustering est encore mis à jour en 2021 avec RNAMotifContrast [159]. Les clusters ne sont plus obtenus sur la métrique de distance de RNAMotifScanX uniquement, mais prennent en compte une nouvelle mesure de distance, qui utilise elle-même l'alignement de séquences et le RMSD en 3D. Le dataset obtenu classe les motifs en quelques très grandes familles de boucles IL et HL (Sarcine-ricine, boucle T, Kink-turn, GNRA, ...)

- Le "BGSU RNA Group" (composé de Petrov, Zirbel, Leontis, et d'autres collaborateurs) propose en 2008 le logiciel d'annotation et de recherche de motifs, appelé FR3D [295]. En 2013, le groupe met en ligne le "RNA 3D Motif Atlas" [255], basé sur FR3D. Cette collection de modules repose sur le regroupement des modules d'abord par similarité géométrique (isostéricité), et ensuite seulement sur la séparation des clusters si des différences existent dans les appariements impliqués. Dans leur modèle, seuls les nucléotides appariés (canoniques et non-canoniques) sont pris en compte pour la superposition, ce qui permet d'obtenir au sein d'un même cluster des boucles de taille différente si certaines bases pointent à l'extérieur, et que celles appariées restent bien isostériques. L'Atlas ne s'intéresse malheureusement qu'aux boucles HL et IL cependant. Le site internet propose des superpositions 3D visuelles

des membres du cluster, des statistiques sur la séquence des modules, et est mis à jour tous les mois depuis 2013 avec les données les plus récentes.

- En 2014, le groupe de Bujnicki propose la base de données RNA Bricks [57], regroupant des motifs de type boucle, hélice, ou fragment, conservés dans leurs contextes (eau, ions, ligands, protéines en contact...). Les structures 3D d'ARN sont décomposées en SSE. Les SSE sont triés par groupes (de taille) puis superposés deux à deux et clustérisés sur la base du RMSD. En particulier, les interactions présentes dans les membres de chaque cluster peuvent différer, mais leur forme géométrique sera proche (<1 Angström de RMSD). Un programme de recherche de motifs 3D similaires à un template dans un ensemble de structures est également proposé et utilise aussi la superposition géométrique. La base de données est mise à jour en RNA Bricks 2 en 2019 (support des structures au format mmCIF, utilisation de ClaRNA pour l'annotation), et est continuellement mise à jour depuis.

- Dix ans après Rna3dMotifs, en 2018, Denise et d'autres collaborateurs étendent ce modèle aux sous-graphes d'interaction entre deux SSE. Il en ressort la base de données CaRNAval [273], qui propose 337 réseaux d'interactions récurrents (RIN) organisés sous forme d'arbre, les sous-graphes fils étant des extensions des sous-graphes pères. Le principal avantage de CaRNAval sur ses compétiteurs est la prise en compte de motifs d'interaction longue-distance, entre deux SSE même très distants dans la structure. On peut y retrouver par exemple des motifs d'interaction entre une boucle HL et une boucle IL via des appariements non canoniques. CaRNAval 2 [318] est proposé en 2020 et étend encore le modèle à un nombre arbitraire de SSE, en plus de grosses améliorations de performance. Enfin, VeRNAl [249] (également proposé en 2020) est une variante qui s'autorise à regrouper des réseaux d'interactions « flous » et non parfaitement isomorphes.

Le modèle s'imposant lentement comme référence est celui du BGSU RNA Group, de par la rigueur de son modèle, ses mises à jour régulières et son intégration avec la Nucleic Acid Database [65]. Néanmoins, les modèles de RNAMotifContrast, CaRNAval 2, et la base RNA Bricks 2 continuellement mise à jour sont également récents et pertinents.

La Figure 1.12 illustre quelques modules bien décrits et faisant consensus selon les modèles précédemment cités.

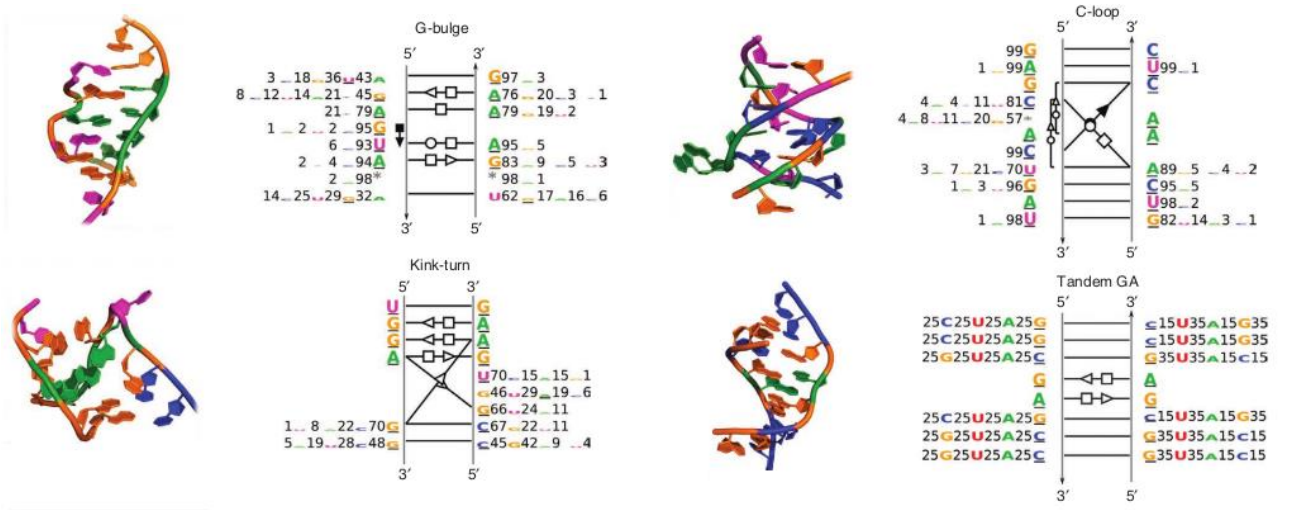


Figure 1.12 – **Les modules d'ARN.** *Illustration de quatre modules existants à l'intérieur de boucles internes (IL). Ces boucles apparaissent au milieu d'une hélice canonique plus grande. Pour chacune, on peut écrire le graphe d'interactions non canoniques, et les fréquences nucléotidiques associées à chaque noeud du graphe.*

Figure extraite de *Cruz, J. A. and Westhof, E. (2011).* Sequence-based identification of 3d structural modules in RNA with RMDetect[75].

I.3 Paysage énergétique et états métastables

Une molécule d'ARN n'a pas *une* structure, mais une infinité d'états métastables. Certains sont largement plus fréquemment observables que d'autres, souvent parce qu'ils sont plus stables. Certains de ces états peuvent être considérés par un biologiste comme étant « bien repliées », par opposition à des formes mal repliées. Ils prennent alors le nom de structure « native ». Certains parmi les états natifs peuvent même être utiles à une fonction, et on les appellera « structure active », en particulier lorsqu'il y a opposition avec une forme inactive (par exemple dans les riboswitches). Il n'y a pas toujours de correspondance entre la forme native (ou active) et la solution la plus stable (de plus basse énergie).

Quand on parle de « la » structure d'un ARN, parce qu'on souhaite l'étudier pour une application biotechnologique, la compréhension d'un mécanisme cellulaire, on fait donc une approximation de la réalité. En réalité, on souhaite étudier la ou les principales formes stables de l'ARN, et parfois, les mécanismes de transition entre elles. Quand on parle de « la » structure d'un ARN parce qu'on l'a trouvée dans une base de données, il faut se souvenir qu'il ne s'agit que d'une forme, figée, obtenue par une méthode expérimentale qui contraint peut-être l'ARN à se replier dans une conformation différente de la forme native ou active (par exemple dans le cas de la cristallographie aux rayons X). Dans cette thèse plus orientée vers l'informatique que les applications en biotechnologie, on s'intéresse à l'exercice de prédiction de structures en soi. On considère donc les structures présentes dans les bases de données comme la vérité à obtenir, en ayant conscience de l'approximation réalisée.

En physique, on sait relier la probabilité d'existence des conformations à une grandeur appelée énergie libre de la structure. On cherche donc à décrire un « paysage énergétique » des conformations, pour estimer lesquelles sont probables.

I.3.1 Energie d'une structure

L'énergie interne d'une structure est une grandeur physique associée à la molécule. En effet, les atomes qui constituent la molécule interagissent entre eux par des interactions chimiques bien décrites (liaisons covalentes, liaisons hydrogène, coordination de cations Mg^{2+}, interactions de van-der-Waals…). Chacune de ces interactions possède une géométrie optimale (typiquement, une distance d'équilibre ou un angle d'équilibre), qui minimise l'énergie interne des atomes interagissant entre eux. Dans une structure d'ARN donnée et figée, la géométrie des interactions n'est pas optimale : en effet, par exemple, la formation d'un appariement stabilise suffisamment la structure pour qu'elle « s'autorise » à dévier des valeurs d'angles et longueurs standard entre liaisons covalentes. La molécule d'ARN a besoin de moins d'énergie pour exister dans cette conformation avec appariement que sans, même si l'énergie individuelle de certaines liaisons covalentes n'est pas minimale. L'énergie interne U d'une conformation est donc additive des énergies internes relatives aux différentes interactions, mais son minimum ne correspond pas à la somme des minima individuels, à cause de l'impossibilité de satisfaire les contraintes géométriques associées.

A l'inverse de l'énergie interne U, l'énergie libre F d'une structure n'est pas additive. Un terme d'entropie S doit être pris en compte, et ce terme ne peut pas être calculé individuellement pour chaque interaction interne. Il doit être estimé de manière globale. Les structures les plus stables sont celles d'énergie libre F minimale, et non pas celles d'énergie interne U minimale. C'est pourquoi minimiser l'énergie libre devrait donner de meilleures performances, et certains modèles s'y sont essayés, comme le modèle VFold [51].

En thermodynamique, l'énergie interne F est utilisée pour décrire la variation d'entropie reçue par l'extérieur lors de la formation du système considéré (ici la structure d'un ARN). En effet, le repliement d'un ARN dans un cube de solvant considéré autour de lui est une transformation isotherme (température T constante) et isochore (volume V constant). Selon le premier principe de la thermodynamique, la variation d'énergie interne U s'écrit $\delta U = \delta W + \delta Q = \delta Q$ (car le terme habituel δW est nul, le système ne fournit pas de travail mécanique, seule reste la chaleur échangée δQ). Selon le deuxième principe de la thermodynamique, l'entropie créée par la transformation est

$$dS_{créée} = dS_{sys} + dS_{ext} = \frac{\delta Q_{sys}}{T_{sys}} + \frac{\delta Q_{ext}}{T_{ext}} > 0.$$

La température T est la même pour le système et l'extérieur. Le système donne une chaleur δQ au milieu extérieur, et le milieu extérieur donne donc $-\delta Q$. Donc $dS_{ext} = \frac{\delta Q_{ext}}{T_{ext}} = \frac{-\delta Q}{T} = \frac{-\delta U}{T}$, d'où $dS_{créée} = dS_{sys} - \frac{\delta U}{T} > 0$. En multipliant par -T (négatif), on obtient $-T.dS_{sys} + \delta U < 0$, et c'est cette quantité qu'on appelle F, l'énergie libre (l'équation utilisant les grandeurs non différentielles est $F = U - T.S$, on retrouve bien $dF = \delta U - T.dS$ avec les différentielles). Aussi, l'énergie libre d'une structure est dépendante de la température. Un certain nombre de méthodes de prédiction, notamment celles qui utilisent des paramètres thermodynamiques mesurés, font l'hypothèse d'une température de travail (298K ou 300K le plus souvent). Ceci a également pour conséquence une difficulté supplémentaire pour l'identification des conformations biologiquement actives des ARN : puisqu'à 37°C, de l'énergie est disponible dans le milieu (mais également pour d'autres raisons, comme la cinétique), les conformations réelles des ARN observés dans la cellule ne sont pas forcément ni les conformations d'énergie libre minimale, ni les conformations les plus stables.

Enfin, L'énergie libre dépend des concentrations ioniques. Quand on modélise une structure seule, on parle d'énergie libre et pas d'enthalpie libre (la pression et le volume ne sont ici pas définis). Cependant en réalité, l'ARN est dans un solvant (l'eau) contenant souvent des solutés, dont des cations positifs qui viennent stabiliser les charges négatives portées par les phosphates du squelette (à pH raisonnable). Ces ions ne sont souvent pas explicitement considérés, mais on les modélise en disant que l'énergie libre dépend de leur concentration.

I.3.2 Modèles d'énergie et champs de forces

La détermination exacte de la fonction d'énergie dépend de calculs de mécanique quantique trop couteux pour être appliqués à des systèmes de plus d'une centaine d'atomes. Cependant, des modèles de cette fonction ont été proposés à l'échelle supérieure, dont les paramètres sont eux-mêmes estimés par mécanique quantique sur des petits systèmes. On appelle ces modèles de la fonction d'énergie des « champs de forces ». Ils approximent la fonction d'énergie à l'échelle du groupe d'atomes ou même plus, selon leur « granularité » (on parle de modèle « à grains fins » contre modèles « gros-grains »).

Dans la suite de l'introduction, on mentionnera de nombreux champs de forces plus ou moins gros grains et appliqués à l'ARN. Néanmoins, quelques modèles très généraux et non spécifiques de l'ARN méritent une introduction préalable. Ils sont parfois utilisés dans les outils mentionnés plus bas, en particulier pour « raffiner » les modèles de structure après une prédiction par une méthode quelconque.

Depuis 40 ans, deux gros modèles s'affrontent : AMBER et CHARMM, similaires dans le principe mais différents dans la façon dont les paramètres sont déterminés. Quelques reviews récentes [316,79] permettent de lister les principales différences entre leurs versions.

- AMBER (*Assisted Model Building with Energy Refinement*) est sujet à de nombreuses branches divergentes, en particulier pour la modélisation des acides nucléiques.
    - (1995) AMBER-ff94 [70] est la première version à supporter les ARN en solvant explicite,
    - (2000) AMBER-ff99 [348] possède un paramètre pour l'orientation de la base déterminé par des calculs de mécanique quantique (QM),
    - (2007) AMBER-ff99 supporte maintenant les nucléotides modifiés [3],
    - (2007) AMBER-ff99$_{BSC0}$ [254] est paramétrisé sur des expériences de QM de meilleure qualité, qui corrige des artefacts dans les angles de torsion α et γ,
    - (2010) AMBER-ff99- χ_{YIL} [371] améliore les prédictions des angles χ (torsion entre le ribose et la base),
    - (2011) AMBER-ff99$_{BSC0}$-χ_{OL3} [376] est une autre amélioration des angles χ, similaire à AMBER-ff99- χ_{YIL}, mais proposée par un autre groupe de recherche, et aboutissant à de meilleures hélices (forme A),
    - (2012) Steinbrecher [322] améliore AMBER-ff99$_{BSC0}$-χ_{OL3} avec une modification des paramètres de van-der-Waals des phosphates. C'est la version recommandée par le manuel d'AMBER,
    - (2012) AMBER-ff99$_{TOR}$ [370] est une extension de AMBER-ff99- χ_{YIL} qui remodélise tous les angles de torsion du squelette,
    - (2013) Chen & Garcia [54] proposent une version sans nom, dont les paramètres reproduisent mieux les énergies des empilements, et encore des valeurs de χ personnalisées,
    - (2015) AMBER-ff99$_{OPC+vdWbb}$ [30] modifie les paramètres de van-der-Waals des phosphates et ajoute le modèle OPC pour les molécules d'eau,

- (2017) AMBER-ff99$_{OPC+vdWYP}$ [366] propose une autre correction des paramètres de van-der-Waals des phosphates avec le modèle OPC,
- (2017) AMBER-ff99-χ+α/γ [346] est une révision récente similaire à AMBER-ff99$_{BSC0}$-χ$_{OL3}$,
- (2017) Mathews [22] rajuste tous les angles de torsion de AMBER-ff99$_{BSC0}$-χ$_{OL3}$, cette version est connue sous le nom de « Rochester » dans le manuel,
- (2018) Shaw [329] améliore AMBER-ff99$_{BSC0}$-χ$_{OL3}$ en rajustant les charges, les paramètres de Lennard-Jones (distances d'équilibre) et les torsions γ, χ et ζ, version connue sous le nom « D E Shaw »,
- (2019 et 2020) Deux termes additionnels, gHBfix [187] et tHBfix [242], sont proposés pour mieux modéliser les liaisons hydrogène.

- CHARMM (*Chemistry at HARvard Macromolecular Mechanics*), dont la version commerciale est vendue au sein de la suite logicielle BioVia de Dassault-Systèmes (anciennement Accelerys).
  - (1995) CHARMM22 [214] est une première version supportant les ARN en solvant explicite,
  - (2001) CHARMM27 [122] révise ses paramètres,
  - (2011) CHARMM36 [92] est une révision concentrée sur le 2'-OH de l'ARN et sur ses effets sur la structure, à la différence de l'ADN. L'ARN est bien mieux modélisé,
  - (2016) CHARMM36m embarque de nouveaux paramètres pour les nucléotides modifiés.

D'autres exemples sont le vieux GROMOS [317] (datant de 2005, et peu utilisé), le modèle AMOEBA [276], le modèle MMFF de Merck, et le récent OPLS-AA (*Optimized Potential for Liquid Simulations*, version All-Atom) [283]. Les autres champs de forces mentionnés plus loin, ou même utilisés dans cette thèse, peuvent donc être considérés comme des approximations, ou parfois des « lissages » de ces champs de forces très détaillés.

I.3.3 Notion de paysage énergétique et états métastables

La fonction mathématique (ou son modèle de champ-de-forces) décrivant l'énergie (interne ou libre) d'une structure dépend donc entre autres de la température, des concentrations ioniques, des distances et angles entre atomes, et de la topologie globale de la molécule, notamment pour mesurer les contributions entropiques et d'exclusion de solvant. Elle dépend donc d'un très grand nombre de variables. En analyse conformationnelle, on appelle chacune de ces variables une « coordonnée de réaction », et on essaie parfois de représenter des modèles de la fonction d'énergie en fonction d'une ou deux coordonnées de réaction. On peut aussi observer une trajectoire réelle d'un système dans l'espace des conformations, et appeler l'avancée sur cette trajectoire les « coordonnées réactionnelles », comme sur la Figure 1.13. La fonction d'énergie étant fortement dépendante d'angles, de nombreux termes trigonométriques donnent aux courbes

mathématiques associées des aspects très vallonnés avec des « montagnes », des « vallées » et des « bassins » ou « puits d'énergie ».

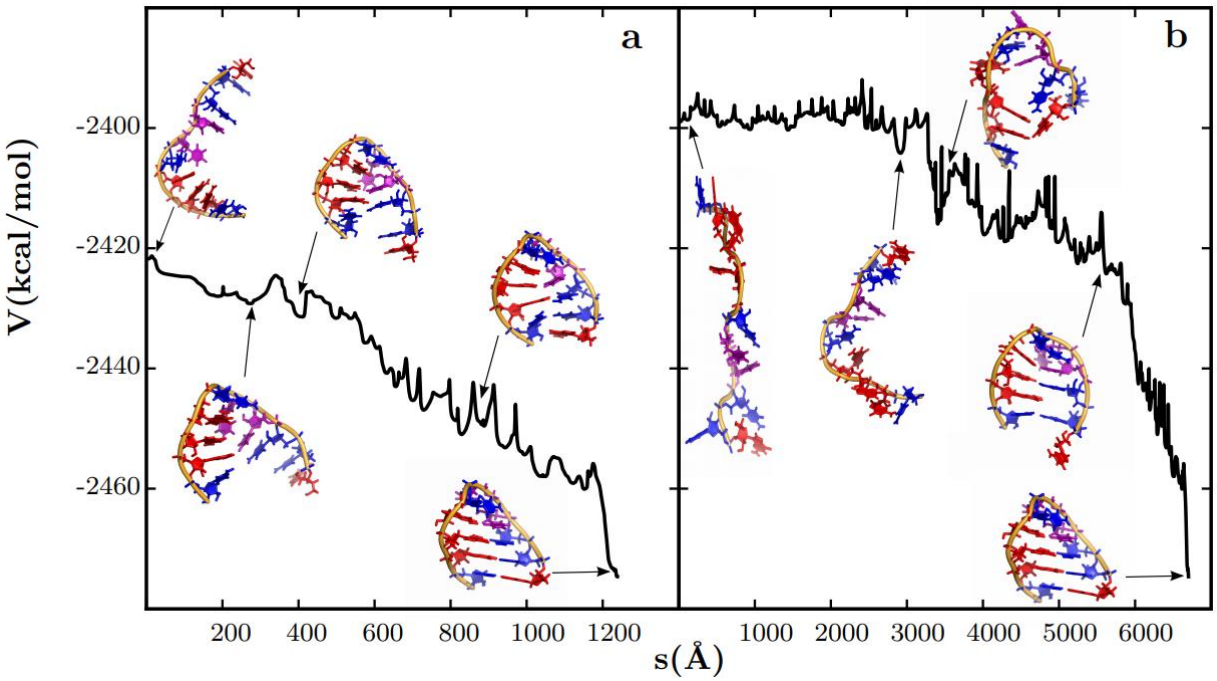


Figure 1.13 – Trajectoire de repliement d'une HL à partir de deux états initiaux. *Ces trajectoires ont été obtenues par simulation numérique* [53]*. La courbe d'énergie (V) est représentée en noir en fonction des coordonnées réactionnelles. À partir de la conformation à gauche, la molécule traverse un grand nombre d'états avant de finir dans la conformation stable de basse énergie en bas à droite. (a) et (b) diffèrent par l'état initial. En particulier, de nombreux minima locaux d'énergie sont traversés.*

Figure extraite de la référence [53] avec la permission d'ACS Publications.

La fonction d'énergie est donc fortement non linéaire et dépendante d'un grand nombre de variables. Elle admet donc de très nombreux minima locaux, des conformations localement stables. Le problème d'optimisation associé, qui consiste à identifier la conformation d'énergie (libre) minimale, est donc extrêmement difficile, et dans la plupart des cas impossible à résoudre de façon exacte. La Figure 1.14 permet de se rendre compte du nombre de ces minima locaux même pour un système simple, et de la « rugosité » de la fonction d'énergie. L'utilisation de modèles gros grains permet de lisser un peu cette fonction et de faire apparaitre les puits (ou bassins) d'énergie plus distinctement.

Malgré la difficulté du problème, de nombreuses méthodes en bio-informatique structurale se donnent pour tâche d'explorer vraiment le paysage énergétique plutôt que de se contenter d'une simple prédiction d'une ou plusieurs structures. Ces méthodes cherchent alors à identifier les principaux bassins d'énergie, appelés états métastables, et les barrières énergétiques entre eux, ou encore les conformations très peu stables mais temporairement nécessaires pour passer d'un état stable A dans un bassin à un état stable B dans un autre bassin.

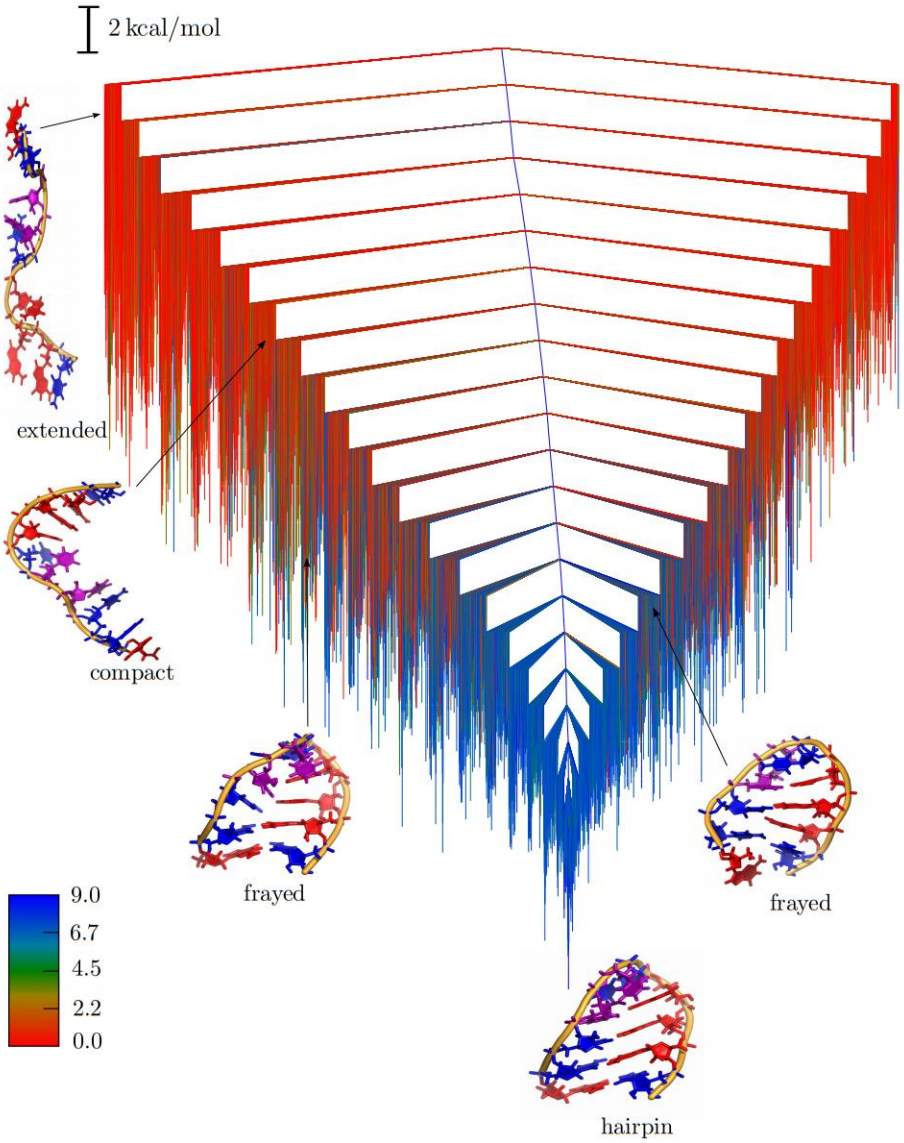


Figure 1.14 – Bassins d'énergie d'une HL simple. *Regroupement des minima locaux d'énergie d'une structure simple par proximité des uns aux autres. En particulier, la longueur des branches de l'arbre indique la valeur du minimum local d'énergie, et la distance à parcourir pour aller d'un état à un autre indique la barrière d'énergie à fournir pour effectuer le changement de forme. La couleur indique le nombre de nucléotides appariés. Quelques conformations sont illustrées en 3D.*

Figure extraite de la référence [53] avec la permission d'ACS Publications.

I.3.4 Ensemble de Boltzmann

On vient de le voir, de nombreux minima locaux existent dans le paysage énergétique, et le milieu étant suffisamment chaud, les structures peuvent passer quelques barrières énergétiques et changer de conformation entre différents états métastables.

Ainsi, si l'on considère une infinité de molécules d'ARN partageant une même séquence à une température donnée, et qu'on regarde à un instant donné la conformation figée des molécules, on observera un très grand nombre de structures, mais dans des proportions différentes, certaines de plus basse énergie étant fortement majoritaires. Les proportions en question suivent ce qu'on appelle une distribution de Boltzmann, c'est-à-dire qu'elles sont directement reliées à l'énergie de chaque structure. La distribution de Boltzmann relie la fréquence de chaque structure *s* à son énergie E(s) selon la formule :

$$\frac{N(s)}{N} \approx p(s) = \frac{e^{-E(s)/k_B T}}{Z}$$

où k_B est la constante de Boltzmann, T la température, et Z la fonction de partition :

$$Z = \sum_{s} e^{-E(s)/k_B T}$$

Un échantillon de structures de taille finie et respectant ces proportions est appelé ensemble de Boltzmann. En énumérant toutes les structures secondaires possibles pour une séquence d'ARN, puis en les pondérant par leur probabilité d'existence dans un ensemble de Boltzmann, on peut calculer la fréquence d'existence de chaque appariement possible entre deux nucléotides (i,j) de la séquence. On appelle cette fréquence la probabilité de l'appariement (i,j). On peut alors résumer l'ensemble de Boltzmann par une matrice triangulaire supérieure de probabilités d'appariement, où par des graphes dont les arêtes sont grisées en fonction de leur probabilité d'existence, comme sur la Figure 1.15.

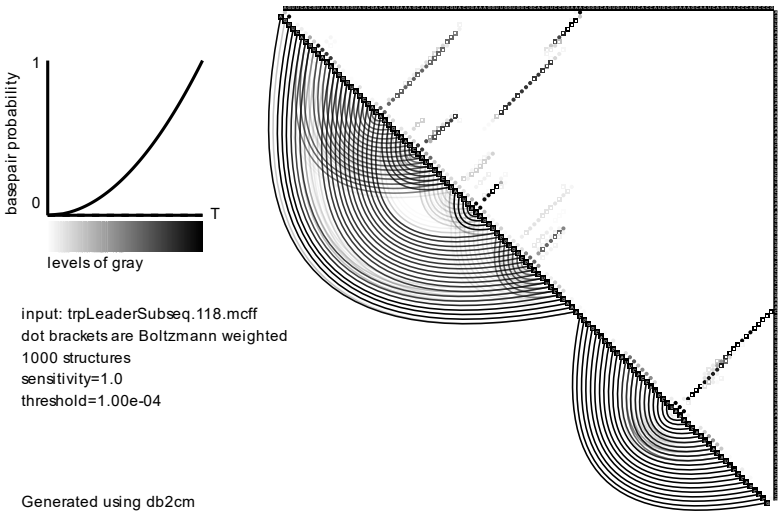


Figure 1.15 – L'ensemble de Boltzmann des structures secondaires. *Représentation conjointe de la matrice de probabilité d'appariement et du graphe de structure secondaire associé, en niveaux de gris corrélés aux probabilités d'appariement.*

Source de la figure : exemple fourni en téléchargement avec le logiciel MC-Flashfold (https://major.iric.ca/MajorLabEn/MC-Tools.html)

I.3.5 Estimateurs de la structure secondaire

Ainsi, les méthodes de prédiction de structure secondaire, si elles renvoient une seule solution, font forcément une approximation de la réalité. On appelle cette approximation un *estimateur* de la structure secondaire.

Les estimateurs les plus connus sont :

- MFE (*Minimum Free Energy*), la structure d'énergie minimale. Le fait qu'elle soit d'énergie minimale n'indique ni sa fréquence dans l'ensemble de Boltzmann, ni si c'est cette structure qui est responsable de l'activité biologique. En revanche, elle est la structure la plus probable dans l'ensemble, et également la plus stable ;
- MEA [209] (*Maximum Expected Accuracy*), la structure qui maximise l'espérance de l'exactitude entre structure prédite et structure observée, en supposant qu'on pioche la structure observée aléatoirement dans l'ensemble de Boltzmann ;
- l'estimateur centroïde [96], la structure qui minimise le nombre maximal de différences avec toute structure de l'ensemble, en pondérant les différences par la probabilité d'existence de ces structures ;
- … et on peut en imaginer d'autres [140].

Un exemple est illustré sur la Figure 1.16. Comme on peut le voir dans l'exemple, aucun des estimateurs ne rend parfaitement compte de la diversité des topologies possibles pour la molécule. De ce fait, les outils de prédiction de structure secondaire modernes essaient toujours de renvoyer un ensemble de solutions même sous-optimales plutôt qu'une solution unique. Par exemple, RNAsubopt [360] renvoie toutes les structures dans un certain intervalle de distance énergétique à la structure MFE (relatif ou absolu).

I.3.6 L'optimisation multicritère et la diversité des solutions

Le chapitre II introduira plus en détail le champ de l'optimisation en mathématiques et notamment de l'optimisation multicritère. Simplement, j'insisterai ici sur l'un des intérêts de cette discipline dans le cadre de la prédiction de structures d'ARN.

La nature même des problèmes d'optimisation multicritères fait que bien souvent, il n'est pas possible d'identifier une solution unique et parfaite au problème. On liste un ensemble de solutions, l'ensemble de Pareto, dans lequel aucune des solutions ne domine les autres sur tous les critères à la fois. Le développement d'outils utilisant l'optimisation multicritère conduit donc naturellement à l'obtention de plusieurs solutions, et donc à un échantillonnage de l'ensemble de Boltzmann. Un défi de ce genre de travaux consiste donc à choisir les critères de façon que les solutions retrouvées dans l'ensemble de Pareto soient également fréquentes dans l'ensemble de Boltzmann. Il n'y a, en effet, aucune garantie là-dessus à priori pour des critères arbitraires.

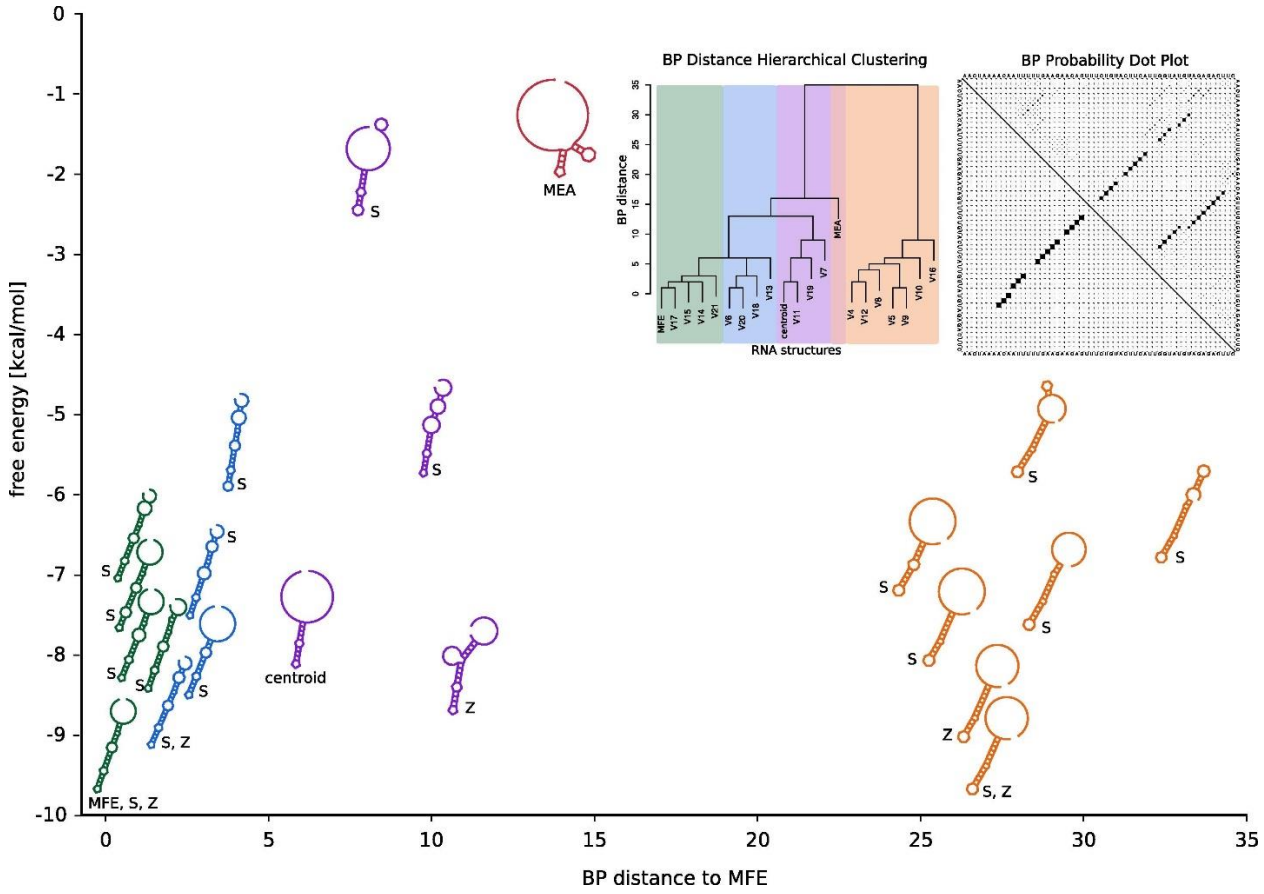


Figure 1.16 – Espace conformationnel des structures secondaires et leurs estimateurs. *La figure présente la structure MFE d'un ARN en bas à gauche, puis positionne quelques structures sous-optimales en fonction de leur nombre de différences à la MFE en abscisse et leur énergie en ordonnée. La figure fait également apparaitre l'estimation centroïde (en violet au centre) et l'estimation MEA (en rouge en haut).*
Figure extraite de la référence [205], sous licence CC-BY.

Par exemple, l'outil BiokoP [192] résolvant un problème d'optimisation bi-objectif (développé au laboratoire IBISC au début de ma thèse par Audrey Legendre) échantillonne des solutions sur un front de Pareto s'étendant entre les solutions MFE et MEA de l'espace des structures secondaires. Sur la Figure 1.16, l'ensemble de Pareto retourné par BiokoP s'étendrait en ligne droite entre les deux estimateurs. Ceci constitue donc une amélioration par rapport à la prédiction seule de l'un des deux estimateurs. En revanche, il n'y a aucune garantie que la fréquence associée à ces structures dans l'ensemble de Boltzmann soit élevée, c'est-à-dire aucune garantie que ces structures existent. En supposant que la structure native (biologiquement active ou d'intérêt) d'un ARN soit l'une des solutions sous-optimales les plus représentées dans l'ensemble de Boltzmann, elle peut aussi bien se trouver dans le front de Pareto ou non, et si elle y est, c'est uniquement par chance.

Un des points importants dans nos méthodes sera donc d'assurer une grande diversité des structures dans le front de Pareto, mais aussi un faible nombre de solutions, si l'on veut avoir une chance d'y trouver la structure native.

I.4 Méthodes de prédiction de la structure des ARN

I.4.1 Familles de méthodes

Il existe plusieurs axes sur lesquels on peut classifier les outils de prédiction de structure.

1) L'échelle : structure secondaire, tertiaire gros-grains (*coarse-grained*) ou tertiaire "all-atom" : Certaines méthodes ont pour objectif de prédire une structure secondaire, c'est-à-dire un graphe ou une liste d'appariements (canoniques ou non). D'autres vont un peu plus loin et proposent une forme du squelette en 3D, parfois avec l'orientation des bases. Ces modèles sont dits "gros grains", c'est-à-dire qu'ils ne modélisent pas tous les atomes, mais seulement un, deux, trois, et jusqu'à sept atomes ou pseudo-atomes pour modéliser un nucléotide. Ces modèles permettent de modéliser suffisamment bien la forme de l'ARN en réduisant drastiquement le temps de calcul nécessaire et sont donc largement utilisés. En général, une étape à la fin permet de reconstruire des modèles avec tous les atomes. Enfin, certaines méthodes permettent directement de prédire des structures avec tous les atomes, on les appelle *all-atom methods* en anglais.

2) La méthode d'évaluation (*scoring*) de la qualité d'une structure : Quelque soit l'échelle à laquelle une structure est modélisée, chaque méthode de prédiction utilise en interne une fonction attribuant un score, une note, une mesure de qualité aux structures possibles. C'est sur la base de ce critère que seront triées les solutions "les meilleures". L'expression mathématique de la méthode d'évaluation correspond donc à la fonction objectif à optimiser ou « critère » d'un problème d'optimisation, au sens défini au paragraphe précédent. D'ailleurs, l'un des objectifs de cette thèse est d'utiliser plusieurs critères à la fois, ce qui n'est pas courant, nous y reviendrons dans les chapitres suivants.

Tous les modèles d'évaluation utilisent des paramètres (statistiques, thermodynamiques, etc.). Ces paramètres sont soit dérivés de modèles physiques, obtenus par le calcul ou mesurés expérimentalement. On parle alors de méthode d'évaluation basée sur la théorie, ou basée sur la physique (*theory-based scoring, physics-based scoring*). On peut citer, par exemple, les règles de Turner [227], les champs de forces AMBER, CHARMM, GROMOS, OPLS, MMFF... Dans d'autres méthodes, ces paramètres sont évalués en estimant les valeurs qu'ils devraient prendre pour que le modèle fonctionne bien avec des données existantes par des méthodes Bayésiennes ou d'apprentissage supervisé. On parle alors de méthodes statistiques, d'évaluation basée sur les données ou sur la connaissance existante (*data-based scoring, knowledge-based scoring*, abrégé en *KB*). Ces méthodes rejoignent, d'un point de vue théorique, le champ de recherche de l'apprentissage automatique. On cite, par exemple, CONTRAFold [102], 3dRNAscore [350], NAST [166], ou la fonction d'énergie de Rosetta [7] utilisée dans l'outil FARFAR [85,355].

Enfin, de nombreux outils utilisent une fonction hybride qui utilise des paramètres thermodynamiques mesurés quand ils le peuvent, et utilisent des données pour estimer ceux

pour lesquels la mesure n'est pas disponible. On parle de méthodes hybrides, et cela regroupe une majorité d'outils dans la pratique.

3) La méthode d'échantillonnage (*sampling*) : En plus du module d'évaluation des structures, les méthodes de prédiction disposent d'un module de proposition de nouvelles structures à évaluer. C'est ce qu'on appelle l'échantillonnage. Il existe plusieurs grandes classes de méthodes d'échantillonnage.

- L'énumération exhaustive des structures possibles, fortement représentée dans les outils de prédiction de structure secondaire basés sur la programmation dynamique.
- La dynamique moléculaire (MD pour *Molecular Dynamics*), et ses variantes (*Replica-Exchange Molecular Dynamics*, *Langevin Dynamics*...), où à partir d'une structure initiale, le calcul des forces sur chaque atome permet d'utiliser la loi de Newton pour calculer sa vitesse. Le déplacement des atomes d'une itération sur l'autre génère de nouvelles structures. Ainsi, il est possible de simuler le temps.
- La résolution d'un problème d'optimisation. Souvent, mais pas toujours, la fonction à optimiser est celle proposée par le modèle d'évaluation, de façon à déterminer la ou les structures de plus basse énergie. Les itérations du programme d'optimisation permettent de proposer des structures à évaluer. La méthode d'optimisation varie d'un outil à l'autre, mais on peut lister rapidement : (1) des méthodes de descente de gradient, de gradient conjugué, de recherche locale, en particulier dans les phases de "raffinement" des solutions pour aboutir à un minimum local d'énergie, (2) des méthodes méta-heuristiques de type algorithme génétique (GA), recuit simulé, et méthodes de Monte-Carlo (MC) en général. Ces algorithmes sont en effet nécessaires dans de nombreux cas où le problème d'optimisation est trop peu linéaire, ou non continu/non dérivable. Les méthodes de Monte-Carlo bien construites permettent en plus d'isoler un échantillon de solutions dont les énergies suivent une distribution de Boltzmann. Les méthodes de Monte-Carlo nécessitent d'avoir inventé une façon de faire un "mouvement" ou un "petit pas" (en anglais un *move-set*) à partir d'une structure pour en obtenir une autre (changer un angle, un appariement...).

Enfin, on peut rassembler certaines méthodes dans la famille des méthodes d'assemblage de fragments (FARNA [84], FARFAR [85,355], MC-Sym [250], Vfold3D [51], RNA-Composer [261], 3dRNA [349,383]...). Néanmoins cette classification n'exclut pas les méthodes d'échantillonnage précédentes. En effet certaines méthodes vont suggérer tous les assemblages de fragments possibles, constituant alors une méthode d'échantillonnage proche de l'énumération exhaustive des structures. Mais d'autres proposent un assemblage de fragments qui sera évalué puis accepté ou rejeté selon l'algorithme de Metropolis. Si l'assemblage est rejeté, un autre est tenté à la place. Ce sont donc simplement des méthodes de Monte-Carlo dont le *move-set* est le remplacement d'un (ou plusieurs) fragments par un autre.

4) Les différents types de données en entrée (*inputs*) : Enfin, les méthodes diffèrent en fonction du support sur lequel elles vont faire leur prédiction. Les méthodes *comparatives* utilisent l'information d'homologie entre plusieurs séquences d'ARN (parfois alignées) pour

proposer une structure. À l'opposé, les méthodes *de-novo* proposent des structures à partir de la seule séquence.

Il existe aussi des méthodes prenant en entrée des données expérimentales, comme des résultats d'expériences de sondage chimique de la structure (SAXS, SHAPE, MAP, FRET, ...) ou même de cristallographie aux rayons X, résonnance magnétique nucléaire ou cryo-électro-microscopie, et qui proposent un modèle de structure cohérent. Cette thèse se concentre sur le problème de prédiction à partir de la séquence, on ignorera donc ici les méthodes reposant majoritairement sur les données expérimentales. Néanmoins, un grand nombre d'outils de prédiction *de-novo* proposent d'utiliser ces informations expérimentales, si elles existent, comme données auxiliaires réduisant l'espace de recherche conformationnelle et facilitant la prédiction.

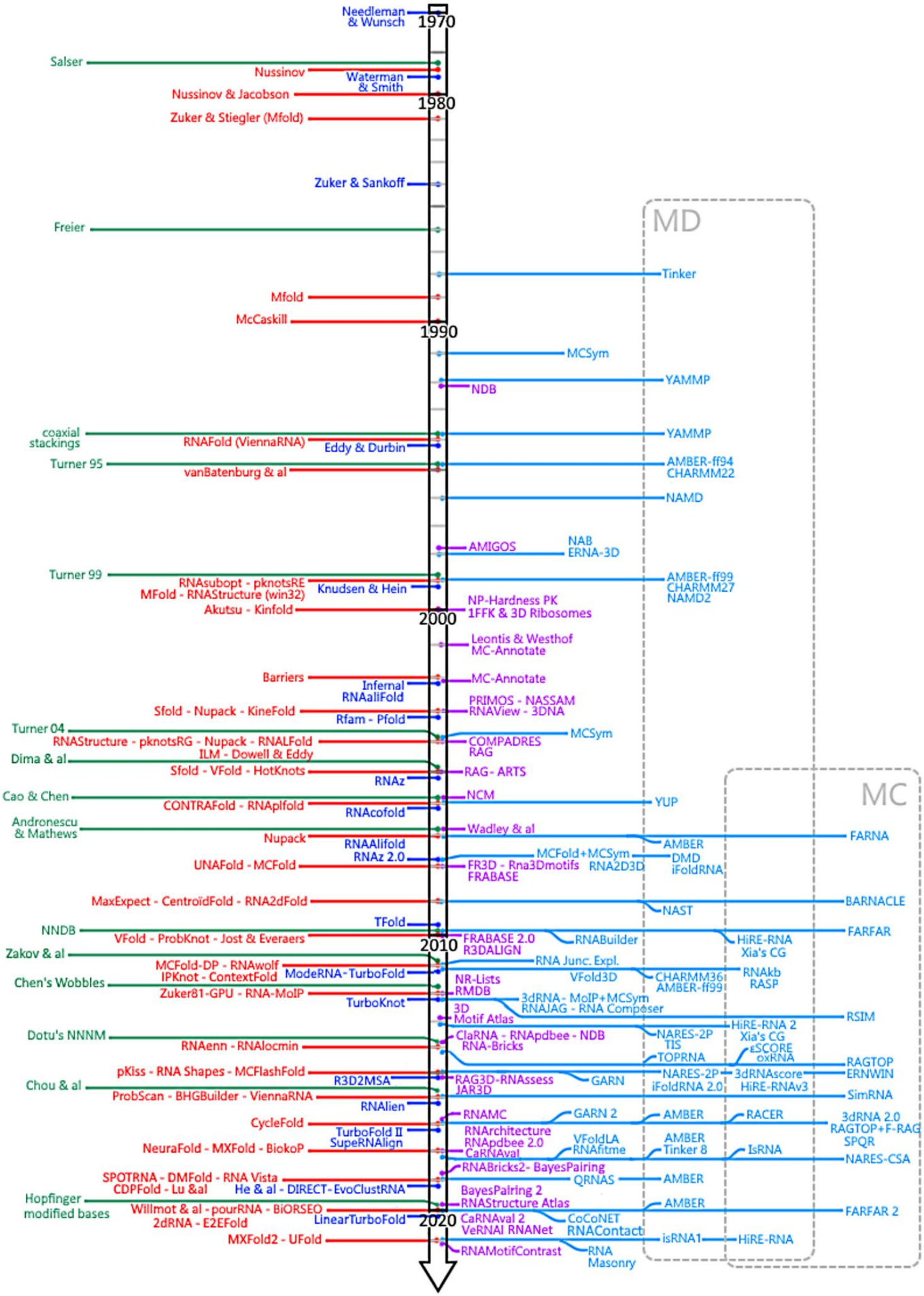
I.4.2 Historique commenté du champ de recherche

Dans cette partie, on listera les principaux travaux à connaitre, à cause de leur célébrité, performance, ou pertinence par rapport aux travaux présentés dans cette thèse. On les classera en plusieurs catégories de travaux :

1. Les avancées générales du champ de recherche, incluant les nouvelles idées de modélisation de l'ARN, les outils d'annotation, les bases de données publiques,
2. Les avancées proposant de nouveaux paramètres thermodynamiques ou statistiques,
3. Les méthodes d'alignement ou de prédiction de structure basées sur l'homologie entre de multiples ARN, en 2D comme en 3D. On les appelle méthodes comparatives.
4. Les méthodes de prédiction et outils d'analyse des structures secondaires (à partir d'une seule séquence),
5. Les méthodes de prédiction et outils d'analyse des structures 3D (à partir d'une séquence ou séquence et structure secondaire).

Cette liste n'est bien entendu pas exhaustive. Sur les trois premiers points, une sélection seulement est présentée. Sur les méthodes de prédiction, la majorité des travaux portés à notre connaissance est présentée, à l'exception de vieux algorithmes et outils des années 1980 et avant.

Figure 1.17 – Chronologie du champ de recherche. *Outils, logiciels, et modèles pertinents pour étudier la structure de l'ARN par ordinateur organisés par date approximative de publication. En vert les principaux travaux relatifs aux paramètres thermodynamiques. En rouge des méthodes de prédiction de structure secondaire, ou d'étude du paysage énergétique des structures secondaires. En bleu foncé des méthodes comparatives et méthodes d'alignement de séquences ou de structures (fortement non exhaustif). En violet, des travaux d'annotation et de modélisation descriptive de l'ARN et de ses modules, et d'autres bases de données et événements d'intérêt. En bleu clair, des méthodes de prédiction, de simulation, ou d'étude des structures 3D et de leur paysage énergétique. Certains noms apparaissent plusieurs fois lorsqu'un outil ou base de données subit une mise à jour d'importance.*


1970
1980
1990
2000
2010
2020
MD
MC
Needleman & Wunsch
Salser
Nussinov
Waterman & Smith
Nussinov & Jacobson
Zuker & Stiegler (Mfold)
Zuker & Sankoff
Freier
Tinker
Mfold
McCaskill
MCSym
YAMMP
NDB
coaxial stackings
RNAFold (ViennaRNA)
Eddy & Durbin
Turner 95
vanBatenburg & al
AMBER-ff94
CHARMM22
NAMD
AMIGOS
NAB
ERNA-3D
Turner 99
RNAsubopt - pknotsRE
MFold - RNAStructure (win32)
Knudsen & Hein
AMBER-ff99
CHARMM27
NAMD2
Akutsu - Kinfold
NP-Hardness PK
1FFK & 3D Ribosomes
Leontis & Westhof
MC-Annotate
Barriers
Infernal
RNAaliFold
MC-Annotate
Sfold - Nupack - KineFold
Rfam - Pfold
PRIMOS - NASSAM
RNAView - 3DNA
Turner 04
RNAStructure - pknotsRG - Nupack - RNALFold
ILM - Dowell & Eddy
MCSym
COMPADRES
RAG
Dima & al
Sfold - VFold - HotKnots
RNAz
RAG- ARTS
Cao & Chen
CONTRAFold - RNAplfold
RNAcofold
NCM
YUP
Andronescu & Mathews
Nupack
RNAAlifold
Wadley & al
AMBER
FARNA
RNAz 2.0
MCFold+MCSym
RNA2D3D
DMD
iFoldRNA
UNAFold - MCFold
FR3D - Rna3Dmotifs
FRABASE
MaxExpect - CentroïdFold - RNA2dFold
NAST
BARNACLE
TFold
NNDB
VFold - ProbKnot - Jost & Everaers
FRABASE 2.0
RNABuilder
HiRE-RNA
Xia's CG
FARFAR
Zakov & al
R3DALIGN
RNA Junc. Expl.
MCFold-DP - RNAwolf
IPKnot - ContextFold
ModeRNA - TurboFold
VFold3D
CHARMM36
AMBER-ff99
RNAkb
RASP
Chen's Wobbles
Zuker81-GPU - RNA-MoIP
TurboKnot
NR-Lists
RMDB
3dRNA - MoIP+MCSym
RNAJAG - RNA Composer
3D Motif Atlas
RSIM
Dotu's NNNM
RNAenn - RNAlocmin
ClaRNA - RNApdbee - NDB
RNA-Bricks
NARES-2P
TIS
TOPRNA
HiRE-RNA 2
Xia's CG
eSCORE
oxRNA
RAGTOP
ERNWIN
pKiss - RNA Shapes - MCFlashFold
R3D2MSA
RAG3D-RNAssess
JAR3D
GARN
NARES-2P
iFoldRNA 2.0
3dRNAscore
HiRE-RNAv3
Chou & al
ProbScan - BHGBuilder - ViennaRNA
RNAlien
SimRNA
CycleFold
TurboFold II
SupeRNAlign
RNAMC
GARN 2
AMBER
RACER
3dRNA 2.0
RAGTOP+F-RAG
SPQR
NeuraFold - MXFold - BiokoP
RNArchitecture
RNApdbee 2.0
CaRNAval
VFoldLA
RNAfitme
AMBER
Tinker 8
IsRNA
NARES-CSA
RNABricks2- BayesPairing
SPOTRNA - DMFold - RNA Vista
CDPFold - Lu &al
He & al - DIRECT - EvoClustRNA
QRNAS
AMBER
BayesPairing 2
RNAStructure Atlas
Hopfinger
modified bases
Willmot & al - pourRNA - BiORSEO
2dRNA - E2EFold
LinearTurboFold
AMBER
FARFAR 2
CaRNAval 2
VeRNAl RNANet
CoCoNET
RNAContact
isRNA1
HiRE-RNA
MXFold2 - UFold
RNAMotifContrast
RNA Masonry

L'approche de présentation choisie est l'approche chronologique. En effet, de nombreux outils sont mis à jour à quelques années d'intervalle au fur et à mesure de la progression des travaux de leurs auteurs. Ils ne correspondent donc pas aux mêmes méthodes selon l'année de publication de l'article où l'on trouve leur nom. Un grand nombre d'articles de synthèse des méthodes (par exemple les références [278,86,79]) réalisent des catalogues sans prendre en compte l'aspect temporel, et dans mon expérience, ceci complique la compréhension en introduisant des contradictions lorsqu'on compare des synthèses d'années de publication différentes. Par exemple, pour répondre à des questions comme "*Est-ce que RNAsubopt supporte les contraintes externes ?*", "*Quel est le move-set de l'algorithme FARNA ?"* ou "*Est-ce qu'iFoldRNA rend bien des modèles all-atom ?*" : cela dépend de l'année à laquelle on se place. Il est également intéressant de voir comment les travaux d'un auteur influencent ceux des autres, et de voir se répandre les nouvelles idées dans les outils. Une vue générale de la chronologie est proposée sur la Figure 1.17, mais chaque modèle sera brièvement présenté avec sa référence dans la suite de cette partie.

Dans les chronologies qui suivent, certains auteurs sont sélectionnés et listés parmi les collaborateurs d'un travail sur la base de la célébrité subjective de leur nom dans le monde de la bioinformatique des ARN. Ceci a pour objectif de faciliter le suivi mental de la continuité du travail au sein des groupes de recherche, mais ne donne aucune information sur les attributions. Mes excuses aux collaborateurs non directement cités.

I.4.2.1 Avancées générales, bases de données, outils de description et d'annotation, et nouvelles idées de modélisation de l'ARN et de ses motifs

1992 – La Nucleic Acid Database [31] répertorie les structures 3D d'ARN existantes.

1998 – Duarte & Pyle proposent le modèle de pseudo-liaisons P-C_4' et les angles de torsion associés η/θ pour modéliser la structure du squelette, ainsi que le logiciel d'annotation AMIGOS pour les calculer [106]. Les premiers "Pyle plots" sont publiés, avec une base de données de 52 ARN existants en 3D (résolus par RMN ou XRC).

Entre 1999 et 2000 – Résolution successive des structures 3D des ribosomes eucaryote et procaryote, sous-unité par sous-unité, puis finalement en une fois, par cristallographie aux rayons X. Notamment, la structure 1FFK [23], qui restera une des références de la sous-unité 70S avec une résolution de 2.4 Angströms. Ce sont les premières structures 3D de gros ARN non codants et structurés disponibles. Les structures hautes résolution du ribosome procaryote seront d'ailleurs récompensées, avec d'autres travaux sur les ribosomes, par le prix Nobel de Chimie en 2009.

2000 – Le problème d'identification de la structure secondaire d'énergie minimale avec pseudonoeuds arbitraires est prouvé comme étant NP-difficile [6].

2001 – Leontis & Westhof utilisent les récentes structures 3D d'ARN ribosomaux pour

proposer leur classification des appariements [197]. Ils étudient aussi leurs isostéricités respectives [196].

Gendron, Lemieux et Major et collaborateurs développent MC-Annotate [132] pour annoter les structures 3D, et repérer les conformations des nucléotides.

2002 – Lemieux et Major et collaborateurs mettent à jour MC-Annotate [195] avec un algorithme probabiliste.

2003 – Wadley et Pyle utilisent PRIMOS pour rechercher des motifs conformationnels dans les structures d'ARN (sur la base des angles eta/theta) [107].

NASSAM [143] propose aussi de faire de la recherche de motifs, en prenant en compte les bases , pas seulement la conformation du squelette.

Leontis & Westhof proposent le programme d'annotation RNAView [367] pour annoter (et aussi visualiser) les structures 3D en 2D,

Xiang-Jun Lu propose 3DNA dans le même but [207]. L'outil 3DNA peut annoter et reconstruire une structure à partir de son annotation quasi-parfaitement.

Tamar Schlick propose sa modélisation des structures secondaires par le graphe dual du graphe habituel, et l'utilise pour estimer l'étendue de l'espace conformationnel possible des structures d'ARN [128]. Elle prouve qu'il est bien plus facile à explorer que l'espace des séquences.

2004 - L'équipe de Schlick propose la base de données RAG (RNAs As Graphs) [127]. Elle décrit toutes les topologies possibles des structures d'ARN de deux à sept SSE, et liste celles découvertes dans la nature, s'il y en a.

Wadley et Pyle proposent COMPADRES [343] qui permet d'automatiser la découverte de nouveaux motifs dans un jeu de données.

2005 – ARTS [105] permet de superposer deux structures d'ARN en 3D, ce qui permet de les comparer.

2006 – Lemieux & Major [194] proposent d'utiliser les NCM comme briques de base pour l'étude des structures 3D. Ils extraient et regroupent les NCM de taille identique selon une mesure de distance corrélée au RMSD.

2007 – Wadley, Keating, Duarte et Pyle [342] publient leurs « Pyle plots » (l'équivalent du Ramachandran plot) pour l'ARN avec les angles η/θ.

2008 – Sarver, Zirbel, Leontis proposent FR3D [295], qui est à la fois un outil d'annotation de structures et un outil de recherche de motifs (symbolique, géométrique, ou les deux).

Djelloul & Denise [101] proposent leur modèle de motifs et le jeu de données RNA3DMotifs.

Popenda, Adamiak et collaborateurs proposent la base de données FRABASE [260], qui annote les structures 3D disponibles avec RNAView (appariements, *sugar-puckers*, angles de torsion...) et dispose d'un moteur de recherche sur la base de la séquence ou des

propriétés annotées.

Le « RNA Ontology Consortium » propose une nomenclature pour les conformations du squelette du nucléotide, via l'identification de clusters [277].

Le logiciel VARNA [81] est proposé par Darty, Denise & Ponty, il permet de visualiser les structures secondaires d'ARN (y compris étendues) selon différentes représentations sous forme de graphe. C'est une alternative à RNAview, rapidement adoptée par la communauté.

2010 – Popenda, Adamiak et collaborateurs mettent à jour RNA FRABASE 2.0 [262], avec de nouvelles annotations venant de 3DNA [207] et un moteur de recherche de fragments en 3D.

2012 - Pyle propose un jeu de données de nucléotides préparés avec différentes valeurs des pseudotorsions η/θ [158].

Zirbel et collaborateurs proposent R3D Align [268] (supposément supérieur à ARTS), pour superposer deux larges structures 3D sur la base de sous-alignements locaux et prenant en compte la flexibilité locale des structures.

Leontis et Zirbel proposent des listes d'ARN non redondants [198]. Les ARN sont regroupés par classes d'équivalence, et un représentant de chaque classe d'équivalence est proposé (en fonction de la qualité de la structure 3D). Ces listes sont mises à jour de façon hebdomadaire depuis, et sont devenues une référence pour éviter la redondance des jeux de données de structures 3D. Néanmoins, ces listes n'incluent pas de structures obtenues par RMN ou cryo-électro-microscopie, elles ne représentent donc pas l'intégralité des données disponibles.

La base de données RMDB [67] répertorie les résultats de sondage chimique publics.

La première expérience communautaire RNA-Puzzles [74] a lieu, dans l'esprit de l'expérience CASP (Critical Assessment of Structure Predictions) organisée pour la prédiction des structures de protéines. Des structures 3D nouvellement résolues expérimentalement mais non encore publiées sont tenues secrètes, et on propose aux groupes de recherche d'effectuer des prédictions de structure 3D dans un temps limité.

2013 - Le BGSU RNA Group propose le RNA 3D Motif Atlas [255], sur la base de FR3D.
RNAlyzer [210] propose différents outils automatisés pour comparer des modèles 3D entre eux, par exemple une structure native et des prédictions.

2014 – Bujnicki et collaborateurs [344] proposent ClaRNA, une méthode d'annotation permettant de trouver un consensus entre les annotations des divers autres outils que sont FR3D, MC-Annotate, et RNAView. Les mêmes proposent RNABricks [57], base de données de modules qui garde les modules dans leur contexte d'interaction avec d'autres constituants.

Antczak, Adamiak et collaborateurs proposent le serveur RNApdbee [20], qui permet d'annoter et visualiser les structures à partir des outils des autres, en supportant les pseudonoeuds.

La Nucleic Acid Database [65] est mise à jour, elle inclut maintenant les outils du BGSU RNA Group : des annotations des structures par les modules du RNA 3D Motif Atlas, les

classes d'équivalence de BGSU, la possibilité d'aligner des structures avec R3D-Align, ou de rechercher des modules avec WebFR3D. Une visualisation de structure secondaire par RNAView et 3D par 3DNA est aussi automatiquement proposée.

2015 – RNAssess [211] remplace RNAlyzer pour la comparaison de modèles 3D entre eux.

JAR3D [388] modélise les distributions de probabilité des séquences pour chaque module du 3D Motif Atlas. Ceci permet de détecter, à partir de la séquence de chaque boucle d'un ARN, les modules les plus probables qu'elle pourrait former.

Schlick améliore le catalogue de topologies RAG en RAG-3D [372], qui contient maintenant une collection de « graphes duaux » en 3D, les nœuds étant placés dans l'espace au centre des boucles, et les arêtes étant de taille bien définie et dans l'axe des hélices. Ces graphes décrivent la topologie globale de la molécule d'ARN. Un outil de recherche de sous-structures est proposé, sur la base d'un algorithme de recherche de sous-graphes.

Le groupe de Vienne propose un nouvel outil de visualisation de structures intégrable dans une page web disposant de fonctions de prédiction et d'annotation, FORNA[174].

2018 – RNArchitecture [40] est une base de données de familles d'ARN groupés par similarité de structure 3D.

Adamiak et collaborateurs mettent à jour RNApdbee 2.0 [390] pour supporter les interactions non canoniques dans les structures secondaires proposées (et les visualisations) et un nouvel algorithme d'annotation des pseudonoeuds.

2019 – BayesPairing [293] transforme les modules en réseaux bayésiens et utilise ces réseaux pour rechercher les modules dans les séquences.

2020 - BayesPairing 2 [294] est une grosse amélioration de performance par rapport au premier, la détection utilise maintenant un échantillon stochastique de structures secondaires de l'ARN cible, et éventuellement un alignement de séquences si fourni.

Le BGSU RNA group propose une nouvelle base de données en ligne intégrant tous leurs outils, le RNA Structure Atlas (non publié). Elle permet la visualisation, la recherche, et la navigation entre les structures, les motifs, et les outils du groupe.

RNANet [27], mon jeu de données intégrant des descripteurs des structures 3D à toutes les échelles ainsi que des données d'homologie est publié et partagé. Il est longuement présenté dans la section IV.2.

Un autre jeu de données [265] de structures 3D lié à des familles Rfam est également développé.

I.4.2.2 Avancées dans la mesure ou l'estimation des paramètres énergétiques de l'ARN

1977 – Les paramètres énergétiques de Salser comprennent les énergies d'empilement et des pénalités spécifiques pour certaines boucles connues [291].

1986 – Freier *et al.* (groupe de Turner) proposent de nouveaux paramètres thermodynamiques pour les hélices et, nouveauté, pour les extrémités pendouillantes [124].

1994 – Walter, Turner, Mathews et Zuker proposent des paramètres thermodynamiques pour les empilements coaxiaux d'hélices [347].

1995 – Serra & Turner [304] récapitulent les paramètres thermodynamiques de plusieurs publications et détaillent une méthode de calcul de l'énergie pour chaque SSE (en prenant en compte l'asymétrie des boucles, leur longueur, etc...).

1997-1999 – Mathews, Turner, Zuker ajustent les paramètres thermodynamiques pour les rendre plus dépendants de la séquence de chaque SSE [225,227]. La fonction d'évaluation de l'énergie résultante devient connue sous le nom de *"Turner rules"*.

2004 – Un modèle dit de « Turner 2004 » reparamétrise certaines valeurs à partir d'un ensemble de publications récentes.

2005 – Dima *et al.* [94] utilisent ViennaRNA avec un potentiel statistique basé sur les empilements et montrent que les performances sont équivalentes aux potentiels thermodynamiques.

2006 - Cao & Chen [50] proposent un modèle thermodynamique des valeurs d'entropie pour les pseudonoeuds, calculées selon le modèle Vfold (voir en I.4.2.4 en 2005). Ils proposent de les utiliser au lieu des estimations assez aléatoires d'autres auteurs.

2007 - Andronescu, Mathews et autres collaborateurs publient une façon automatisée d'apprendre les paramètres thermodynamiques utiles au modèle de Turner 99 en les estimant sur un jeu de données [15,16]. La méthode permet d'utiliser aussi les mesures thermodynamiques existantes, et résout itérativement un problème d'optimisation pour converger vers une solution. Ils commentent sur le récent outil CONTRAFold.

2008 – Sharma *et al.* [306] proposent une étude de QM et déterminent des paramètres théoriques pour les appariements trans-Watson-Watson.

2010 - Mathews lance la NNDB [339] (Nearest-Neighbour Data Base), une base de données répertoriant des jeux de paramètres thermodynamiques. L'idée était pertinente, mais la base n'est malheureusement pas mise à jour depuis, ce qui perd de son intérêt.

2011 – Zakov *et al.* [373] proposent d'utiliser un modèle du plus proche voisin très étendu (à de lointains voisins) et dont les paramètres sont dépendants des séquences. Ils proposent des modèles de complexité croissante, de 226 à 70 000 paramètres, estimés sur la base de structures RNAStrand [14] par un algorithme d'apprentissage dit « passif-agressif ».

2012 – Chen *et al.* [55] étudient en détail les appariements « Wobble », c'est-à-dire G-U cWW. Ils mesurent expérimentalement des paramètres pour le modèle du plus proche voisin.

2014 – Dotu *et al.* [103] proposent un modèle du « plus proche voisin et le suivant » (*next-*

nearest neighbour model) où les énergies des éléments de structure dépendent de la séquence jusqu'à un nucléotide plus loin (en considérant par exemple les triplets de nucléotides empilés pour les hélices).

2016 – L'équipe de Das montre que leur fonction d'évaluation RECCES-Rosetta [59], qui prend en compte une évaluation de l'entropie, est capable de retrouver avec une bonne précision les paramètres thermodynamiques du modèle du plus proche voisin. Ils réalisent également des prédictions de paramètres non encore mesurés et valident l'approche en mesurant certains d'entre eux. Les prédictions sont bonnes.

2018 – Zuber *et al.* [392] mesurent l'impact d'erreurs dans les expériences de dissociation de l'ARN sur les erreurs dans les paramètres thermodynamiques, et sur les erreurs dans les prédictions de structure secondaire obtenues au final.

2020 – Hopfinger *et al.* [155] proposent une étude de QM et des paramètres pour les nucléotides modifiés.

Sakuraba *et al.* [290] utilisent la FEP (*Free Energy Perturbation*) pour calculer les paramètres pour l'inosine (un nucléotide assez rare). Le protocole pourrait être appliqué à d'autres nucléotides modifiés.

I.4.2.3 Méthodes comparatives de prédiction de structure ou d'alignement de séquences et de structures

Cette liste d'outils est loin d'être exhaustive, mais permet d'avoir quelques repères temporels à propos des outils de référence.

1970 et 1978 : Needleman & Wunsch puis Waterman & Smith appliquent la programmation dynamique à l'alignement de séquences. Ceci inspirera les algorithmes d'énumération de structures (voir référence [395]).

1984 – Zuker & Sankoff : Un algorithme par programmation dynamique est proposé pour fonctionner à partir de séquences homologues en entrée [395]. Les séquences sont à la fois alignées et utilisées pour la prédiction de structure secondaire.

1994 – Eddy & Durbin adaptent les HMM (*Hidden Markov Models* ou modèles de Markov cachés) et CM (*Covariance Models* pour modèles de covariance) de leur outil Hmmer (hmmer.org) pour l'ARN, permettant de proposer une structure secondaire "consensus" pour des séquences homologues entre elles [109]. L'implémentation s'appelle COVE.

1999 – Knudsen & Hein utilisent une SCFG entrainée sur des séquences homologues pour apprendre une distribution de probabilité postérieure de Bayes des structures secondaires possibles [183]. L'outil permet de prédire la structure commune aux séquences d'un alignement, de façon supposément supérieure aux HMM qui prendraient mal en compte les interactions longue-distance.

2002 – Eddy propose Infernal [108] pour aligner des séquences d'ARN homologues en prenant en compte leur CM, et donc la structure secondaire supposée. Cette implémentation remplace COVE, et embarque un deuxième algorithme utilisable en option, basé sur CYK (aussi connu sous le nom d'algorithme de Viterbi, avec une faible consommation de mémoire, mais une plus grosse complexité temporelle).

RNAalifold [152] rejoint le ViennaRNA package pour produire une structure consensus à partir de séquences homologues en calculant la MFE possible pour toutes les séquences.

Tahi *et al.* proposent DCFold [328], un outil déterminant une structure secondaire consensus à partir d'un ensemble de séquences alignées, basé sur la recherche de palindromes conservés dans ces séquences.

2003 - Griffith-Jones et Eddy créent Rfam [138], la base de données classifiant les ARN non codants en "familles", et proposant pour chacune des alignements, CM, et structure secondaire consensus.

Knudsen & Hein améliorent leur SCFG et sortent l'outil Pfold [182].

Tahi *et al.* améliorent DCFold en P-DCFold [327], qui supporte maintenant les pseudonoeuds simples.

2005 – RNAz [353] prédit des structures à partir d'alignements en prenant en compte à la fois l'énergie et la covariation, et en renvoyant un score séparé pour chaque (énergie + Z-score).

2006 – RNAcofold [36] prédit la structure secondaire de dimères d'ARN.

2008 - RNAz 2.0 [139] s'améliore (prédiction de structure secondaire consensus sur des bases de covariation structurale et de stabilité thermodynamique).

RNAalifold [35] est amélioré et utilise des matrices RIBOSUM-like pour modéliser les mutations de transition entre nucléotides.

2010 – Engelen et Tahi proposent TFold [115], un outil de prédiction de structure secondaire pour une séquence donnée, mais alignée avec d'autres également fournies en entrée. TFold recherche la position des hélices stables, sur la même base algorithmique que P-DCFold.

2011 – TurboFold [142] propose une structure secondaire consensus pour des séquences homologues, sur la base des probabilités d'appariement dans chaque séquence individuelle.

ModeRNA [285] permet de faire de la modélisation comparative de structure 3D. À partir d'un patron 3D et d'un alignement des séquences du patron et de l'ARN à modéliser, un modèle 3D peut être proposé en remplaçant les bases.

2012 – TurboKnot [301] propose des structures consensus pour un ensemble de séquences, avec support des pseudonœuds.

2015 - R3D-2-MSA [177] permet de transformer des alignements structuraux par R3D-Align en alignements de séquences.

2016 – RNAlien [110] permet la construction automatique de familles d'ARN homologues et éventuellement d'alignements à partir d'une seule séquence. Il repose sur les outils d'Infernal, RNAz, et d'autres.

2017 - TurboFold II [333] utilise un HMM pour prédire une structure secondaire consensus sur la base de critères internes à chaque séquence (MEA/MFE) et externes (probabilités, covariation).

SupeRNAlign [256] superpose des structures d'ARN en 3D pour obtenir des alignements de séquences basés sur l'alignement des structures 3D.

2019 – EvoClustRNA [215] réalise des modèles 3D pour chaque séquence d'un alignement (via FARFAR ou SimRNA) et retient une structure 3D consensus (centroïde du cluster).

DIRECT [164] améliore la méthode classique d'analyse des covariations (DCA) avec une Machine de Boltzmann Restreinte (RBM) et des données 3D. Le modèle peut ainsi prédire les contacts nucléotidiques à la fois sur la base de séquences homologues et de structures homologues. Leur jeu de données est fabriqué à la main à partir de familles Rfam et de structures 3D associées.

He & Xiao [146] proposent un réseau de convolution pour faire de l'analyse des covariations.

2020 - L'algorithme de LinearFold (voir section I.4.2.4 en 2019) est adapté pour TurboFold II, ce qui aboutit à LinearTurboFold [199].

I.4.2.4 Méthodes de prédiction de structure secondaire à partir d'une séquence, ou d'étude de leur paysage énergétique

1977 – La thèse de Ruth Nussinov introduit un algorithme de prédiction de structures secondaires par programmation dynamique. Chaque appariement compte pour un score de +1, et le programme renvoie donc la structure avec le plus d'appariements.

1980 – Algorithme de Nussinov-Jacobson [248] : cette version utilise l'énergie des appariements à la place des scores de +1.

1981 – Zuker & Stiegler proposent Mfold et le modèle du plus proche voisin (prise en compte des appariements, empilements et pénalités spécifiques des boucles) avec une complexité algorithmique de $O(n^3)$ [396].

1989 – Mfold renvoie des solutions sous-optimales en plus de la MFE, mais pas toutes (élimination volontaire des solutions trop proches) [394].

1990 – McCaskill adapte l'algorithme au calcul des probabilités d'appariement [229].

1994 – Hofacker propose RNAfold, une ré-implémentation "moderne" de Mfold, parallélisée et plus rapide [153]. Le package ViennaRNA naissant contient aussi de nouveaux outils : RNAinverse et RNAdistance.

1995 - Van Batenburg propose un algorithme génétique pour prédire les pseudonoeuds [24].

1997-1999 – Mathews, Turner, et Zuker améliorent techniquement Mfold [225,227] dans une série de publications, et prennent en compte les paramètres thermodynamiques les plus récents.

1999 – Wutchy & Fontana proposent une méthode d'énumération exhaustive des structures secondaires sous optimales entre la MFE et une limite donnée, et l'implémentent dans l'outil RNAsubopt [360].

Rivas & Eddy proposent un schéma de programmation dynamique qui supporte les pseudonoeuds simples [279]. Le même article introduit les "diagrammes de Feynman" de structure secondaire pour décrire les schémas de programmation dynamique. L'outil correspondant est pKnots (parfois appelé pKnotsRE).

Mathews propose encore une nouvelle implémentation, RNAstructure, identique à Mfold mais conçue pour Windows 32 bits (Mfold est pour Unix) [227].

2000 – Akutsu propose un autre schéma de programmation dynamique avec pseudonoeuds (sans avoir connaissance de celui de Rivas, non cité) [6]. L'article contient aussi une preuve de difficulté NP du problème d'énumération des structures avec pseudonoeuds arbitraires.

Hofacker et Schuster proposent Kinfold [119], une simulation de Monte-Carlo qui renvoie des trajectoires de formation des structures secondaires. Cet outil ne doit pas être confondu avec KineFold [361], sorti plus tard pour faire la même chose.

2002 - Hofacker et Stadler proposent Barriers [120]. À partir d'une liste de conformations (structures secondaires), le programme calcule les minima locaux et les barrières énergétiques pour passer des uns aux autres. Il produit des images du paysage énergétique sous forme d'arbre.

2003 - Dirks & Pierce créent un troisième schéma de programmation dynamique supportant les pseudonoeuds simples [99]. Il n'apporte rien de nouveau en soi par rapport à Pknots, mais l'année suivante, ils proposent également l'adaptation de l'algorithme de McCaskill pour obtenir les probabilités d'appariements avec pseudonoeuds, ce que leur outil Nupack [100] est le seul à proposer (à l'époque).

Ding & Lawrence proposent une méthode d'échantillonnage de solutions de structure secondaire sous-optimales suivant une distribution de Boltzmann [97], toujours par programmation dynamique. Le programme, Sfold, calcule aussi une *Boltzmann-weighted density-of-states*, c'est-à-dire la distribution des valeurs d'énergie dans une population. Un webservice prévu pour le design est proposé l'année suivante.

KineFold [361] (à ne pas confondre avec Kinfold[119], bien que les outils soient

similaires) est un outil de prédiction de structures secondaires avec pseudonoeuds utilisant un échantillonnage de Monte-Carlo.

2004 – Reeder & Giegerich [271] proposent un nouveau schéma de programmation dynamique pour une classe simple de pseudonoeuds, de complexité plus faible que celui de Rivas & Eddy. Ils le nomment pknotsRG (pour Reeder & Giegerich) par opposition au pknotsRE (Rivas & Eddy).

Mathews modifie RNAstructure (le programme *fold*) avec une mise à jour de paramètres et de modélisation des empilements coaxiaux d'hélices [226]. Il rajoute aussi le support des nucléotides modifiés, et réimplémente l'algorithme de McCaskill dans son logiciel pour donner des intervalles de confiance sur chaque appariement de la MFE renvoyée.

Hofacker propose RNALfold [154], une adaptation de l'algorithme de Zuker qui calcule des structures secondaires locales dans les longs ARN, avec une contrainte de distance maximale sur la portée des appariements possibles pour un nucléotide donné.

ILM [286] propose de prédire des structures avec pseudonoeuds en résolvant plusieurs fois le schéma de programmation dynamique sans support de pseudonoeuds, mais en retirant de la séquence d'une récursion à l'autre les nucléotides détectés comme appariés avec un très bon score.

Dowell & Eddy [104] proposent un article pédagogique, une synthèse et un benchmark à propos des méthodes utilisant les SCFG, indispensable à la compréhension de ces outils. L'étude confirme la qualité de l'outil Pfold de Knudsen & Hein [182]. L'article fait le lien entre SCFG et programmation dynamique, et le lien entre SCFG et CM.

2005 – Ding & Lawrence mettent à jour Sfold : l'ensemble de Boltzmann est clustérisé en 3.2 clusters en moyenne et des centroïdes sont retournés en tant que structures représentatives.

Cao & Chen proposent Vfold [49], leur modèle d'estimation de l'entropie des structures, qui sera amélioré au fil des années pour devenir l'un des plus performants à ce jour (selon le benchmark RNA-Puzzles [239]). Les enthalpies sont estimées comme la somme des enthalpies des empilements. Les boucles ont donc une enthalpie nulle. Les enthalpies et entropies des hélices sont celles du modèle de Turner 1995. Pour estimer les entropies des boucles, le programme énumère les conformations possibles d'un segment du bon nombre de nucléotides. Le fragment de chaine est modélisé par les pseudo-liaisons P-C4', et les configurations énumérées en 3D sur une grille en diamant. La contribution entropique de la boucle est calculée comme une fonction du rapport entre le nombre de conformations possibles pour la boucle et le nombre de conformation possibles pour le fragment sans contrainte. Le calcul d'une MFE se fait en calculant les sous-structures optimales en ajoutant des nucléotides progressivement, de 3' vers 5'. Pour les auteurs, le « VFold model » décrit l'utilisation des pseudo-liaisons P-C4' (ce qu'ils ne sont pas les seuls à faire pourtant). Il me semble cependant que pour la communauté, le terme « VFold model » désigne plutôt la prise en compte de ces contributions entropiques par énumération des conformations sur une grille.

HotKnots [275] utilise une heuristique pour proposer des structures secondaires avec

des pseudonoeuds de complexité arbitraire.

2006 - Do *et al.* proposent l'estimateur MEA dans un outil basé sur une SCFG, CONTRAFold [102]. C'est un modèle purement entrainé sur des données et n'utilisant aucun paramètre thermodynamique (nouveau). De plus, la méthode prend un paramètre qui permet d'ajuster le niveau de sensibilité/spécificité. CONTRAFold fait un bond en performance (selon son propre article) par rapport aux algorithmes existants (notamment grâce à l'estimateur MEA).

Bernhart, Hofacker & Stadler proposent RNAplfold [34], un outil prédisant des appariements locaux seulement mais supportant de très longs ARN. L'outil est ajouté à ViennaRNA.

2007 - Dirks & Pierce [98] proposent une adaptation de leur modèle pour les complexes d'ARN dans Nupack.

2008 - Au Mathews lab, UNAFold [221] succède à Mfold et combine en un seul exécutable la prédiction d'une ou plusieurs structures secondaires, le calcul des probabilités d'appariement, et les calculs de température de dissociation des appariements.

2009 – En plus de l'estimateur MFE, l'estimateur MEA proposé par CONTRAFold en 2006 est réimplémenté par le groupe de Mathews dans MaxExpect [209], et d'autres estimateurs centroïdes sont proposés par Sato et Asai [140]. Cette dernière référence permet de bien saisir les enjeux de ce que représente une estimation de structure secondaire parmi une population.

RNA2DFold [203] permet de visualiser le paysage énergétique et les structures secondaires voisines ainsi que leurs énergies, autour de structures secondaires de référence passées en entrée. Des visualisations 2D du paysage sont proposées.

2010 – ProbKnot [29] propose des structures secondaires avec pseudonoeuds identifiés en comparant les structures MEA sous optimales. L'idée est intéressante, mais la performance mauvaise (~60% d'exactitude, ou *accuracy*).

Le modèle VFold [200] s'étend à tout type de boucle comprise entre deux hélices, et au support des pseudonoeuds de type H. L'article insiste aussi sur les approximations faites par le modèle ; notamment, les contributions entropiques des boucles peuvent être additionnées car on néglige l'effet de volume exclu (faible) qu'elles pourraient avoir les unes sur les autres.

Jost et Everaers [167] proposent un modèle de prédiction de structure secondaire énumérant les structures en 3D gros grains (1 bille par nucléotide) sur une grille discrète, de manière similaire à VFold, pour estimer l'entropie. Il supporte les pseudonoeuds. Les auteurs critiquent au passage les modèles de VFold, Kinefold, Nupack, et pknotsRG.

2011 – Zu Siederdissen et collaborateurs proposent MC-Fold-DP [309], une reformulation du modèle MC-Fold de Major en utilisant une SCFG et un schéma de programmation dynamique, n'arrivant malheureusement pas à reproduire les résultats de l'original. Les

mêmes proposent également RNAwolf [309], qui étend le modèle précédent à la prédiction de structure secondaire étendue en général, et au support des appariements triples.

IPKnot [297] utilise avec succès une formulation du problème de prédiction de structure secondaire par MEA avec support des pseudonoeuds par la programmation linéaire en nombres entiers (ILP). Malgré l'utilisation du terme « MEA », ils montrent que leur fonction d'évaluation est pourtant équivalente à celle de CentroïdFold [140] (mais avec support des pseudonoeuds).

Zakov *et al.* proposent ContextFold [373], un outil de prédiction qui utilise leur modèle énergétique du plus proche voisin étendu, dont les paramètres sont estimés sur la base de de données de structures secondaires RNAStrand [14].

2012 - Une version de l'algorithme de Zuker & Stiegler (1981) est proposée pour GPU [193].

2014 – RNAlocmin [186] effectue une descente de gradient à partir d'une structure secondaire pour l'optimiser. La modélisation interne résume le paysage énergétique en un « Basin-hopping graph ». L'outil BHGBuilder proposé permet de visualiser ce graphe.

RNAenn [103] prédit une structure MFE sur la base du modèle du plus proche voisin étendu proposé par les mêmes auteurs dans le même article.

2015 - Le Vienna RNA Group et le BGSU Group tentent d'utiliser l'identification de modules 3D pour aider à la prédiction de structure secondaire [336].

Paul Dallaire & Major proposent MC-FlashFold [76,77], leur propre ré-implémentation de MC-Fold sous la forme d'une SCFG et avec un algorithme de programmation dynamique de complexité $O(n^3)$. La thèse de Paul Dallaire [76] permet de mieux comprendre le modèle MC-Fold par rapport à l'article, ainsi que les différences avec RNAwolf et MC-Fold-DP.

pKiss succède à pknotsRG en supportant les pseudonoeuds de type HHH. Avec deux autres outils, il forme le package RNA Shapes Studio [163].

2016 - Les outils de ViennaRNA se dotent d'une « couche de support des contraintes » [204] qui permettent de leur rajouter à tous des contraintes de compatibilité avec des résultats de sondage chimique par exemple.

BHGBuilder supporte les pseudonoeuds [185]. L'article introduit une nouvelle façon de classer les pseudonoeuds par rapport à celle de Pseudobase++.

ProbScan [315] définit une fonction de partition estimant la probabilité qu'une portion de séquence forme un élément de structure secondaire particulier. Cet outil permet de calculer des structures secondaires directement en prédisant les SSE (et non pas les appariements).

2017 – Sloma et Mathews proposent CycleFold [314], une implémentation similaire à MC-FlashFold, qui énumère des assemblages de NCM et renvoie des structures secondaires étendues. Le programme supporte les « contraintes évolutives » en intégrant TurboFold.

2018 – BiokoP [192], développé dans notre laboratoire, prédit des structures secondaires s'étendant entre les estimateurs MFE et MEA. L'outil est basé sur un problème d'optimisation

ILP bi-objectif, et supporte donc les pseudonoeuds.

NeuraFold [4] est proposé par le groupe de Sato (développeurs d'IPKnot). Au lieu du traditionnel algorithme de McCaskill (ou d'une variante), les probabilités d'appariement sont prédites avec un réseau de neurones. À partir de ces probabilités, une prédiction MEA sans pseudonoeuds peut être faite par un algorithme classique, ou avec pseudonoeuds en reprenant la formulation d'IPKnot. L'utilisation d'un réseau de neurones permet de se passer de modèle et de paramètres thermodynamiques, et aussi de supporter des ARN assez longs (~1000 bases). Deux architectures sont proposées. Un RNN bi-dimensionnel ne supportant pas les pseudonoeuds et un réseau feed-forward profond les supportant. Ce dernier est plus intéressant, et montre des performances supérieures aux outils de l'état de l'art supportant les pseudonoeuds.

Les mêmes auteurs (Sato et al) proposent aussi MXFold [5], qui prédit la structure secondaire en utilisant cette fois des paramètres thermodynamiques. Les structures secondaires sont encodées par un vecteur de caractéristiques donnant le nombre de chaque élément associé à une contribution thermodynamique dans le modèle du plus proche voisin (empilement, appariement, boucle). La fonction d'évaluation associée est presque bi-objective : c'est une somme d'un terme associant des poids thermodynamiques à chacune des caractéristiques (modèle du plus proche voisin), et d'un terme associant des poids entrainables qui seront mis à jour par l'optimisation du réseau de neurones.

2019 – RNAvista [18] prédit la structure secondaire étendue à partir de la séquence. Six outils de prédiction de structure secondaire sont proposés pour une première prédiction de structure canonique. Ensuite, RNA Composer (voir section I.4.2.5 en 2012) prédit une structure 3D, ce qui a pour effet d'incorporer des interactions non-canoniques. Enfin, RNApdbee réannote la structure 3D produite en structure secondaire étendue.

Lu *et al.* [206] proposent un réseau LSTM pour prédire les probabilités d'appariement, puis une couche supplémentaire identifie une structure de type MFE.

Singh *et al.* [312] proposent SPOT-RNA, inspiré par les programmes RaptorX [129] et SPOT-Contact [141] utilisés pour les protéines, notamment l'utilisation de ResNets (réseaux de convolution résiduels) et de LSTM bidirectionnels. SPOT-RNA est donc un algorithme de prédiction de contacts (prévu pour la 3D), rendant une structure secondaire étendue en sortie, et supportant les pseudonoeuds. Le modèle utilise le *transfer*-learning, c'est-à-dire des entrainements successifs sur des types de données différents. Un ResNet et un LSTM bi-directionnel sont d'abord entrainés sur la base de données de structures secondaires bpRNA-1m [78] (contenant une dizaine de milliers de structures secondaires non redondantes). Puis, l'entrainement du modèle est affiné sur un jeu de données de structures 3D haute résolution, mais de faible taille. L'outil final supporte donc les appariements non canoniques et les pseudonoeuds, avec une performance similaire aux outils qui ne les supportent pas (F1 score ~0.8).

DMFold [351] utilise également un réseau LSTM bidirectionnel triple couche entrainé sur l'équivalent de bpRNA pour prédire une structure secondaire, suivi d'une unité de correction qui impose à la structure d'être correcte. Les tenseurs utilisés ont des dimensions fixes, on suppose donc que les ARN sont tronqués ou colmatés pour correspondre à une taille de 300 bases.

CDPFold [379] est un CNN qui prédit les probabilités d'appariement, accompagné d'une couche de programmation dynamique qui rend une structure MEA. L'outil apporte plusieurs idées intéressantes : la structure secondaire est encodée sous forme d'une matrice non pas de contacts (0 ou 1) mais de nombre de liaisons hydrogène (0 pour non-apparié, 2 pour AU, 3 pour CG, et x<2 pour GU, le paramètre est étudié). Ceci permet d'embarquer l'information de séquence et de structure dans une petite structure de données. Ensuite, les empilements sont pris en compte (détection d'appariements voisins en 3' ou 5'). Une fenêtre glissante de 3 x d est utilisée, et aboutit à une prédiction de 3 probabilités, une pour chaque symbole de la notation dot-bracket (à savoir '(', '.' ou ')', sans pseudonoeuds). Enfin, le modèle est entrainé une fois pour chaque famille d'ARN (ce qui est dommage, nous verrons pourquoi en section IV.4).

LinearFold [157] utilise un schéma de programmation dynamique approché. La prédiction de structures secondaires sans pseudonoeuds est faisable en temps linéaire en gardant une bonne performance.

2020 – L'outil BiORSEO [26], présenté en détail au chapitre III de cette thèse, est publié. Il prédit un ensemble de structures secondaires en exploitant la détection la présence potentielle de modules 3D dans la séquence, en plus du modèle énergétique habituel.

pourRNA [117] remplace « barriers » pour la description du paysage énergétique sous forme d'arbre de barrières énergétiques.

LinearFold est également adapté en LinearPartition [380] pour renvoyer les probabilités d'appariement.

Willmot *et al.* [358] proposent un autre LSTM bidirectionnel prédisant l'état d'appariement des nucléotides d'une séquence. Il ne s'agit même pas d'une prédiction de structure secondaire, mais juste de l'état apparié ou non, et pourrait plutôt s'apparenter à la prédiction d'une valeur SHAPE ou de sondage chimique.

2dRNA [219] (par le Xiao lab) est une combinaison d'un LSTM et d'un U-Net (une classe de CNN) pour prédire la structure secondaire. À partir de la séquence encodée 1-hot, 2 couches de LSTM bidirectionnel prédisent l'état de chaque nucléotide parmi les 7 symboles des notations dot-bracket (3 niveaux de pseudonoeuds autorisés). À ce niveau, les appariements ne sont pas forcément symétriques. Le CNN prend donc cette prédiction en entrée et prédit une structure cohérente sous forme de matrice de contacts symétrique.

E2EFold [56] se distingue par l'utilisation d'une architecture « Transformer » qui prédit une matrice de contacts, suivie d'une couche permettant de résoudre un problème de satisfaction de contraintes à l'intérieur même du réseau (différentiable), pour que la structure secondaire finale soit plausible. Les performances vantées sont bonnes, mais on se méfiera : le jeu de données utilisé est petit, et les comparaisons sont faites à des outils qui ne prédisent pas les pseudonoeuds.

2021 – Sato *et al.* mettent à jour MXFold en MXFold2 [239]. Le terme thermodynamique supposé prédire l'énergie d'une structure est maintenant reformulé comme une « régularisation thermodynamique » (en plus de la régularisation L1). Son objectif est de maintenir le score associé à une structure aussi proche que possible de l'énergie de ladite structure (calculée selon les règles de Turner 99). L'architecture a changé, les blocs utilisent

maintenant de la convolution 1D, des LSTM bidirectionnels, de la convolution 2D, et un MLP final. La prédiction de la sortie du réseau est un ensemble de 4 scores pour chaque nucléotide pour 4 contributions énergétiques selon 4 états possibles (non apparié, dans une hélice, ouvrant une hélice ou fermant une hélice). Une étape finale de programmation dynamique renvoie la MFE sur la base de ces scores. La performance de prédiction est similaire aux méthodes de l'état de l'art. La corrélation des scores avec l'énergie des structures reportée est de 0.8 environ, contre 0.53 sans le terme de régularisation thermodynamique.

UFold [126] est une autre méthode de prédiction de structure secondaire par architecture U-Net (CNN). Une différence notable qu'elle apporte est la façon d'encoder l'entrée : au lieu d'un encodage 1-hot de la séquence, pour chaque type d'appariement possible ({A,C,G,U}×{A,C,G,U}), une matrice 2D des positions des appariements possibles de ce type est construite, formant une « image à 16 canaux ». Un canal supplémentaire donne la probabilité d'appariement (calculée selon les règles de CDPFold [379]). Une matrice de contacts est prédite en sortie. UFold peut prédire de longues structures de plusieurs milliers de bases en peu de temps.

I.4.2.5 Méthodes de prédiction de structures 3D, ou d'étude de leur paysage énergétique

1987 – Tinker [258] est un logiciel généraliste de dynamique moléculaire et de minimisation, acceptant la plupart des champs de force et des algorithmes, prévu pour la simulation des macromolécules. À cette époque, les champs de force ne sont pas spécifiques de l'ARN.

1991 – Major propose la première version de MC-Sym [217] pour générer des structures 3D à partir de la séquence et de contraintes (décrivant plus ou moins ce qu'on sait de la structure secondaire). Les contraintes portent sur les appariements canoniques ou reverse-Hoogsteen, les empilements, et le respect des structures en hélices pour les nucléotides consécutifs des hélices. À partir d'une dizaine de configurations « templates » de nucléotides (valeurs discrètes des angles de torsion), un algorithme de satisfaction de contraintes assemble des nucléotides en structure 3D. La structure est générée séquentiellement à partir du premier nucléotide, plusieurs branches sont maintenues en parallèle tant qu'elles ne violent pas les contraintes, et tous les templates sont essayés (algorithme de type *branch-and-cut*). Une minimisation est ensuite effectuée dans CHARMM.

1992 – YAMMP [330] propose une collection de routines, procédures et scripts pour manipuler les structures 3D, les champs de forces, fabriquer des modèles gros grains, et des algorithmes de minimisation ou de dynamique moléculaire.

1994 – YAMMP [218] propose une modélisation très gros grains de l'ARN pour aider à résoudre des structures 3D basse résolution. Une première passe modélise la structure avec 1 bille/SSE, une 2e à 5 billes/hélice, et une dernière avec 1 bille par nucléotide (le phosphate).

1995 – Le champ de forces AMBER ff94 [70] supporte officiellement les acides nucléiques, avec une première version des paramètres prévue pour le cas général. Le champ de forces

CHARMM22 [214] suit et supporte lui aussi officiellement les acides nucléiques.

1996 – Le logiciel de dynamique moléculaire NAMD [247] est proposé, intégrant CHARMM et l'outil de visualisation VMD.

1998 – NAB (Nucleic Acid Builder) est proposé, pour construire des structures 3D All-Atom d'hélices à partir de la séquence (avec raffinement dans AMBER) [213].

ERNA-3D est un logiciel graphique développé par le Max Planck Institute pour modifier des structures 3D « à la main ».

1999 - AMBER ff99 contient des paramètres d'orientation des bases calculés par QM. Il est cependant prévu pour l'ADN, pas l'ARN. À partir de ff99, de multiples variantes existent, différant par le protocole de calcul des paramètres ou par l'ajout de termes spécifiques. Il n'existe pas de consensus clair sur la meilleure version pour le cas général.

NAMD2 [240] est proposé, rendant possible des simulations de dynamique moléculaire dans CHARMM sur des clusters d'ordinateurs et supercalculateurs.

2000 - CHARMM27 [122] propose, comme Amber ff99, un calcul par QM des angles de torsion du squelette et de l'orientation des bases.

2003 - Major met à jour MC-Sym [216], amélioré avec une plus large collection de templates annotés par le nouveau MC-Annotate (voir section I.4.2.1 en 2001). Il supporte l'entrée d'une structure secondaire étendue en input. Cette structure est découpée en cycles (NCM) qui sont modélisés individuellement par l'algorithme de satisfaction de contraintes, puis ré-assemblés. Un *back-tracking* de type Las-Vegas est implémenté (si un nœud de l'arbre est inconsistant avec les contraintes, on ne remonte pas de 1 mais d'un nombre aléatoire de nœuds). Ceci accélère la convergence.

2006 – YUP [331] succède à YAMP, porté en Python.

2007 - Das et Baker [84] proposent FARNA (Fragment Assembly of RNA), première méthode d'assemblage de fragments permettant d'obtenir des structures all-atom en 3D à partir de la séquence. Des fragments de chaine de 3 nucléotides sont collectés (dans la structure 1FFK). A partir d'une structure linéaire, un segment de 3 nucléotides est sélectionné aléatoirement et ses torsions sont remplacées par celles observées dans un fragment pioché dans la 1FFK, correspondant à la bonne séquence. Le move-set est accepté ou rejeté selon le critère de Metropolis. La fonction d'évaluation gros grains favorise l'aspect compact (radius of gyration) pénalise les clashs, et modélise les appariements et empilements. Pour les appariements, un "potentiel spatial" pour savoir où trouver le A à proximité du U est appris sur la structure du ribosome. L'algorithme réalise 1000 insertions sans évaluation (chauffage) puis 50000 insertions (le terme de coplanarité est pondéré à 0, ½ et 1 dans les 3 tiers de la simulation). Le top 1% en énergie est clustérisé puis les centroïdes des 5 plus larges clusters sont retournés. L'outil supporte les triplets et les pseudonoeuds.

AMBER supporte les nucléotides modifiés [3] naturels trouvés dans l'ARN.

2008 - Parisien & Major [250] proposent MC-Fold, et le pipeline MC-Fold/MC-Sym permettant d'obtenir des structures 3D par assemblage de 15 classes de NCM. À partir de la séquence, le programme choisit tous les sites possibles d'initiation d'une HL (complexité en $O(N^2)$) puis énumère pour chaque site en les empilant récursivement, tous les empilements possibles de NCM à partir de cette HL d'initiation (complexité en $O(15^{N/2})$ puisqu'il y a 15 classes de NCMs). En comptant une contribution énergétique par NCM, MC-Fold peut renvoyer la structure de plus faible énergie. La fonction d'évaluation est une pseudo-énergie dépendante de la probabilité d'observation des caractéristiques d'une structure sachant la séquence, $p(NCM \mid séquence).p(jonctions \mid NCM).p(appariements\ fermants \mid jonctions)$. Malheureusement, sa formulation dans le document supplémentaire accompagnant l'article est difficile à comprendre, je recommande (sur les conseils de F. Major) de lire plutôt la thèse de Paul Dallaire [76] pour comprendre MC-Fold. Des dispositions spécifiques modélisent les pseudonoeuds de type H et les empilements coaxiaux d'hélices.

RNA2D3D [223] est un logiciel graphique proposant des modèles 3D directement construits à partir d'une structure secondaire. À partir du graphe de structure secondaire, les « nœuds » sont remplacés par des modèles all-atom de chaque nucléotide. Puis, les hélices sont modélisées dans leur forme A, en gardant les boucles au mieux dans le plan initial du graphe. Les pseudonoeuds sont toujours rendus avec les deux hélices empilées de façon coaxiale. Des corrections manuelles peuvent être apportées. Des raffinements dans TINKER (CHARMM) ou ff99 sont possibles à la fin, de façon locale ou globale.

Ding et collaborateurs proposent DMD [95] (Discrete Molecular Dynamics), un modèle gros grains avec 3 billes par nucléotide. L'échantillonnage se fait par dynamique moléculaire « discrète » avec échange de réplicas, supposément plus efficace que la dynamique classique. Les forces considérées sont les appariements, les empilements, l'effet électrostatique (Debye-Hückel) et l'effet hydrophobe. Une estimation rapide de l'entropie est faite à partir des paramètres de la structure secondaire de Mathews 1999. Ils améliorent ensuite DMD en iFoldRNA [307], en rajoutant une reconstruction all-atom à la fin.

2009 – NAST [166] est un modèle gros grains à une bille par nucléotide (le C3') et un champ de forces KB associé. À l'aide de contraintes de structure secondaire (obligatoires) et de contacts 3D connus (facultatifs), la simulation prédit des structures natives. L'usage de données SAXS est également possible. La fonction prend en compte les distances, angles, torsions, une répulsion entre bases éloignées de plus de 3 nucléotides (potentiel de Lennard-Jones), une géométrie particulière pour les nucléotides appariés (distance, angles), et des contraintes pour les contacts distants fournis par l'utilisateur.

BARNACLE [125] est une première méthode d'échantillonnage continu des structures 3D, via les angles de torsion du squelette. Le modèle utilise un « réseau bayésien dynamique » (qui fait penser à un LSTM avant que cette architecture soit populaire) pour prédire les angles de torsions. L'outil apprend donc les distributions de probabilité des angles, et est capable d'échantillonner de manière continue (et de reconstruire des structures 3D par l'algorithme NeRF [252]), ce qui s'oppose aux méthodes existantes d'assemblage de fragments, qui font de l'échantillonnage discret (les fragments). Ceci permet de satisfaire la réversibilité des algorithmes MCMC (voir section II.1.4.1). La fonction d'évaluation de la

simulation MC mesure la distance de la structure 3D avec la structure secondaire cible (somme des distances aux distances optimales pour chaque liaison H de chaque appariement). L'utilisation de BARNACLE fait mieux que l'échantillonnage des valeurs d'angles dans une distribution uniforme, ou dans un GMM (modèle mixte Gaussien) entrainé sur les mêmes données d'entrainement (mais sans dépendance d'un nucléotide au suivant). L'article est également intéressant à propos de la constitution d'un jeu de données d'entrainement.

2010 – FARFAR [85] (FARna with Full Atom Refinement) améliore FARNA. Il ajoute une minimisation finale par la fonction d'énergie de Rosetta. La fonction gros-grains de FARNA possède maintenant des termes pour les interactions base-squelette et squelette-squelette. La procédure intègre maintenant 3 étapes de FARNA avec des fragments de taille 3, puis 2, puis 1 nucléotides avant le raffinement final.

Pasquali propose le modèle HiRE-RNA [253], un modèle gros grains précis avec 6 ou 7 billes par nucléotide, capable de modéliser les empilements et appariements. Il est prévu pour des simulations de REMD. La fonction d'énergie ressemble à AMBER pour les coordonnées internes, et un potentiel personnalisé est proposé pour les interactions non liées, dont les appariements. Les paramètres sont appris sur des structures 3D disponibles et optimisés à la main. Des contraintes issues d'une prédiction de structure secondaire peuvent être ajoutées.

Xia et collaborateurs présentent un autre modèle gros grains haute résolution, CG [363]. La modélisation des nucléotides est un peu différente et la fonction d'évaluation aussi, mais ses paramètres sont également appris sur des structures existantes. Le modèle est prévu pour la dynamique moléculaire en solvant implicite, et pour le recuit simulé. Des contraintes sur la structure secondaire peuvent également être fournies.

RNABuilder [121] est un outil de génération de modèles 3D non-automatique via un outil graphique. Il permet de générer tous les appariements non canoniques. Les modèles finaux sont raffinés dans AMBER. Plus tard, l'outil supportera aussi l'ADN et les protéines, et sera renommé Macro-Molecular Builder (MMB) au sein du package SimTK.

Schlick propose RNA Junction Explorer [190], un outil d'apprentissage supervisé capable de prédire la classe des jonctions multiples (ML) et l'empilement coaxial de leurs hélices. C'est un outil unique en son genre, spécifique de cette tâche difficile. La performance est assez moyenne, mais le jeu de données d'entrainement était petit.

2011 - Bernauer et collaborateurs [33] proposent deux modèles d'évaluation des structures sur la base des distances interatomiques, l'un all-atom (RNAkb) et l'autre à 5 billes/nucléotide. Un modèle de distance est créé pour chaque paire d'atomes possible en piochant dans les 4 nucléotides. La courbe des fréquences associées à la distance entre ces atomes présentera plusieurs modes, l'un pouvant correspondre à ces deux atomes dans deux nucléotides empilés, l'autre aux deux nucléotides appariés. Les distributions de ces distances sont estimées par des GMM dans une population observée et une population de référence, puis un potentiel statistique est calculé, entièrement différentiable et utilisable en descente de gradient ou MD. Ils montrent que cela suffit à identifier les structures natives, mieux que la fonction de FARFAR, même en gros grains et sans termes physiques pour

l'électrostatique ou le solvant implicite.

RASP [52] est un potentiel all-atom KB, basé sur les distributions de probabilité des distances entre atomes interagissant.

CHARMM36 All-Atom [92] possède des paramètres estimés pour tous les angles de torsion de l'ARN.

VFold devient un pipeline VFold-3D ou VFold2D+3D [51]. À partir de l'ensemble des structures secondaires calculé par VFold, celles de basse énergie sont modélisées en 3D gros grains par une recherche de fragments (sur la base de la séquence), SSE par SSE. La base de données de SSE utilisée sera baptisée VFoldMTF plus tard. La reconstruction all-atom est faite en remplaçant les nucléotides par ceux d'une hélice de forme A. Les structures finales sont minimisées dans AMBER.

Mathews [300] propose un pipeline de prédiction de structure 3D All-atom, par recuit simulé dans AMBER-ff99 en solvant implicite à partir d'une structure linéaire, soumis à des contraintes calculées à partir d'une prédiction de structure secondaire, *direct coupling analysis* (DCA), prédiction d'empilements coaxiaux des hélices, et d'autres.

2012 – 3dRNA [383] propose des modèles all-atom par assemblage de fragments connus, sélectionnés avec une heuristique, ou générés si absents de la base de fragments. Il n'y a pas d'échantillonnage/évaluation à proprement parler. La structure finale est minimisée dans AMBER-ff98.

Reinharz, Waldispühl, et Major proposent RNA-MoIP [272], programme qui utilise une formulation ILP pour raffiner une ou plusieurs structures secondaires connues ou prédites pour un ARN. Il propose en sortie une structure secondaire dans laquelle sont insérés des modules 3D de RNA3dMotifs [101]. Le programme génère aussi un script de contraintes pour MC-Sym, ce qui aboutit à des prédictions 3D avec une bonne performance.

Popenda, Adamiak et collaborateurs proposent RNA Composer [261], un système automatique pour construire une structure 3D à partir de la 2D. Les SSE sont modélisés séparément. Des templates sont sélectionnés dans FRABASE 2.0 (voir section I.4.2.1) sur la base de la forme et similarité de séquence, le remplacement de certaines bases peut être nécessaire. Si un fragment n'est pas trouvé, il est généré avec NAB [213] (pour les hélices) ou CYANA (pour les boucles). Bien que les fragments de FRABASE soient pré-minimisés dans CHARMM (par le programme XPLOR), un raffinement final relaxe la structure proposée, d'abord dans CYANA pour les angles de torsion, puis dans XPLOR (CHARMM) pour les coordonnées cartésiennes.

RNAJAG [189] (de l'équipe de Schlick) modélise les jonctions multiples en prédisant non seulement la classe de topologie de la boucle ML, les empilements coaxiaux, mais aussi l'orientation des hélices voisines dans l'espace.

Rsim [37] est un pipeline de prédiction de structures 3D. Des structures secondaires sont prédites par RNAsubopt, puis modélisées en 3D par assemblage de fragments. Une procédure analytique énumère les conformations possibles et des structures 3D correspondantes sont produites en assemblant des fragments de 3 nucléotides. On obtient plusieurs solutions par structure secondaire. Ces structures 3D subissent une optimisation MC améliorée sur la base de FARNA. L'échantillonnage se fait avec des *closed moves*, c'est-à-dire qu'après la casse de la structure secondaire par l'insertion d'un fragment, un autre

fragment est inséré ailleurs de façon à compenser et maintenir la structure secondaire. Un pré-calcul des compatibilités entre fragments pour préserver chaque structure secondaire a été réalisé pour accélérer la procédure. L'évaluation se fait dans un potentiel statistique all-atom et knowledge-based (KB) basé sur les distances interatomiques au sein de paires de nucléotides appariés et appris sur une base de données de dimères. En plus, un terme mesure l'aspect compact, et un terme mesure la probabilité d'existence des liaisons hydrogène en fonction de la position des donneurs et accepteurs. Le programme rend un graphe reliant les structures finales entre elles et permettant d'identifier des clusters de structure secondaire, et des chemins dans le paysage énergétique.

2013 - HiRE-RNA v2 [72] est légèrement modifié (changement des termes modélisant les liaisons hydrogène), et se dote d'une reconstruction all-atom.

Le modèle « CG » de Xia & Ren est également mis à jour [362] avec de nouveaux termes pour prendre en compte l'électrostatique,

TIS [91] est un nouveau champ de forces gros grains à trois billes par nucléotide, mais pas uniquement KB, puisqu'il utilise des paramètres thermodynamiques mesurés expérimentalement pour les appariements et empilements. Il est utilisé pour modéliser trois systèmes, par dynamique de Langevin.

NARES-2P [148] est un champ de forces gros grains à 2 sites d'interaction par nucléotide, utilisé en dynamique moléculaire. Sa paramétrisation originale est inconnue car protégée par un paywall. Une nouvelle paramétrisation sera proposée en 2015.

WebRASP est un webservice utilisant RASP pour estimer la qualité d'une structure 3D (le serveur effectue une évaluation de la structure dans RASP).

2014 – RAGTOP [177] améliore RNAJAG avec un potentiel statistique et un algorithme de recuit simulé pour réellement échantillonner des graphes de SSE en 3D (des graphes RAG-3D) à partir d'une structure secondaire et des prévisions de RNAJAG pour la forme des ML. Il peut donc prédire la topologie globale de l'ARN.

TOPRNA [245] propose des structures 3D à partir de structures 2D, en modélisant l'ARN par un modèle gros grains à 3 billes par nucléotide. Les hélices sont modélisées semi-rigides, de petites variations de twist étant possibles. Les jonctions monocaténaires entre hélices peuvent tourner (les angles de torsion sont modifiables). Les longueurs entre billes sont modélisées par un potentiel harmonique, différent si le nucléotide est apparié ou non. Les angles de torsion suivent un potentiel statistique (KB). La forme de la fonction est celle de CHARMM, mais appliquée à un modèle gros grains dont les paramètres sont obtenus à partir de RNA FRABASE.

oxRNA [324] est un potentiel gros grains inspiré d'oxDNA, utilisant deux billes par nucléotide mais modélisant les empilements et appariements. Ses paramètres sont ajustés pour correspondre au modèle de Turner. Les auteurs ont une volonté de mesurer l'énergie libre. Des codes utilisant oxRNA dans des simulations MC et MD sont fournis.

Bottaro propose εSCORE [42], une fonction d'évaluation KB basée sur l'apprentissage des distributions de probabilité des positions relatives entre bases proches entre elles (la distribution capture à la fois les appariements et les empilements). Les auteurs montrent qu'elle est capable d'identifier la structure native dans un ensemble de propositions, mieux

que RASP ou FARFAR [43]. L'article présente également εRMSD, une métrique de comparaison de structures, et l'utilisent pour faire de la recherche de modules.

2015 - NARES-2P [147] est reparamétré.

Denise et collaborateurs proposent GARN [45], une méthode originale d'échantillonnage pour transformer des structures 2D en 3D via la recherche d'un équilibre de Nash par minimisation du regret. Les SSE sont les « joueurs » d'un jeu d'échantillonnage, et peuvent évoluer sur une grille 3D triangulaire. Le graphe des SSE (les joueurs) est construit, et chaque joueur peut se déplacer vers un nœud de grille voisin, avec une probabilité donnée par le potentiel statistique. Le graphe utilisé n'est pas vraiment un graphe de SSE, les hélices possèdent un joueur pour 5(x2) nucléotides, et les jonctions triples sont modélisées par deux joueurs (un pour représenter deux hélices empilées coaxialement, et le second pour l'hélice branchante). L'algorithme qui va faire prendre la décision au joueur est soit UCB (meilleure moyenne des positions passées + meilleure position actuelle) soit EXP3 (mouvement aléatoire dans le voisinage de la solution, choisi à l'aide du potentiel KB). La fonction KB est une sorte de potentiel de Lennard-Jones pour SSEs (dépendante du type des SSE en jeu), qui décrit globalement une distance optimale entre eux. Plusieurs idées sont testées. À la fin, le programme rend une topologie 3D (graphe de SSE en 3D). La méthode est efficace computationnellement, mais très « gros grains ».

Xiao et collaborateurs [350] proposent 3dRNAscore, une fonction d'évaluation all-atom KB, rigoureusement entrainée, et basée sur les distances interatomiques et les 6 angles de torsions du squelette. Elle prétend identifier les structures natives mieux que d'autres fonctions all-atom KB, RASP, RNAkb [33] ou FARFAR [85].

iFoldRNA v2.0 [184] propose maintenant des contraintes issues de prédictions de structure secondaire ou de sondage chimique.

ERNWIN [175] construit des structures 3D gros grains en assemblant des templates de SSE. Une fonction KB les évalue, comprenant un terme « physique » pour empêcher les interpénétrations des éléments.

HiRE-RNA v3 [73] est une amélioration proposant des nouveaux termes pour les appariements et empilements. Les appariements non canoniques et multiples sont mieux modélisés. Les paramètres sont optimisés par un algorithme génétique, sur la base de ceux appris sur des données réelles.

2016 – SimRNA [41] est un modèle de gros grains assez précis. Un algorithme de replica-exchange Monte-Carlo échantillonne des structures et un potentiel statistique bien construit les évalue. Le programme supporte les contraintes externes. Il surveille l'énergie de la structure secondaire au cours de la procédure d'optimisation, et renvoie toute la trajectoire de repliement à partir d'une structure initiale circulaire. Les structures finales sont clustérisées et les centroïdes des 3 plus grands clusters sont renvoyés. Une reconstruction all-atom et un raffinement final sont effectués.

2017 - Mathews propose une révision des paramètres d'angles de torsion pour AMBER [22].

3dRNA 2.0 [349] supporte des contraintes produites par *direct coupling analysis* (DCA). La famille Rfam de la séquence à modéliser est inférée, l'alignement de séquences

téléchargé et normalisé statistiquement avec des poids. Les contraintes sont calculées dessus. La prédiction brute de 3dRNA v1 (sans raffinement) est soumise à un recuit simulé sur une représentation gros-grains. Les hélices et petites boucles sont rigides, seule leur orientation change. En dehors, les pseudo-torsions η/θ peuvent changer notamment. La fonction d'évaluation est un modèle harmonique sur longueurs et angles du squelette P-C4', un terme pour les torsions, un terme pour les empilements, un pour les appariements, plus un terme avec une grosse pénalité si une contrainte d'appariement n'est pas respectée. L'outil génère 1000 modèles, les clustérise par DBSCAN, et les meilleurs scores 3dRNAscore sont retenus. Un raffinement par AMBER est possible en option.

Schlick développe F-RAG [162] pour reconstruire des modèles all-atom après les prédictions de topologie de RAGTOP. Le programme fouille la base de données RAG-3D à la recherche de templates de même topologie. La combinaison des deux permet d'obtenir des modèles all-atom en sortie [25].

SPQR [257] est un modèle gros grains à trois billes par nucléotide, reprenant et améliorant la fonction εSCORE et ajoutant un algorithme MC d'échantillonnage. Il prend en compte le volume exclu, les appariements et empilements, les interactions base-phosphate, et angles entre atomes du squelette en fonction des sugar-puckers des nucléotides voisins. Chaque terme est modélisé par une distribution de probabilité en fonction de la localisation dans l'espace des nucléotides intervenant dans une interaction, de façon similaire à εSCORE. Le programme termine par une reconstruction all-atom et une petite dynamique moléculaire dans AMBER-ff99, en maintenant des contraintes pour garder la position des bases (ce sont plutôt les atomes du squelette qui s'ajustent).

Le groupe de Denise améliore GARN en GARN 2 [53]. Il est beaucoup plus rapide, et supporte toutes les jonctions, au prix d'un modèle de graphe un peu plus complexe. Il échantillonne deux structures au lieu d'une, sur la base du score et de l'aspect compact. Plusieurs fonctions d'évaluation (toutes KB, portant sur les distances) sont testées, une forme Lennard-Jones, une forme Lennard-Jones sans le terme positif, et une forme de Gauss. Le benchmark final montre cependant qu'ERNWIN a systématiquement une meilleure performance (modèle et sorties similaires).

Le modèle CG de Xia & Ren est mis à jour en RACER [28], avec un terme modélisant les interactions de van-der-Waals et une re-paramétrisation des termes électrostatiques.

2018 - Le groupe d'Adamiak propose RNAfitme [19], qui réajuste la position des bases d'un modèle 3D sans changer son squelette. Cela permet aussi de générer un modèle all-atom à partir d'un modèle du squelette et d'une séquence au format FASTA. L'algorithme utilise des jeux de templates au choix, et finit par un raffinement dans CHARMM (par NAMD).

NARES-CSA [311] succède à NARES-2P, avec un algorithme génétique pour l'échantillonnage. Il introduit aussi le support des contraintes de structure secondaire.

Le logiciel de dynamique moléculaire Tinker sort dans sa version 8 [267], avec une interface graphique, et le support des GPU.

VFoldLA [365] est une variante de VFold3D, différant surtout par la méthode de recherche de fragments à assembler pour la reconstruction 3D. Au lieu de boucles, des fragments monocaténaires sont recherchés et assemblés (par exemple, les deux brins d'une IL séparément). On perd en qualité sur la compatibilité structurale des brins entre eux, mais

la recherche de fragments réussit bien plus souvent, ce qui peut être vu comme un avantage. Les pseudonoeuds sont aussi mieux modélisés. Néanmoins, les benchmarks montrent que VFold3D reste souvent meilleur.

Chen et collaborateurs critiquent la façon dont doit être construit un champ de forces KB, notamment sur la question du choix des distributions de référence, et proposent IsRNA [377], un champ de forces KB gros grains à 4 ou 5 billes par nucléotide. Le modèle prend en compte les corrélations entre paramètres, par une estimation itérative des états de référence, ce qui est unique en son genre. La fonction en elle-même modélise les coordonnées internes (longueurs, angles), les appariements, l'électrostatique (interactions avec le squelette) et le volume exclu.

2019 – QRNAS [321] est un outil de raffinement des modèles, basé sur AMBER, supportant les ions, ligands, structures hybrides, etc. Il permet de construire des pipelines où une autre méthode de prédiction rend un modèle (all-atom) non raffiné, puis est raffiné par QRNAS directement.

2020 - FARFAR 2 [355] intègre un nouveau move-set de construction de boucles et une base de données de fragments mise à jour. Le pipeline accepte des templates de fragments connus en entrée, si on en connait. Des filtres éliminent les structures avec un faible score gros-grains (FARNA) pour éviter de perdre du temps à les raffiner en all-atom.

CoCoNet [375] est un CNN entrainé sur ce jeu de données, apprenant les contacts entre nucléotides au sein d'ARN homologues et dont les séquences sont alignées. Une première couche calcule les covariations, les suivantes font de la convolution, et la dernière applique un seuil pour déterminer si oui ou non les nucléotides sont en contact.

RNAContact [325] fait la même chose, mais à partir de caractéristiques issues d'une prédiction de structure secondaire, et à partir d'un alignement de séquences produit par Infernal. Les auteurs utilisent Infernal pour construire un alignement à partir d'une séquence et de l'archive NCBI.

2021 – RNAMasonry [58] assemble des fragments 3D de SSE issus de RNA Bricks, et les évalue avec la fonction de SimRNA . Il supporte les contraintes externes (sondage chimique, contacts connus)

HiRE-RNA supporte des contraintes externes de type SAXS [228].

IsRNA1 [295] améliore la fonction IsRNA et l'utilise dans des simulations de MD, avec contraintes de structure secondaire. Le terme modélisant les appariements est modifié, et la fonction reparamétrisée sur un plus gros jeu de données.

I.4.3 Champs de recherche voisins

L'historique et état-de-l'art présenté à la partie précédente se concentrait sur les méthodes de prédiction de structures d'ARN *in silico*, pour un ARN simple dont on voudrait connaitre le ou les repliements dans l'espace.

Il y a également d'autres problèmes très étudiés en biologie computationnelle de l'ARN, on peut citer :

- le problème d'alignement des séquences, en prenant en compte la structure secondaire des séquences à aligner,
- la découverte de nouvelles « familles » d'ARN non codants, que ce soit sur la base de l'homologie de séquence, de la similarité de structure, ou d'une fonction biologique,
- les méthodes d'aide à l'interprétation de données expérimentales pour la résolution des structures 3D (SAXS, protocoles de sondage chimique, résonnance magnétique nucléaire, cryo-EM),
- l'étude de la cinétique de repliement des ARN,
- l'étude de la dynamique d'une structure et de sa stabilité, pour connaitre les temps de séjour moyens dans chaque état métastable, connaitre les zones rigides et celles plus flexibles,
- le problème du « design » : concevoir rationnellement une séquence pour que la structure (ou l'ensemble des structures) réponde à certaines contraintes, ou aie la capacité de faire certaines interactions souhaitées,
- le problème du « docking », c'est-à-dire la prédiction des sites, modes et forces d'interaction entre un ARN et une autre macromolécule ou un ligand,
- la recherche sur les nucléotides modifiés et artificiels, dans le but de stocker de l'information dans l'ARN, d'améliorer sa stabilité, ou d'étendre les possibilités d'interaction,
- et probablement d'autres…

CHAPITRE II :
Introduction à l'optimisation

L'optimisation est le domaine des mathématiques qui cherche à trouver les extrema d'une ou plusieurs fonctions appelées objectifs ou critères. Dans toute la suite, on supposera que l'on cherche des minima, par exemple, les minima de fonctions décrivant l'énergie d'une molécule d'ARN. Une molécule de basse énergie est communément associée, en physique, à une plus grande stabilité et donc à une plus grande probabilité d'existence.

Dans ce chapitre, je commencerai par présenter les méthodes d'optimisation continue, linéaires, non linéaires, et les métaheuristiques utilisées pour l'exploration de paysages énergétiques. Je présenterai ensuite quelques méthodes d'optimisation discrète, en particulier la programmation linéaire en nombres entiers utilisée dans cette thèse. Enfin la dernière section propose un état-de-l'art des méthodes d'optimisation multi-objectif.

II.1 L'optimisation continue

Dans cette classe de problèmes, on s'intéresse aux cas où la fonction f est continue et supposée dérivable (au moins dans une majorité du domaine). Le minimum de la fonction est recherché dans un sous-domaine de l'espace $\mathbb{R}^m$ bien défini, parfois à l'aide de contraintes d'égalité ou d'inégalité (optimisation sous contraintes). De façon formelle, appelons $f : x \mapsto f(x)$ la fonction à minimiser sous k_1 contraintes d'égalité et k_2 contraintes d'inégalité. On peut rassembler les contraintes sous la forme de deux vecteurs de fonctions g et h tels que le problème soit défini par :

$$\underset{x \in \mathbb{R}^m}{\operatorname{argmin}} f(x) \qquad g(x) \leq \vec{0} \in \mathbb{R}^{k_1},\ h(x) = \vec{0} \in \mathbb{R}^{k_2}$$

où $\vec{0}$ est le vecteur nul (toutes ses composantes sont nulles). De cette définition très large, on peut sous-lister de très nombreuses classes de problèmes d'optimisation.

Une fonction f est dite convexe si tout segment reliant deux points arbitraires de la courbe de f se situe dans ou au-dessus de la courbe de f. Un espace est dit convexe si tout segment reliant deux points arbitraires de l'espace se trouve dans l'espace. Si les fonctions à minimiser sont convexes et les contraintes également, l'espace de recherche est convexe. On dit alors que le problème est convexe, il existe alors un unique minimum global. Lorsque le problème n'est pas convexe, il peut exister des minima locaux. Un minima local est une solution x telle qu'il existe une boule de rayon $\varepsilon > 0$ centrée sur x telle que x est une meilleure solution à l'intérieur de cette boule. On dit aussi que x est la meilleure solution dans son voisinage.

Les propriétés et méthodes présentées ici sont admises et connues dans le champ de recherche et difficilement attribuables à un auteur. De nombreux livres les présentent. Pour ma part, je citerai le cours en anglais du Pr. Jean-Charles Gilbert (ENSTA-Université Paris-Saclay) [135] ou son livre en français [134]. Prises ensemble, ces deux références complémentaires couvrent bien le champ de recherche. Elles redirigent également vers de nombreuses autres publications.

II.1.1 Conditions d'optimalité

Pour résoudre des problèmes d'optimisation continue, avant même de parler des méthodes de recherche, on commence par se demander comment identifier une solution optimale. La réponse dépend de la structure du problème et des contraintes. La recherche en optimisation a donc découvert et listé des « conditions d'optimalité » qui, si elles sont remplies, permettent de conclure à l'optimalité d'une solution.

On distingue des conditions nécessaires (toutes les solutions optimales la respectent, mais certaines non-optimales aussi) et des conditions suffisantes (toutes les solutions qui la respectent sont optimales, mais certaines optimales peuvent ne pas la respecter).

On peut donc utiliser les conditions suffisantes pour essayer de trouver les optima. Et si on n'a pas de condition suffisante à disposition, on peut uniquement chercher à vérifier les conditions nécessaires, mais il faudra encore éliminer certaines des solutions trouvées.
Certaines conditions font appel seulement à la notion de gradient (dérivée) de la fonction, on les appelle conditions du premier ordre. Certaines, plus difficiles à vérifier, font appel à la matrice Hessienne $\nabla^2 f(x)$ (les dérivées secondes) de la fonction. On les appelle conditions du second ordre. Par exemple, pour l'optimisation sans contraintes, on peut citer :

- La condition nécessaire de criticité (ou théorème de Peano-Kantorovitch) : si un point x^* est un minimum local, alors $\forall d \in \mathbb{R}^m,\ \nabla f(x^*) \cdot d \geq 0$. Ceci revient à dire que quelque soit la direction d dans laquelle on regarde à partir de x^*, la pente est positive, c'est-à-dire que la courbe de f remonte. On a aussi le corolaire $\nabla f(x^*) = 0$ qu'on appelle « stationnarité » (ce ne serait plus le cas avec des contraintes). Dans le cas où f est convexe, le problème est convexe, cette condition est alors suffisante. Les solutions qui vérifient cette condition sont appelées points stationnaires (si $\nabla f(x^*) = 0$), ou critiques ($\forall d \in \mathbb{R}^m, \nabla f(x^*) \cdot d \geq 0$). Pour la plupart des fonctions continues décemment formées, stationnarité et criticité sont équivalents.
- La condition nécessaire du second ordre : si un point x* est un minimum local, alors $\nabla f(x^*) = 0$ d'une part, et d'autre part $\nabla^2 f(x^*)$ est semi-définie positive. « Semi-définie positive » veut dire symétrique et à valeurs propres non négatives, ou encore que $\forall d \in \mathbb{R}^m,\ d^T \cdot \nabla^2 f(x^*) \cdot d \geq 0$. On peut aussi utiliser le fait qu'une solution ne vérifie pas cette condition nécessaire pour dire qu'elle n'est pas un minimum local.
- La condition suffisante du second ordre : si x^* est un point stationnaire ($\nabla f(x^*) = 0$) et $\nabla^2 f(x^*)$ est définie positive, alors x^* est un minimum local. « Définie positive » veut dire symétrique et à valeurs propres strictement positives, ou encore semi-définie positive et inversible, ou encore que $\forall d \in \mathbb{R}^{m*},\ d^T \cdot \nabla^2 f(x^*) \cdot d > 0$.

Avec des contraintes d'égalité, on se place dans le cas où l'on cherche le minimum d'une fonction $f(x)$ ou x est un vecteur de m variables vérifiant k contraintes d'égalité, c'est-à-dire vérifiant $h(x) = 0$ où h est un vecteur de k fonctions.

$$\underset{x \in \mathbb{R}^m}{\operatorname{argmin}} f(x) \qquad h(x) = \vec{0} \in \mathbb{R}^k$$

Dans ce nouveau cas, on obtient :

- La condition nécessaire de Lagrange : si un point x^* est un minimum local et que les contraintes $h(x)$ sont qualifiées, alors il existe un vecteur λ^* tel que le gradient du Lagrangien $\nabla L(x^*, \lambda^*) = \nabla f(x^*) + \lambda^* \cdot \nabla h(x^*) = 0$. Si le problème est convexe, cette condition est suffisante.
- Il existe des conditions nécessaires et suffisantes du second ordre, qu'on ignorera ici.

Le gradient de f est un vecteur de m valeurs, λ^* est un vecteur ligne de k « multiplicateurs de Lagrange », et ∇h est la Jacobienne des contraintes $h(x)$ à k lignes et m colonnes.

La « qualification de contraintes en x^*» est considérée comme valide si x^* valide au moins l'une des contraintes de qualification, comme LCQ, LICQ, MFCQ, CRCQ, GCQ, KTCQ, les conditions de Slater, et/ou d'autres [220]. Si les contraintes sont qualifiées avec SMFCQ ou une condition plus forte comme LICQ, λ^* existe et est unique. Avec MFCQ seule, le domaine des λ^* possibles est non vide et borné. La qualification des contraintes signifie qu'une propriété locale de la situation en x^* permet d'affirmer que l'approximation linéaire de ces contraintes est une bonne approximation. En pratique, les contraintes sont très souvent qualifiées (surtout si elles sont linéaires, du coup). Cette vérification de qualification est surtout nécessaire à la justification de l'échec des conditions nécessaires et suffisantes dans certains cas pathologiques ou les contraintes « ne sont pas qualifiées ».

Le plus souvent, on note la condition de Lagrange sous la forme $\nabla f(x^*) + \sum {\lambda^*}_i \cdot \nabla h_i(x^*) = 0$ accompagnée de $h(x) = 0$, pour faire apparaitre un système à $m + k$ variables et inconnues. On voit qu'avec des contraintes linéaires, $\nabla f(x^*)$ peut être non nul (≠ Peano-Kantorovitch), mais dans ce cas égal à une combinaison linéaire des gradients des contraintes. Ce n'est plus le gradient de la fonction qui s'annule, mais celui du Lagrangien.

Enfin, avec des contraintes d'égalité et d'inégalité (le cas général), on possède deux ensembles de contraintes $g(x)$ et $h(x)$. Parmi ces contraintes, certaines sont dites « actives » ou « saturées » en x^*, cela signifie qu'elles contraignent effectivement les mouvements possibles à partir de x^* dans le domaine à explorer. Donc, une contrainte $c(x)$ est active au point x^* si $c(x^*) = 0$, même si $c(x)$ est une contrainte d'inégalité normalement formulée comme $c(x) \leq 0$. Cela signifie que x^* est sur un bord du domaine réalisable, à partir de x^* on ne peut pas se déplacer dans toutes les directions puisque la contrainte x^* en interdit probablement certaines. Les contraintes d'égalité $h(x*) = 0$ sont toujours actives, on attribue plutôt ce qualificatif aux contraintes d'inégalité pour montrer qu'elles sont respectées avec égalité. Ainsi, parmi les contraintes d'inégalité, certaines sont actives en x^* : on peut les transformer en contraintes d'égalité. D'autres sont inactives : elles n'ont aucun effet sur les directions possibles de déplacement à partir de x^* : on peut les supprimer du problème (quand on considère le problème en x^*).

On note $c(x^*)$ l'ensemble des contraintes d'égalité et contraintes actives en x^*.

$$\underset{x \in \mathbb{R}^m}{\operatorname{argmin}} f(x) \qquad g(x) \leq \vec{0} \in \mathbb{R}^{k_1},\ h(x) = \vec{0} \in \mathbb{R}^{k_2}$$

Certaines conditions d'optimalité connues sont alors :

- Les conditions nécessaires de Karush-Kuhn-Tucker (KKT) : si un point x^* est un minimum local et que les contraintes sont qualifiées, alors :
    - il existe des vecteurs $\lambda^* \geq 0$ et μ^* tels que le gradient du Lagrangien $\nabla L(x^*, \lambda^*, \mu^*) = \nabla f(x^*) + \lambda^* \cdot \nabla g(x^*) + \mu^* \cdot \nabla h_i(x^*) = 0$,
    - avec au moins un λ_i^* non nul,
    - et vérifiant $\sum \lambda_i^* \cdot g_i(x^*) = 0$.

Si le problème est convexe, la condition est suffisante.

Par rapport à la condition de Lagrange, on rajoute des multiplicateurs devant les contraintes d'inégalité. Ces multiplicateurs doivent être positifs si la contrainte est active, et nuls si elle est inactive, comme l'impose la *complementary slackness condition* (le fait que $\sum \lambda_i^* \cdot g_i(x^*) = 0$). Comme précédemment, ces multiplicateurs de Lagrange sont uniques ou non, selon la qualification des contraintes (uniques si les contraintes LICQ ou SMFQC sont qualifiées, voir la référence [220] pour leurs définitions). Dans tous les cas, il existe une garantie que l'ensemble des vecteurs de multiplicateurs possibles est un polyèdre convexe (fermé) de $\mathbb{R}^k$. Si MFCQ est respectée, c'est aussi un ensemble borné.

- La condition nécessaire de Fritz-John : si un point x^* est un minimum local, alors il existe des vecteurs $\lambda^* \geq 0$ et μ^* tels que $\lambda_0^* \cdot \nabla f(x^*) + \sum \lambda_i^* \cdot \nabla g_i(x^*) + \sum \mu_i^* \cdot \nabla h_i(x^*) = 0$, avec au moins un λ_i^* non nul.

La différence avec les conditions KKT réside dans le facteur λ_0^* présent devant le gradient de f. Cette condition a l'avantage de ne pas nécessiter de qualification de contraintes. Si $\lambda_0^* > 0$, elle est équivalente aux KKT (il suffirait de diviser les autres λ_i^* par λ_0^*). Mais si $\lambda_0^* = 0$, on est dans une situation où les contraintes ne sont pas qualifiées (en particulier, MFQC n'est pas respectée).

II.1.2 Optimisation linéaire

Cette courte section est brièvement mentionnée pour la complétude de la vue d'ensemble, mais il n'y a pas de problème linéaire continu posé dans cette thèse.

Dans cette sous-section, on s'intéresse aux problèmes dont la fonction à optimiser est linéaire, et dont les contraintes (d'égalité ou d'inégalité) sont également linéaires.
En anglais, on parle de LP ou *Linear Programming*.

Les contraintes étant linéaires, l'espace de recherche est une intersection d'hyperplans, il est donc convexe (parce qu'un hyperplan est convexe et que l'intersection de plusieurs ensembles convexes est toujours convexe). S'il est borné, c'est un polyèdre convexe. Grâce à ceci, on sait que l'un des points extrêmes (les sommets du polyèdre) correspond forcément au minimum ou à l'un des minima (une arête entière peut constituer l'ensemble des minima). Les algorithmes doivent donc se concentrer sur l'énumération intelligente des points extrêmes du polyèdre.

Pour tout problème linéaire, il existe un problème dual que l'on peut parfois résoudre à la place du primal, quand on trouve cela plus simple. C'est souvent le cas si le nombre de contraintes est supérieur au nombre de variables ($k \geq m$), on a intérêt à résoudre plutôt le problème dual, puis à retrouver la solution du primal à partir du dual.

L'algorithme le plus connu est l'algorithme du simplexe (et ses variantes) [80]. On peut aussi citer l'algorithme en croix, la méthode du gradient projeté, la méthode du lagrangien

augmenté, la méthode de l'ellipsoïde, les méthodes affines, les méthodes de points intérieurs, et d'autres. En particulier, l'algorithme du simplexe est connu pour des raisons historiques, mais il n'est ni le plus rapide en termes de convergence (en nombre d'itérations) ni le plus adapté aux larges problèmes. Pour la vitesse de convergence, on regardera la méthode des points intérieurs. Pour les grands problèmes, on peut utiliser les méthodes de décomposition de Benders ou de Dantzig-Wolfe, ou encore la méthode des plans sécants.

II.1.3 Optimisation non linéaire par descente de gradient

Si on se situe dans le cas où la fonction est convexe et les contraintes convexes, l'espace réalisable est un convexe et on dit que le problème est convexe. Dans ces conditions, il n'y a qu'un seul minimum local et global, et le trouver permet de résoudre le problème. Ces problèmes sont donc considérés 'faciles'.

Bien qu'il existe de nombreuses méthodes dédiées aux problèmes convexes, on citera :

- L'approximation de la fonction f convexe par un paraboloïde (par un développement de Taylor à l'ordre 2 de f au voisinage de x_t) suivie de la résolution analytique du minimum x_{t+1} de ce paraboloïde. On peut recommencer jusqu'à convergence de la suite (x_t).
- Les méthodes de descente de gradient, elles convergeront vers le minimum, mais ne seront pas forcément les plus rapides.

Cependant dans de très nombreux problèmes concrets de recherche ou d'ingénierie, les fonctions à optimiser ne sont ni linéaires ni convexes. Elles contiennent donc plusieurs (voir de très nombreux) minima locaux. Pour ces problèmes, la descente de gradient est une méthode rapide, mais qui conduit quasiment toujours à une solution sous-optimale, un minimum local. Répéter l'entrainement avec des conditions d'initialisation différentes permet parfois, par chance et en fonction de la difficulté du problème, de tomber sur des minima proches du minimum global. C'est donc de cette façon que sont entrainés l'écrasante majorité des réseaux de neurones artificiels utilisés en apprentissage profond, et dont nous ferons usage dans cette thèse. En effet, l'entrainement d'un réseau de neurones est un problème d'optimisation : on chercher à trouver le vecteur de poids et de biais qui minimise la fonction de perte (ou *loss*, l'erreur entre la prédiction du réseau et la réponse attendue). On effectue donc une descente de gradient de cette erreur jusqu'à converger vers les poids et biais qui la minimisent, puis on considère alors que le réseau est entrainé.

En général, ces méthodes se déroulent en deux temps : tout d'abord, il faut choisir une « direction de descente », puis la « taille du pas » que l'on va faire dans la direction choisie.

II.1.3.1 Méthode de la plus forte pente (Steepest Descent)

On prend comme direction de descente $d = -\nabla f(x)$. Ceci correspond à la direction de plus forte pente, car si l'on se place sur chaque axe dans l'espace des variables, on a deux directions possibles (vers les positifs, ou vers les négatifs), et $\nabla f(x)$ indique la direction où la fonction croît. La valeur numérique des composantes de d sur chaque axe n'est pas réellement importante puisqu'en raisonnant ainsi, on approxime f par ∇f au voisinage de x. C'est une approximation linéaire, donc quel que soit le coefficient de proportionnalité qu'on pourrait mettre sur chaque axe, le coté où ça monte donnera des valeurs de $f(x)$ plus élevées, et le coté où ça descend des valeurs plus basses. Si ceci devient faux, c'est que l'approximation linéaire de f au voisinage de x n'est plus bonne.

Pour déterminer la longueur du déplacement dans la direction d, on utilise un coefficient α_k tel que $x_{k+1} = x_k + \alpha_k . d$. Ce coefficient peut être choisi constant, ou dépendre du numéro de l'itération k, ou encore être déterminé de façon à minimiser $f(x_k + \alpha_k . d)$, ce qui constitue un nouveau problème d'optimisation. Si l'on est en train de résoudre un problème d'optimisation avec contraintes, on veillera à ce que la mise à jour ne sorte pas du domaine réalisable. En apprentissage automatique, et plus particulièrement dans l'optimisation des poids des réseaux de neurones, on appelle α_k le *learning-rate* ou « vitesse d'apprentissage » et on le note plutôt η.

II.1.3.2 Stochastic gradient descent (SGD), batch gradient descent et mini-batch gradient descent

Dans le cadre de l'entrainement de réseaux de neurones, on peut choisir d'appliquer la méthode de la plus forte pente en choisissant à chaque itération un point de données aléatoire (et non encore utilisé) pour calculer le gradient de la loss en fonction des poids à optimiser (les variables de décision). Le calcul du gradient est donc rapide (puisque calculé sur un point), mais la descente dans son ensemble sera très fluctuante. Ceci peut aider à passer certaines barrières énergétiques, mais la convergence vers l'optimum sera plus lente. Utiliser un learning-rate décroissant avec le temps permet de régler le problème.

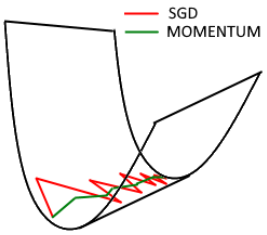
A l'inverse, toujours lors de l'entrainement des réseaux de neurones, on peut choisir d'effectuer la descente de gradient en choisissant une direction d qui ne représente pas forcément la direction de plus forte pente pour chaque donnée, mais « une bonne direction » de descente pour un petit ensemble de données appelé « paquet » ou *batch*. Ainsi, on choisit souvent $d = -\frac{1}{n}\sum \nabla f(x_i)$, l'opposé de la moyenne des i gradients calculés pour les chaque point de données x_i du batch de taille n (les gradients sont bien calculés par rapport aux poids des neurones ici et pas par rapport aux entrées x_i).
La *batch gradient descent* (batch-GD) est supposée utiliser toutes les données, quand la *mini-batch gradient descent* utilise des mini-batches de taille n inférieure à la taille du jeu de données.

La *batch gradient descent* est lente à réaliser une itération sur CPU, mais adaptée aux GPU ou processeurs équipés d'instructions SIMD, à partir du moment où le batch tient en entier en mémoire. L'utilisation des mini-batches est un bon compromis entre les avantages

de SGD et de la batch-GD. Cela permet d'être à la fois assez rapide et assez stable (la stabilité augmente avec la taille du batch). Un *learning-rate* trop grand fera fluctuer l'algorithme longtemps autour du minimum, un *learning-rate* trop petit rendra la convergence de la descente très lente. Il est recommandé d'utiliser des schémas de décroissance.

II.1.3.3 Améliorations de l'algorithme pour l'apprentissage profond

Pour accélérer la vitesse de convergence lors de l'entrainement des réseaux de neurones, de nombreuses améliorations ont été proposées [287] (et de nombreuses continuent d'être proposées) :



- Momentum : Permet d'accélérer la convergence de la descente de gradient en évitant les rebonds dans les « ravins » (les zones abruptes sur une direction mais plates dans les autres directions). La direction d est calculée en fonction du gradient mais aussi en fonction de la direction à l'itération précédente : $d_{t+1} = \gamma d_t - \eta \nabla f(w)$ avec γ un hyperparamètre qu'on doit choisir du même ordre de grandeur que le learning-rate η. Ceci force la descente de gradient à faire des virages très peu serrés.
- Nesterov Accelerated Gradient (NAG) : Au lieu de calculer le gradient en x^*, cette méthode le calcule en $x^* - \gamma d_{t-1}$(elle fait une pré-projection en utilisant la direction précédente). Ceci a pour effet d'anticiper les remontées et de ralentir la descente à l'approche du minimum. On peut combiner NAG et Momentum, ce qui donne : $d_{t+1} = \gamma d_t - \eta \nabla f(x^* - \gamma d_t)$.
- Adagrad et RMSProp (Root-Mean-Square Prop) adaptent le learning-rate à chaque axe au lieu d'utiliser une valeur constante.
- Adadelta propose une méthode capable d'utiliser le bon learning-rate automatiquement (plus besoin de proposer le paramètre à l'algorithme).
- Adam combine Adadelta et Momentum. Adam est très efficace pour « coller » au fond des minima locaux et ne pas remonter la pente de l'autre côté, ce qui en fait un algorithme de choix lorsque chaque itération compte.
- Nadam combine Adam et le NAG, utilisant une pré-projection.
- AMSGrad (2019) est utilisé dans les rares cas où Adam n'est pas meilleur qu'un simple SGD+Momentum. Au lieu de garder une moyenne évolutive des gradients passés, on utilise le maximum de ces gradients passés. En pratique, on ne sait pas encore si AMSGrad est toujours meilleur qu'Adam ou seulement sur ces cas où Adam est mauvais.

II.1.4 Métaheuristiques pour l'exploration des paysages énergétiques

Dans le cas des fonctions d'énergie utilisées pour évaluer les structures moléculaires en bio-informatique, les méthodes de descente de gradient ne sont pas une option (ou alors juste pour « raffiner » une structure). En effet, les très nombreux minima locaux du paysage énergétique piègent ces algorithmes et ne leur permettent pas d'explorer suffisamment

l'espace des conformations possibles. Les méthodes d'échantillonnage qui ont recours à l'optimisation d'une fonction d'énergie hautement non linéaire utilisent alors des « métaheuristiques ». Ce sont des algorithmes d'optimisation non-déterministes et sans garantie sur la qualité des solutions renvoyées, mais qui permettent d'explorer plus largement l'espace des conformations.
Le terme « métaheuristique » désigne en général une famille d'algorithmes plutôt qu'une procédure précise, qui a besoin d'être adaptée au problème concret à résoudre (pour le coté méta-). Les algorithmes en question utilisent l'aléatoire (ils sont non déterministes), et on ne peut donc pas garantir l'optimalité de la solution (coté heuristique).

II.1.4.1 Méthodes de Monte-Carlo

Un algorithme de Monte-Carlo est une méthode stochastique de calcul d'une valeur, de recherche d'un minimum de fonction, ou d'échantillonnage de solutions dans l'espace, qui peut donner une réponse approchée dans un certain intervalle de confiance, en un temps de calcul court. On distingue et on leur oppose les algorithmes de Las Vegas, une sous classe de Monte Carlo qui donnera un résultat exact avec un temps de calcul aussi long que nécessaire.

Parmi les méthodes de Monte-Carlo, on en trouve certaines appliquées à l'échantillonnage de points dans des distributions de probabilité complexes et arbitraires, comme celles issues d'un paysage énergétique d'une structure moléculaire. On les appelle méthodes de Monte-Carlo par chaines de Markov (ou MCMC pour Monte-Carlo Markov Chain), car elles reposent sur un processus de « marche aléatoire » entre les états possibles d'une chaine de Markov. Ces méthodes nécessitent, à partir d'un état donné (une solution), de définir des voisins de cet état qu'il est possible d'atteindre par des mouvements unitaires que l'on appelle *move-sets*. Ces move-sets sont spécifiques du problème étudié. La marche aléatoire consiste alors en une trajectoire dans le graphe des états en passant d'un voisin à l'autre. Les probabilités de transition entre états diffèrent selon la méthode, mais garantissent que la distribution finale des états par lesquels le processus est passé suive la distribution de probabilité voulue. Je vais maintenant présenter les plus pertinentes d'entre elles en bioinformatique structurale.

L'algorithme de Metropolis [236] permet d'échantillonner des solutions dans une distribution de probabilité arbitraire et de grande dimension (comme celle des structures 3D par exemple), sans même avoir besoin de la connaitre. On a besoin de connaitre seulement une fonction $f(x)$ qui lui est proportionnelle (à une constante près). On peut ainsi utiliser pour une molécule x la fonction $f(x) = e^{-E(x)/k_B T}$ où $E(x)$ est son énergie (T la température absolue et k_B la constante de Boltzmann). On s'affranchit ainsi du calcul difficile de la fonction de partition (constante) qu'il faudrait rajouter au dénominateur de cette expression pour calculer la probabilité exacte, qui nécessite l'énumération exhaustive des solutions possibles. Dans cet algorithme, les probabilités de transition d'un état à un autre sont symétriques. À chaque itération, on tire un candidat y voisin de l'état actuel x (selon les

probabilités de transition). Puis on l'accepte comme nouvel état actuel avec une probabilité $f(y)/f(x)$. Si ce quotient est plus grand que 1, on l'accepte toujours.

L'algorithme de Metropolis-Hastings [144] généralise l'algorithme aux cas où les probabilités de transition entre deux états ne sont pas symétriques, c'est-à-dire que $p(y_{t+1}|x_t) \neq p(x_{t+1}|y_t)$. Dans ce cas, on doit choisir les move-sets tels qu'ils respectent la condition de réversibilité $p(y_{t+1}|x_t)f(x) = p(x_{t+1}|y_t)f(y)$. Ceci implique qu'on ne peut pas choisir n'importe quelle modification de structure comme move-set. Si cette condition est validée, et que la probabilité d'acceptation d'un état candidat y est modifiée en $\frac{f(y)p(x|y)}{f(x)p(y|x)}$, l'algorithme échantillonnera bien dans la distribution de probabilité souhaitée. Cette formule, aussi appelée critère de Metropolis-Hastings, est ré-utilisée dans de nombreuses méthodes et variantes. Le critère porte sur $A(y|x)$ la probabilité d'accepter le nouvel état candidat y sachant l'état actuel x, et se note plus formellement :

$$A(y|x) = \min\left(1, \frac{f(y)p(x|y)}{f(x)p(y|x)}\right).$$

L'échantillonnage de Gibbs [131] est un autre algorithme, particulièrement adapté aux distributions de probabilité fortement multivariées. Il est très utilisé dans les méthodes d'inférence Bayésiennes. Dans cette méthode, la transition d'un état vers l'autre se fait variable par variable. Supposons qu'une structure moléculaire à l'itération t soit exprimée par k paramètres formant le vecteur x^t. On utilise k distributions de probabilité indépendantes pour chaque paramètre i, dépendantes des $(i-1)$ premiers paramètres déjà échantillonnés, et des valeurs précédentes des $(k-i)$ paramètres restants. À chaque mise à jour d'un paramètre i, on calcule donc $p(x_i^{t+1} \mid x_1^{t+1}, x_2^{t+1}, \dots, x_{i-1}^{t+1}, x_i^t, \dots, x_k^t)$. On répète le processus un grand nombre de fois, et à la fin, la distribution des solutions approxime la distribution jointe de tous les paramètres considérés ensemble. L'échantillonnage de Gibbs n'a pas été appliqué à l'exploration de paysages énergétiques, car l'estimation de ces distributions de probabilités marginales est quasiment impossible. Cependant, on le retrouve dans des méthodes comme DIRECT [164], pour échantillonner des solutions (des prédictions de matrices de contacts entre nucléotides) à partir de distributions de probabilité estimées sur des données.

L'échantillonnage de Langevin [282] est parfois utilisé pour échantillonner des structures 3D à l'issue d'une dynamique moléculaire de Langevin. La simulation du système dynamique propose des nouveaux états (par dynamique de Langevin, une variante de dynamique moléculaire contenant un terme stochastique modélisant les effets de viscosité du solvant). Il n'y a pas de probabilités de transition ici, la succession des états est dictée par la simulation du temps. Cependant, chaque nouvel état est accepté ou non selon le critère de Metropolis. Si l'état est refusé, on re-simule le passage du temps depuis l'état précédent. Le processus étant stochastique, une nouvelle proposition d'état pourra être faite. Plusieurs outils de simulation de l'ARN en 3D utilisent la dynamique de Langevin (par exemple TIS [91]), et pourraient échantillonner des structures par cette méthode (bien que ce ne soit pas toujours le but de l'outil).

Les algorithmes de recuit simulé [1] sont une famille de méthodes inspirées d'un processus métallurgique, le recuit, visant à obtenir l'état cristallin parfait par un refroidissement lent et progressif du métal. Une chute trop brusque de la température conduit à des états cristallins sous-optimaux. Dans les années 80, des ingénieurs d'IBM ont ré-utilisé ce modèle pour proposer un algorithme de Metropolis capable de localiser le minimum global d'une fonction non convexe sans se laisser piéger dans des minima locaux sous-optimaux. Les algorithmes de recuit simulé fonctionnent tous plus ou moins selon la procédure suivante :

1. On choisit un point de départ x au hasard, et une température de départ T.
2. On calcule un voisin de ce point y aléatoirement (par un move-set),
3. On évalue ce point avec la fonction de coût $f(y)$ et on calcule l'écart avec $f(x)$,
4. Si l'écart est négatif (on a amélioré la solution), on accepte la solution y comme nouveau point de départ x, et on recommence à l'étape 2. Si l'écart est positif, on peut quand même accepter y, mais avec une probabilité $e^{-(f(y)-f(x))/T}$. Si y n'est pas accepté, on retourne à l'étape 2 sans avoir changé de solution x.

Au fur et à mesure des itérations de l'algorithme, on diminue la température T (souvent par paliers), et avec elle la probabilité d'accepter une solution qui empire l'état x actuel. Au bout d'un certain temps, la solution x converge vers le minimum global (mais sans garantie). À haute température, $e^{-(f(y)-f(x))/T}$ est proche de 1, donc le système est libre d'explorer l'espace des possibles assez facilement. Si la largeur des bassins d'énergie est à peu près proportionnelle à la profondeur de leur minimum, l'algorithme va se retrouver dans le bon bassin au hasard des itérations, puis converger vers le minimum global. À l'inverse, cette méthode fonctionne moins bien pour les minima très « profonds » dans des bassins très peu larges.

L'algorithme par saut de bassins (ou *Basin Hopping* en anglais) [345] est une variante proposée par David Wales en 1997 particulièrement adaptée à l'exploration des conformations moléculaires. La principale différence avec un recuit simulé est l'application systématique d'une descente de gradient simple avant d'évaluer $f(y)$. On compare ainsi d'une itération sur l'autre non plus différents états quelconques du système, mais différents minima locaux entre eux.

II.1.4.2 La recherche locale (ou algorithme Tabou)

L'algorithme est proposé par Glover [136] en 1986. On définit un voisinage d'une solution x comme l'ensemble des solutions qui peuvent être atteintes à partir de x par un move-set. Si cet ensemble est trop grand, on sélectionne comme voisinage un échantillon des états possibles seulement. On évalue ensuite tous les éléments du voisinage, et on remplace x par son voisin qui a la valeur de f la plus basse (même si cette valeur est plus grande que $f(x)$, ce qui permet à l'algorithme de remonter pour sortir d'un minimum local). Pour éviter à l'algorithme de redescendre dans un minimum local qu'il viendrait de quitter, on entretient une liste d'états « tabous » et contenant les états les plus récemment traversés, qu'on interdit au programme de considérer à nouveau.

Néanmoins cet algorithme est difficile à appliquer à l'exploration des paysages énergétiques, puisqu'il s'applique plutôt mieux à des problèmes discrets que continus.

II.1.4.3 Les algorithmes génétiques

La famille des algorithmes génétiques s'inspire du processus de reproduction sexuée des êtres vivants, suivi d'une étape de sélection naturelle en fonction de leurs qualités.

On définit un individu (une solution) par son « génome » (les valeurs des variables de cette solution) et par sa *fitness* (la qualité de la solution, mesurée par une fonction objectif).

La procédure d'optimisation est la suivante :

1. À partir d'une population de N solutions, on crée N nouvelles solutions en réalisant des croisements : on sélectionne deux « parents » (au hasard, ou parfois sur la base de leur fitness) et on échange les valeurs d'un sous ensemble de leurs variables.
2. Pour chaque « enfant », on effectue une mutation, c'est-à-dire que l'on va changer (de façon absolue ou relative) la valeur de l'une (ou plusieurs) de ses variables. On ajoute les « enfants » ainsi créés à la population initiale, désormais de taille $2N$.
3. La phase de sélection soumet les individus à un processus pour déterminer s'ils seront conservés ou supprimés. De nombreuses stratégies de sélection existent, mais on compte par exemple :
    a. La sélection élitiste : on conserve les N individus de meilleure fitness,
    b. La sélection par tournoi : on forme des paires de façon aléatoire, et on conserve le meilleur individu de chaque paire,
    c. La sélection par rang : on trie les individus par fitness croissante, et chaque individu survit à la sélection avec une probabilité rang/N,
    d. …. et d'autres.
4. On retourne à l'étape 1 jusqu'à satisfaction d'une condition de convergence.

Les algorithmes génétiques sont appréciés pour leur facilité d'implémentation sur tous les problèmes, même avec des fonctions objectif très complexes, des contraintes peu standard, des domaines réalisables discontinus ou non quantitatifs, etc. Cependant, leur convergence est lente et non garantie, et la qualité des solutions retournées est également non garantie et aléatoire. En général, on les considère donc plutôt comme des outils de dernier recours.

II.2 L'optimisation discrète et combinatoire

Lorsque chaque variable peut prendre un nombre dénombrable de valeurs, on parle d'optimisation discrète. Le terme optimisation combinatoire renvoie aux problèmes où l'on cherche à trouver une permutation d'un ensemble ordonnée qui minimise la fonction objectif, ou un objet dans un ensemble. Ils peuvent se transformer en problèmes d'optimisation discrète. Même si la variable x prend des valeurs discrètes, on peut distinguer de nouveau les sous classes de problèmes linéaires, quadratiques, non-linéaire convexes, et non-linéaires non convexes. Dans cette thèse, on s'intéresse particulièrement au cas linéaire, puisque plusieurs outils de prédiction de structures d'ARN utilisent des modèles de programmation linéaire en nombre entiers (*Integer Linear Programming* ou ILP).

II.2.1 La programmation linéaire en nombres entiers (ILP)

Les problèmes linéaires à variables discrètes sont appelés « programmes linéaires en nombres entiers ». Une sous-classe de ces problèmes contient les problèmes à variables binaires, où les variables peuvent seulement prendre les valeurs 0 ou 1. Ils ont la forme générale $\min_{x \in N^m} c^T \cdot x$ avec $c \in \mathbb{R}^m$ dépendant du problème, et sont également soumis à des contraintes linéaires de type $A \cdot x = b$ (pour k contraintes, on a les dimensions $A \in M_{k,m}$ et $b \in \mathbb{R}^m$). Pour une solution x donnée, on appelle le produit à minimiser $c^T \cdot x = z$.

Une façon simple de résoudre un problème linéaire en nombre entiers est de résoudre sa relaxation, c'est-à-dire le même problème mais en autorisant les variables de décision à prendre des valeurs non-entières. Cependant, le domaine réalisable de la relaxation est plus large que celui du problème discret : il est possible de trouver une solution « trop » optimale. Arrondir la solution de la relaxation à des valeurs discrètes ne garantit pas l'obtention d'une solution optimale : parfois, le résultat est juste faux, comme illustré sur la Figure 2.1.

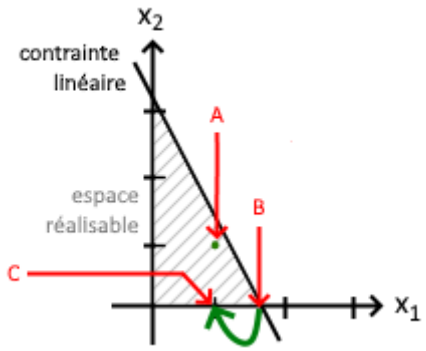


Figure 2.1 – Solutions d'un exemple d'ILP et de sa relaxation. *On considère un exemple arbitraire d'espace de recherche à deux variables x_1 et x_2, sous les contraintes $x_1 \geq 0$, $x_2 \geq 0$, et une troisième contrainte arbitraire représentée par la droite noire. On suppose l'existence d'une fonction de ces variables à minimiser telle que la solution optimale soit le point A si l'on considère seulement les valeurs entières des variables, et le point B si on autorise les valeurs non-entières. Alors, arrondir la solution B à l'entier le plus proche donne le point C, qui n'est pas le point A.*

La famille d'algorithmes couramment utilisée pour résoudre les problèmes linéaires en nombres entiers s'appelle *branch-and-bound*. Ces algorithmes permettent d'énumérer les solutions potentielles en parcourant un arbre de décision, et fonctionnent donc également avec des problèmes d'optimisation combinatoire ou des problèmes mixtes (dont certaines variables doivent être entières et d'autres réelles). Chaque branche de l'arbre considère typiquement un intervalle borné ou une valeur fixe pour l'une des variables. Par un mécanisme de calcul de bornes, l'algorithme identifie les branches qui mèneront nécessairement à des solutions non optimales et empêche leur exploration. L'algorithme général suit le déroulement suivant :

1. Poser $z^* = +\infty$
2. Choisir une variable de branchement x_j parmi les variables entières, dont la valeur optimale x_j^* dans la relaxation est décimale. Créer deux sous-problèmes en ajoutant respectivement les contraintes $x_j \leq \lfloor x_j^* \rfloor$ et $x_j \geq \lceil x_j^* \rceil$.
3. Pour chaque sous-problème, calculer le $z^{*\prime}$ de la branche : c'est une borne inférieure pour z^* pour toutes les valeurs optimales qu'on pourrait obtenir avec cette branche.
4. Marquer les sous-problèmes comme traités si :
    a. On a un $z^{*\prime} \geq z^*$ (la branche est inintéressante),
    b. Il n'y a pas de solution à la relaxation (la branche est infaisable),
    c. La solution a des valeurs entières sur toutes les variables souhaitées, on met alors à jour $z^* = z^{*\prime}$.
5. Tant qu'il y a des sous-problèmes non traités, retourner à l'étape 2 pour chacun. Sinon, quitter, et z^* est optimal.

De nombreuses variantes de l'algorithme existent, notamment des versions parallélisées sur CPU multicœur, certaines étant plus efficaces que d'autres dans le cas moyen. Ces algorithmes sont implémentés dans des solveurs généralistes (CPLEX, Gurobi, GNU GLPK, et d'autres), et dans le cas des solveurs propriétaires, ils sont parfois tenus secrets.

II.2.2 Modélisation de structures d'ARN avec l'ILP

Le problème de minimisation de l'énergie de la structure secondaire d'un ARN de séquence donnée a été formulé comme un programme linéaire en nombres entiers pour la première fois par Poolsap, Kato et Akutsu [259] en 2009. Une version améliorée est à l'origine d'IPKnot [297], par le même groupe de recherche en 2011. L'année suivante, RNA-MoIP [272] réutilise le modèle, puis plus récemment BiokoP développé à IBISC en 2018 [192], et moi-même avec BiORSEO [26] présenté au chapitre suivant.

Ces formulations sous forme de programme linéaire en nombre entiers permettent la recherche dans l'ensemble exhaustif des structures secondaires avec pseudonoeuds arbitraires, ce qui n'a jamais été supporté par les schémas de programmation dynamique dérivés de l'algorithme de Zuker [396] utilisé dans les . Au cours de l'algorithme de branch-and-bound, seules les branches intéressantes sont explorées et l'énumération exhaustive

n’est donc pas nécessaire. L’ILP constitue donc un cadre théorique prometteur pour les méthodes de prédiction de structures secondaires.

Cependant, en pratique, les outils précédemment cités utilisaient des formulations simples (linéaires) comme fonction d’énergie à optimiser. Ils se contentent de comptabiliser les contributions énergétiques des appariements canoniques. Il n’y a à ce jour pas de formulation aussi détaillée que le modèle énergétique de Turner 1999.
De plus, la résolution du problème ILP est en pratique souvent plus lente que les algorithmes polynomiaux par programmation dynamique.

II.3 L'optimisation multicritère

Dans cette partie, on s'intéresse au cas où la fonction à optimiser est à valeurs dans $\mathbb{R}^n$ avec $n > 1$, c'est-à-dire lorsqu'on essaie d'optimiser plusieurs fonctions en même temps. La partie difficile du problème est d'arriver à « ne pas choisir » entre les différentes fonctions, à ne pas accorder de priorité arbitraire à l'une ou l'autre.
Les problèmes étudiés dans la suite du document peuvent s'écrire sous la forme :

$$\underset{x \in X}{\operatorname{argmin}} F(x), \quad X \subseteq \mathbb{R}^m, \quad F(x) \in \mathbb{R}^n,$$

avec X un sous-espace de $\mathbb{R}^m$, défini par exemple par des contraintes. On a donc m variables, et n fonctions objectifs F_1 à F_n, dont les images d'une solution x sont rassemblées en un vecteur $F(x)$ de $\mathbb{R}^n$.

Tout comme nous avions déjà étudié des conditions d'optimalité nécessaires et suffisantes dans la première partie, les méthodes présentées dans celle-ci sont parfois :

- des formulations *« nécessaires »*, qui permettent de trouver des points optimaux ou non, mais tous les optimaux sont supposément atteignables par la méthode,
- des formulations *« suffisantes »*, qui permettent d'atteindre des points dont l'optimalité est garantie. En revanche, elles ne permettent pas toujours de trouver tous les points optimaux.

Les formulations suffisantes sont les plus utiles en pratique. Malheureusement, il existe surtout des méthodes nécessaires qui identifient des points « faiblement » optimaux ou pire.

On distingue deux grandes familles de résolution : des méthodes *a priori* ou méthodes de scalarisation, qui définissent une fonction d'utilité à partir des différents critères et retournent une unique solution. La fonction d'utilité demande généralement le choix de poids ou coefficients qui sont une façon de donner la préférence *a priori* à certains critères par rapport à d'autres, d'où le nom de ces méthodes. Résoudre successivement le problème en faisant varier les coefficients permet de trouver plusieurs solutions et de dessiner le front de Pareto (voir définitions au paragraphe suivant). À l'opposé, les méthodes *a posteriori* cherchent à déterminer le front de Pareto (ou une approximation) sans organiser les préférences à l'avance ni définir de poids.

Une vue d'ensemble de ces méthodes rappelées ici est présentée plus en détail par exemple dans les références ([66,114,222]). Une référence fortement citée par les précédentes est le livre de Miettinen [240].

II.3.1 Vocabulaire de l'approche de Pareto

On définit un certain nombre de termes utilisés tout le long de cette thèse, et dont certains sont illustrés sur la Figure 2.2.

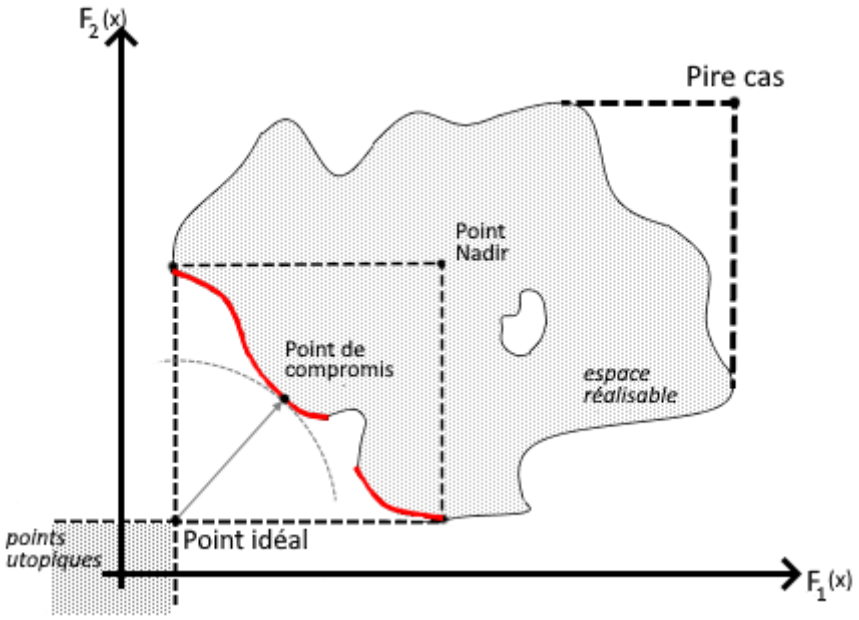


Figure 2.2 – Points d'intérêt dans l'espace des critères. *Visualisation d'un espace de recherche non convexe arbitraire pour un problème d'optimisation bi-objectif fictif. Le front de Pareto est mis en valeur en rouge.*

Solutions et points : On appelle solution un élément de l'ensemble X. Le terme point peut désigner aussi bien un élément de X que l'image de cet élément par F. Généralement, l'usage du mot « point » désigne les deux à la fois.
Dominance : Une solution A domine une solution B si les valeurs de ses objectifs sont aussi bonnes ou meilleures que celles de B, avec au moins une valeur strictement meilleure sur au moins un critère.
Pareto-optimalité (forte) : Un point x^* est dit Pareto optimal s'il n'existe pas d'autre point x tel que $F(x) \leq F(x^*)$ et $F_i(x) < F_i(x^*)$ pour au moins un indice i. En d'autres termes, aucune autre solution ne fait mieux que x^* sur tous les objectifs à la fois. Ou encore, aucune autre solution ne permet d'améliorer x^* sur au moins un objectif sans perdre sur un ou plusieurs autres. On appelle ces points les points « Pareto-optimaux », ou parfois « points efficaces », ou encore « points non dominés ».
Efficacité et dominance : Quand on parle de point, on parle à la fois de l'antécédent dans l'espace des variables et de son image dans l'espace des critères. Dans l'espace des variables, les antécédents forment l'ensemble de Pareto (*Pareto set*) et sont appelés solutions « efficaces » lorsque le point est Pareto-optimal. Leurs images dans l'espace des fonctions forment le front de Pareto (*Pareto front*) et sont appelés points « dominants ». Lorsqu'on parle d'un point en général, les termes « efficace » et « dominant » sont équivalents.
Pareto-optimalité faible : Un point x^* est dit faiblement Pareto-optimal, ou faiblement efficace, s'il n'existe pas x tel que $F(x) < F(x^*)$. C'est-à-dire, aucun autre point x n'est meilleur sur tous les objectifs en même temps. On peut quand même trouver des points qui améliorent l'un des $F_i(x)$ sans changer la valeur des autres objectifs.
Pareto-optimalité propre : Un point x^* Pareto-optimal est proprement Pareto optimal si en plus, il existe $M > 0$ tel que pour toutes les autres solutions x pour lesquelles on peut améliorer l'un des objectifs $F_i(x) < F_i(x^*)$, on dégrade au moins un autre objectif d'au moins $\Delta F_i/M$. On appelle cette relation de dominance la « dominance au sens de Geoffrion ».

Autrement dit, un point est proprement Pareto-optimal si $\frac{\Delta F_j}{\Delta F_i} \geq M$, pour tout (i, j), i étant un critère amélioré en passant de x^* à x et j un critère dégradé.

Points extrêmes : Parfois appelés points d'ancrage, ce sont les points du front de Pareto obtenus lorsqu'on minimise chaque critère de façon individuelle. Ils sont donc aux « extrémités » du front, d'où leur nom.

Points idéal et utopiques : Le point idéal d'un problème d'optimisation est obtenu en prenant les meilleures valeurs obtenables sur chacun des objectifs pris indépendamment. Le point idéal domine tout le front de Pareto et tout l'espace réalisable. Il n'est en général pas réalisable lui-même. L'ensemble des points formé par le point idéal et tous les points qui le domineraient est appelé ensemble des *points utopiques*. La notation $F°$ désigne un point utopique.

Point Nadir : Le point Nadir est obtenu en retenant les pires valeurs des différents critères observées sur l'ensemble des points qui sont les meilleurs sur l'un des critères lorsque pris tout seul. Ensemble, le point idéal, le point Nadir, et les points extrêmes du front de Pareto délimitent un hypercube contenant tout le front de Pareto. Il est inutile de chercher des solutions en dehors de cet hypercube.

Point de compromis : Le point le plus proche du point idéal dans le domaine réalisable. Il fait partie du front de Pareto. La mesure de distance couramment utilisée est la distance Euclidienne.

Rang de Pareto : Si une solution appartient à l'ensemble de Pareto, son rang vaut 1. Si on oublie ou met de côté ces solutions, d'autres deviennent alors non dominées : elles forment l'ensemble de Pareto de rang 2. On peut à nouveau les oublier pour déterminer un ensemble de rang 3, et ainsi de suite.

Pour s'affranchir des effets d'échelle liés aux unités des différents critères, il est parfois utile de normaliser les critères avant de chercher à résoudre le problème multi-objectif. Plusieurs normalisations existent, par exemple :

- $F_i^{trans}(x) = \frac{F_i(x)}{\max\limits_{1 \leq j \leq n} F_i(x_j^*)}$, où x_j^* est la solution optimale du critère j ($x_j^* = \underset{x \in X}{\operatorname{argmin}} F_j(x)$).
- $F_i^{trans}(x) = \frac{F_i(x) - F_i°}{F_i°}$, où $F°$ est un point utopique, à valeurs strictement positives (la « différence relative »),
- $F_i^{trans}(x) = \frac{F_i(x) - F_i°}{\max\limits_{1 \leq j \leq n} F_i(x_j^*) - F_i°}$, un mélange des deux précédentes (la « normalisation »).

Dans les sections qui suivent, la notation $F(x)$ est utilisée, mais il faut plutôt optimiser l'une de ces trois normalisations de F et non pas F elle-même, pour s'affranchir des différences d'échelle.

II.3.2 Les méthodes de scalarisation (ou méthodes à priori)

Ces méthodes permettent de transformer le problème multi-objectif en problème mono-objectif à l'aide d'une « fonction d'utilité » (une fonction des fonctions objectifs) à minimiser. Elles permettent de renvoyer une unique solution. On peut faire varier leurs poids et paramètres pour obtenir différentes solutions, et en répétant suffisamment de fois l'opération, échantillonner le front de Pareto.

II.3.2.1 Pondération polynomiale des objectifs, « compromise programming » et « goal programming »

Ces approches consistent à optimiser une somme pondérée des objectifs, c'est-à-dire choisir des poids w_i et résoudre $\underset{x \in X}{\operatorname{argmin}} \sum w_i \cdot F_i(x)^p$, avec p un nombre entier constant, le degré du polynôme. Le cas le plus fréquemment observé est $p = 1$ (pondération linéaire des objectifs), avec des poids positifs dont la somme vaut 1. Le choix des w_i n'est pas trivial, et nécessite une connaissance *à priori* de l'importance relative que l'on donne aux critères. Chaque choix donnera une solution différente. L'optimisation de la fonction d'utilité (linéaire, ou polynomiale) donne une valeur Pareto-optimale.
Si $F(x) \geq 0$, cette méthode est nécessaire et suffisante, c'est-à-dire que pour tout point optimal il existe un p et un ensemble de poids qui conduisent à trouver cette solution [222]. Néanmoins quand p est faible (dont $p = 1$), la méthode ne permet pas de trouver les solutions Pareto-optimales présentes dans les zones concaves du front de Pareto. On appelle ces solutions des solutions « non supportées ». Une représentation visuelle des limites des pondérations à faible p est présentée sur la Figure 2.3.

Dans le cas où l'on n'a pas $F(x) \geq 0$ pour tout x, on peut transformer les critères en leur soustrayant la valeur du point idéal. On ne pondère alors plus $F(x)$ mais la distance $F(x) - F°$. Cette formulation est équivalente à la recherche d'un point de compromis avec une distance de norme d'ordre *p* (le *compromise programming*) [222]. La fonction d'utilité a donc la forme :

$$\underset{x \in X}{\operatorname{argmin}}\, U = \sqrt[p]{\sum_{i=1}^{n} w_i (F_i(x) - F_i^{\circ})^p}$$

Dans de nombreux cas, par exemple lorsque l'un des critères est fortement non linéaire, il est difficile de déterminer de façon exacte le point idéal $F°$. Dans ce cas, on utilise un point « d'aspiration », point de « référence » où point « cible », toutes ces expressions désignant une approximation du point $F°$. Le *compromise programming* est nécessaire et suffisant tant que cette estimation de $F°$ est bien en dehors du domaine réalisable dans l'espace des critères. Dans ces derniers cas, U est parfois appelée « *achievement function* ». Notons que le calcul de la racine n'apporte rien et que pour des raisons d'efficacité, l'ignorer ne change ni la vitesse de convergence ni la qualité des solutions obtenues.

On peut aussi trouver dans la littérature le terme de *« goal programming ».* Cette

méthode cherche à minimiser la déviation de chaque critère $F_i(x)$ à un objectif b_i au moyen de deux variable positives mesurant la déviation positive et négative d_i^+ et d_i^- :

$$\underset{x\in X, d_i^+, d_i^-}{\operatorname{argmin}}\ U = \sum_{i=1}^{n} (\, d_i^+ + d_i^-)$$

soumis aux contraintes $d_i^+, d_i^- \geq 0$, $d_i^+ \cdot d_i^- = 0$, $F_i(x) + d_i^+ - d_i^- = b_i$ pour tout critère i.

En pratique, lorsque le « but » b utilisé est le point $F°$, les d_i^- sont tous nuls et le problème est mathématiquement équivalent au *compromise programming* avec une pondération égalitaire. Si le but choisi n'est pas $F°$ ou un point utopique, les solutions obtenues ne seront pas forcément Pareto-optimales.

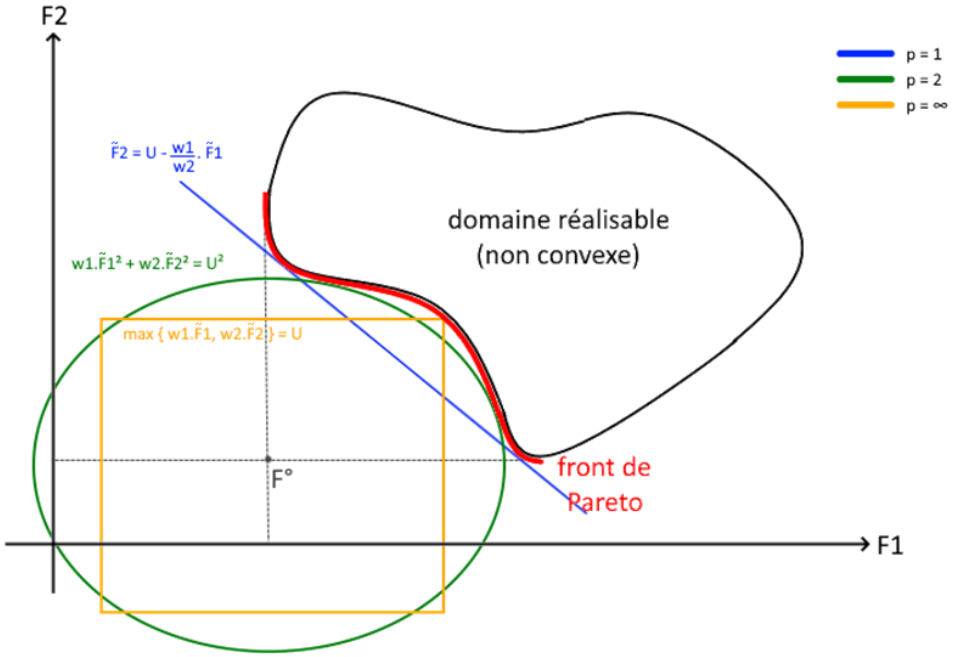


Figure 2.3 - Méthodes de pondération polynomiale des objectifs avec un domaine réalisable non convexe. *La notation $\tilde{F}_i$ dénote la différence $F_i - F_i^\circ$. La pondération linéaire cherche une droite dans le plan adjacente au front de Pareto et de plus petite ordonnée à l'origine, et renvoie le ou les points d'adjacence. Faire varier les poids fait varier la pente de la droite et permet de découvrir d'autres solutions. On remarque que de nombreuses solutions dans la partie concave ne peuvent pas être découvertes par cette méthode (solutions non supportées). La pondération quadratique cherche la plus petite ellipse dans le plan, elle aussi adjacente au front de Pareto. Elle permet de découvrir un peu plus de solutions, mais il reste toujours des solutions non supportées. Plus on augmente p, plus on va pouvoir pénétrer la partie concave de l'espace réalisable. Enfin, la norme infinie (ou min-max, ou méthode de Tcheybycheff) est nécessaire et suffisante : tous les points du front de Pareto sont atteignables. Ici, les courbes U représentées sont des fonctions de $\tilde{F}_i = F_i - F_i° \geq 0$ et non pas directement de F_i. Elles sont donc toujours suffisantes, et deviennent nécessaires au fur et à mesure que p augmente, selon la concavité du domaine réalisable (il existe un p minimum tel que U est une formulation nécessaire).*

II.3.2.2 Le Min-Max pondéré (ou méthode de Tcheybycheff)

Si l'on pousse le *compromise programming* à l'extrême en augmentant p à l'infini, on obtient une norme infinie (mais pondérée), qui s'exprime avec des min et max (remplaçant les exposants et racines) :

$$\underset{x \in X}{\operatorname{argmin}}\, U = \max_i \; w_i \, (F_i(x) - F_i^{\circ})$$

Cette méthode est nécessaire : tous les points optimaux peuvent être atteints avec le bon ensemble de poids, elle ne manque pas de solutions non supportées, tant que l'estimation de $F°$ est bien en dehors du domaine. Elle est aussi suffisante pour la Pareto-dominance faible : tous les points qu'elle trouve sont au moins faiblement Pareto-optimaux [222].

II.3.2.3 La résolution lexicographique

Dans cette approche, on trie les objectifs par ordre de priorité, puis on résout une succession de problèmes d'optimisation mono-objectifs, en utilisant les minima obtenus sur les critères précédents comme contraintes d'égalité sur les critères suivants. Cette méthode aboutit aux points extrêmes du front de Pareto (et uniquement à eux si la résolution monocritère est exacte).
Une variation classique, la « méthode hiérarchique », n'utilise pas des contraintes d'égalité mais autorise une certaine variation (directe, ou relative) des précédents critères par rapport à leur valeur optimale observée. Ceci permet d'obtenir une solution « proche » d'un point extrême du front de Pareto, mais pas le point extrême en lui-même (souvent jugé « trop » extrême, et qu'on peut obtenir simplement en résolvant le problème d'optimisation monocritère concernant le critère de plus grande priorité).

II.3.2.4 La méthode ε-contrainte

Pour n critères, cette méthode consiste à choisir un ensemble de $(n-1)$ paramètres ε_i qui seront des bornes supérieures dont on interdit le dépassement sur $(n-1)$ critères. On résout ensuite le problème d'optimisation monocritère sur le critère restant, jugé le plus important, les autres critères étant utilisés dans de simples contraintes d'inégalité. Répéter l'opération en faisant varier les paramètres ε_i permet de dessiner le front de Pareto.
Formellement, on résout :

$$\underset{x \in X}{\operatorname{argmin}}\, U = \; f_1(x), \qquad \text{soumise à } f_i(x) \leq \varepsilon_i \quad \forall i \in [\![2, n]\!].$$

La méthode est nécessaire, et suffisante pour la Pareto-dominance faible [222]. Elle est très utilisée car simple d'implémentation en pratique.

II.3.2.5 L'intersection de frontière (Boundary Intersection)

On choisit un vecteur $\vec{d}$ qu'on applique au point $F°$, et on regarde à quel endroit la direction définie par ce vecteur et passant par $F°$ intersecte le front de Pareto.

On cherche donc à résoudre :

$$\underset{x\epsilon X}{\operatorname{argmin}} \min_{t\geq 0} t \text{ sous la contrainte } F(x) = t \cdot \vec{d} + F°.$$

C'est une méthode a priori, en effet, le vecteur $\vec{d}$ correspond à un vecteur de poids w_1, … , w_n accordant plus ou moins d'importance à certains critères.

Une variante ne fait pas varier $\vec{d}$ pour obtenir de nouvelles solutions, mais propose plutôt d'appliquer $\vec{d}$ à tout point (ou à un échantillon de points) d'un hyperplan de dimension $(n-1)$ passant par $F°$.

Il n'est pas garanti que la direction choisie intersecte réellement le front de Pareto si celui-ci est discontinu. La résolution peut échouer. La résolution peut aussi aboutir à un point sur la frontière du domaine réalisable dans une portion qui ne fait pas partie du front de Pareto. Cette méthode est donc nécessaire (il existe toujours une direction reliant $F°$ et un point Pareto optimal), mais pas suffisante.

II.3.2.6 Pondération exponentielle des critères

La fonction d'utilité U à minimiser est une pondération des exponentielles des critères.

$$\underset{x\epsilon X}{\operatorname{argmin}} U = \sum_{i=1}^{n} (e^{p\cdot w_i} - 1) e^{p\cdot F_i(x)}$$

Cette méthode est nécessaire et suffisante [222], ce qui en fait une méthode d'intérêt. Cependant, des problèmes numériques (de type *overflow*, ou de précision) sont fréquemment rencontrés.

II.3.2.7 Pondération du produit

Cette approche est peu testée dans la littérature [222]. Elle est utilisable quand les fonctions sont de magnitude différente mais pas normalisées (ou difficilement normalisables).

$$\underset{x\epsilon X}{\operatorname{argmin}} U = \prod_{i=1}^{n} F_i(x)^{w_i}$$

II.3.2.8 Programmation physique

La « programmation physique » [232] est un cadre de travail incorporant un maximum d'informations disponibles dans le problème sans utiliser de poids.

Le domaine de définition de chaque critère est découpé en régions (hautement inacceptable, inacceptable, tolérable, désirable et hautement désirable). On définit ensuite une fonction de préférence à partir de ces régions, des contraintes, et de toute information disponible. L'algorithme propose ensuite la solution optimale sans avoir besoin d'un processus itératif. Les solutions renvoyées sont Pareto-optimales [235] (la méthode est suffisante).

II.3.3 Les méthodes d'énumération (ou méthodes a posteriori)

Ces méthodes cherchent à déterminer l'ensemble du front de Pareto pour proposer le plus de solutions possibles en tant que résultat de la procédure d'optimisation.
Libre au modélisateur de choisir ensuite les régions du front de Pareto qu'il considère comme intéressantes.

De façon générale, il est possible d'utiliser les méthodes *a priori* avec un schéma d'exploration générique de leurs paramètres, ce qui permet d'obtenir un échantillonnage de solutions sur le front de Pareto. Les méthodes présentées ici expliquent comment choisir ce « schéma d'exploration », pour obtenir des points régulièrement espacés sur le front de Pareto, ce qui ne va pas de soi.

II.3.3.1 Normal Boundary Intersection (NBI)

La méthode NBI [82] propose une façon de transformer le vecteur de poids pour que des variations régulières de ce vecteur aboutissent à des espacements réguliers entre points sur le front de Pareto, même si celui-ci est non-convexe.

À partir des points extrêmes du front (les minima individuels), on calcule l'équation de l'hyperplan qui contient l'enveloppe convexe des minima individuels (abrégée en CHIM), qui est un simplexe. En 2 dimensions, il s'agit d'un segment entre les deux points extrêmes, en 3D, un triangle, et en dimension $k+1$, un k-simplexe. La méthode NBI propose un système de coordonnées sur ce simplexe permettant d'y échantillonner des points. C'est ce qui la différencie de l'intersection de frontière « à priori » où l'on doit choisir arbitrairement des directions dans lesquelles chercher les intersections, ce qui revient à choisir des pondérations des critères. Ici, NBI propose une façon de choisir ces directions qui ne demande pas d'ordonner de préférences sur les critères. Pour chaque point sur le simplexe, on considère la droite normale à l'hyperplan : dans les cas bien construits, elle coupe le front de Pareto en un point. On résout donc le problème d'optimisation sous la contrainte que la solution soit la plus proche possible de l'hyperplan sur la droite normale au simplexe, à partir d'un de ces points. Une illustration est proposée sur la Figure 2.4A.
Mathématiquement, l'hyperplan est défini par une matrice Φ dont les colonnes sont les coordonnées des points extrêmes, et les coefficients de la diagonale mis à zéro. Un vecteur β à coefficients positif ou nuls tels que $\sum \beta_i = 1$ définit les coordonnées d'un point dans la le simplexe. Les coordonnées de ce point dans l'espace des critères sont $\Phi\beta$. On définit ensuite n le vecteur unitaire normal à la CHIM et pointant vers $F°$ (l'origine). On cherche alors à s'éloigner le plus possible de la CHIM dans la direction de n en résolvant :

$$\underset{x\in X, t\in R}{\operatorname{argmax}}\, t \qquad \text{soumis à } \Phi\beta - tn = F(x)$$

En pratique, la méthode échantillonne des points sur toute la surface de l'espace réalisable, et comme pour l'intersection de frontière, elle peut rendre des points qui ne sont pas Pareto-optimaux dans des portions de la frontière du domaine qui seraient dominées par d'autres. Si le domaine réalisable n'est pas continu, certains points peuvent aussi échouer

(et ne pas trouver de solution faisable). Enfin, en dimension supérieure à 2, la méthode peut rater certaines solutions Pareto-optimales proches des axes. La méthode n'est donc ni nécessaire, ni suffisante. De nombreuses variantes et améliorations existent.

II.3.3.2 Normal Constraint (NC)

Afin d'améliorer NBI, Messac *et al.* ont proposé la méthode NC [233]. Les critères y sont normalisés de façon à placer le point idéal à l'origine et les différents points extrêmes aux coordonnées 1.0 des axes sur lesquels ils ne sont pas optimaux.
Comme dans NBI, un ensemble de points est échantillonné régulièrement dans le simplexe formé par la CHIM. Chacun de ces points est défini par $(n-1)$ coordonnées. En faisant varier indépendamment chacune des coordonnées, on obtient $(n-1)$ hyperplans perpendiculaires à la CHIM : on les utilise comme contraintes pour résoudre le problème d'optimisation, et on obtient une solution sur le front de Pareto. En itérant sur tous les points échantillonnés dans le simplexe, on dessine le front de Pareto, avec un espacement régulier. L'algorithme contient un « filtre » qui élimine au fur et à mesure les solutions lorsqu'on découvre qu'elles sont dominées. Avec ces ajustements, cette méthode est nécessaire et suffisante.

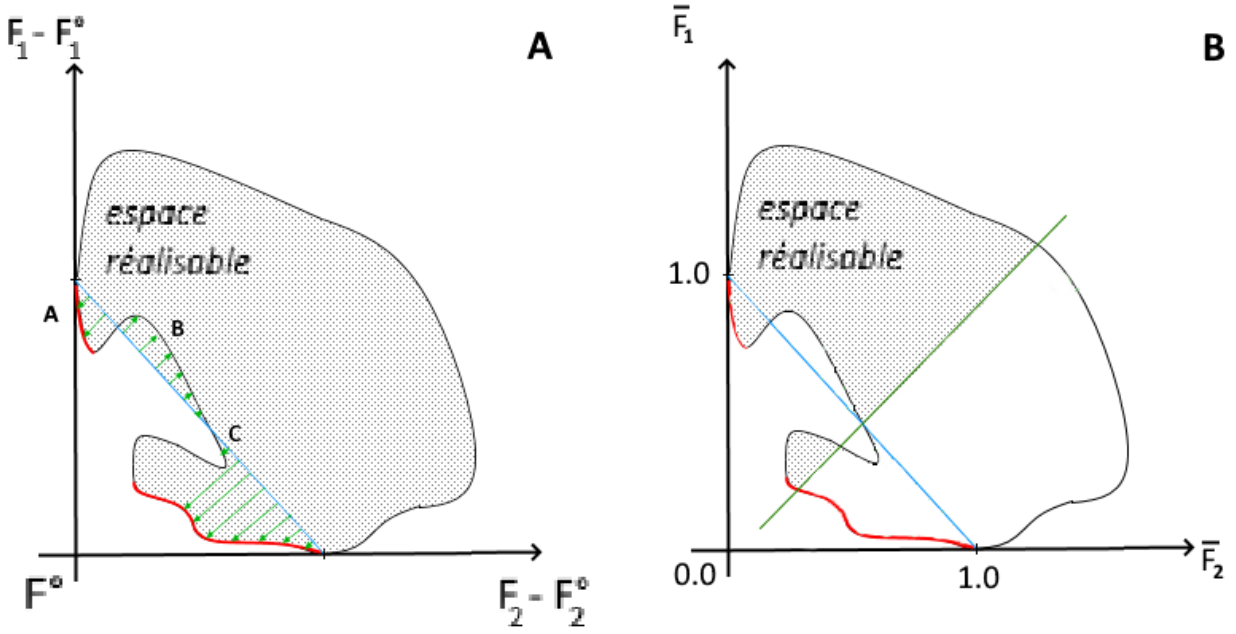


Figure 2.4 : Intersections normales de frontière et contrainte normale. (A) *La méthode NBI échantillonne régulièrement des points sur la CHIM (ici le segment bleu). Les solutions sont cherchées sur les droites normales à ce segment en partant de la CHIM (en vert), et trouvées à l'intersection de la frontière du domaine réalisable. Au point A, des solutions Pareto-optimales sont trouvées. En B, les solutions trouvées sont dominées. En C, la méthode trouve un point qui n'est pas optimal alors que la vraie solution n'est pas trouvée.* ***(B)*** *Pour contourner le problème, NC utilise chaque droite normale comme une contrainte (par exemple ici sur F_2) et résout le problème monocritère sur le premier critère (ici F_1).*

II.3.3.3 Autres

Une extension de la programmation physique (présentée en section 2.3.2.8) a été proposée par son auteur Messac [234] pour proposer une approche a posteriori, qui génère le front de Pareto. La méthode est nécessaire et suffisante.

On citera également les algorithmes SPO [244] (*Successive Pareto Optimization*, 2009) parfois aussi appelé SBG [46], DSD [118] (*Directed Search Domain*, 2011), ou mCHIM [133] (*modified Convex-Hull of individual Minima*, 2015), moins connus mais tous motivés par l'amélioration de NBI.

II.3.4 Méta-heuristiques multicritères

On peut facilement adapter les métaheuristiques utilisées en optimisation monocritère au cas multicritère, pour résoudre des problèmes à critères ou contraintes non linéaires, non différentiables, ou des problèmes difficiles en général. On n'a alors souvent pas de garanties sur leur convergence ou sur la qualité des solutions obtenues.

II.3.4.1 Recuit simulé multicritères

En optimisation multicritères, les méthodes doivent proposer une façon de déterminer si une nouvelle solution « améliore » ou « détériore » l'état actuel. Il faut donc définir de nouveaux opérateurs de comparaison entre solutions. Ces opérateurs peuvent aussi dépendre du contexte global (population de solutions, estimation actuelle du front de Pareto) et ne se limite pas aux deux seules solutions comparées.

PASA (Pareto-Archived Simulated Annealing) [116] **:** Cette méthode utilise une fonction d'utilité pour estimer la qualité d'une solution :

$$U(x) = \sum_{i=1}^{n} \ln f_i(x)$$

Aussi, la différence d'utilité entre l'état actuel x et la nouvelle solution y peut s'exprimer par :

$$\Delta U = U(y) - U(x) = \sum_{i=1}^{n} \ln\left(\frac{f_i(y)}{f_i(x)}\right)$$

En fonction de cette valeur (et de la température actuellement en vigueur), la solution est conservée ou non comme dans le recuit simulé monocritère. Ensuite, l'algorithme entretient une « archive de solutions » pour stocker les points non dominés ayant été observés. De temps à autre, un nouveau départ dans une région aléatoire permet d'explorer plus largement l'espace.

MOSA (Multi-Objective Simulated Annealing) [341] **:** A l'inverse de la méthode PASA, cette méthode définit une probabilité d'acceptation π_i par critère qui vaut 1 si le critère est amélioré, et une probabilité dépendante de la température sinon.
Pour la sélection finale, deux variantes existent :

- soit on calcule une probabilité globale π en faisant le produit des probabilités π_i,

- soit on prend la plus petite probabilité π_i comme probabilité globale π.

Certains introduisent une préférence lexicographique en pondérant les probabilités de chaque critère par des exposants λ_i.

Enfin, la sélection est réalisée à l'aide d'une fonction d'utilité de type combinaison linéaire :

$$U(x) = \sum_{i=1}^{n} w_i . f_i(x)$$

Donc la différence entre l'état actuel x et une nouvelle solution y vaut

$$\Delta U = \sum_{i=1}^{n} w_i . (f_i(y) - f_i(x)).$$

Si $\Delta U < 0$, on accepte y, sinon, on l'accepte avec la probabilité globale π.
Comme dans PASA, une archive permet de garder les solutions dominantes identifiées.

Une méthode qui fonctionne bien est de créer plusieurs populations obéissant à plusieurs vecteurs de pondération w_i, mais de vérifier la dominance en prenant en compte l'ensemble des populations en une seule fois.

II.3.4.2 Algorithmes génétiques multicritères

On distingue de nombreuses variantes d'algorithmes génétiques adaptées au cas multicritère, présentées ci-dessous. Elles diffèrent surtout par leur méthode de sélection, les croisements et mutations étant plutôt spécifiques du problème étudié. Un groupe de méthodes utilise un classement des solutions basé à la fois sur leur rang de Pareto et sur leur contribution à la diversité des solutions (NSGA [320], NSGA-II [88], NPGA [156], SPEA, SPEA2 [389]). D'autres méthodes se basent sur un indicateur unique pour déterminer la fitness d'une solution (VEGA [64], MOGA [123], WARGA [66]). D'autres ne trient pas les solutions de la même façon en fonction de leur rang de Pareto (MOEA/D [381], NSGA-III [87]). On présente brièvement quelques variantes :

- **VEGA (Vector-Evaluated Genetic Algorithm)** [64] : L'algorithme VEGA repose sur un mode de sélection particulier. La population est divisée en sous-groupes de tailles égales, la fitness des individus de chaque groupe étant évaluée sur une fonction objectif différente. On remélange ensuite les sous-groupes pour effectuer la sélection. La performance est assez faible.

- **MOGA (Multi-Objective Genetic Algorithm)** [123] : Cette famille d'algorithmes génétiques utilise la relation de dominance pour déterminer l'efficacité d'un individu. Pour un individu x, on définit son rang comme 1 plus le nombre d'individus qui le dominent. La probabilité de sélection de l'individu est modélisée par une fonction du rang, par exemple une fonction affine décroissante.

- **WARGA (Weighted Average Ranking Genetic Algorithm)** [66] : Une méthode similaire à MOGA, mais le rang est défini différemment, en considérant la dominance sur chaque critère individuellement. Le rang d'un individu x est la somme sur chaque

critère pris indépendamment des nombres d'individus meilleurs que x sur les critères considérés. Ainsi, même les solutions de l'ensemble de Pareto ont un rang élevé (souvent supérieur à 1).

- **NSGA (Non-dominated Sorting Genetic Algorithm)** [320], **NGSA-II, et NGSA-III (Deb & Jain 2014** [87]**)** : Dans NSGA, on utilise une fonction d'efficacité inversement proportionnelle au rang de Pareto. Les solutions non dominées ont donc le rang 1 (et la meilleure efficacité), celles qui ne sont dominées que par des solutions de rang 1 sont de rang 2, et ainsi de suite. Au sein d'un même front de Pareto de rang k, on organise les solutions de façon à maximiser leur diversité. L'efficacité de chaque individu, d'abord issue du rang de Pareto, est pénalisée par une mesure de la proximité à d'autres individus. La sélection se fait ensuite de façon proportionnelle à l'efficacité. NGSA-II est une amélioration publiée en 2002 [88], qui reprend les principes du premier, mais avec une procédure de tri de complexité plus faible ($O(nN^2)$ contre $O(nN^3)$ pour NSGA, n étant le nombre de critères et N le nombre de solutions à trier), une sélection un peu différente, et le support de problèmes d'optimisation avec contraintes. Cette méthode est encore recommandée de nos jours pour les problèmes bi-objectifs. Lors de la sélection, les individus des premiers rangs de Pareto sont tous gardés, puis lorsqu'on arrive au rang où seuls certains seront conservés, c'est la métrique de distance à leurs voisins qui les départage. Enfin, NSGA III est une mise à jour qui rend NSGA plus efficace pour les problèmes avec de nombreux objectifs (4 ou plus), et qui en fait une méthode de référence pour ces problèmes. Un ensemble de points de référence régulièrement espacés sont calculés dans l'hyperplan contenant la CHIM des points extrêmes du front de Pareto en cours de construction, de la même façon que dans la méthode NBI. Des droites de référence reliant le point idéal à chaque point de référence sont également calculées. Comme dans NSGA-II, lors de la sélection, les premiers rangs de Pareto sont tous gardés. Cependant la méthode de sélection pour le dernier rang change : la diversité n'est plus assurée par une métrique de distance aux solutions voisines à prendre en compte dans le calcul de l'efficacité. On utilise à la place une mesure de proximité aux droites de référence, en favorisant les solutions proches de droites de référence dans des régions peu explorées.

- **NPGA (Niched Pareto Genetic Algorithm)** [156] : Une méthode similaire à NSGA, mais la sélection se fait différemment. Elle ressemble à la sélection par duel (une paire d'individus est comparée et un seul des deux sera sélectionné). Le rang de ces deux individus est estimé sur la base de la dominance au sein d'un groupe d'individus de taille fixée (par exemple 10) sélectionnés aléatoirement. Si l'un des deux individus a un meilleur rang au sein du groupe que l'autre, il est sélectionné. S'ils sont à égalité, on les juge sur la métrique de diversité. Cette méthode n'appliquant pas la comparaison à l'ensemble de la population, elle est très rapide d'exécution.

- **SPEA (Strength Pareto Evolutionary Algorithm), SPEA2** [389] **et SPEA2+** [176] **:** À chaque itération, les solutions dominées et non dominées sont séparées dans des

populations différentes. Lors de la reproduction (le croisement), l'un des parents est systématiquement choisi parmi les solutions dominées et l'autre parmi les non-dominées. La population initiale est composée de copies des points extrêmes. L'efficacité des solutions est basée sur le rang, comme dans MOGA. Dans SPEA2, le calcul du rang prend en compte non seulement le nombre de solutions dominant celle qu'on considère, mais aussi le nombre d'autres solutions que celle-ci domine. Ceci limite les cas d'égalité de rang entre solutions dominées par les mêmes individus. En plus de ceci, une mesure de la densité locale est calculée pour chaque solution (environ égale à l'inverse de la distance au $k^{\text{ième}}$ voisin le plus proche). L'efficacité d'une solution est ensuite calculée comme la somme de ces deux termes (rang et densité). SPEA2+ rajoute ensuite le croisement forcé entre individus proches les uns des autres dans l'espace des critères.

- **SMS-EMOA** [113] **(S-metric Selection Evolutionary Multi-Objective Algorithm)** : Cette méthode utilise l'hypervolume (une mesure de la taille de l'espace dominé par le front de Pareto) comme mesure de qualité du front de Pareto estimé dans une population. À partir d'une population de solutions « mères » et « filles », les individus sélectionnés seront l'échantillon de taille N maximisant l'hypervolume de l'espace dominé par l'échantillon.

- **MOEA/D** [381] **(Multi-Objective Evolutionary Algorithm based on Decomposition)** : Cette méthode découpe l'espace de recherche en de nombreuses régions régulières et propose des solutions en résolvant le problème d'optimisation multicritère par une autre méthode (par exemple, la méthode de Tchebycheff) dans chacune des régions. On commence par définir des points de référence dont les coordonnées dans l'espace de recherche sont régulièrement espacées. On détermine aussi un point idéal $F°$. Pour chaque point de référence, on précalcule aussi qui sont ses points de référence voisins. À partir d'une population aléatoire, la phase de sélection se fait en sélectionnant, pour chaque point de référence, deux solutions dans le voisinage et en les croisant. Le résultat est optimisé localement pour se rapprocher du point de référence (sur tous les objectifs). Si ce nouveau candidat est plus proche du point de référence que l'ancien, il le remplace. On met éventuellement à jour l'estimation du point idéal, et on itère. Pendant ce temps, les solutions non dominées sont stockées dans une archive externe.

- **PAES (Pareto Archived Evolution Strategy)** [180] **et M-PAES** [181] **(Mimetic PAES)** : PAES est une idée très générale consistant à utiliser une population de taille 1, à la faire muter, et à accepter la solution « fille » seulement si elle domine la « mère » ou si elle passe un test (des variantes existent) la comparant aux solutions d'une archive de solutions non dominées précédemment rencontrées. Le test en question peut faire intervenir des conditions de dominance, de diversité, ou d'autres. Cependant, on ne peut pas vraiment parler d'algorithme génétique avec une population de taille 1, il s'agit plutôt d'une recherche locale. Des variantes avec λ mutants générés à partir de μ parents sont proposées, mais on identifie alors plus de

spécificité de la méthode si ce n'est l'archivage des solutions non dominées, puisque tout devient générique. M-PAES introduit une étape de recombinaison et une de sélection. Il s'agit alors d'un vrai algorithme génétique, mais incorporant une phase de recherche locale pour toutes les solutions proposées et non dominées.

- **PESA (Pareto Enveloppe-based Selection Algorithm)** [69] **et PESA-II** [68] : Cette variante réalise des croisements uniquement à partir de parents non dominés, mais chaque parent a une faible probabilité d'être muté avant le croisement. Le choix des parents n'est pas fait au hasard, mais en fonction de la densité de solution autour de chaque candidat. Cette densité est définie avec une grille virtuelle quadrillant l'espace des critères comme le nombre d'autres individus dans la même case de la grille. Lors de la sélection d'un parent, deux candidats sont piochés au hasard et seul celui avec la plus faible densité locale sera utilisé. De plus l'archive de points non dominés a une taille maximale. Si la place vient à manquer, certaines solutions des zones très denses sont éliminées pour favoriser le stockage de nouvelles dans des zones moins denses. Dans PESA-II, ce ne sont plus les individus à qui on assigne une fitness mais directement les cases de la grille. Le processus de sélection choisit donc certaines cases, et un seul individu est choisi aléatoirement dans chacune des régions choisies. Ceci permet, selon les auteurs, de faciliter l'expansion rapide du front de Pareto entre les points extrêmes.

CHAPITRE III : Un programme bi-objectif pour la prédiction de structures secondaires

Ce chapitre s'intéresse à un modèle bi-critère ayant servi de travail introductif à la thèse. La démarche et l'algorithme sont fortement inspirés de BiokoP, un outil bi-critère développé dans l'équipe et publié en 2018 [192], qui recherchait les structures secondaires d'un ou plusieurs fronts de Pareto. Le problème bi-objectif utilisait à la fois un critère d'énergie à minimiser (modèle MFE), et un critère sommant les probabilités d'existence des appariements à maximiser (modèle MEA). Dans BiORSEO, on s'intéressera aux modules 3D et à leur détection dans les séquences. Le travail présenté dans ce chapitre a également été publié en Janvier 2020 dans le journal *Bioinformatics* (Oxford Press) [26]. La section III.1 présente les motivations et la démarche. La section III.2 formalise le problème d'optimisation bi-objectif linéaire en nombre entiers, et présente l'algorithme de recherche du front de Pareto. La section III.3 présente des résultats (benchmarks, statistiques, et étude de la position des structures natives sur les fronts de Pareto). On conclut et donne des perspectives en section III.4.

III.1 Prédictions de structure secondaire par la détection de modules

III.1.1 Hypothèse de travail

Comme introduit en section I.2.5, plusieurs groupes de recherche ont proposé des modèles de "modules" (ou motifs 3D lorsque l'isostéricité géométrique est prise en compte, ou réseaux d'interactions lorsque seul le graphe d'interactions non-canoniques compte). Les premiers modèles (notamment Rna3dMotifs [101] et le RNA 3D Motif Atlas [255]) utilisent tous comme base des modules les éléments de structure secondaire (SSE). Ainsi, un module est un modèle plus précis d'une hairpin-loop (HL), internal-loop (IL) ou multiloop (ML), qui intègre des interactions non canoniques, et dans le cas du 3D Motif Atlas, correspond à une certaine occupation de l'espace.

On fait alors l'hypothèse que l'identification à partir de la séquence de la présence d'un module à une certaine position donne un indice sur le type de structure secondaire qu'il serait possible de former à cet endroit. La Figure 3.1 illustre ceci avec une HL. Imaginons (comme sur la figure) que par exemple l'instance HL_59710.1 du 3D Motif Atlas, et son modèle statistique de séquence, corresponde bien à une partie d'une séquence dont on souhaite déterminer la structure secondaire. Cette correspondance est un indice quant à la formation d'une HL à cet endroit. Donc le fait de détecter une séquence déjà observée dans

une boucle n'est un indice pour la prédiction de la structure secondaire qu'en supposant que les bonnes hélices existent de part et d'autre de la boucle. Malheureusement, cette hypothèse est à priori faiblement fondée, puisque la stabilisation de la structure est plutôt due à la présence des hélices, pas à celle des boucles. Les boucles sont également fréquemment influencées par l'environnement direct de l'ARN (interactions avec des ions, ligands, ou autres macromolécules). Cet environnement n'est pas pris en compte dans les approches de prédiction de structure secondaire.

Une première partie de ce travail de thèse a donc consisté à tester cette hypothèse en développant un outil qui utiliserait la détection de modules pour proposer des structures secondaires. Ceci permettrait de conclure, via des benchmarks, à l'utilité ou non de ce critère et donc à la pertinence de l'hypothèse de travail.

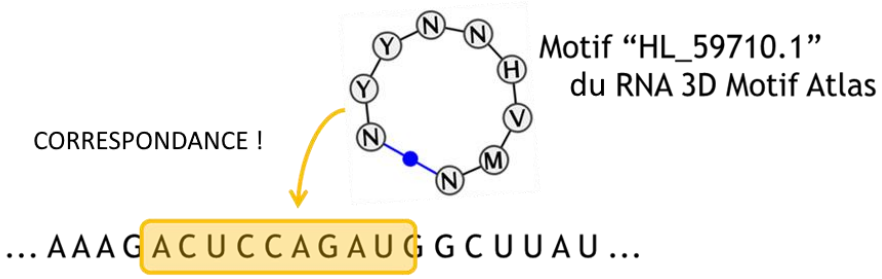


Figure 3.1 – Identification d'une *hairpin-loop* grâce à un module. – *Les lettres N, Y, H, V et M dans la nomenclature IUPAC indiquent la possibilité d'avoir n'importe quel nucléotide (N), un nucléotide à 1 cycle (Y), tout sauf G (H), tout sauf U (V), et A ou C (M). On a ici une correspondance de séquence qui laisse supposer que cette portion peut se replier sous forme de hairpin loop (à supposer que des appariements existent autour pour la stabiliser).*

III.1.2 Outils de détection de modules dans les séquences

Pour plusieurs modèles de modules, des outils ont été développés pour construire des distributions de probabilité des séquences possibles pour chaque module. Ces distributions peuvent ensuite être utilisées pour attribuer un score de vraisemblance (ou une mesure similaire) à une sous-séquence d'ARN qu'on imaginerait se conformer dans l'espace selon un certain module.

Le modèle JAR3D [388] permet d'attribuer à chaque sous-séquence dont on sait si c'est une HL ou une IL les instances les plus probables de modules du 3D Motif Atlas, en y ajoutant un score. Le modèle probabiliste utilisé prend en compte la variabilité possible de la longueur de séquence, inhérente au modèle de modules du 3D Motif Atlas. L'outil est efficace et simple d'utilisation, mais il possède deux défauts : il nécessite d'avoir déjà identifié les HL et IL dans la séquence, et il ne supporte pas les ML (le Motif Atlas n'en contient pas).

Une seconde approche existante est celle des réseaux Bayésiens, qui modélisent les distributions de probabilité des nucléotides sur chaque nœud du graphe d'interactions d'un module, en prenant en compte les dépendances. En effet, les bases ne sont pas indépendantes dans un module, puisqu'un appariement se fera préférentiellement entre

certaines. Les préférences les plus connues d'une base pour une autre sont les appariements canoniques, où A s'associe avec U et G avec C ou U.
En 2011, RMDetect [75] permet de détecter quatre modules connus dans les séquences, à l'aide de réseaux bayésiens construits à la main. Un outil appelé RMBuild (fourni avec) permet d'entrainer de nouveaux réseaux à la main, sur la base d'un fragment de structure 3D correspondant au module plus un alignement de séquences censées former le module. En 2013, metaRNAmodules [335] montre que l'automatisation de la construction de réseaux bayésiens avec RMBuild est possible. C'est un pipeline qui repère les structures 3D de la PDB qui sont à la fois mappées à une famille Rfam (via une table des correspondances fournie par Rfam et construite avec l'outil `cmscan` d'Infernal [246]), et qui contiennent un ou plusieurs modules identifiés dans le 3D Motif Atlas [255]. Le pipeline utilise ensuite les alignements de séquences fournis par Rfam et le fragment de structure 3D identifié comme "module" pour construire un réseau bayésien avec RMBuild. Les réseaux Bayésiens obtenus montrent une bonne sensitivité et spécificité pour détecter les modules en question dans les séquences. Cependant, metaRNAmodules n'est pas prévu comme un outil de détection de modules dans une séquence, mais plutôt comme un benchmark montrant la performance de l'approche. Plus tard en 2019, BayesPairing [293] automatise la construction de réseaux Bayésiens pour tout modèle de module arbitraire (ou plus précisément, de réseau d'interactions) y compris les boucles multiples, et est prévu comme outil de recherche de modules dans les séquences. BayesPairing est très vite amélioré en BayesPairing 2 [294] qui implémente des changements techniques, supporte la recherche à partir d'une séquence seule ou d'un alignement de séquences, et augmente sa performance.

III.1.3 Utilisation des modules pour prédire la structure secondaire dans la littérature

L'utilisation des modules comme source d'informations sur la structure secondaire a déjà été utilisée, notamment par RNA-MoIP dans Reinharz *et al.* [272] en 2012 (qui a inspiré cette étude) et Theis *et al.* en 2015 [336] (découvert pendant la thèse).

RNA-MoIP [272] est un outil proposé par Reinharz, Major et Waldispühl, prenant des propositions de structures secondaires en entrée (par exemple, proposées par un outil comme RNAsubopt [360] ou Sfold [97]). Il est capable de les modifier pour y insérer des modules de Rna3dMotifs qu'il aura détectés sur la base d'une correspondance de séquence. Il renvoie une unique structure secondaire "étendue" (avec des interactions non canoniques venant des modules insérés). L'outil est prévu pour être utilisé en entrée de MC-Sym [217] qui reconstruit ensuite un modèle 3D à partir de la prédiction de RNA-MoIP. L'article montre une bonne performance du pipeline.

En 2015, JAR3D et metaRNAmodules sont utilisés ensemble et avec RNAz [139] pour prédire la position des éléments de structure secondaire de 13 génomes entiers alignés entre eux [336]. L'étude montre que la détection de modules sur les éléments de structure secondaire permet de réduire le nombre de faux positifs découverts (par rapport à une prédiction basée sur RNAz ou RNAalifold [35] seuls).

Cependant, dans ces deux études, l'obtention de structures secondaires se fait d'abord par un outil de prédiction externe, basé sur un autre critère que la présence de modules (l'énergie), les modules ne sont utilisés que pour faire des « corrections ».

III.1.4 Insertion de modules dans les structures par RNA-MoIP

RNA-MoIP a inspiré ce travail, on développe donc ici son fonctionnement un peu plus en détail. Deux aspects du modèle RNA-MoIP sont intéressants : d'une part, la détection des sites d'insertion de modules à l'aide d'expressions régulières dans la séquence. D'autre part, l'optimisation de l'énergie de la structure secondaire des ARN en utilisant un programme linéaire en nombre entiers (ILP), comme les outils IPKnot et BiokoP déjà présentés en section I.4.2.4. RNA-MoIP cherche à optimiser deux critères, l'un visant à respecter la structure secondaire fournie en entrée (qui peut avoir été obtenue en optimisant un modèle énergétique par exemple, comme nous le faisons avec RNAsubopt). L'autre maximise l'inclusion de motifs dans la structure. L'approche « multi-objectif » utilisée est une combinaison linéaire des deux critères. A partir d'une structure donnée en entrée, RNA-MoIP cherche donc à « casser » certains appariements, juste suffisamment pour insérer des modules connus. En insérant des modules dans chacune des propositions de structures données en entrée, RNA-MoIP sélectionne la « meilleure » selon la combinaison linéaire de ses deux critères, et retourne une unique structure secondaire étendue (avec les interactions non canoniques des modules insérés). Cette structure peut ensuite servir de contrainte pour la modélisation 3D, ce qui est l'usage prévu pour RNA-MoIP.

Si l'on arrête le pipeline RNA-MoIP à la prédiction de structures secondaires, ce qui n'est pas la tâche pour laquelle il a été conçue initialement, on constate des imperfections du modèle utilisé. En effet, lorsque RNA-MoIP "casse" certains appariements dans une structure secondaire qu'il modifie pour y insérer des modules, il détruit parfois des appariements importants de la structure secondaire. Sans aucun doute, cette perte peut être justifiée par l'insertion de nouveaux appariements canoniques et non-canoniques et la modélisation en 3D. Cependant, si l'on considère seulement les structures secondaires canoniques, elles en sortent détériorées par rapport à celles fournies en entrée par RNAsubopt, comme montré par le benchmark de la Figure 3.2.

Ensuite, on sait que cette approche d'optimisation multi-objectif par pondération linéaire des critères manque des solutions dites "non-supportées" lorsque le front de Pareto n'est pas convexe (voir l'illustration sur la Figure 2.3). De plus, RNA-MoIP ne cherche pas à faire varier les poids associés à chaque terme, l'optimisation conjointe renvoie donc une seule solution. La pondération choisie l'a été de façon arbitraire et n'a pas été explorée. Il est donc possible que l'outil manque fréquemment des solutions potentiellement intéressantes.

Nous avons donc décidé de tester une approche réellement bi-objectif, à savoir la méthode dite "ε-contrainte", qui cherchera conjointement à optimiser la structure secondaire et l'insertion de modules, et non pas successivement.

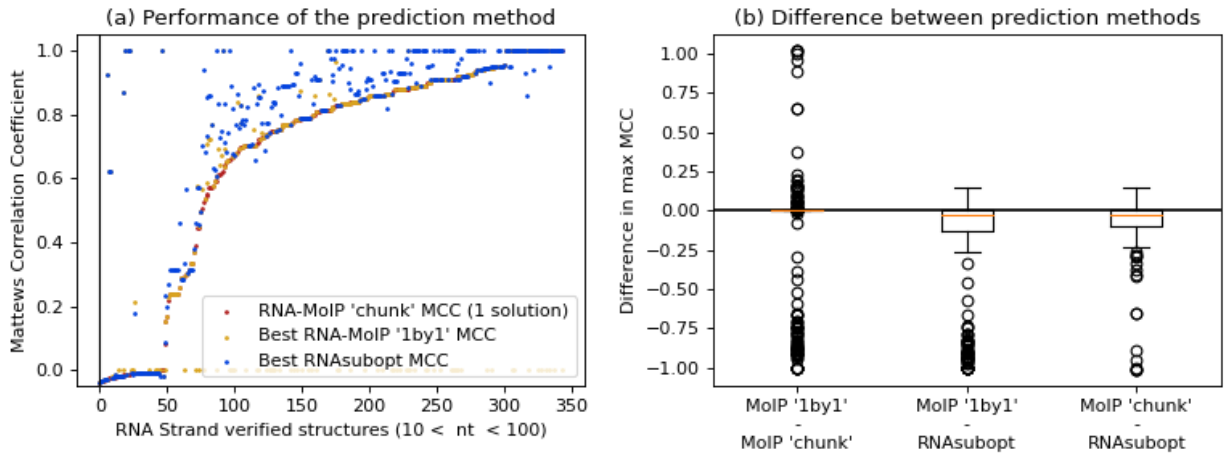


Figure 3.2 – RNA-MoIP détériore les solutions de RNAsubopt. *Statistiques de performance de prédictions de structures secondaires pour 350 petits ARN de la base RNAStrand 2.0* [14]*. À partir d'une "vraie" structure secondaire et d'une prédiction, les nombres de vrais et faux appariements positifs et négatifs détectés sont calculés. Pour prendre en compte le large déséquilibre de classes entre positifs et négatifs (il y a toujours beaucoup plus de paires de nucléotides non-appariés, puisque chaque nucléotide ne peut au plus qu'un seul appariement), on choisit le coefficient de corrélation de Matthews comme métrique de performance (redéfini en section III.3.1). RNAsubopt renvoie plusieurs solutions sous optimales, et seule celle ayant le meilleur MCC est gardée. RNA-MoIP '1-by-1' est la statistique obtenue en donnant chaque structure de RNAsubopt seule à RNA-MoIP, puis en retenant le meilleur MCC après modification par RNA-MoIP. RNA-MoIP 'chunk' est la statistique obtenue en laissant RNA-MoIP déterminer la meilleure solution en lui donnant toutes les structures de RNAsubopt simultanément.* ***(a)*** *Sur l'axe des abscisses sont étalées les 350 molécules d'ARN ordonnées par performance de RNA-MoIP croissante. Une majorité des solutions retournées par RNA-MoIP ont un moins bon MCC que la meilleure proposition de RNAsubopt prise en entrée.* ***(b)*** *Soustractions des MCC comparant les méthodes deux à deux. Chaque point correspond à une molécule d'ARN. La première colonne montre que la sélection de la meilleure solution par RNA-MoIP est relativement aléatoire. Les deux suivantes montrent que RNA-MoIP a tendance à légèrement détériorer les prédictions faites par RNAsubopt.*

C'est la comparaison des meilleurs MCC obtenus par RNAsubopt et par RNA-MoIP '1-by-1' (à nombre de solutions égales, donc) qui montre que RNA-MoIP détériore la meilleure solution qui a été proposée par RNAsubopt. On rappelle que l'utilisateur ne sait pas identifier cette solution dans l'ensemble des solutions sous-optimales renvoyées, dans les deux cas. Le résultat de RNA-MoIP 'chunk' (usage normal de RNA-MoIP) est fourni pour comparaison mais n'apporte rien à la présente analyse. Il renvoie une seule solution qui permet à l'utilisateur de faire un choix parmi les solutions sous-optimales de RNAsubopt ; choix qui n'est donc souvent pas le meilleur, mais au moins on n'a plus qu'une seule structure.

Un des objectifs de notre approche bi-objective est de garder la réduction du nombres de solutions (comme RNA-MoIP) mais sans également détériorer la meilleure solution de l'ensemble des solutions sous-optimales.

III.2 Modèle d'optimisation

III.2.1 Programme linéaire en nombres entiers

L'utilisation avec succès de la programmation linéaire en nombres entiers par les outils IPKnot [297], RNA-MoIP [272], et plus récemment BiokoP [192] dans notre laboratoire nous a naturellement tourné vers la réutilisation de cette méthode de modélisation. En effet, les algorithmes d'ILP ne reposant pas sur la programmation dynamique et l'énumération des structures "nichées" les unes dans les autres, ces outils modélisent naturellement les pseudonoeuds en permettant l'appariement de tout nucléotide avec tout autre (éloigné d'au moins 3 bases). En contrepartie, le modèle d'énergie utilisé avec l'ILP dans ces outils est un modèle simpliste par rapport au modèle de Turner, qui repose sur la sommation de contributions négatives liées aux appariements comptés indépendamment. Il n'y a pas de prise en compte des empilements, empilements coaxiaux, contributions énergétiques connues de certaines boucles, etc…

On définit les notations suivantes (fortement inspirées de RNA-MoIP, IPknot et BiokoP) :

- n le nombre de bases dans la séquence s donnée en entrée,
- p_{uv} la probabilité d'existence de l'appariement entre deux nucléotides u et v, calculée à l'avance par l'outil Nupack 3.2 basé sur la méthode de Dirks & Pierce [100], proche de celle de McCaskill [229] mais choisie pour son support des pseudonoeuds simples,
- M l'ensemble des modules potentiellement insérables quelque part dans s,
- pour chaque module x de M, $\|x\|$ le nombre de composantes distinctes de x, c'est-à-dire le nombre de suites de nucléotides contigus reliant deux hélices (1 pour une HL, 2 pour une IL, k pour une k-ML). Dans le cas des modules de CaRNAval, elles ne relient pas forcément des hélices, ce sont juste des suites de nucléotides contigus.
- $p(x)$ le score d'insertion associé à x renvoyé par JAR3D ou BayesPairing (présentés en III.1.2),
- $P_{x,i}$ la position dans s où l'on pourrait insérer la $i$ème composante de x. Si un module peut potentiellement être inséré à plusieurs endroits différents dans s, on considère qu'il s'agit de plusieurs instances x de M différentes, même si l'instance du module dans la base de données de modules est la même,
- $k_{x,i}$ le nombre de bases dans la $i$ème composante de x,
- pour les modules de CaRNAval, Y_x l'ensemble des appariements canoniques (u,v) forcés par l'insertion du module x. Pour les autres bases de données de modules, qui correspondent à des boucles (HL, IL, ML), cet ensemble contient les appariements fermant la boucle,
- pour les modules de CaRNAval toujours, $A_{x,i}$ le nombre de nucléotides de la $i$ème composante de x impliqués dans un appariement (de Y_x). Pour les autres types de modules, ce nombre vaut toujours 2.

On introduit ensuite les variables de décision de notre problème d'optimisation linéaire en nombre entiers :

- Soit y_v^u la variable de décision booléenne valant 1 ou 0 selon que l'on insère ou non un appariement canonique entre les nucléotides $s[u]$ et $s[v]$. On ne s'autorise à former des appariements que si $v > u + 3$. De plus, comme dans Ipknot et BiokoP, pour réduire le nombre de variables peu utiles, on ne définira une variable y_v^u que si p_{uv} est supérieure à un paramètre seuil θ.
- Soit C_i^x la variable de décision booléenne valant 1 ou 0 selon que l'on insère ou non la $i$ème composante du module x à la position $P_{x,i}$.

Notons que l'on autorise les appariements que si $v > u + 3$, et que l'on n'a pas besoin de deux variables y_v^u et y_u^v pour modéliser le même appariement. Nous avons donc $\sum_{i=4}^{n}(n-i)$ variables de décision de type y_v^u (environ $\frac{1}{2}n^2$). Si l'on ajoute à celà une moyenne de μ modules potentiellement insérables par séquence, les modules ayant en moyenne γ composantes, pouvant chacune être placée à en moyenne π positions dans s, on a également $\mu \times \pi^\gamma$ variables de type C_i^x, soit environ $\frac{1}{2}n^2 + \mu\pi^\gamma$ variables de décision en tout. En pratique, μ dépend complètement de la longueur de la séquence donc sa valeur est difficile à généraliser (mais nos résultats sur la Figure 28A pour des séquences courtes montrent que μ ne dépasse pas 5 environ). Le nombre de composantes moyen γ dépend du jeu de données de modules considéré ($\gamma \in \{1,2\}$ pour le 3D Motif Atlas, $\gamma \in \{1,2,3\}$ pour Rna3Dmotifs, $\gamma \in N^*$ pour CaRNAval). Et enfin, le nombre moyen de sites d'insertions possibles pour chaque composante est également très variable, le plus souvent de 1 pour des composantes longues dans des séquences courtes, il peut atteindre plusieurs dizaines pour des composantes courtes dans des séquences longues pour certains modules.

Formulons maintenant des contraintes sur ces variables, pour qu'elles restent cohérentes entre elles :

1. On n'autorise au plus qu'un seul appariement par nucléotide :

$$\sum_{v<u} y_u^v + \sum_{v>u} y_v^u \leq 1, \quad \forall u \in [\![1,n]\!].$$

2. On interdit les appariements solitaires qui ne sont pas stabilisés par un empilement :

$$y_{v+1}^{u-1} - y_v^u + y_{v-1}^{u+1} \geq 0, \quad \forall (u,v) \in \{(u,v) \in [\![1,n]\!]^2 |\ u+3 < v\}$$

Ainsi, un appariement doit être obligatoirement accompagné par l'un de ses voisins ($u-1, v+1$) ou ($u+1, v-1$). En théorie, ceci pourrait ajouter jusqu'à $\frac{1}{2}n^2$ contraintes, mais en pratique ce nombre reste raisonnable puisqu'on ne considère pas les n^2 appariements possibles en éliminant ceux dont la probabilité est inférieure à un seuil θ, et avec cette condition également ceux qui n'ont pas de voisin dont la probabilité d'existence est supérieure à θ. Cette contrainte n'est pas imposée si les modules utilisés viennent de CaRNAval, car ils contiennent souvent des appariements solitaires.

3. On interdit aux composantes d'un même module de se chevaucher :

$$\sum_{x \in M} \sum_{i=1}^{\|x\|} C_i^x \times I\left[P_{x,i} < u < P_{x,i} + k_{x,i} - 1\right] \leq 1, \qquad \forall u \in [\![1, n]\!].$$

On utilise $I\left[P_{x,i} < u < P_{x,i} + k_{x,i} - 1\right]$, une variable booléenne indicatrice dont la valeur (0 ou 1) dépend de la véracité de la condition. Ici, la condition impose que tout nucléotide u n'appartienne qu'à zéro ou une seule composante de module.

On force le respect de la structure des motifs à plusieurs composantes à l'aide de trois contraintes supplémentaires :

4. On force l'insertion de soit toutes les composantes d'un module, soit aucune :

$$\sum_{i=2}^{\|x\|} C_i^x = (\|x\| - 1) \times C_1^x, \qquad \forall x \in \{x \in M \mid \|x\| \geq 2\}$$

5. On force ensuite les appariements imposés par la structure du module x. Pour les boucles (HL, IL, ML), les nucléotides fermant les hélices adjacentes à la boucle font partie du module. Il faut donc apparier le dernier nucléotide de chaque composante au premier de la suivante et le dernier de la dernière au premier de la première.

$$C_1^x \leq y_{P_{x,\|x\|}+k_{x,\|x\|}-1}^{P_{x,1}} \qquad \forall x \in \{x \in M \mid \|x\| \geq 2\}$$

$$C_1^x \leq y_{P_{x,j+1}}^{P_{x,j}+k_{x,j}-1} \qquad \forall x \in \{x \in M \mid \|x\| \geq 2\}, \qquad \forall j \in [\![1, \|x\|[\![$$

Pour les modules de CaRNAval, qui autorisent certains appariements entre des positions arbitraires du module, qui ne correspondent pas toujours à une boucle, on utilise plutôt :

$$A_{x,i} \times C_i^x \leq \sum_{u=P_{x,i}}^{P_{x,i}+k_{x,i}} \sum_{\substack{v \\ (u,v) \in Y_x}} y_v^u \qquad \forall x \in M \ \ \forall i \in [\![1, \|x\|]\!],$$

ce qui n'autorise l'insertion de la ième composante de x que si tous les appariements (u, v) avec u un nucléotide apparié de la composante sont respectés (dans l'équation les y_v^u valent tous 1, et leur somme $A_{x,i}$).

On décrit aussi des contraintes facultatives à utiliser au besoin :

6. On peut, éventuellement, interdire les pseudonoeuds (contrainte utilisée à des fins de test et de comparaison) :
$y_v^u + y_l^k \leq 1, \qquad \forall u, v, k, l$ tels que $1 \leq u < k < v < l \leq n, \mathrm{u} + 3 < \mathrm{v}, \mathrm{k} + 3 < \mathrm{l}.$
Ici encore, le nombre de contraintes est limité par l'utilisation d'un seuil de probabilité d'existence minimale de l'appariement nécessaire à la définition d'une

variable y_v^u.

7. On peut interdire expressément une structure s^*, notamment pour interdire au programme de renvoyer comme solution une structure déjà trouvée :

$$\sum_{u<v}(1-y_v^u)\times I\left[y_{s^*[v]}^{s^*[u]}=1\right]+y_v^u\times I\left[y_{s^*[v]}^{s^*[u]}=0\right] + \sum_x (1-C_i^x)\times I[C_i^x=1 \text{ dans } s^*]+C_i^x\times I[C_i^x=0 \text{ dans } s^*]\geq 1.$$

La contrainte exprime qu'il doit y avoir au moins une différence entre l'état des variables de décision dans la solution et l'état qu'elles ont dans s^*.

8. On peut vouloir interdire les appariements sur les nucléotides non appariés d'un module inséré (pour forcer le non-appariement). Cette contrainte permet de respecter parfaitement la structure du module, mais empêche la formation de pseudonoeuds à longue distance (de type kissing-hairpins par exemple). On utilise :

$$(k_{x,i}\ -\ 2)\times C_i^x+\sum_{u=P_{x,i}+1}^{P_{x,i}+k_{x,i}-1}\sum_{v=0}^{n} y_v^u\leq (k_{x,i}\ -\ 2)\qquad \forall x\in M\ \ \forall i\in[\![1,\|x\|]\!].$$

Pour les modules de CaRNAval, on utilise plutôt

$$k_{x,i}\times C_i^x+\sum_{u=P_{x,i}}^{P_{x,i}+k_{x,i}}\sum_{\substack{v\\(u,v)\notin Y_x}} y_v^u\leq k_{x,i}\qquad \forall x\in M\ \ \forall i\in[\![1,\|x\|]\!],$$

ce qui n'autorise l'insertion de la ième composante de x que si tous les appariements (u,v) avec u un nucléotide non apparié de la composante sont absents (dans l'équation les y_v^u seraient nuls).

Maintenant que les variables et les contraintes sont définies, il faut encore définir la ou les fonction(s) objectif(s) à optimiser pour obtenir un programme linéaire complet. Dans ce modèle, on se donne deux objectifs : maximiser la probabilité d'existence de la structure, et maximiser l'insertion de modules connus.

Pour le critère maximisant la probabilité d'existence de la structure, on utilise la fonction suivante :

$$f_{prob}(s)=\sum_{(u,v)\in\{(u,v)\mid p_{uv}>\theta\}} y_v^u\times p_{uv}\,,$$

soit la somme des probabilités des appariements effectifs dans la structure s. C'est une grandeur sans unité qui mesure l'espérance de l'exactitude des appariements (estimateur MEA).
Pour le critère d'insertion de modules, nous proposons plusieurs approches qui sont décrites dans la section suivante.

III.2.2 Propositions de critères d'insertion de modules

On cherche ici à construire une bonne fonction d'évaluation de l'insertion de modules. On constate que plus le nombre de nucléotides faisant partie d'un module augmente, plus on a d'informations sur la structure secondaire de la molécule obtenues grâce aux modules. On peut donc chercher à maximiser le nombre de modules inclus. Malheureusement, cette idée pénalise fortement l'insertion de modules complexes comme des boucles multiples avec un grand nombre de composantes, car la place qu'elles occupent rend impossible l'insertion d'autres petits motifs dans la séquence.
Un modèle plus juste serait alors de maximiser le nombre de nucléotides concernés par l'insertion d'un module. Ainsi, RNA-MoIP utilise la somme des carrés des nombres de nucléotides de chaque composante de module insérée. Nous gardons cette idée comme première proposition avec la fonction suivante :

$$f_A(s) = \sum_{x \in M} \sum_{i=1}^{\|x\|} {k_{x,i}}^2 \times C_1^x.$$

Le défaut à priori de cette fonction est de favoriser de larges boucles HL et IL, ce qui n'est pas très fréquent, et qui laisse moins de place pour d'autres modules. On propose alors une seconde fonction, qui cherche plutôt à insérer des motifs complexes (ML plutôt qu'IL et HL) mais de petite taille, en pénalisant chaque module par le logarithme de son nombre de nucléotides :

$$f_B(s) = \sum_{x \in M} \left[\frac{\|x\|}{\log_2\left(\sum_{i=1}^{\|x\|} k_{x,i}\right)} \times C_1^x \right].$$

Mais on sait aussi que toutes les insertions de modules ne sont pas équiprobables, et qu'on s'expose à un grand risque de faux positifs. Aussi, une information précieuse à utiliser est le score du "hit" $p(x)$ renvoyé par les outils BayesPairing ou JAR3D à propos d'un site d'insertion de module. On propose donc une fonction uniquement basée sur ce score :

$$f_C(s) = \sum_{x \in M} p(x) \times C_1^x.$$

Et enfin, on propose une dernière fonction pondérant les termes de la fonction f_B par les scores $p(x)$:

$$f_D(s) = \sum_{x \in M} \left[\frac{\|x\|}{\log_2\left(\sum_{i=1}^{\|x\|} k_{x,i}\right)} \times p(x) \times C_1^x \right].$$

En résumé ;

- f_A insère de gros motifs,
- f_B insère de petits motifs à grand nombre de composantes,
- f_C insère des motifs bien notés,
- f_D insère de petits motifs à grand nombre de composantes et bien notés.

Les différentes versions du problème d'optimisation seront testées, et on cherchera à observer si l'une des fonctions objectif est meilleure.

III.2.3 Modules et méthodes d'insertion considérées

Le problème est donc maintenant formellement défini. Cependant, il nous faut encore en pratique spécifier :

- une source de données (quels modules utiliser),
- une façon de découvrir les sites d'insertions possibles de modules candidats x dans la séquence s.

À propos des méthodes de placement des modules, on se propose d'utiliser toutes les techniques existantes pour les comparer :

- Le *pattern-matching* simple, c'est-à-dire l'utilisation d'une expression régulière pour rechercher un motif de séquence dans s. C'est la méthode utilisée par RNA-MoIP. Comme dans RNA-MoIP, les modules avec de petites composantes de 1 ou 2 bases sont transformés : on crée toutes leurs extensions possibles de façon à ce que toutes les composantes aient une taille minimale de 3 bases. Cependant, cette méthode de placement ne renvoie pas de score de confiance $p(x)$ et ne peut donc pas être utilisée avec les fonctions f_C et f_D.
- L'utilisation de JAR3D. Comme JAR3D demande d'identifier la position de boucles dans la séquence pour pouvoir y chercher des modules, on réalise d'abord une première prédiction de structures secondaires par RNAsubopt et on utilise les solutions renvoyées pour lister les sites possibles pour les HL et IL. On les donne ensuite en entrée à JAR3D.
- L'utilisation de BayesPairing, qui ne demande aucun pré-traitement particulier.

De ces méthodes découle naturellement le choix des bases de données de modules à utiliser. Le *pattern-matching* n'est possible que sur des modèles de modules à la séquence bien définie, nous le testerons donc avec Rna3Dmotifs. JAR3D n'est compatible qu'avec les modules du 3D Motif Atlas. BayesPairing supporte les deux bases de données. La première version de BiORSEO supportera donc ces deux types de modules en entrée. Une seconde version de BiORSEO sera proposée fin 2020 étendant le support aux modules de CaRNAval par *pattern-matching.* Dans la seconde version de BiORSEO, nous avons donc une longue liste de variantes possibles du problème :

- modules de Rna3Dmotifs, insérés par *pattern-matching*, et évalués par f_A ou f_B,
- modules de Rna3Dmotifs, insérés par BayesPairing, et évalués par f_A, f_B, f_C ou f_D,
- modules du 3D Motif Atlas, insérés par JAR3D, et évalués par f_A, f_B, f_C ou f_D,
- modules du 3D Motif Atlas, insérés par BayesPairing, et évalués par f_A, f_B, f_C ou f_D,
- modules de CaRNAval, insérés par *pattern-matching*, et évalués par f_A ou f_B,

… soit 16 variantes au total.

III.2.4 Algorithme bi-objectif

On a maintenant entièrement défini le problème d'optimisation bi-objectif en nombres entiers. On propose ici un algorithme pour le résoudre. Cet algorithme a été développé par simplification de l'algorithme dichotomique de BiokoP, auquel on a supprimé

l'organisation des solutions en ensembles successifs de rang de Pareto identique, et en ajoutant quelques optimisations de performance.

L'algorithme rejoint la famille des méthodes dites "epsilon-contraintes", présentée en section II.3.2.4. On résout itérativement un problème d'ILP monocritère en obligeant le second critère à rester dans un intervalle via des contraintes supplémentaires. Dans une méthode epsilon-contrainte classique, l'intervalle en question se restreint d'une petite quantité epsilon d'une itération sur l'autre, ce qui permet de trouver de nouvelles solutions petit à petit et de dessiner le front de Pareto. Cependant, on a ici un problème d'optimisation discret, on n'a donc pas besoin de définir epsilon. On interdit simplement au programme de retrouver les solutions précédentes d'une itération sur l'autre, et on définit l'intervalle de recherche par rapport à la solution précédemment trouvée, comme illustré sur la Figure 3.3.

Le fait de ne pas diminuer l'intervalle de recherche d'une quantité epsilon fixée garantit alors l'obtention du front de Pareto exhaustif et exact, à partir du moment où la résolution du problème linéaire monocritère est exacte. On ne manque ainsi aucune solution.

Pour décrire l'algorithme, on note f_1 la fonction d'évaluation de l'insertion de modules (f_A, f_B, f_C ou f_C) et f_2 la fonction f_{prob} :

1. Résoudre les problèmes monocritères $L_1 \coloneqq \underset{s}{\operatorname{argmax}}\, f_1(\mathrm{s})$ et $L_2 \coloneqq \underset{s}{\operatorname{argmax}}\, f_2(\mathrm{s})$.
2. Définir l'ensemble $R \coloneqq \{L_1\}$ des solutions de l'ensemble de Pareto déjà découvertes et y ajouter l'une des extrémités de l'ensemble de Pareto, par exemple L_1.
3. Définir un ensemble $F \coloneqq \{L_1\}$ l'ensemble des solutions interdites et y insérer celle qu'on a mis dans R.
4. Lancer la procédure chercher_entre($f_2(L_1), f_2(L_2)$).
5. Lancer la procédure chercher_entre($-\infty, f_2(L_1)$).

La procédure chercher_entre(min, max) résout le problème d'optimisation monocritère sur f_1, f_2 étant contrainte dans l'intervalle [min, max]. Plus précisément :

Procédure `chercher_entre(`$\lambda_{\mathtt{min}}$`, `$\lambda_{\mathtt{max}}$`)` :

1. Résoudre le problème monocritère $s \coloneqq \underset{s \in \{s \mid f_2(s) \in [\lambda_{min}, \lambda_{max}]\}}{\operatorname{argmax}} f_1(\mathrm{s})$. Si aucune solution n'est trouvée telle que $f_2(s) \in [\lambda_{min}, \lambda_{max}]$, terminer la procédure.
2. Ajouter s à l'ensemble F.
3. Si la solution s est dominée par une solution de R, terminer la procédure.
4. Retirer de R les solutions qui seraient nouvellement dominées par s.
5. Lancer la procédure chercher_entre($f_2(s), \lambda_{max}$).
6. Si $\lambda_{max} - \lambda_{min} > 0$, lancer la procédure chercher_entre($\lambda_{min}, f_2(\mathrm{s})$).

La 6e étape permet d'identifier des solutions superposées à s. Elle n'est pas relancée si l'on est déjà en train de chercher entre deux solutions superposées ($\lambda_{max} = \lambda_{min}$), l'étape 5 suffit. La figure 3.3 ci-dessous illustre cet algorithme avec un cas simple et fictif.

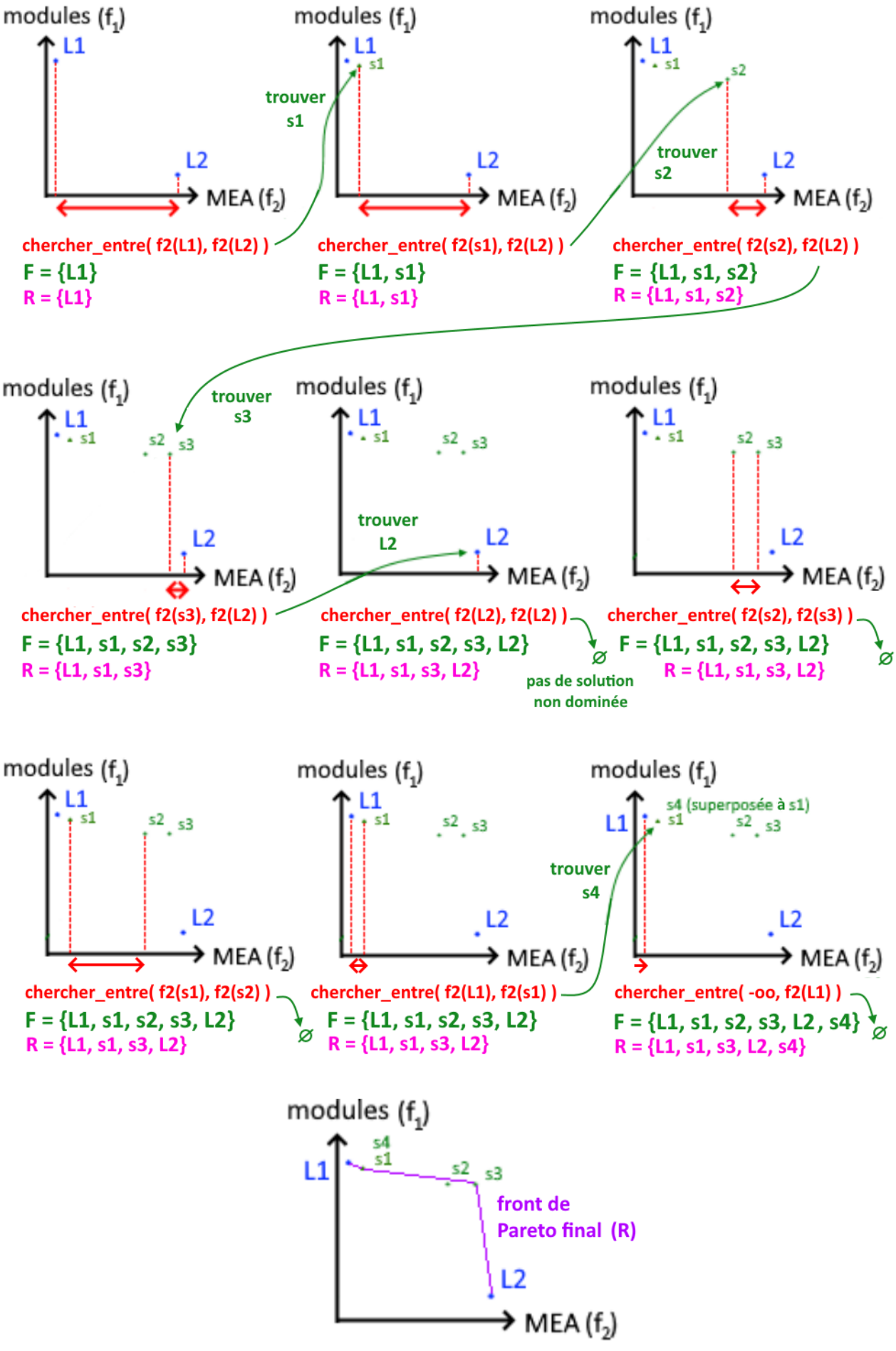

modules (f1)
L1
L2
MEA (f2)
trouver s1
s1
chercher_entre(f2(L1), f2(L2))
F = {L1}
R = {L1}
chercher_entre(f2(s1), f2(L2))
F = {L1, s1}
R = {L1, s1}
trouver s2
s2
chercher_entre(f2(s2), f2(L2))
F = {L1, s1, s2}
R = {L1, s1, s2}
trouver s3
s3
chercher_entre(f2(s3), f2(L2))
F = {L1, s1, s2, s3}
R = {L1, s1, s3}
trouver L2
chercher_entre(f2(L2), f2(L2))
F = {L1, s1, s2, s3, L2}
R = {L1, s1, s3, L2}
pas de solution non dominée
chercher_entre(f2(s2), f2(s3))
F = {L1, s1, s2, s3, L2}
R = {L1, s1, s3, L2}
chercher_entre(f2(s1), f2(s2))
F = {L1, s1, s2, s3, L2}
R = {L1, s1, s3, L2}
trouver s4
s4 (superposée à s1)
chercher_entre(f2(L1), f2(s1))
F = {L1, s1, s2, s3, L2}
R = {L1, s1, s3, L2}
chercher_entre(-oo, f2(L1))
F = {L1, s1, s2, s3, L2, s4}
R = {L1, s1, s3, L2, s4}
front de Pareto final (R)

Figure 3.3 – Recherche dichotomique et obtention du front de Pareto. *On a deux fonctions f_1 et f_2 à maximiser. On commence par identifier les maxima L_1 et L_2 sur chaque axe, de façon indépendante. On définit l'ensemble F des solutions qu'on interdit au programme d'optimisation de retrouver, via la contrainte 7 présentée précédemment. On définit aussi l'ensemble R des solutions constituant l'ensemble de Pareto. La recherche dichotomique cherche d'abord « à droite » de chaque nouvelle solution trouvée (sur les cinq premières étapes), puis dans un deuxième temps revient chercher « à gauche » des solutions trouvées pour identifier d'éventuelles nouvelles solutions superposées à celles déjà trouvées (étapes 6 à 9). L'ensemble de Pareto final est constitué des éléments de R à la fin de l'algorithme. On note que la solution s2 de l'exemple n'en fait pas partie, elle est dominée.*

III.2.5 Implémentation

Le programme d'optimisation bi-objectif est implémenté en langage C++17, et repose sur l'interface de programmation C++ de plusieurs dépendances :

- l'interface "`concert`" du logiciel d'optimisation CPLEX d'IBM, obtenu sous licence académique pour la résolution du problème ILP mono-objectif,
- l'interface de Nupack pour le calcul de probabilités d'appariement (dans BiORSEO 1.0), substituée par celle de ViennaRNA et sa bibliothèque C libRNA [202] dans BiORSEO 2.0. Le choix de libRNA par rapport à Nupack dans la seconde version repose sur sa moins grande complexité en mémoire, sans qu'aucune influence sur les performances n'ait été observée.

Le paramètre θ utilisé par défaut est fixé à une probabilité minimale de 0,001. Les appariements moins probables ne sont pas considérés du tout.

Le programme est accompagné d'un script Python 3 chargé de superviser le pipeline (lancer RNAsubopt, JAR3D ou BayesPairing). L'outil porte le nom de BiORSEO (Bi-Objective RNA Structure Efficient Optimizer) et est disponible en téléchargement sur le site EvryRNA, plateforme de dépot des outils bioinformatiques dédiés à l'ARN du laboratoire IBISC : https://evryrna.ibisc.univ-evry.fr/evryrna/biorseo

Afin de faciliter l'installation et l'utilisation, une image Docker préconfigurée est également proposée en téléchargement.
La version actuelle de BiORSEO est la version 2.0, supportant la base de données CaRNAval (ce qui n'était pas le cas dans l'article initial de BiORSEO [26]). Il n'y a pas de différence majeure entre les deux versions, la principale autre différence étant le remplacement de Nupack par la libRNA pour le calcul des probabilités d'appariement, ce qui n'a pas d'impact perceptible sur les résultats. Des évolutions futures sont possibles, pour supporter la version 2.0 de BayesPairing [294], et de nouvelles bases de données de modules.

III.3 Résultats

III.3.1 Performance des prédictions de structure secondaire

On teste la performance de l'outil de prédiction BiORSEO sur plusieurs jeux de données de séquences. Le premier jeu provient de la base de données RNASTRAND 2.0 [14]. On y récupère 344 séquences de 10 à 100 nucléotides, sans nucléotides modifiés, et pour lesquelles la structure secondaire est obtenue expérimentalement (et pas par une méthode computationnelle). Les familles d'ARN sont variées, et 74 d'entre eux contiennent un (ou plusieurs) pseudonoeuds. Le second jeu de données est extrait de la base Pseudobase++ [334] et contient exclusivement des ARN avec pseudonoeuds, de différentes classes de complexité. Enfin, un nouveau benchmark est recalculé en 2021 après la sortie de la compilation de données bpRNA [78]. On récupère 12 000 structures toujours entre 10 et 100 bases ayant moins de 90% d'identité de séquence du jeu bpRNA-1m_90. Malheureusement, pour des raisons pratiques, les calculs ne seront pas terminés pour les 12 000 structures et on dispose « seulement » de quelques milliers de prédictions pour chaque variante (un nombre différent pour chaque variante). Ceci suffit amplement pour tirer des statistiques puis des conclusions sur la performance de l'outil. RNA-STRAND et bpRNA sont des compilations de compilations malheureusement assez opaques sur leurs sources, il est difficile de retrouver les informations expérimentales pour chaque structure. Néanmoins elles ne semblent pas contenir de structures provenant d'annotations des structures 3D d'ARN de la PDB, il n'y a donc pas de fuites de données par rapport aux jeux de données d'entrainement de JAR3D et BayesPairing qui en sont dérivés. On note aussi que ces bases de données modélisent uniquement la structure secondaire simple, on n'a pas d'informations ni sur les appariements non canoniques réels ni sur les motifs réels.

On estime la performance d'une prédiction de structure en comptant les nombres d'appariements réels correctement prédits (vrais positifs ou VP), d'appariements réels non prédits (faux négatifs FN), de paires de nucléotides non-appariés et non prédits comme appariés (vrais négatifs VN), et de paires de nucléotides non-appariés prédits comme l'étant (faux positifs FP). La performance est résumée par une valeur, le coefficient de corrélation de Matthews, qui permet de prendre en compte le déséquilibre important des classes négatives et positives (il y a toujours beaucoup plus de négatifs) :

$$MCC = \frac{VP \times VN - FP \times FN}{\sqrt{(VP + FP)(VP + FN)(VN + FP)(VN + FN)}}.$$

Plus cette métrique est proche de 1, meilleure est la prédiction. Plus elle est proche de 0, pire est la prédiction (à 0.5, la prédiction est aléatoire).

La performance qui nous intéresse est la réponse à la question : "Est-ce que la structure recherchée se trouve parmi les solutions renvoyées par l'outil ?". Ainsi, pour les outils renvoyant plusieurs solutions (BiORSEO, RNAsubopt, et RNA-MoIP '1-by-1'), on conserve uniquement la valeur du MCC maximum observé parmi les solutions renvoyées,

celle se rapprochant le plus de la structure recherchée (native, active, ou l'une des structures actives, en tout cas, celle reportée dans la base de données). Les résultats de prédiction sont compilés sur les Figures 3.4 et 3.5. La première est la figure originale publiée avec BiORSEO [26], qui utilise la base de données RNA Strand 2.0 [14]. La seconde est la figure mise à jour à l'occasion de la sortie de la base de données bpRNA [78] et de l'ajout du support des réseaux d'interactions récurrents de CaRNAval [273] dans BiORSEO 2.0.

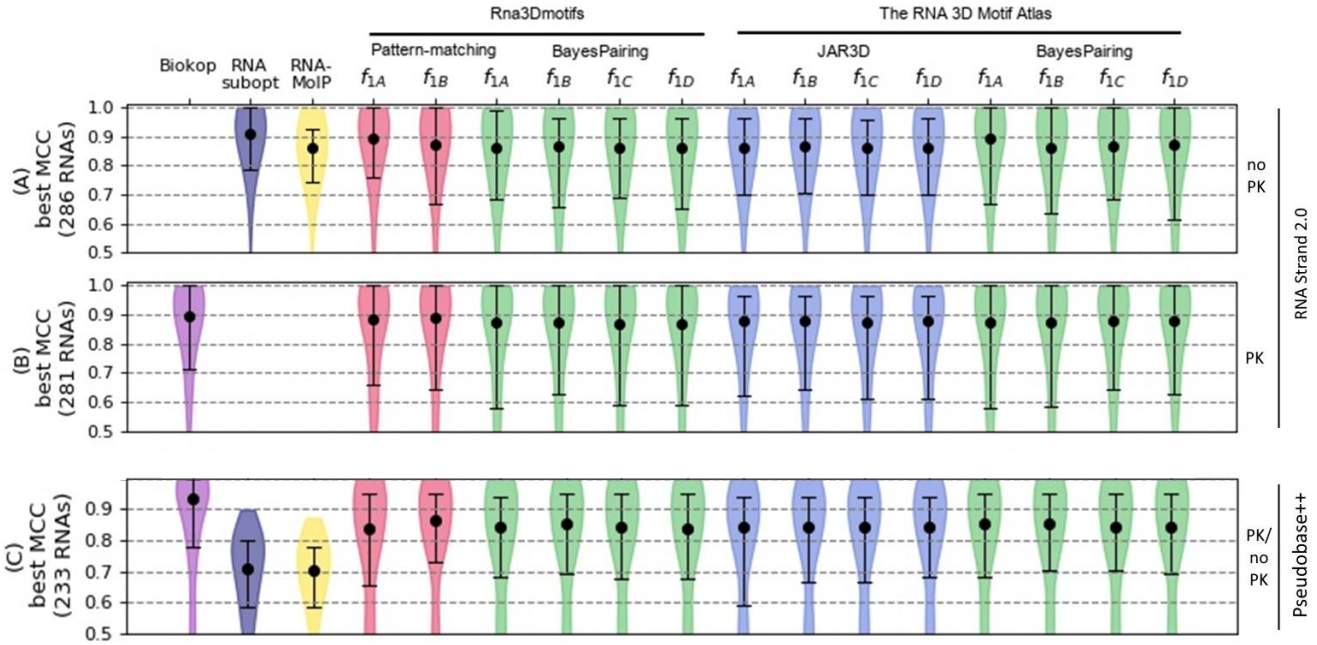


Figure 3.4 – Benchmarks de prédiction de structure secondaire. *La figure est organisée en 3 lignes, correspondant à un benchmark chacune, et en 17 colonnes, correspondant à 17 méthodes de prédiction différentes. De gauche à droite : BiokoP, RNAsubopt, RNA-MoIP '1-by-1', puis les 14 variantes (originales) de BiORSEO.* ***(A)*** *Benchmark sur 344 ARN courts extraits de la base de données RNA Strand, en interdisant à BiORSEO de prédire des pseudonoeuds à l'aide des contraintes (6). La comparaison est possible avec RNAsubopt et RNA-MoIP qui n'en prédisent pas. Seuls 286 ARN ont pu être prédits par les 14 variantes.* ***(B)*** *Benchmark sur le même jeu de données, en autorisant BiORSEO à proposer des pseudonoeuds. Seuls 281 ARN ont pu être prédits par tous les outils.* ***(C)*** *Benchmark sur le jeu de données Pseudobase++, en autorisant BiORSEO à trouver des pseudonoeuds. Les outils ne les supportant pas font des prédictions sans pseudonoeuds, ce qui les pénalise.*

Ces résultats permettent de relever plusieurs observations :

- Le modèle de placement et évaluation des motifs initial de RNA-MoIP semble bon. En effet, la variante de BiORSEO plaçant les motifs par pattern-matching et les évaluant avec f_A a une performance supérieure ou égale aux autres variantes. Ces deux idées ont été reprises à RNA-MoIP et elles semblent les plus efficaces.
- Toutefois, l'approche de Pareto améliore la performance par rapport à la pondération polynomiale des objectifs pratiquée par RNA-MoIP, ce dont témoigne la performance supérieure de BiORSEO avec pattern-matching et la fonction objectif f_A sur RNA-MoIP dans le benchmark (A).
- Le support des structures avec pseudonoeuds permet d'augmenter encore cette

performance, ceci est mis en évidence par les benchmarks (B) et (C).

- Mais cependant, la performance n'est toujours pas réellement améliorée par rapport aux méthodes existantes dans le cas général. En effet, RNAsubopt fait aussi bien dans le benchmark (A), et BiokoP fait aussi bien dans le benchmark (C).

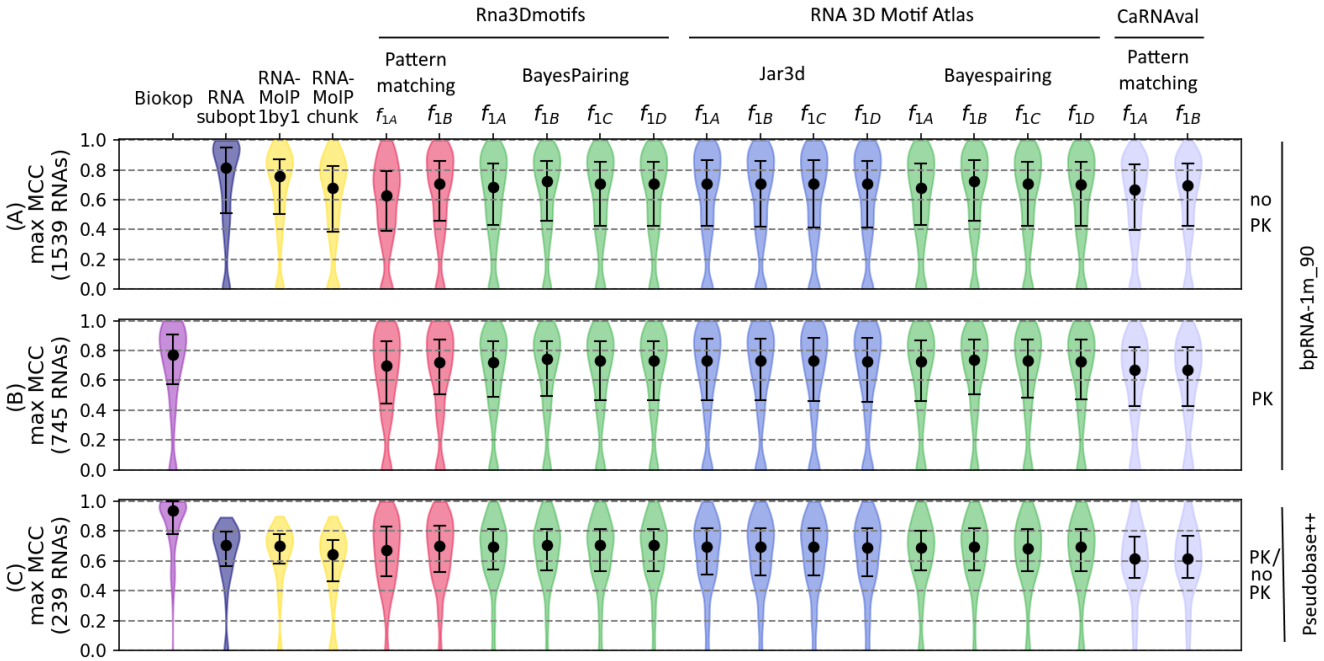


Figure 3.5 – Benchmarks de prédiction de structure secondaire (mise à jour avec BiORSEO 2.0 et CaRNAval). *Benchmark similaire sur un plus gros jeu de données. Attention, l'échelle de l'axe des ordonnées a changé par souci de lisibilité. Deux nouvelles variantes de BiORSEO sont ajoutées à droite, utilisant les RIN de CaRNAval placés par pattern-matching et utilisant les fonctions objectifs A ou B. On a également séparé les deux façons d'utiliser RNA-MoIP. RNA Strand a été remplacée sur les deux premières lignes par bpRNA-1m_90, dans laquelle elle est incluse. Comme dans la figure précédente, on ne considère que les ARN pour lesquels on a pu obtenir une prédiction par toutes les variantes et outils. Leur nombre est indiqué à gauche de chaque ligne.*

Mais plus récemment, le benchmark sur bpRNA remet en question ces premières conclusions :

- la performance de toutes les méthodes, BiORSEO et outils concurrents compris, est globlement moins bonne (environ 10 points de MCC inférieure),
- en particulier la combinaison "BiORSEO + pattern-matching + f_A" identifiée comme étant une amélioration directe de RNA-MoIP, dont la performance a chuté encore plus que les autres, sans que l'on sache expliquer pourquoi,
- les variantes utilisant CaRNAval ne se démarquent pas particulièrement. Ce n'est pas très surprenant, CaRNAval contient des motifs avec interactions « longue distance » et pas de simples boucles (HL, IL, ML) qui auraient contraint directement la structure secondaire.

On conserve tout de même une performance similaire à RNA-MoIP (tout en restant inférieure à BiokoP), mais en supportant en plus la prédiction des pseudonoeuds.

III.3.2 Visualisation des fronts de Pareto

Avec les prédictions du benchmark de bpRNA (avec pseudonoeuds autorisés), pour chacune des variantes de BiORSEO, et pour chaque ARN, un front de Pareto a été obtenu avec une ou plusieurs solutions de structures secondaires. On normalise ces front des Pareto dans l'intervalle $[0,1]^2$ pour pouvoir comparer leur forme. On identifie dans chacun la position de la structure de plus grand MCC (la plus proche de la structure native), et on superpose les points obtenus pour les différents ARN sur un seul graphique par variante. Le nombre de points étant de plusieurs milliers, on résume leur densité par une estimation de densité avec un kernel Gaussien. Les résultats sont présentés sur la Figure 3.6.

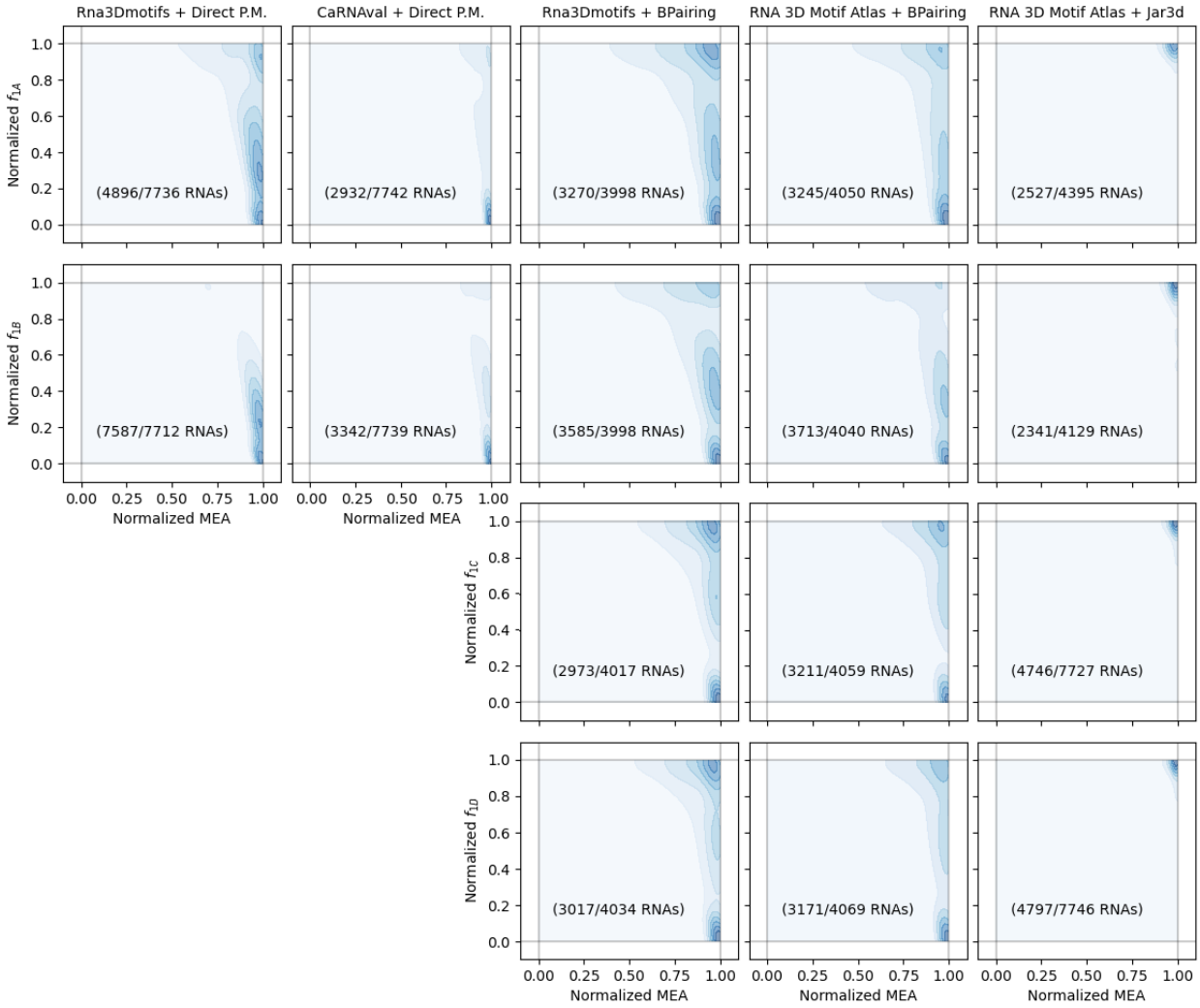


Figure 3.6 – **Distribution des structures natives sur les fronts de Pareto.** *En ligne, les 4 fonctions objectifs f_A, f_B, f_C, et f_D. En colonne, les variantes de placement de modules. Chaque point ayant servi à estimer les distributions représentées correspond à une solution du front de Pareto, celle possédant le meilleur MCC. Pour chaque sous-graphique, l'axe des abscisses mesure la qualité de la solution native selon le critère MEA, et celui des ordonnées la qualité selon le critère des modules (f_A, f_B, f_C, ou f_D). Les fronts de Pareto composés d'un unique point de coordonnées (1,1) ne sont pas informatifs et ne sont pas représentés. Le nombre d'ARN représentés, ceux possédant un front de Pareto de deux structures ou plus, est indiqué sur la figure.*

La répartition des meilleures solutions dans les fronts de Pareto permet de commenter sur l'utilité respective des différents critères. On distingue plusieurs répartitions possibles :

- Cas A : Si les meilleures solutions se situent sur la droite $y = x$, alors les deux critères sont corrélés, et l'optimisation multicritère n'est pas réellement pertinente.
- Cas B : Si les meilleures solutions se concentrent vers le point de coordonnées (1,1), alors les deux critères sont de bonne qualité pour trouver les solutions natives. Cependant, ils sont toujours corrélés et on pourrait se contenter d'un seul.
- Cas C : Si les meilleures solutions se répartissent le long d'une arête x=1 ou y=1 mais pas l'autre, alors l'un des critères est pertinent et l'autre non (la meilleure solution ayant une valeur aléatoire sur ce second critère).
- Cas D : Si les meilleures solutions sont bien réparties le long des arêtes x=1 et y=1 du carré, alors l'utilisation conjointe des deux critères est utile : tantôt c'est l'un des deux critères qui permet d'identifier la bonne solution, tantôt c'est l'autre.

Selon la variante de BiORSEO considérée, on se situe quelque part entre les cas B et C. De nombreux fronts de Pareto se résument à un unique point, constituant une solution optimale sur les deux critères (cas B, ils n'ont donc pas été représentés sur la Figure 3.6). Les fronts restants montrent aussi des meilleures solutions proches de (1,1) dans les variantes utilisant JAR3D. Ce n'est pas étonnant, puisque l'insertion de modules par JAR3D repose sur la découverte des boucles par RNAsubopt, qui repose sur le même modèle énergétique que le critère MEA.
Pour les variantes utilisant BayesPairing, on est plutôt dans le cas D : les solutions qui ne sont pas proches de (1,1) s'étalent tout le long de l'axe des modules, en gardant un score MEA normalisé proche de 1. Ceci montre que c'est largement le critère MEA qui aboutit à la prédiction de structures proches de la structure native. Enfin, les variantes utilisant CaRNAval sont dans un cas extrême : les solutions natives ont toujours ou presque un score nul sur le critère d'insertion des modules. On a ici un argument pour ne pas recommander CaRNAval pour cet usage. Dans tous les cas l'utilisation d'un modèle complexe, bi-critère, utilisant des modules, n'est pas justifiée : ceci ne permet ni l'amélioration des performances de prédiction par rapport à un modèle MEA seul (Figure 3.5), ni d'identifier des structures natives que MEA aurait manquées (Figure 3.6).

III.3.3 Nombre et composition des solutions

Pour caractériser un peu mieux les solutions renvoyées par les outils, on indique également quelques statistiques sur le nombre de solutions retournées par les différentes méthodes concurrentes sur la Figure 3.7, et sur le nombre maximum de motifs insérés dans les solutions sur la Figure 3.8A. Sur la Figure 3.8B, on montre les ratios entre le nombre de motifs dans la meilleure solution (toujours celle de MCC maximum avec la structure native) et le nombre de motifs maximum insérés dans l'une des solutions du front de Pareto (celui de la Figure 3.8A). Si ce ratio vaut 1, la meilleure solution est celle (ou fait partie de celles) qui ont le plus de modules.

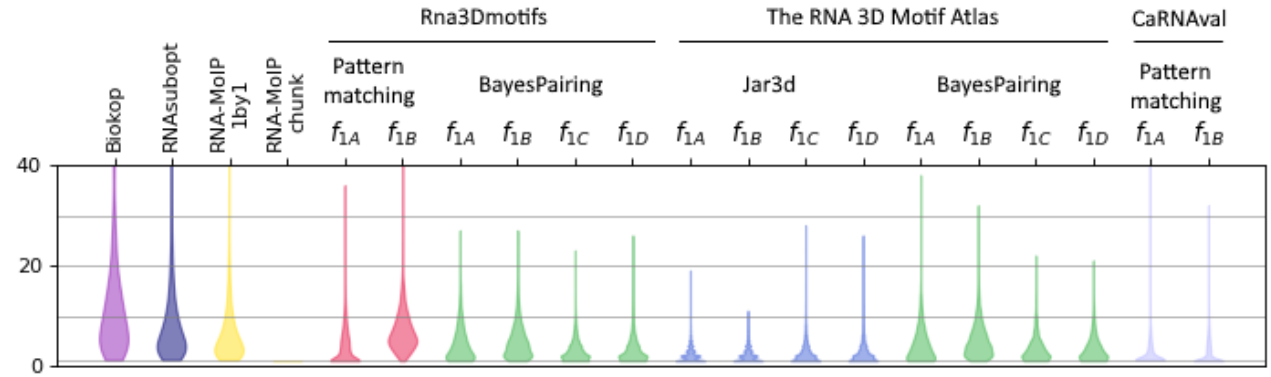


Figure 3.7 – Nombre de solutions retournées. *Pour chaque ARN dans le benchmark bpRNA-1m90, on compte le nombre de solutions obtenues dans le front de Pareto ou l'ensemble de solutions, et on affiche la distribution de ce nombre pour chaque variante. Les pseudonoeuds sont autorisés pour les méthodes qui les supportent.*

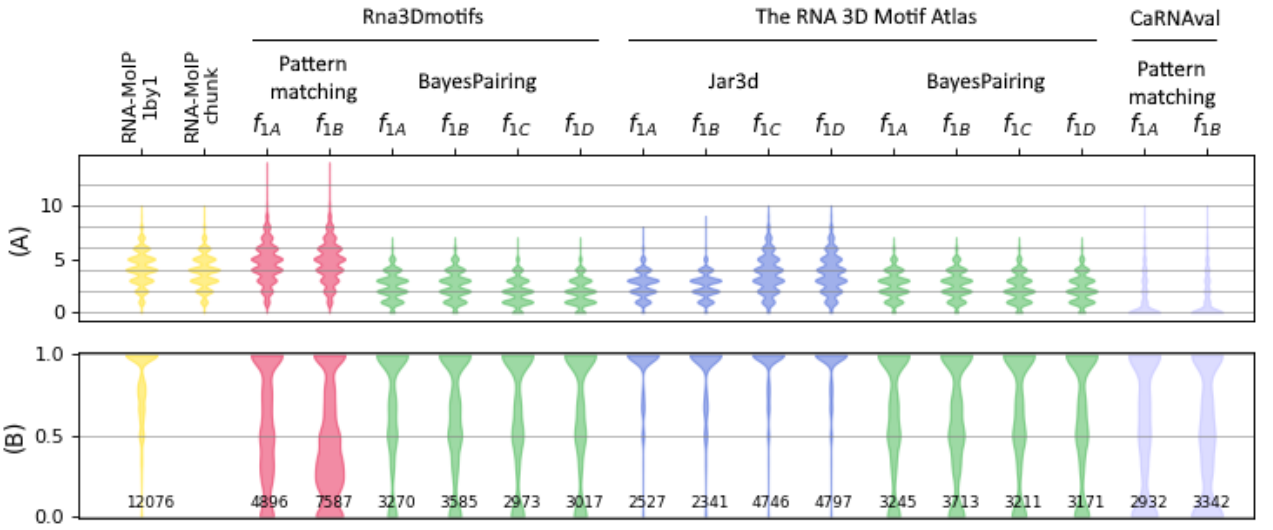


Figure 3.8 – Nombre de modules inclus. *Pour les méthodes qui insèrent des modules (RNA-MoIP et BiORSEO), on compte : **(A)** Le nombre maximum de modules insérés dans une solution pour chaque ARN. **(B)** Le ratio entre le nombre de modules dans la meilleure solution et le nombre maximum de motifs pouvant être insérés dans l'une des solutions. Le nombre d'ARN utilisés pour tracer la figure (issus du benchmark bpRNA-1m_90) est indiqué en bas.*

Pour rappel, BiokoP renvoie un front de Pareto (comme BiORSEO), et RNAsubopt renvoie les solutions dont l'énergie est à moins de 1 kcal/mol de la solution d'énergie minimale (par défaut, modifiable à d'autres intervalles). RNA-MoIP utilise les solutions de RNAsubopt et les renvoie toutes (mode une-par-une) ou n'en renvoie qu'une seule (mode "chunk"). En moyenne, les variantes de BiORSEO retournent moins de solutions que les autres outils, rarement plus de 5. Pourtant cet ensemble de solutions plus petit contient une solution de bonne qualité aussi souvent ou presque (voir les performances de prédiction sur les Figures 3.4 et 3.5). Ceci est un avantage, puisque l'utilisateur souhaitant déterminer une structure aura à choisir entre (ou à étudier plus en détail) un nombre plus réduit de candidates. On remplit donc partiellement l'objectif de RNA-MoIP qui était de sélectionner la bonne solution parmi un ensemble de solutions sous-optimales d'un point de vue énergétique.

L'analyse du nombre de motifs insérés montre sans surprise que les variantes reposant sur le pattern-matching de séquences peuvent insérer de nombreux motifs, jusqu'à 10 ou 15 par séquence de moins de 100 bases. Ce n'est cependant pas fréquemment le cas avec les modules de CaRNAval, qui ne sont pas des éléments de structure secondaire (HL, IL, ML) mais des réseaux d'interactions avec des contraintes d'appariement plus complexes. Les variantes qui utilisent un outil de placement (BayesPairing, JAR3D) insèrent plutôt 2 à 5 modules en moyenne.

Ceci permet d'expliquer deux autres défauts des variantes "pattern-matching" avec Rna3Dmotifs : la facilité d'insertion d'un module fait exploser la combinatoire des solutions possibles. Il en découle deux conséquences fâcheuses :

- le temps de calcul est plus long, même pour des séquences courtes (jusqu'à 1h30),
- les calculs échouent plus souvent, car nous arrêtons volontairement le programme à partir de 500 solutions dans le front de Pareto (pour cause de présomption d'explosion combinatoire).

On ne recommandera donc pas non plus ces variantes, qui ont plus de difficultés à rendre des solutions, et ces solutions sont en trop grand nombre. On voit sur la Figure 3.7B que la meilleure structure n'est pas celle ayant le plus de modules pour ces variantes-là.

III.4 Conclusion et perspectives

L'hypothèse qui sous-tend ce travail était que repérer les sites d'insertion de modules pouvait donner des indices sur la structure secondaire des ARN. Un outil se basait déjà sur cette supposition, RNA-MoIP. Souhaitant aller plus loin que l'approche de pondération polynomiale de RNA-MoIP, nous avons développé un outil capable de prédire des structures secondaires pour une séquence d'ARN tout en y insérant des modules connus. Quatre critères d'insertion ont été proposés et testés, ainsi que trois bases de données de modules, placés dans les séquences par trois méthodes de l'état de l'art.

Le programme renvoie les structures secondaires correspondant au front de Pareto du problème d'optimisation bi-objectif entre le critère MEA et le critère d'insertion de modules choisi. La modélisation choisie repose sur la programmation linéaire en nombres entiers, et la résolution (exacte) du problème est réalisée par le solveur CPLEX d'IBM.

Nous avons évalué la performance des méthodes en recherchant les meilleurs MCC observés entre la structure réelle d'un ARN et les structures de ce front de Pareto, pour plusieurs centaines d'ARN du jeu de données bpRNA-1m_90 [78].

Les différentes variantes ont une performance correcte et comparable aux autres outils, sans pour autant faire mieux. Toutefois, on note qu'en moyenne, les variantes de BiORSEO retournent moins de solutions que les autres outils, tout en proposant des solutions de qualité identique, ce qui est plutôt mieux.

Enfin, les variantes simplement basées sur le pattern-matching de séquence ont tendance à considérer plus de sites d'insertion que les autres et donnent parfois lieu à une explosion combinatoire, ce qui rend impossible la résolution du problème. On préférera donc utiliser un outil de placement des modules.

Différents travaux d'extension sont envisagés, …

- …pour étendre la compatibilité à de nouvelles bases de données de motifs (CaRNAval 2.0, RNAMC, RNA-Bricks 2.0, ou d'autres),
- …pour étendre le support de nouvelles formes de contraintes sur la structure secondaire. En particulier, pour l'instant, les "modules" concernés sont des sous-modèles d'éléments de structure secondaire. La contrainte qu'ils imposent sur la structure secondaire consiste à forcer les appariements entre composantes de chaque boucle (appariements fermants les hélices). Cependant on peut imaginer d'autres formes de contraintes. Avec le support de CaRNAval, les appariements peuvent être forcés sur tout couple de nucléotides appartenant à toute composante du module. D'autres formes de modules pourraient encore être envisagées : par exemple, des modules dont le graphe d'interactions est non connexe, mais dont les deux sous-graphes connexes doivent être observés ensemble dans la structure ;
- …pour étendre le pipeline à la modélisation 3D, à la façon de RNA-MoIP+MCsym. Les modules insérés pourraient être utilisés comme templates pour faire de l'assemblage de fragments ;

- …pour réfléchir à de nouvelles formes de fonction objectif pertinentes pour l'insertion de modules ;
- …pour étendre le nombre de critères à plus de deux, en réintégrant par exemple le critère MFE de BiokoP ;
- …pour unifier (en mélangeant tous les modules), ou automatiser le choix de l'une des variantes de BiORSEO en fonction d'un algorithme arbitraire dépendant des propriétés de la séquence fournie en entrée ;
- …pour utiliser des motifs d'interaction ARN-protéine, de façon à détecter dans les ARN les emplacements des sites d'interaction, tout en gardant une structure secondaire plausible (thématique en cours d'investigation).

CHAPITRE IV : Appliquer l'apprentissage profond à la prédiction de structures 3D d'ARN

Cette thèse a débuté en septembre 2018, époque présentant deux avantages :

- aucune méthode de prédiction de structures d'ARN par apprentissage automatique récente n'avait encore été proposée (si ce n'est les travaux exploratoires de Laing *et al.* comme RNA Junction Explorer et RNAJAG et entrainés sur très peu de données),
- des méthodes d'apprentissage profond étaient déjà en place dans le champ de recherche voisin de la prédiction de structures de protéines [10,129,141,352], et montraient des performances très bonnes pour des temps de calcul bien moindres que les méthodes de simulation classiques.

En décembre 2018, le succès du modèle AlphaFold lors de l'expérience CASP13, et les nombreuses discussions et commentaires de chercheurs [8] autour de leurs méthodes (qui ne seront pas publiées avant janvier 2020 [302]) ont renforcé l'attrait pour tenter d'adapter ces algorithmes à l'ARN.

Bien entendu, de nombreux autres groupes de recherche ont eu cette même idée en même temps, et la majorité des outils de prédiction de structures secondaires sortis depuis 2018 utilisent une architecture d'apprentissage profond. Ces travaux, s'ils s'écartent légèrement du sujet de cette thèse, s'inscrivent donc dans le cadre d'une tendance générale du champ de recherche. Ils pourront éventuellement aussi être inclus dans des modèles d'optimisation multicritère.

La première section (IV.1) de ce chapitre présente l'usage de l'apprentissage profond en bioinformatique structurale. La section IV.2 présente le jeu de données RNANet, conçu pendant cette thèse et servant de base aux autres travaux de cette thèse basés sur l'utilisation de données d'ARN. Viennent ensuite deux essais d'application de l'apprentissage profond à la prédiction de structures d'ARN : en section IV.3, la prédiction de la forme du squelette par réseau récurrent avec l'architecture RGN. En complément, la section IV.4 présente le travail de stagiaires à qui j'ai proposé ces sujets et que j'ai encadrés pendant leurs stages. Cette section s'intéresse à la prédiction des distances inter-résidus et des angles de torsion à l'aide de réseaux de convolution. Enfin, on conclut sur l'utilisation de l'apprentissage profond pour la prédiction de structures d'ARN.

IV.1 L'apprentissage profond s'impose en bioinformatique structurale

IV.1.1 Principe de l'apprentissage profond et familles d'architectures de réseaux

Les algorithmes d'apprentissage profond (ou méthodes de *deep-learning*) sont sous-divisés en grandes familles selon le type d'architecture du réseau de neurones utilisé.

IV.1.1.1 Rappels du principe de l'apprentissage supervisé par réseau de neurones

On part d'un couple de données (x, y), x étant un vecteur de caractéristiques d'un point de données que l'on connait (des descripteurs), et y étant une étiquette que l'on voudrait faire prédire par le réseau de neurones. Dans les problèmes de régression, y est un nombre (ou un vecteur de nombres). Dans les problèmes de classification, y est une étiquette de classe (0 ou 1, parfois -1, 0 ou 1, ou encore un vecteur de probabilités de la taille du nombre de classes possibles).

On appelle neurone virtuel une fonction mathématique, qui prend un nombre unique x_i (scalaire) en entrée, et lui applique deux transformations successives : d'abord une transformation affine (calcul de $w_i * x_i + b_i$ à l'aide de deux coefficients, le « poids » w_i et le « biais » b_i), puis une seconde par une « fonction d'activation » dérivable quelconque qui dépend de la tâche. Les neurones sont organisés entre eux, la sortie des uns étant reliés à l'entrée des autres, formant une architecture de réseau. Différentes architectures de réseau existent selon le type de problème que l'on veut résoudre. L'architecture la plus simple, dénotée MLP pour *Multi-Layer-Perceptron*, consiste à organiser les neurones en couches. La sortie de chaque neurone d'une couche est reliée à toutes les entrées des neurones de la couche suivante.

La sortie du dernier neurone de la dernière couche, après transformation par la dernière fonction d'activation, sera la "prédiction" du réseau de neurones. On compare ce résultat obtenu à la valeur connue de y, à l'aide d'une fonction de coût ou *loss-function* qui mesure l'erreur de prédiction, et qu'on cherche à minimiser. La prédiction étant elle-même dépendante des paramètres du réseau de neurones (vecteurs de poids w et de biais b), on peut exprimer la loss comme une fonction de w et b, notée $L(w, b)$. Bien entendu, pour des valeurs arbitraires des poids et des biais, la différence entre la prédiction et la valeur connue est grande. L'entrainement d'un réseau de neurones consiste à trouver les poids et biais qui minimisent cette erreur $L(w, b)$. C'est un problème d'optimisation continu, la loss étant un critère à minimiser, dépendante de variables que sont les poids et les biais. Habituellement, l'optimisation se fait par descente de gradient : les fonctions affines et fonctions d'activation étant dérivables, on peut calculer le gradient de L par rapport à chaque poids et biais, et modifier sa valeur en fonction de ce gradient, de façon à minimiser L. On répète ce processus en changeant le point de données (x, y) utilisé d'une itération sur l'autre. Quand tous les points de données ont été utilisés, on a achevé une *epoch*, et on recommence en réutilisant des points de données déjà passés. L'entrainement d'un réseau de neurones peut prendre

plusieurs dizaines à centaines d'epochs avant de converger vers une solution de qualité, on dit alors que le réseau est entrainé. On fixe et on sauvegarde alors les valeurs finales des poids et biais.

Si l'entrainement d'un réseau de neurones sur un gros jeu de données peu demander des ressources de calcul importantes et plusieurs heures ou jours de calcul, une fois le réseau entrainé, réaliser une seule prédiction est quasiment instantané. Ces méthodes présentent donc un avantage de rapidité de plusieurs ordres de grandeur par rapport aux méthodes de simulation "classiques" de type Monte-Carlo ou dynamique moléculaire.

Enfin, le principe de l'apprentissage "profond" consiste simplement à utiliser un grand nombre de couches entre l'entrée et la sortie du réseau de neurones. Le problème d'optimisation résultant devient plus difficile à résoudre, mais les capacités du modèle à apprendre des schémas complexes augmentent avec le nombre de paramètres du modèle, ce qui rend l'apprentissage profond très efficace dans les tâches de traitement d'images, de sons, ou de reconnaissance de motifs complexes comme des motifs de séquence (en bioinformatique, ou en traitement syntaxique du langage).

IV.1.1.2 Architectures avancées

Si le principe général de l'apprentissage automatique reste le même pour toutes les architectures, il existe des façons plus élaborées de disposer les neurones entre eux que les couches. Au fur et à mesure que de nouvelles architectures performantes sont découvertes et atteignent leur maturité, on voit les outils logiciels évoluer et passer d'une architecture à la suivante. La recherche en apprentissage étant très active, une nouvelle architecture "à la mode" et surpassant l'ancienne sort environ tous les 18 mois.

En 2018, l'architecture de prédilection pour le traitement des séquences était le réseau de neurones récurrent [112] (RNN). Dans cette famille de méthodes, un neurone ou un groupe de neurones voit sa sortie rebranchée à une seconde entrée (en plus d'une entrée principale recevant les nouvelles données). Ceci lui permet de garder « en mémoire » le ou les états précédents, le contexte, et d'adapter son comportement vis-à-vis de la nouvelle donnée en mélangeant le signal qu'elle apporte avec un état interne, dépendant des données passées. Ceci rend les RNN adaptés à la reconnaissance de motifs dans les séquences, où chaque nouvelle donnée est un des éléments de la séquence, le contexte des éléments précédents étant gardé en mémoire. Les architectures les plus utilisées dans la famille des RNN sont les LSTM [150] (*Long-Short-Term-Memory*) et les GRU [61] (*Gated Recurrent Unit*), ces deux termes désignant des « cellules » (des groupes de neurones) dont la sortie est reliée à une de leurs entrées.

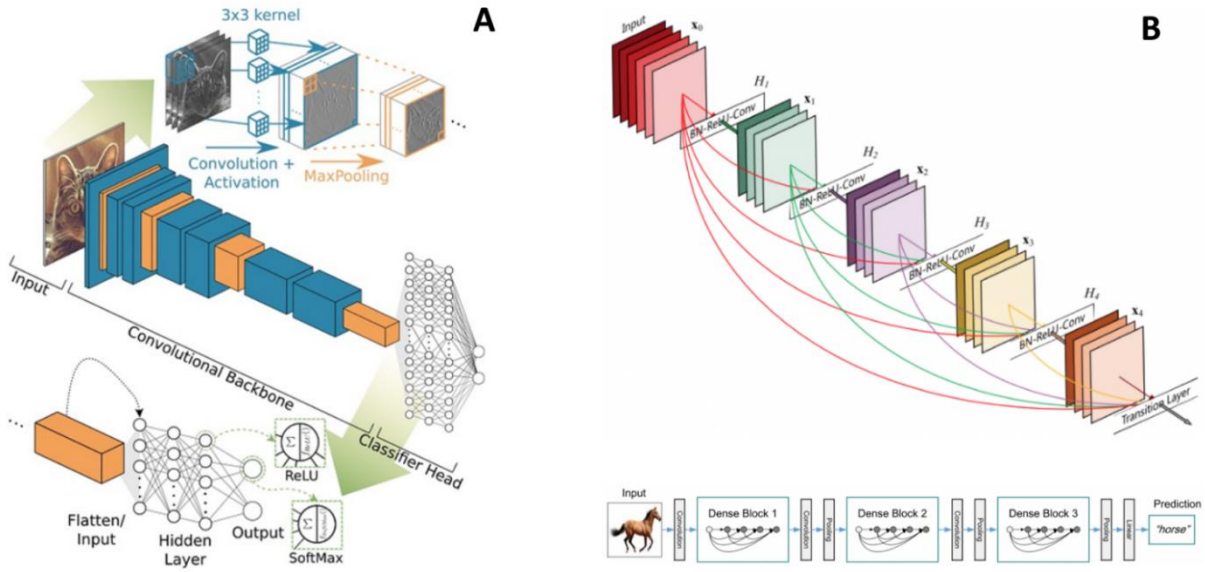


Figure 4.1 : Exemple d'architectures de réseaux de convolution. ***(A)*** *Un réseau de convolution simple appliqué à la classification d'images. On y voit la succession de couches de convolution (en bleu, avec un noyau de taille 3x3) et de couches de pooling (en orange, avec un noyau de 2x2), suivies d'un perceptron de trois couches et de deux neurones de sortie.* ***(B)*** *Un exemple de ResNet. Le signal peut être propagé à distance en sautant des couches, ce qui permet de rendre le réseau très profond. Les sauts peuvent se situer entre couches de convolution (en haut), ou au sein de couches denses disposées entre les couches de convolution (en bas).*

Figure (A) extraite de la référence ([151]) sous licence Creative Commons CC-BY. Figure (B) extraite du blog datascience.eu, visité le 5 mai 2021 à l'adresse https://datascience.eu/fr/apprentissage-automatique/un-apercu-de-resnet-et-de-ses-variantes/.

En 2019, et bien qu'ils aient été présentés dès 1998 par Yann Lecun [191], les réseaux de convolution ou CNN (voir Figure 4.1A) s'imposent comme supérieurs aux RNN. Ce sursaut d'intérêt provient de la démocratisation plus récente des réseaux résiduels profonds (ou ResNets, voir Figure 4.1B) [145], un type de réseau profond qui relie des couches distantes entre elles par des « raccourcis », permettant une meilleure propagation du signal. L'application de cette idée aux CNN leur a permis un bond en performance.

Les CNN classiques font passer le signal de la façon suivante :

1. Une couche de convolution, à l'intérieur de laquelle chaque neurone est relié à plusieurs neurones adjacents de la couche d'entrée (par exemple trois neurones si la couche d'entrée est en 1D, ou un carré de 3x3 si la couche d'entrée est en 2D, etc). On appelle la taille de ce groupe de neurones traités ensemble le noyau de convolution, ou *kernel* en anglais. Les neurones de cette couche possèdent donc des poids entrainables plus un biais, qui sont partagés entre tous les neurones de la couche, quelque soit leur localisation. La fonction d'activation n'est pas encore appliquée à ce niveau.
2. Une couche de *pooling* ou de sous-échantillonnage, qui sert à réduire la dimension. Cette couche ne contient pas de poids et biais entrainables, elle applique juste une opération mathématique à un groupe de neurones adjacents de la couche de convolution. Par exemple, prendre le maximum d'un groupe de 3 neurones, ou d'un

carré de 3x3 neurones (*max-pooling*). On peut aussi prendre la moyenne, ou d'autres mesures. Cette couche permet de résumer l'information lorsque la présence d'un signal n'est pas attendue spécifiquement à un endroit précis mais dans un voisinage. Ceci réduit l'impact mémoire et le nombre de paramètres du modèle, puisque les couches suivantes seront de taille plus faible.

3. Une couche fonction d'activation (typiquement, la *Rectified Linear Unit* ou ReLU, remplaçant simplement les valeurs négatives par 0), qui transforme simplement le signal par une fonction donnée comme le font les neurones d'un perceptron.

La sortie peut ensuite être rebranchée à une autre couche de convolution. Les couches successives et de taille de plus en plus réduites sont finalement reliées à un petit MLP en sortie, qui rend une prédiction finale. Toutes ces couches ne sont pas toujours utilisées ou ordonnées de la même façon, les auteurs précisent généralement le plan de leur architecture avec un diagramme montrant leur organisation.

D'autres architectures récentes se proposent comme successeurs des réseaux de convolutions, notamment les réseaux d'attention et l'architecture « Transformer » [160].

IV.1.2 Applications pour la prédiction de structures

Pour modéliser une structure macromoléculaire sous forme de vecteurs numériques, plusieurs approches existent (et ont été explorées pour la recherche sur les protéines).
Il y a deux choix à faire : celui des caractéristiques utilisées, et celui des étiquettes utilisées.

Pour les caractéristiques, il faut choisir des descripteurs que l'on connait sur la molécule que l'on souhaite replier. La première caractéristique triviale est la séquence. Cependant, les réseaux de neurones ne peuvent pas traiter de chaines de caractères comme "ACGU...". On utilise alors un encodage binaire de la séquence par des vecteurs : (1,0,0,0,0) pour A, (0,1,0,0,0) pour C, (0,0,1,0,0) pour G, (0,0,0,1,0) pour U et (0,0,0,0,1) pour les autres. Ceci donne donc déjà $5L$ valeurs pour une séquence de taille L. Pour les protéines, on utilise un vecteur de taille 22. En plus de la séquence, on peut également encoder la position du nucléotide dans la séquence de façon explicite (position absolue ou relative), pour aider le réseau à se localiser dans la molécule. Enfin, de nombreuses méthodes essaient d'exploiter l'information contenue dans l'homologie des séquences et la covariation des nucléotides dans les alignements multiples de séquences homologues. En effet, des nucléotides qui covarient fortement à deux positions d'un alignement multiple indiquent une pression de sélection de l'un sur l'autre, ce qui trahit une interaction fonctionnelle entre eux et donc une proximité (ils sont en contact). Les méthodes de prédiction de structures secondaires dites comparatives exploitent ce principe et sont très utilisées et efficaces. Donc, en plus de la séquence, il est possible d'utiliser par exemple les fréquences nucléotidiques à chaque position calculées sur un alignement de séquences [10]. D'autres méthodes calculent des caractéristiques plus complexes, comme celles de l'analyse des covariations (DCA) par maximisation de la pseudo-vraisemblance (plmDCA) [111,357]. Ces caractéristiques donnent des scores à chaque position dans la séquence, une mesure de leur implication dans un

contact important (notées h_i), et des scores à chaque couple de positions, mesurant leur covariation (notés J_{ij}). Enfin, d'autres proposent de s'affranchir de ces calculs en donnant directement l'alignement de séquences en entrée et en laissant le réseau de neurones calculer ce dont il aurait besoin [241].

Pour les étiquettes (que l'on veut apprendre au réseau à prédire correctement), on rencontre plusieurs types d'informations : tout d'abord, les contacts entre résidus, indiquant des interactions à distance. Ce genre d'information peut être encodée sous la forme d'une matrice triangulaire supérieure, ou symétrique, contenant des 0 ou 1 selon que les nucléotides correspondant aux indices de la ligne et de la colonne sont en interaction ou non. On citera des outils de prédiction des contacts au sein des protéines [141,352], ainsi que les récentes applications (développées en parallèle de cette thèse sur la période 2018/2021) à la prédiction de contacts ou de structure secondaire d'ARN [219], [220], [222]–[225], [227]–[231], [278], [279].
Ensuite, certains réseaux de neurones essaient de prédire la conformation globale du squelette de la molécule par la prédiction, pour chaque nucléotide, des angles de torsion avec le précédent et le suivant. Cela peut se faire soit par un problème de régression directe des valeurs [129], soit par un problème de classification multiple en prédisant l'appartenance de chaque nucléotide à un alphabet structural donné [10]. Un alphabet structural est une collection de conformations locales fréquemment observées (par exemple, des clusters dans l'espace des angles de torsion). Le réseau de neurones prédit donc une 'lettre' de cet alphabet pour chaque nucléotide, ce qui correspond à une conformation précise. Enfin, la principale innovation technologique apportée fin 2018 par AlphaFold [302], puis directement réutilisée par ses concurrents [364], est la prédiction non plus des contacts entre nucléotides (une mesure binaire oui/non) mais des distances réelles (une valeur numérique). Ces méthodes prédisent donc des matrices (toujours symétriques) de distances entre résidus au lieu de valeurs binaires 0 ou 1.

IV.1.3 Des protéines à l'ARN

Si l'idée d'adapter les algorithmes d'apprentissage profond des protéines à l'ARN est prometteuse, il y a cependant des différences importantes à considérer.

IV.1.3.1 Les différences théoriques

Il y a plusieurs différences théoriques majeures entre la structure des protéines et celles des ARN. Comme on l'a vu en introduction, la structure des ARN est maintenue stable essentiellement grâce aux bases, qui forment des empilements et appariements. C'est l'inverse dans les protéines, où la structure secondaire est stabilisée par des interactions hydrogène distantes entre atomes du squelette. Ainsi, prédire la forme du squelette d'une protéine, c'est prédire ce qui stabilise sa forme globale, alors que prédire la forme du squelette dans l'ARN, c'est passer à côté de ce qui le stabilise (la position des bases). Le squelette adopte la conformation dictée par les bases. Ensuite, le squelette des protéines est modélisé majoritairement par deux angles de torsions par acide aminé (nommés phi et psi

respectivement), la liaison peptidique étant plane, et les angles plans constants. Ce n'est pas le cas dans l'ARN, où le squelette présente 6 angles de torsion par nucléotide, auxquels il faut ajouter la conformation du cycle furanose (le *sugar-pucker*), théoriquement définie par 5 angles mais formant deux conformations majoritaires (nommées C3'-endo et C2'-endo). Heureusement, ce problème peut être contourné par l'utilisation d'angles de pseudo-torsions [173], dont l'équipe de Pyle a montré qu'elles suffisent à bien décrire les conformations possibles du squelette de l'ARN.

Enfin, le nombre minime de résidus (4 contre 21 communs) a aussi pour conséquence de rendre tout motif de séquence beaucoup plus fréquent dans l'ARN que dans les protéines par pur hasard. La reconnaissance d'un motif de séquence dans l'ARN porte donc moins d'information en soit qu'un motif de séquence protéique de même taille, et la comparaison à des distributions de référence est obligatoire. On note aussi la présence rare mais non négligeable de nucléotides modifiés dans l'ARN, qui rajoutent des difficultés dans les modèles quant au choix de traitement à leur appliquer (les ignorer, les transformer en un nucléotide proche, ou les modéliser au prix de modèles plus complexes). On rappelle qu'un nucléotide modifié n'est pas forcément plus proche du nucléotide non-modifié dont il est issu, par exemple lorsqu'une méthylation vient supprimer l'accessibilité d'un atome donneur ou accepteur de liaisons hydrogènes. Il est plus judicieux de substituer au nucléotide modifié le nucléotide canonique capable d'interagir de la même façon que lui. Une liste de référence de substitution des nucléotides modifiés est proposée et utilisée par X3DNA/DSSR [207]. Une ancienne version contenant « seulement » 596 nucléotides modifiés est par exemple disponible en ligne à l'adresse : http://x3dna.org/luxfiles/modified-bases-2013oct18.txt

IV.1.3.2 Les différences pratiques

D'un point de vue pratique, on a d'autre difficultés, qui se révéleront plus pénalisantes que les difficultés théoriques. Premièrement, les données disponibles sur l'ARN sont de moins bonne qualité, pour deux raisons.

1. Leur quantité est plus faible : environ 12 000 structures d'ARN pour 170 000 structures de protéines selon les statistiques de la PDB, qui sont presque exhaustives des données publiques (https://www.rcsb.org/stats/summary). C'est fortement pénalisant pour les modèles d'apprentissage profond qui nécessitent plutôt de grosses quantités de données.
2. Mais surtout, leur diversité est faible. Quelques familles d'ARN, notamment les ARN de transfert (ARNt) et les ARN ribosomiques (ARNr 5S, 5.8, 16S ou 23S) dominent largement les structures disponibles. Non seulement, de nombreuses copies de la même molécule peuvent exister, mais au sein d'une famille, les ARN ont souvent des séquences et structures très proches. Il est donc extrêmement difficile de chercher à éliminer la redondance sans supprimer purement et simplement une large majorité des données disponibles.

Ensuite, une large majorité des familles d'ARN non codants connues n'ont jamais été résolues en 3D (aucun des membres). On a donc aucun exemple de leur repliement.

Il y a également une différence sur les tailles des molécules disponibles. Les ARN comme les protéines varient en taille selon les familles de quelques nucléotides à plusieurs milliers. Cependant, pour les protéines, la majorité est de petite taille, et les très grands spécimens restent des exceptions dans le jeu de données disponibles. Pour les ARN, les grands spécimens sont les ARN ribosomiques, et représentent une large fraction des données disponibles. On doit donc entrainer des modèles capables d'atteindre les milliers de nucléotides (ce qui n'a pas été fait pour les protéines), ou se priver de données disponibles.

Enfin, il n'y a pas de jeu de données standardisé d'ARN disponible, téléchargeable et exploitable rapidement. Les structures sont déposées dans la PDB mêlées à des complexes d'autres éléments et non annotées (pas de structure secondaire, famille, motifs, interactions, etc.). Depuis 2018, quelques jeux de données ont été proposés, notamment exploités par les outils de prédiction de contacts précédemment cités, mais aucun ne valide tous les critères qui en feraient un bon jeu de données exploitable pour l'apprentissage profond :

- contenir un maximum de données disponibles,
- avoir des données nettoyées : avoir extrait et séparé les chaines individuellement, avoir renuméroté les résidus d'une façon standardisée, utiliser un unique format de fichier standard,
- avoir des données annotées : connaître pour chaque chaine sa séquence, sa structure secondaire, les interactions présentes au sein de la molécule sous forme symbolique (représentation sous forme de graphe ou de liste par exemple), et si possible sa famille, afin de pouvoir exploiter des données d'homologie à partir d'alignement de séquences,
- avoir un pré-découpage intelligent en ensembles d'entrainement et de tests qui limite la fuite d'informations entre molécules homologues,
- avoir une population de structures de référence [332] qui permettrait de calculer des potentiels statistiques avec le jeu de données d'entrainement.

Les travaux présentés dans ce chapitre sur l'application de l'apprentissage profond à la prédiction de structures ont donc nécessairement commencé par la résolution du problème du jeu de données standardisé. Ils ont abouti à RNANet [27], présenté dans la partie suivante. Il est à ce jour l'exemple le plus abouti à ma connaissance de jeu de données standardisé d'ARN, et reste la principale contribution de cette thèse au domaine de recherche.

IV.2 Un jeu de données intégratif : RNANet

IV.2.1 Objectifs attendus du jeu de données

La conception est inspirée à l'origine par la publication en 2019 de ProteinNet [12] (d'où la similitude de nom), qui donne des lignes de conduite à suivre. On peut lister les points intéressants suivants :

- une extraction et numérotation standardisée des chaines,
- une construction automatique d'alignements de séquences homologues pour chacune, pour les replacer dans leur contexte de variabilité phylogénétique,
- un pré-calcul de sous-ensembles d'entrainement, de test et de validation garantissant que les structures ne dépassent pas un certain pourcentage de similarité entre elles et entre structures des groupes. La mesure de distance entre point de données utilisée est basée sur un clustering des alignements multiples des séquences homologues de chaque point de données entre eux.
- la sauvegarde dans un format de données directement exploitable pour les réseaux de neurones.

Nos objectifs sont donc initialement les mêmes, avec en plus :

- l'annotation systématique des structures, permettant de produire toutes sortes d'informations qui pourraient être utilisées comme étiquettes à prédire par un réseau de neurones,
- l'automatisation complète du processus de création, de façon à pouvoir automatiser le re-calcul régulier et maintenir le jeu de données à jour,
- la documentation systématique des raisons qui ont poussé à l'ignorance ou la mise de côté d'une certaine structure, nucléotide, famille d'ARN, etc.

IV.2.2 Méthodes de construction de RNANet

On a donc développé un pipeline dont les étapes sont résumées sur la Figure 4.2. Nous avions besoin de plusieurs éléments clés, dont les choix de conception sont détaillés dans les paragraphes suivants.

IV.2.2.1 Jeu de données de structures 3D

Nous avons choisi d'utiliser les listes non redondantes proposées par le BGSU RNA Group [198]. Ce n'est pas la liste la plus complète des structures d'ARN disponibles (elle exclut entre autres les structures obtenues par RMN, qui ne sont pourtant pas négligeables). Cependant ce jeu de données présente de nombreux avantages : il est mis à jour une fois par semaine, il est accessible au téléchargement automatique par une interface de programmation, et surtout, les structures sont groupées par classes d'équivalence avec sélection d'un représentant pour chaque classe, ce qui fournit des informations précieuses sur la redondance du jeu de données. Les structures présentes dans la liste sont donc téléchargées depuis la PDB. Dans la version originale publiée en novembre 2020, elles

l'étaient toutes. Depuis la version 1.4b, seule la structure représentante est téléchargée par défaut. La considération de l'ensemble reste possible via une option de ligne de commande.

Il fallait également choisir un logiciel d'annotation de ces structures pour en extraire des listes d'appariements et interactions, des mesures d'angles, et d'autres descripteurs. Notre choix s'est tourné vers DSSR [208], logiciel continuellement mis à jour, efficace computationnellement, accordant une attention particulière aux nucléotides modifiés, et délivrant le plus grand nombre de descripteurs utiles.

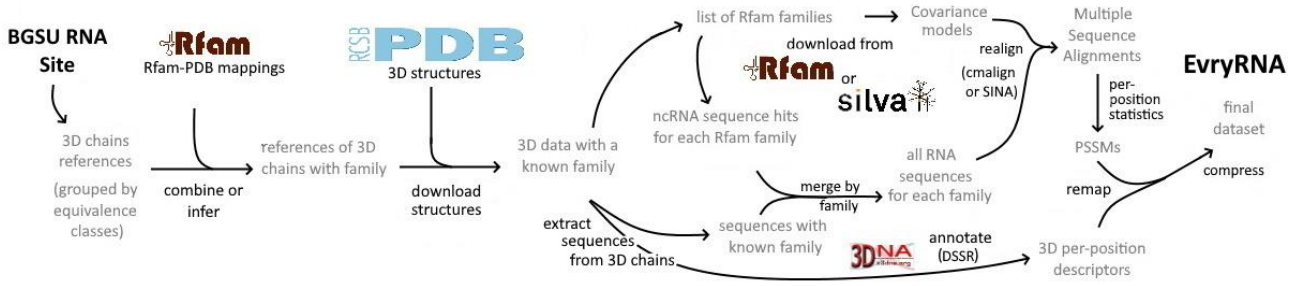


Figure 4.2 – Pipeline de RNANet. *À partir des chaines 3D de la liste de BGSU, on identifie (ou on infère) les chaines pour lesquelles une famille Rfam est connue. On télécharge les structures 3D correspondantes depuis la PDB* [32]*. D'une part, on aligne leurs séquences avec des séquences homologues téléchargées sur Rfam* [172] *ou SILVA* [264] *pour les ARNr. D'autre part, on annote ces structures avec DSSR* [208]*. Enfin, on remappe les indices des nucléotides entre les chaines 3D et les alignements multiples de séquences, sur lesquels on a calculé d'autres descripteurs (fréquences, pourcentages de gaps, structure consensus, plmDCA). On archive enfin les données dans des fichiers CSV et une base de données SQL. La figure présente les étapes du pipeline original, les versions récentes de RNANet (1.4b et plus récentes) n'utilisent plus SINA/SILVA par défaut, et ne considèrent que les structures représentatives d'une classe d'équivalence par défaut.*
Figure extraite de la publication originale de RNANet[27] en Novembre 2020.

IV.2.2.2 Données d'homologie de séquence

Pour pouvoir calculer des fréquences nucléotidiques pour chaque position de chaque séquence, il faut construire un alignement de séquences homologues pour chaque chaine 3D. Notre méthode de référence, ProteinNet, le réalisait avec le logiciel de référence `jackhmmer` de la suite Hmmer (https://hmmer.org). Ce logiciel recherche des séquences homologues à une ou plusieurs séquences données en entrée de façon itérative, ce qui permet rapidement de construire d'importants alignements de séquences avec les séquences les plus homologues présentes dans une base de données. Cependant, jackHmmer est prévu pour les protéines, et dans la suite similaire adaptée aux ARN qu'est Infernal [246], aucun équivalent n'a été porté. D'autres logiciels similaires existent (HHBlits [274]) mais ne supportent pas non plus les ARN. Il existe effectivement une alternative créée par le ViennaRNA group, nommée RNAlien [110], dont nous n'avions malheureusement pas connaissance à cette époque. Cette perspective est explorée seulement maintenant dans l'équipe.

Nous avons donc choisi une solution plus simple pour obtenir des alignements multiples de séquences : utiliser ceux déjà proposés par Rfam et leur classification des ARN non codants en familles, chaque famille étant proposée avec un modèle de covariation (CM), une liste de séquences homologues, et même la liste des structures 3D disponibles dont on sait qu'elles appartiennent à cette famille. Toutes ces informations sont elles aussi accessibles automatiquement via une interface de programmation (API) et un serveur SQL public. La principale difficulté consiste à identifier une famille Rfam correspondant à chaque chaine 3D disponible. On utilise pour cela la liste fournie par Rfam (et issue d'une recherche par l'outil `cmscan` d'Infernal [246]). On étend les résultats de mapping fournis par cette liste en utilisant les classes d'équivalence de BGSU [198] : si un ARN de la classe d'équivalence est mappé à une famille d'ARN, alors on infère que les autres le sont aussi. Cette extension est réalisée même si le pipeline ne considère ensuite que le représentant de la classe. Une option permet également de retourner plusieurs copies de la même chaine si elle est potentiellement identifiée comme faisant partie de plusieurs familles.

IV.2.2.3 Réalignement et calcul des caractéristiques d'homologie

À partir des séquences identifiées comme appartenant à chaque famille Rfam (issues de la base de données Rfamseq [171], une collection de génomes de référence), on veut maintenant calculer un alignement multiple de séquences. Les principaux algorithmes d'alignement à notre disposition sont :

- L'outil `cmalign` d'Infernal avec les options par défaut, très rapide [246], mais nécessitant des quantités de mémoire RAM devenant très importantes lorsque la taille des séquences augmente (plusieurs centaines de giga-octets). Son algorithme supporte les séquences tronquées, donne un indice de confiance sur l'alignement de chaque colonne, et est prévu pour être utilisé avec les CM de chaque famille Rfam. Cet algorithme est une heuristique, mais donne des résultats largement acceptables dans la majorité des cas.
- L'outil `cmalign` avec l'algorithme exact basé sur CYK [108] (algorithme de Cocke-Younger-Kasami). Cet algorithme garantit la solution d'alignement optimale, et troque l'importante complexité en mémoire pour une plus importante complexité en temps. Ainsi, les calculs sont plus longs, mais peuvent se faire avec très peu de mémoire RAM. Malheureusement, cette approche ne permet pas de supporter les séquences tronquées et ne retourne pas de scores de confiance.
- Pour aligner les ARN ribosomiques de grande taille qu'il est difficile d'aligner avec `cmalign`, on peut utiliser une alternative spécialisée, l'aligneur SINA [263]. SINA aligne des séquences cibles avec celles de la base de données SILVA [264], avec une quantité importante mais raisonnable de mémoire. SILVA sépare en deux catégories différentes les grandes sous-unités (LSU) et les petites (SSU). On utilise donc les "clans" Rfam, des méta-familles Rfam, pour identifier certaines familles comme LSU ou SSU. Le clan 111 comprenant les familles {RF00117, RF02542, RF02545, RF01959, RF01960} est identifié comme correspondant aux SSU, et le clan 112 comprenant les familles {RF00002, RF02540, RF02541, RF02543, RF02546} comme les LSU.

  Dans la version initiale de RNANet, le choix avait été fait d'aligner les séquences

d'ARN ribosomiques avec SINA (sans utiliser leur CM venant de Rfam), et les autres familles avec `cmalign`. Ceci permettait d'exécuter le pipeline sur une machine de calcul raisonnable équipée de 48 Go de RAM (voir Figure 4.3). Malheureusement, les alignements d'ARN ribosomaux issus de SINA étaient de piètre qualité. À partir de la version 1.4 beta de RNANet, Infernal (`cmalign`) est donc toujours utilisé par défaut, ce qui nécessite au moins 128 Go de RAM. L'utilisation de SINA peut être demandée à l'aide d'une option de ligne de commande, tout comme l'utilisation de l'algorithme CYK. Ainsi, les trois options sont possibles et laissées au choix de l'utilisateur.

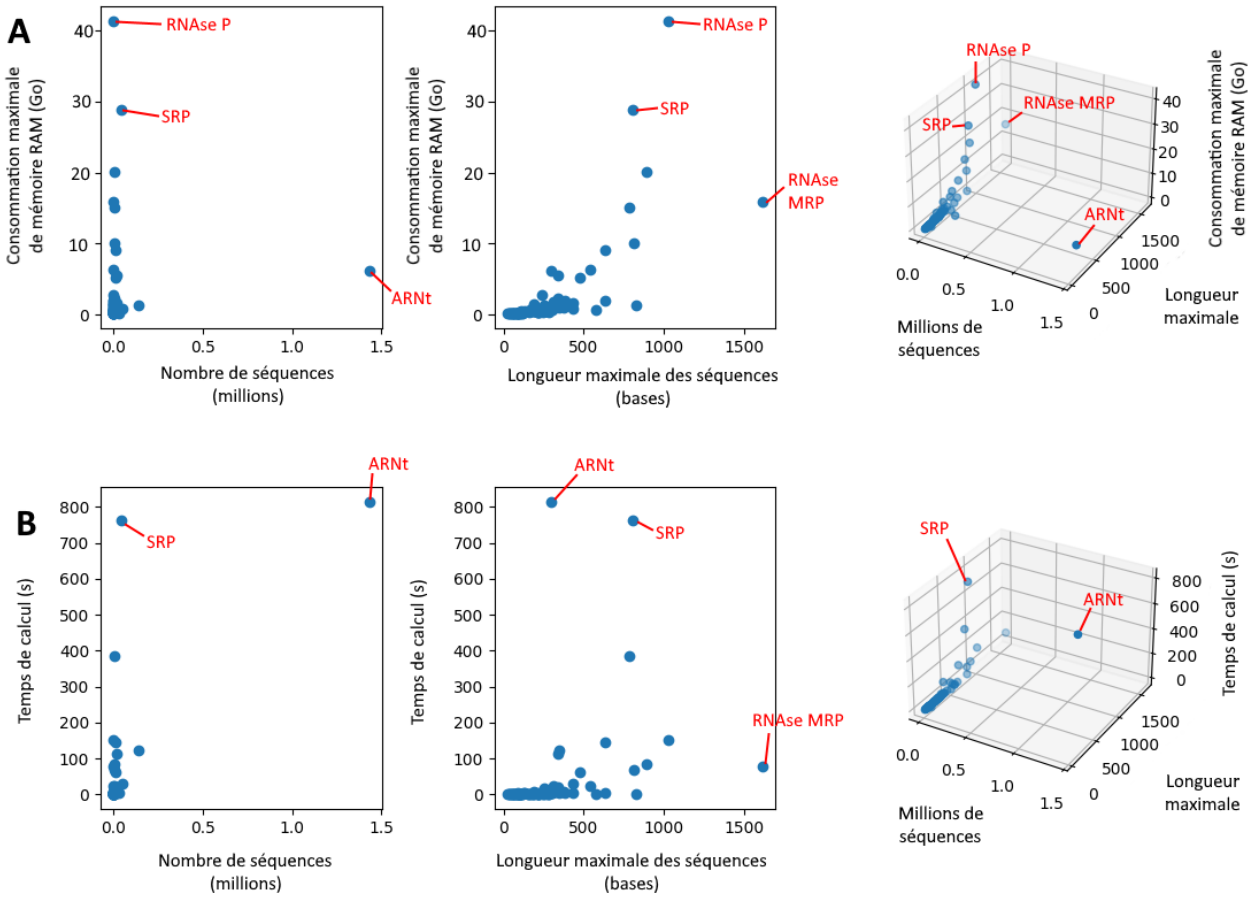


Figure 4.3 – Performances computationnelles de `cmalign` (hors ARNr). ***(A)*** *Consommation de mémoire RAM en fonction du nombre de séquences à aligner et de leur longueur maximale. En excluant les ARNr, on peut exécuter RNANet sur une station de travail équipée de 48 Go de RAM, ce qui reste raisonnable. Les familles les plus gourmandes sont celles des RNAses P (RF00009) et des particules de reconnaissance du signal (ou SRP, RF00017).* ***(B)*** *Temps de calcul en fonction du nombre de séquences à aligner et de leur taille maximale. En excluant les ARNr, chaque famille peut être alignée en moins d'un quart d'heure (sur notre serveur équipé de 60 cœurs de CPU Intel Xeon E7-4850v4 à 2.10GHz). Les familles les plus longues à traiter sont les ARN de transfert (RF00005, avec 1,43 million de séquences à aligner) et les SRP à nouveau (RF00017).*

Une fois l'alignement calculé, un calcul des fréquences nucléotidiques (et fréquence des gaps) est calculé pour chaque colonne de chaque alignement. Dans les versions récentes de RNANet (1.5 beta et supérieures), les alignements sont également soumis à une analyse

des covariations (par maximisation de pseudo-vraisemblance) par l'outil pyDCA [374]. On retient alors pour chaque famille les caractéristiques h_i et J_{ij} de la plmDCA, qu'on a présentées en section 4.1.2. On enregistre également pour chaque colonne un nucléotide consensus pour la position, et si l'alignement a été fait par `cmalign`, une structure secondaire consensus.

IV.2.2.4 Re-mapping des positions entre nucléotides 3D et colonnes de l'alignement

Une fois les séquences issues des structures 3D réalignées avec leurs séquences homologues, il faut calculer les correspondances d'indices dans les séquences 3D et séquences réalignées. Dans les séquences 3D, des gaps symbolisés par des tirets ('-') peuvent être présents, et représentent des nucléotides dont on suppose ou connait l'existence mais qui n'ont pas été résolus en 3D. Dans les séquences alignées, deux types de gaps peuvent être trouvés. Les gaps d'insertion par rapport au CM, notés par un point ('.'), n'indiquent rien de spécial, et servent à combler le vide créé par une insertion dans une autre séquence de l'alignement. Les gaps de délétion, notés par un tiret ('-'), indiquent la délétion d'un nucléotide manquant par rapport au CM et à la séquence consensus de la famille d'ARN. Les alignements créés par SINA n'utilisent que des gaps tirets.

On procède de la façon suivante : les correspondances de nucléotides sont mappées entre elles. Les correspondances de gaps sont également mappées entre elles (tiret contre tiret). Si un nucléotide dans la structure 3D est trouvé en face d'un gap dans la séquence alignée (quelque soit le type de gap), on ignore les colonnes de l'alignement jusqu'à tomber sur le nucléotide en question. Si un gap (tiret) dans la structure 3D ne se trouve pas en face d'un gap tiret dans la séquence alignée, on ignore les colonnes de l'alignement jusqu'à trouver un gap tiret. Si on n'en trouve pas (on tombe sur un nucléotide réel avant d'en trouver un), on mappe ce gap au prochain gap "point". Si on n'en trouve pas non plus (le gap est en face d'un nucléotide dans la séquence alignée), alors l'alignement est localement de mauvaise qualité. On ne mappe le gap à aucune colonne de l'alignement. Si un nucléotide dans la structure 3D se retrouve en face d'un nucléotide qui ne correspond pas, une erreur est renvoyée (en pratique, ceci n'arrive pas).

Les informations des colonnes qui ne sont mappées à aucune chaine 3D ne sont pas calculées. En les retirant, on peut reconstituer un alignement uniquement constitué des chaines 3D de la même famille.

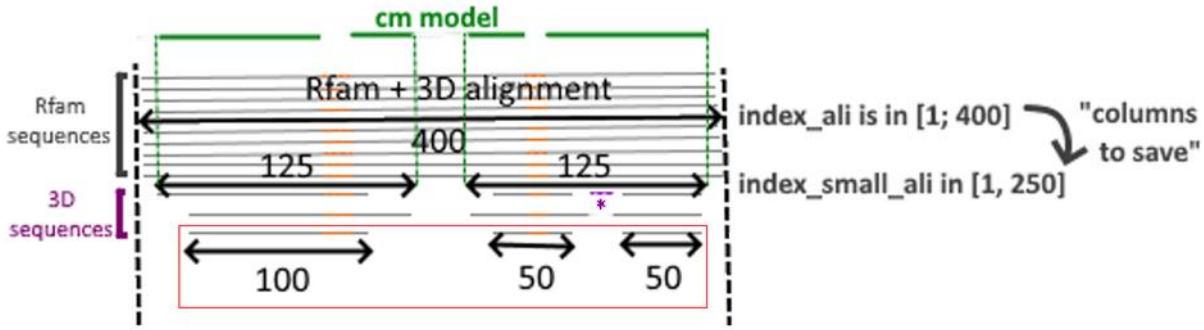


Figure 4.4 – Systèmes d'indices de RNANet. *Exemple fictif d'alignement multiple de séquences homologues. À partir de l'alignement complet (séquences de Rfam et séquences issues des chaines 3D), seules certaines positions seront conservées dans le "petit" alignement, contenant uniquement les chaines 3D. Certains nucléotides sont alignés avec le modèle consensus de la famille (CM en vert, de taille 220 par exemple), d'autres constituent des insertions par rapport au modèle consensus (en orange), et certains nucléotides manquent par rapport au consensus (astérisque violette). Si l'on considère une unique chaine d'ARN 3D, par exemple celle encadrée en rouge et de taille 200, on aura les indices suivants : index_chain sera dans l'intervalle [1, 200], index_ali dans [1, 400], index_small_ali dans [1, 250], et enfin cm_coord sera dans [1, 220] et non défini dans les zones orange.*

Après cette étape, chaque nucléotide d'une chaine d'ARN se retrouve donc identifié par plusieurs indices, illustrés sur la Figure 4.4. On liste, en utilisant les noms des descripteurs utilisés dans la base de données :

- `index_chain` : La position du nucléotide dans la chaine, entre 1 et N. Ce nombre est issu de notre processus de normalisation de la numérotation,
- `old_nt_resnum` : L'ancien numéro du nucléotide dans le fichier 3D. Cet indice peut prendre toutes sortes de valeurs, positives, nulles, négatives, contenant des lettres, etc…, ceci prouvant la nécessité d'une numérotation normalisée,
- `index_ali` : La position du nucléotide dans l'alignement de séquences, déterminée par la procédure décrite dans ce paragraphe,
- `index_small_ali` : La position du nucléotide dans l'alignement des séquences 3D seules (en éliminant les colonnes uniquement utiles aux séquences de Rfamseq pour lesquelles on n'a pas de structure 3D),
- `cm_coord` : La position du nucléotide dans le modèle consensus de la famille d'ARN.

IV.2.2.5 Mode sans homologie

Une option dans RNANet permet d'ignorer la recherche de séquences homologues, l'alignement et le re-mapping. Cette option a l'avantage de permettre la considération d'un plus grand nombre de chaines 3D d'ARN, notamment toutes celles pour lesquelles on ne peut pas identifier d'appartenance à une famille Rfam. Les autres étapes du pipeline sont donc appliquées : les structures sont téléchargées, annotées par DSSR, et les chaines extraites et renumérotées de façon standardisée.

IV.2.2.6 Calcul de statistiques

Une fois chaque chaine d'ARN 3D correctement mappée à une famille d'ARN, le pipeline calcule différentes statistiques sur les nombres de chaines disponibles et les familles Rfam représentées en fonction de la résolution des structures 3D. Une procédure réalise un histogramme de la taille des chaines disponibles. Une autre calcule les fréquences nucléotidiques intra-chaine.

Si RNANet a été exécuté au moins une fois avec homologie, les distances entre paires de nucléotides de chaque structure 3D sont mesurées et rangées dans une matrice de distances (pour chaque chaine 3D). Des matrices de moyennes et d'écarts-type sont calculées par famille d'ARN, en considérant uniquement les positions du modèle de covariation (CM) ; les insertions sont ignorées, et les délétions prennent la valeur "`NaN`". Une procédure calcule la fréquence d'apparition de chaque type de nucléotide (en incluant les nucléotides modifiés) dans chaque famille d'ARN, et la fréquence totale. Une autre procédure mesure la fréquence de chaque type d'interaction non-canonique, également par famille puis sur l'ensemble du jeu de données. Une troisième procédure calcule des matrices d'identité de séquences à partir des alignements multiples de séquences contenant toutes les chaines 3D de la même famille.

Si RNANet a été exécuté au moins une fois en mode sans-homologie (et ce pour disposer du plus large jeu de données possible), une procédure réalise une analyse statistique des paramètres géométriques observés dans le jeu de données. Par type de nucléotide, on mesure les distances interatomiques, les angles, les torsions entre atomes. Les mesures qui concernent le squelette sont mélangées de façon indépendante de la base. Une procédure automatique mesure et rassemble en clusters les pseudo-torsions eta et thêta (et leurs variantes), de façon à reproduire automatiquement le travail de Wadley & Pyle 2007 [342]. La distribution statistique de chaque paramètre est estimée à l'aide d'un modèle mixte Gaussien. Des résultats seront présentés en section 4.2.3.

IV.2.2.7 Découpage en jeux de données d'entrainement et de validation

Notre méthode de référence, ProteinNet [12], propose une méthode de construction soignée de sous-ensembles de données (jeu d'entrainement plus jeu de test) dont les chaines ne dépassent pas un certain pourcentage de similarité les unes par rapport aux autres entre elles d'une part, et entre les deux jeux d'autre part.
En effet, la partition aléatoire de l'ensemble de données implique l'hypothèse d'indépendance et de distribution identique des point de données. Cette hypothèse n'est pas valide avec des séquences biologiques : elles sont reliées entre elles par des liens évolutifs et phylogénétiques. Il faut donc manuellement s'assurer que des données ne fuitent pas entre le jeu de données d'entrainement et le jeu de test.

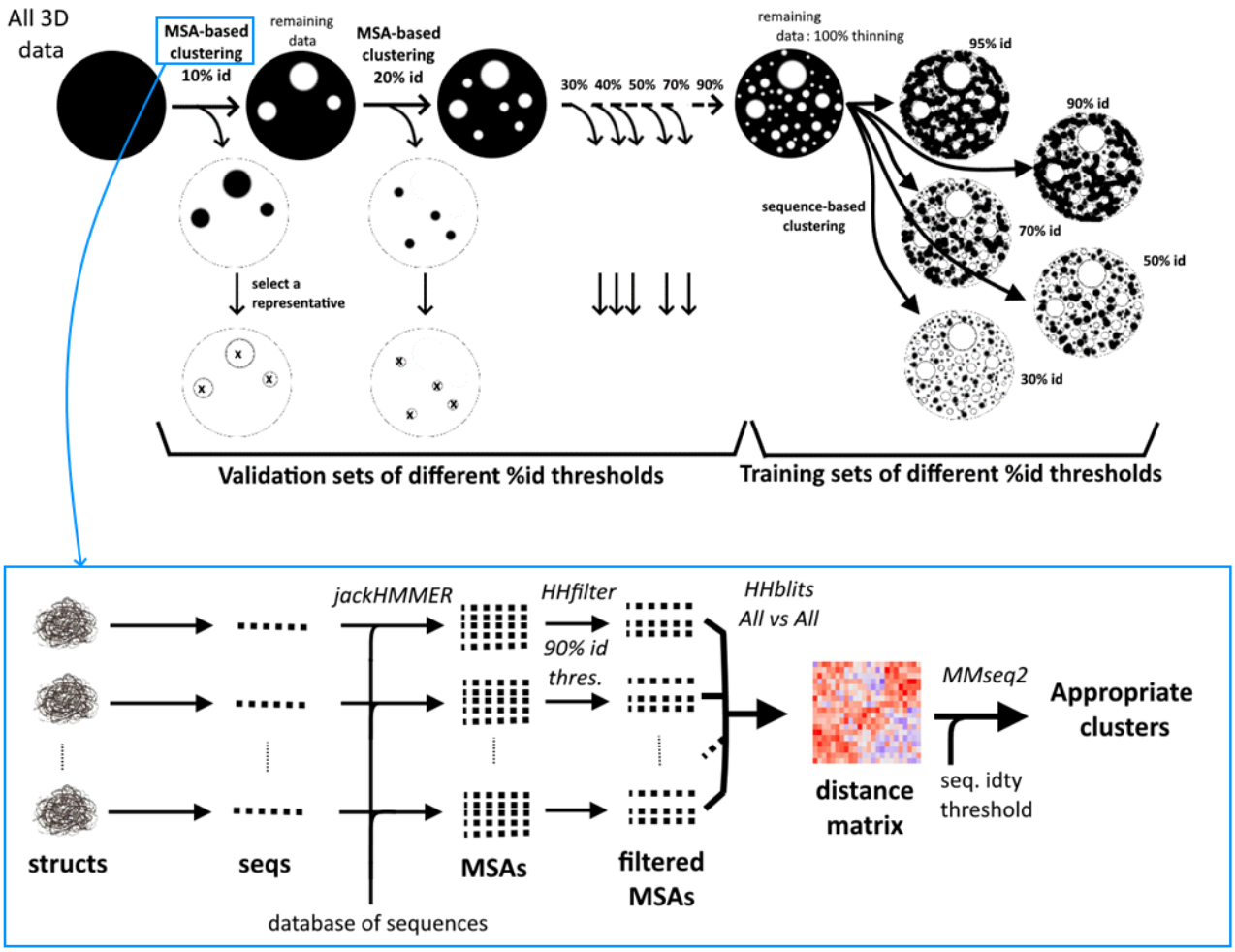


Figure 4.5 – Stratégie de tri des données de ProteinNet. *Les données sont regroupées par pourcentages de similarité successifs, et séparés en même temps en jeux de données d'entrainement et de validation. La première phase consiste à appliquer successivement des étapes de clustering basées sur la similarité des alignements multiples de séquences (MSA), comme illustré dans le cadre bleu. Pour un clustering basé sur les MSA à X% d'identité, la procédure identifie un sous-ensemble des données organisé en clusters, dont la similarité des MSA entre deux molécules d'ARN piochées dans deux clusters différents sera inférieure à X%. Un seul représentant est conservé par cluster pour former le jeu de test, toutes les molécules proches du représentant étant supprimées, ce qui assurera la distance minimale entre jeu de test et jeu d'entrainement. Une fois ces premières chaines d'ARN supprimées du jeu initial de données, un nouveau clustering peut être appliqué avec un X% plus permissif. Lorsqu'on a retiré suffisamment de données du jeu initial pour préparer des jeux de test, on réalise des clusterings simples basés sur l'identité de séquence pour préparer les jeux d'entrainement.*

La méthode de ProteinNet [12] est détaillée sur la Figure 4.5. L'élément clé de celle-ci est le clustering basé non pas sur l'identité de séquence, mais sur l'identité des alignements multiples de séquences. Cette approche augmente la sensibilité de détection des homologies distantes entre séquences par rapport à la comparaison des séquences seules deux à deux. Malheureusement, cette approche est impossible à reproduire avec les données d'ARN. La première raison est l'impossibilité de construire un alignement de séquences différent pour chaque molécule d'ARN (indisponibilité d'outils comme

jackHmmer pour l'ARN). Ensuite, la forte redondance des données d'ARN entraine nécessairement une réduction drastique de la quantité de données si l'on applique un filtre sur le pourcentage d'identité de séquence. Ne pouvant décider à l'avance d'un seuil pertinent ou d'une méthode de filtration (autre que la sélection d'un unique représentant par classe d'équivalence BGSU), nous avons préféré distribuer le jeu non filtré en avertissant l'utilisateur, lui laissant le choix du compromis redondance/quantité qu'il souhaitera.

Cependant, depuis la découverte de RNAlien [110] et depuis la récente mise à jour de la base de données de séquences d'ARN non codants RNACentral [281] qui embarque maintenant un moteur de recherche de séquences homologues, il est devenu possible de générer un alignement de séquences pour chaque séquence d'ARN. Ceci laisse place à l'exploration réelle de l'approche rigoureuse de ProteinNet, dans la limite de la quantité de données disponibles.

IV.2.3 Résultats et description statistique des données actuelles

IV.2.3.1 Implémentation

Le pipeline est implémenté en langage Python 3.8, sous la forme de deux fichiers de script (susceptibles d'évoluer en vrai package Python). Un troisième script Bash gère l'automatisation de la mise à jour du jeu de données mensuelle sur notre serveur. Il dépend des logiciels DSSR (v1.9.9), Infernal (v1.1.4) et SINA (1.6.0) à installer séparément, et de plusieurs packages Python, dont BioPython 1.78 [63] pour manipuler les fichiers 3D au format mmCIF et les alignements de séquences aux formats "aligned-FASTA" et "Stockholm". Le pipeline a été fortement étudié pour optimiser le traitement concurrentiel des tâches et minimiser le temps de calcul. Il est donc adapté aux serveurs de calcul fortement multicœur. Néanmoins, la consommation mémoire augmente linéairement avec le nombre de cœurs disponibles (notamment pendant la phase d'annotation par DSSR), la machine doit donc être équipée de façon proportionnelle avec environ 1,5 Go de RAM par cœur de CPU. Il est possible de limiter par des options le nombre de cœurs CPU. En raison d'entrées/sorties constantes avec le disque, le temps de calcul de RNANet dépend également beaucoup de la vitesse du périphérique de stockage utilisé, l'usage de disques SSD NVMe est fortement recommandé.

Le pipeline fonctionne de manière différentielle : il ne réexécute pas les tâches de téléchargement, annotation ou alignement qui ont été précédemment terminées. Il se contente d'exécuter les tâches qui nécessitent un re-calcul pour mettre à jour la base de données actuelle avec les structures les plus récentes. De nombreuses options de ligne de commande sont disponibles, notamment les plus utiles :

- une option pour donner un seuil de résolution minimale nécessaire pour considérer une structure 3D. La valeur de 4.0 Angströms est utilisée par défaut, et le jeu de données publié sur EvryRNA utilise un seuil très permissif de 20.0 Angströms, le but étant d'avoir un maximum de données précalculées, même de mauvaise qualité (les

utilisateurs d'EvryRNA restent libres de filtrer après leur téléchargement),

- les options de contrôle de la redondance : celle pour autoriser la considération de toutes les structures redondantes d'une classe d'équivalence, et celle pour autoriser les copies d'une même chaine si celle-ci peut être associée à plusieurs familles d'ARN,
- les options de contrôle de l'algorithme d'alignement, pour choisir d'utiliser SINA pour les ARN ribosomaux, ou passer des options à `cmalign`, par exemple pour lui indiquer d'utiliser l'algorithme CYK,
- les options de contrôle des tâches statistiques qui seront effectuées à l'issue des calculs.

Le résultat des calculs est archivé et proposé au téléchargement sur le site EvryRNA (https://evryrna.ibisc.univ-evry.fr/evryrna/rnanet). Les résultats sont proposés sous deux formes différentes. On propose une base de données SQL (SQLite 3) structurant les informations sous forme de tables. La structure de la base de données est illustrée sur la Figure 4.6. Une documentation plus détaillée de la base de données est disponible dans le dépôt de code, dans le dossier "doc/" et plus précisément le fichier `doc/Database.md`, (voir https://forge.ibisc.univ-evry.fr/lbecquey/RNANet/). On peut l'utiliser pour exécuter des requêtes complexes, extraire un sous ensemble du jeu de données, ou calculer des statistiques. L'usage de SQL permet de stocker toutes ces données de façon très compacte (par opposition à des fichiers texte formatés en CSV ou JSON). Cela permet aussi de supporter des entrées/sorties en lecture et écriture simultanées lors de la production et de l'exploitation des données par de nombreux processus parallèles.

En complément de la base de données, pour rendre le jeu de données plus exploitable, on propose au téléchargement des fichiers texte au format CSV résumant toute l'information disponible pour chaque chaine d'ARN (un fichier par chaine). Les métadonnées issues des calculs de statistiques sont également proposées (statistiques sur les familles, les fréquences nucléotidiques et fréquences d'appariements).

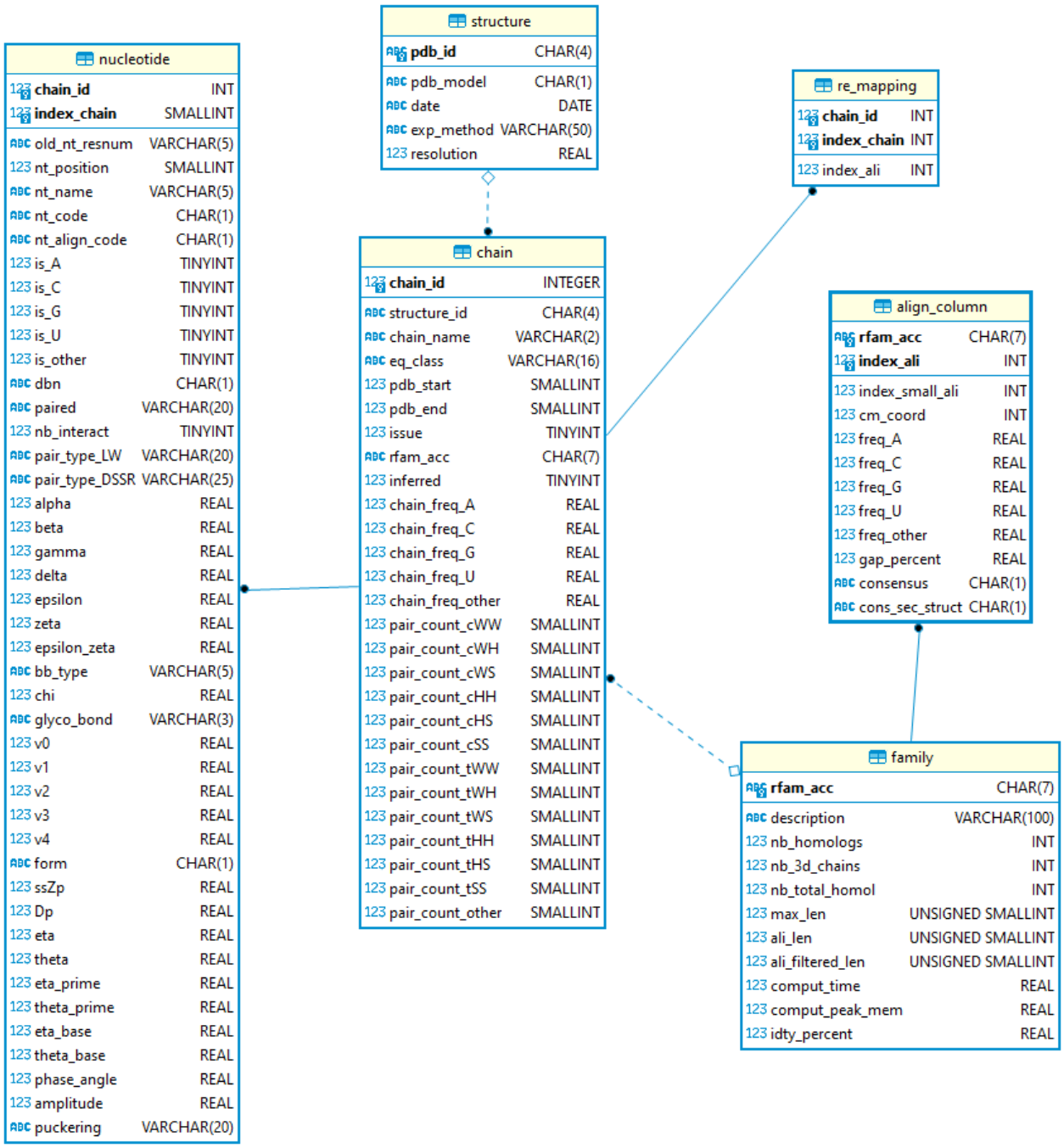


Figure 4.6 – Tables SQL de la base de données RNANet. *La table 'structure' stocke des informations sur les fichiers 3D issus de la PDB (code PDB, date, méthode expérimentale, résolution). À partir de ces fichiers, des portions de chaines sont identifiées comme faisant partie d'une famille, et leurs informations sont stockées dans la table 'chain' (nom, structure d'origine, famille, intervalle mappé à la famille, classe d'équivalence BGSU, fréquences nucléotidiques et fréquences des différents appariements). L'annotation de ces chaines par DSSR produit des descripteurs pour chaque nucléotide, rangés dans la table 'nucleotide'. Le re-mapping entre les nucléotides et les informations sur les colonnes de l'alignement dans la table 'align_column' est donné par la table 're_mapping'. Enfin, la table 'family' donne des informations sur les familles Rfam, comme leur description, le nombre, la taille des ARN de cette famille, et leur pourcentage d'identité. Les fichiers CSV récapitulent les informations de chaque nucléotide (un par ligne), venant des tables 'nucleotide' et 'align_column'.*

IV.2.3.2 Quantité de données

En mai 2021, 12 845 chaines d'ARN issues de 4 595 structures PDB de résolution 20.0 Angströms ou meilleure sont listées par la liste de BGSU. Seules 10 534 chaines d'ARN sont exploitables pour le pipeline, de nombreuses chaines étant éliminées (quand la structure ne contient que les atomes de P, quand elle fait moins de 5 bases, ou quand d'autres erreurs de traitement arrivent). Parmi ces 10 534 chaines, 6 565 seront identifiées comme membres d'au moins une famille Rfam (en considérant toutes les chaines des classes d'équivalence). Au total, 97 familles Rfam sont actuellement représentées. Ces chiffres sont présentés en fonction du seuil de résolution des structures 3D et de la méthode expérimentale sur les Figures 4.7 et 4.8.

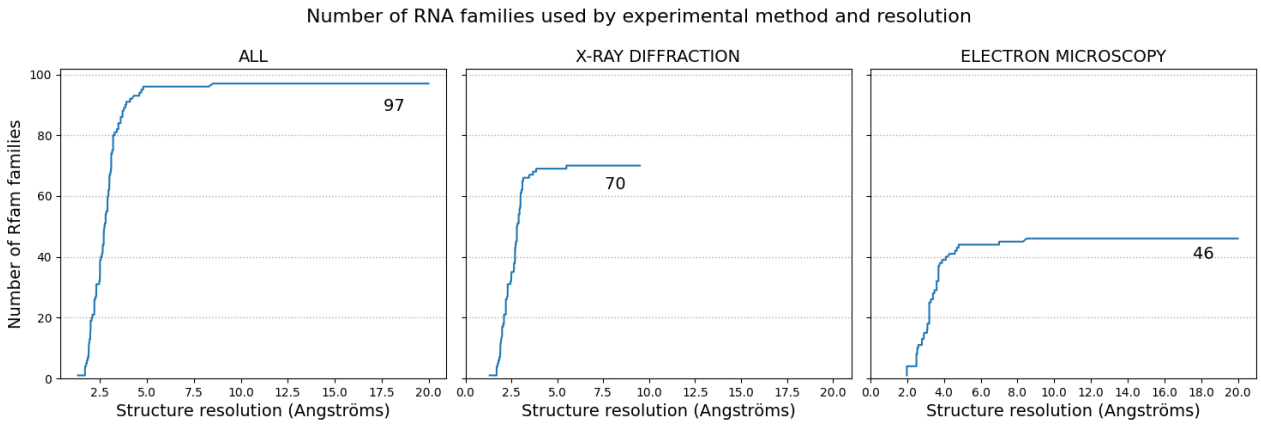


Figure 4.7 – Familles Rfam représentées en fonction de la résolution. *Les trois colonnes comptent les nombres de familles Rfam représentées : dans toutes les structures, dans l0es structures obtenues par XRC, et dans celles obtenues par cryo-EM.*

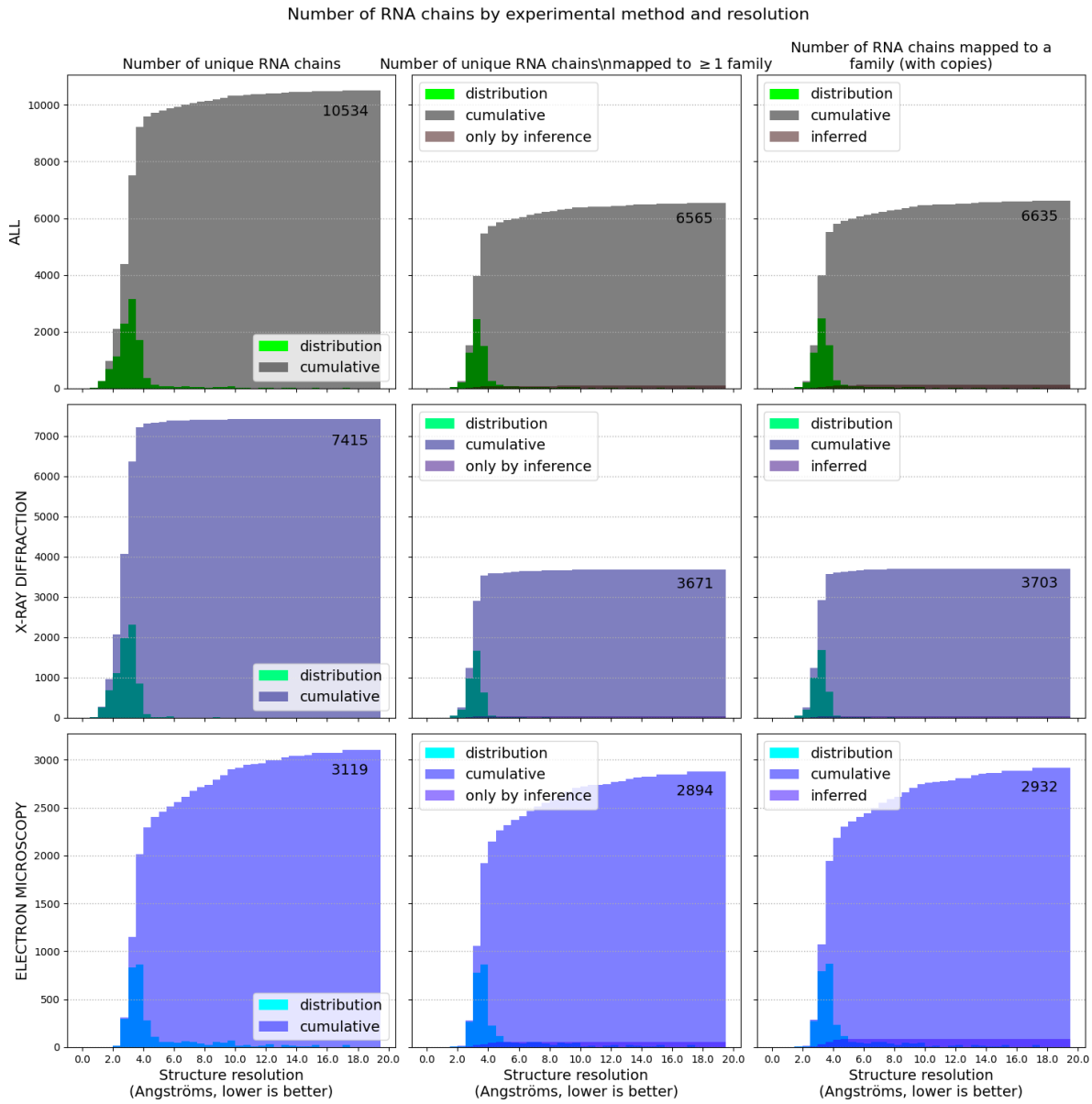


Figure 4.8 – Quantité de données disponibles selon la résolution. *La figure présente trois lignes : la ligne du haut considère toutes les structures, celle du milieu correspondant exclusivement aux structures résolues par XRC, et celle du bas celles résolues par cryo-EM. À gauche : la distribution et la distribution cumulée du nombre de chaines uniques en fonction de la résolution. Au centre : les distributions du nombre de chaines mappées au moins une fois à une famille. Une faible fraction de la distribution cumulée est mise en valeur et correspond aux structures obtenues par inférence en croisant les mappings Rfam-PDB avec les classes d'équivalence de BGSU. À droite : le nombre total de chaines mappées à une famille obtenues si l'on autorise les copies quand une chaine peut être identifiée comme membre de plusieurs familles.*

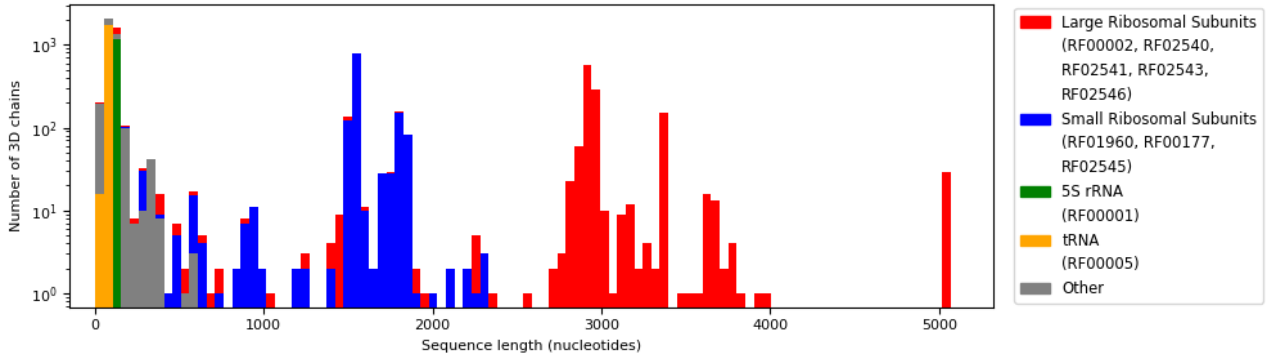


Figure 4.9 – Histogramme des longueurs de séquences. *Le graphique est en échelle logarithmique. Le jeu de données est dominé par les ARN ribosomaux et ARN de transfert. La majorité des ARN non codants en dehors des ARNr sont de taille inférieure à 500 bases. Les petites sous-unités ribosomiques ont des tailles de 500 à 2500 bases, et les grandes sous-unités peuvent atteindre les 5000 bases.*

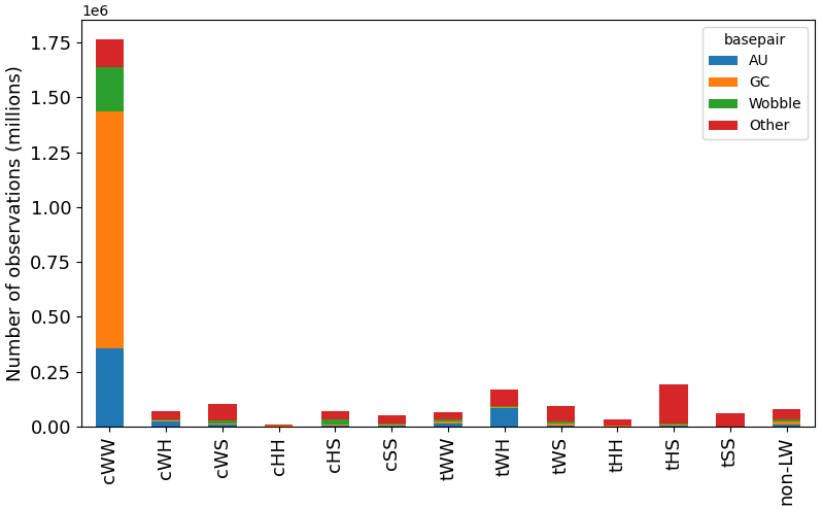


Figure 4.10 – Distribution des types d'appariements non-canoniques. *Les douze plus une colonnes correspondent au douze modes d'interaction entre bases dans la nomenclature de Leontis-Westhof, plus les autres modes. Chaque colonne est subdivisée selon la nature des bases interagissant.*

Les ARNr dominent largement les autres familles d'ARN, que ce soit par la quantité de données disponibles ou par la taille des séquences, illustrées sur la Figure 4.9. D'autres statistiques sur la fréquence des différents appariements sont présentés sur la Figure 4.10. Sans surprise, les appariements canoniques (cWW AU, GC et GU) sont largement majoritaires. La Figure 4.11 présente l'identité des chaines au sein des familles. Les matrices "trop jaunes" sont trop similaires : on suppose que les séquences sont trop proches et seraient vite éliminées par une filtration avec un seuil d'identité de séquence. A l'inverse, les matrices "trop bleues" démontrent des séquences très différentes : soit les alignements sont de mauvaise qualité, soit les séquences ont été classifiées à tort dans la famille. Les matrices vertes sont de la qualité souhaitée : homologues, mais non identiques.

Figure 4.11 – Matrices d'identité de séquences par famille. *Les familles pour lesquelles plus de deux chaines sont disponibles en 3D sont représentées. Plus une valeur dans la matrice est jaune, plus la paire d'ARN considérée est identique. A l'inverse, plus la valeur est bleu foncé, plus les séquences sont éloignées.*

IV.2.3.3 Reproduction des résultats de Wadley & Pyle 2007

Dans Wadley & Pyle 2007 [342], les auteurs essaient de reproduire un diagramme avec les angles de pseudo-torsion η/θ, de façon similaire au célèbre diagramme de Ramachandran des protéines (avec les angles de torsion phi et psi du squelette protéique). Ils commencent par tracer un nuage de points, puis tentent d'en estimer la densité avec une fenêtre de Blackman. En appliquant une coupe à un seuil de densité minimal donné, ils identifient des clusters correspondant à certaines conformations du squelette.
Cependant, leur étude est réalisée sur une petite sélection de structures haute résolution.

On automatise la reproduction de ces résultats en produisant un nuage de points des couples d'angles (η/θ) et (η'/θ'), puis en réalisant une estimation de densité par un kernel Gaussien, méthode un peu plus précise que la fenêtre de Blackman. Les clusters sont les mêmes que ceux précédemment identifiés par Wadley et Pyle, mais leur localisation et leur taille est plus précise. Ils sont représentés sur la Figure 4.12, organisés selon la résolution minimale des nucléotides utilisés. Un nouveau petit cluster est découvert pour les conformations C_2'-endo dans le diagramme (η/θ) [27].

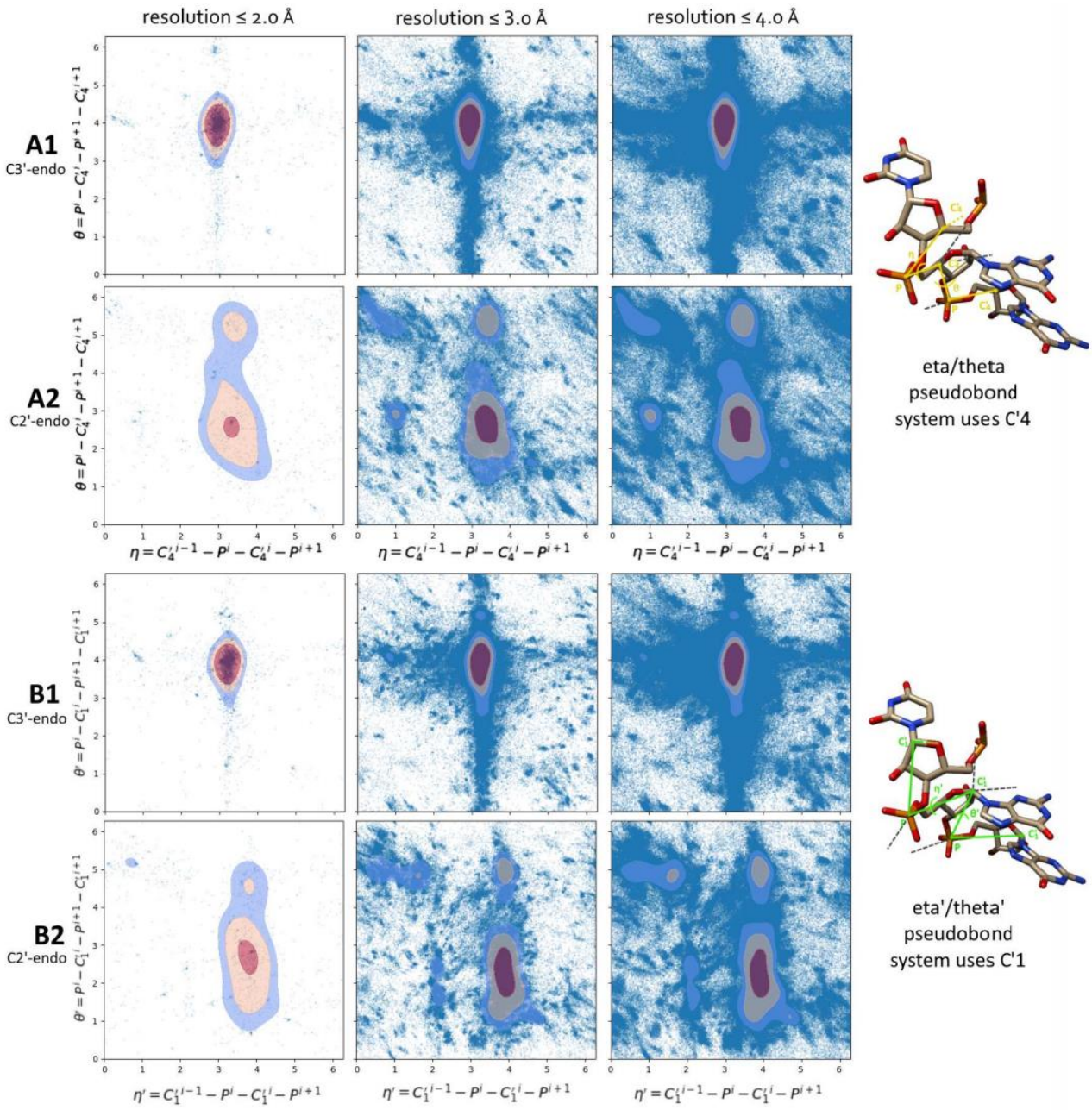


Figure 4.12 – Reproduction des "Pyle plots" à différentes résolutions. *Les 3 colonnes représentent la même analyse, mais en utilisant un jeu de nucléotides de résolution minimale croissante (à gauche le seuil le plus exigeant, à droite le moins exigeant). Les deux lignes du haut correspondent au couple d'angles (η,θ) mesurés sur le pseudo-squelette formé par les atomes P et C₄'. Les deux du bas correspondent aux couples (η',θ'), mesurés sur les atomes P et C₁'. En A1 et B1, le ribose des nucléotides utilisés est en conformation C₃'-endo. En A2 et B2, le ribose est en conformation C₂'-endo. Sur toutes les figures, la première courbe de niveau (bleue) correspond à la moyenne de la densité sur tout le plan plus une fois l'écart-type. La seconde courbe de niveau (orange) correspond à la moyenne plus deux fois l'écart-type, et la troisième (rouge) à la moyenne plus quatre écarts-type.*

IV.2.3.4 Distributions statistiques des paramètres géométriques

On mesure la distribution des différents paramètres géométriques des nucléotides, par exemple, les distances et angles entre atomes. La Figure 4.13 présente les histogrammes des sept angles de torsion. On peut estimer la distribution de chaque histogramme à l'aide d'un ensemble de lois normales dont la somme pondérée forme une distribution de probabilité globale appelée modèle mixte Gaussien (GMM). Le nombre de lois normales ainsi que les paramètres de ces lois normales sont choisis pour maximiser la vraisemblance des données. La difficulté de cette analyse est de prendre en compte la toricité de l'espace des angles lors des calculs de vraisemblance. L'approche utilisée est de copier les observations d'angles de l'intervalle [-π, 0] dans [π, 2π] et les observations de [0, π] dans [-2π, -π]. On calcule ensuite le GMM sur la totalité de l'intervalle [-2π, 2π], contenant toutes les données en double, puis on ne retient que les lois dont le centre se situe dans [-π, π], et on renormalise leurs poids pour que leur somme atteigne 1. Quand les distributions forment des pics, cette méthode fonctionne bien. En revanche quand de nombreuses valeurs d'angles sont observées sur tout l'intervalle, une loi normale de très grande variance et non « contenue » dans l'intervalle [-π, π] peut être proposée par le modèle, ce qui mène à un artefact autour de la valeur 0 qui doit être ignoré (comme pour l'angle de torsion Zeta, illustré sur la Figure 4.13). On remarque que certains sont multimodaux, par exemple les angles de torsion α, γ et δ : il serait faux de les considérer comme constants, il existe au moins deux conformations majoritaires. On réalise ces estimations pour toutes les distances, angles, torsions et pseudo-torsions.

En plus des paramètres géométriques mesurés sur les structures avec tous leurs atomes, on applique la même procédure de calcul à différents modèles gros-grains de l'ARN. On mesure les distances, angles et torsions entre atomes dans le modèle de Wadley & Pyle déjà présenté au paragraphe précédent, pour les deux variantes utilisant les atomes de P et C_4', P et C_1'. On applique aussi la procédure de calcul aux paramètres géométriques du modèle HiRE-RNA [73,253], dont nous aurons l'usage au chapitre V. En plus des distances, angles et torsions, ce modèle demande plusieurs mesures décrivant la géométrie des appariements.

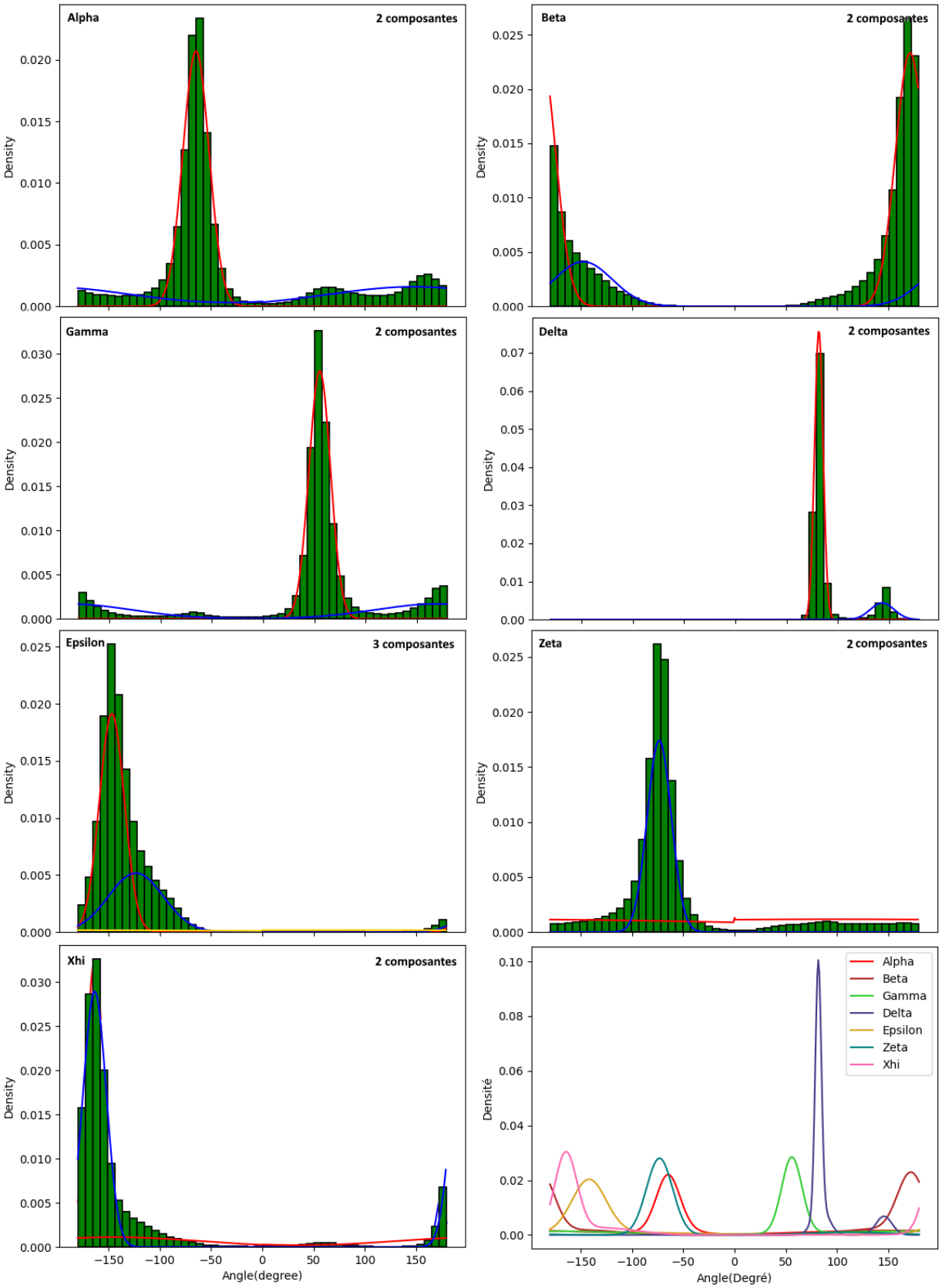


Figure 4.13 – Estimation des densités de probabilité des angles de torsion. *Les histogrammes des 6 angles de torsions du squelette sont représentés, et celui de l'angle Xhi entre la base et le sucre. Sur chacun, on ajuste un modèle mixte Gaussien de 2 ou 3 lois normales, dont les courbes globales sont rassemblées en bas à droite. Pour l'angle Zeta, on note l'artefact dû à une loi de grande variance.*

IV.2.4 Comparaison à ProteinNet et à d'autres jeux de données

Par rapport à ProteinNet, RNANet ne parvient pas à valider les mêmes critères de qualité, non pas à cause de sa propre méthodologie, mais à cause de la pauvreté des données ARN disponibles de toutes façons dans la PDB. La quantité de données disponible est bien plus faible. En plus, l'utilisation des familles Rfam rendent le pipeline plus rapide mais disqualifient environ 37% des chaines 3D. À l'époque de la publication originale de RNANet, nous n'avions pas connaissance de RNAlien [110] (et RNACentral [281] n'avait pas encore de moteur de recherche d'homologues), nous n'avions donc pas le choix.
Ensuite, la forte redondance des données disponibles empêche le pré-calcul raisonnable de jeux de données d'entrainement et de test sans réduire dramatiquement la quantité de chaines disponibles. Le choix du seuil est donc pour l'instant laissé à l'utilisateur.

Néanmoins, RNANet reste à notre connaissance le meilleur jeu de données d'ARN conçu pour la science des données et l'apprentissage automatique. Il est nettoyé et standardisé. Il contient des informations à toutes les échelles de modélisation, sur la séquence, la structure secondaire canonique ou étendue, et la structure 3D avec des descripteurs gros-grains (les contacts) et des descripteurs géométriques précis (angles, conformations, et toutes les annotations de DSSR). Par-dessus tout, le jeu de données embarque les liens connus avec des familles d'ARN et les séquences homologues associées, ce qui nous permet de distribuer des alignements multiples de séquences et des fréquences nucléotidiques pour chaque position.

Par comparaison, tous les outils d'apprentissage profond sortis entre temps (une petite vingtaine en trois ans) utilisent des jeux de données sous-optimaux.

Deux premiers jeux de données, baptisés "Set A" et "Set B" dans NeuraFold et MXFold, sont issus de l'article de Rivas *et al.* [280]. Set A est une compilation de structures secondaires utilisées dans d'autres articles, listant 3 863 structures secondaires. L'origine de ces structures secondaires n'est pas forcément l'annotation de structures 3D connues, elles peuvent avoir été inférées par une méthode comparative, ou estimées par sondage chimique. Set B est constitué de 1 524 séquences de moins de 70% d'identité et de moins de 250 bases, issues de 22 familles Rfam 10.0 pour lesquelles on connait au moins une structure 3D. Ces deux jeux datent de 2011.

Lu *et al.* [206] entrainent leur modèle sur RNAStrand [14], une base de données de structures secondaires hétérogène, dont la source de la structure secondaire est rarement l'annotation d'une structure 3D résolue, et non mise à jour depuis 2008.

SPOT-RNA [312] et SPOT-RNA 2 [313] font du *transfer-learning* en entrainant le réseau d'abord sur la large base de structures secondaires bpRNA-1m [78], plus récente que RNAStrand, mais toujours composée de structures secondaires d'origine et de sources diverses ne provenant majoritairement pas de l'annotation 3D. L'entrainement est poursuivi sur un jeu de 217 (236 pour le SPOT-RNA 2) structures 3D haute résolution, dont la structure est extraite par DSSR. Dans SPOT-RNA 2 cependant, des données d'homologie sont aussi utilisées. Des CM sont construits à partir des 236 séquences à l'aide de BLASTN et Infernal, puis des alignements de séquences sont construits à partir de ces CM et de la base de

séquences NCBI. Des fréquences et des caractéristiques de DCA sont ensuite calculées sur les alignements.

DMFold [351], E2EFold [56] et CDPFold [379] utilisent le jeu de données de 3 948 structures secondaires du Mathews lab, appelé "archive II" dans plusieurs publications. Ce jeu de données contient des ARNr 5S, 16S, 23S, des introns des groupes I et II, des RNAses P, des ARN de transfert, des SRP, des tmRNA et des "Telomerase RNA". En particulier, CDPFold est entrainé une première fois sur 1059 séquences d'ARN 5S issues de ce jeu de données, puis une seconde fois sur toutes les familles mais en retirant les séquences contenant des pseudonoeuds.

Le dataset RNAStrAlign introduit avec TurboFold II [333] en 2017 par le Mathews lab (et également utilisé par E2Efold) est une extension de l'archive II, contenant maintenant 20923 séquences et structures secondaires (soit le double de bpRNA-1m), toujours des mêmes familles précédemment listées. Malheureusement, la constitution de cette base de données devient difficile à retracer, et si les structures secondaires sont prétendument "connues", il est impossible de retrouver pour une structure donnée comment la structure a été déterminée. RNAStrAlign est un mélange de jeux de données (comme bpRNA) qui étaient déjà des mélanges d'autres archives (comme RNAStrand ou Rfam), étant parfois eux-mêmes des compilations de sources de structures (comme CRW [47] ou SRPDB [284]). Les structures passent donc d'un jeu de données à l'autre depuis deux décennies, la traçabilité est difficile, et la documentation sur les méthodes d'obtention de chaque structure secondaire inexistante.

Willmot *et al.* [358] entrainent leur réseau sur les 17 032 ARNr 16S du Comparative RNA Web site (CRW), qui contient de nombreuses séquences et des structures secondaires inférées par analyse des covariations. Il s'agit donc de prédictions, et pas de données réelles.

UFold [126]combine RNAStrAlign, Archive II, bpRNA-1m et bpRNA-1m-new, basé sur la plus récente version de Rfam, v14.1. Ces jeux sont redondants entre eux, c'est donc un mélange peu rigoureux.

MXFold II [296] est entrainé plusieurs fois avec chacun des jeux précédents : les Sets A et B de Rivas, Archive II, RNAStrAlign, et le jeu de SPOT-RNA (basé sur bpRNA-1m). Le jeu "T-Full" utilisé par Andronescu en 2010 [15] est également utilisé, pour estimer des paramètres énergétiques de façon statistique.

DIRECT [164] est entrainé sur une sélection de riboswitches en provenance de Rfam (pour les séquences et structures secondaires), et sur 147 puis 1 023 structures 3D représentantes de classe de taille inférieure à 120 bases, issues de la liste BGSU (pour les structures 3D).

RNAContact [325] est entrainé aussi sur 1 786 chaines représentantes de classe de la liste BGSU (ou plus précisément sur un sous ensemble de 366 chaines de taille comprise entre 30 et 1000 bases de moins de 30% d'identité de séquence entre elles). Les auteurs disent avoir utilisé Infernal pour "chercher des séquences homologues dans la base de séquences NCBI" sans plus de détails, ce qui à notre connaissance n'est pas possible à ce jour. Le code source de l'outil suggère plutôt que leur approche utilise BLASTN sur NCBI pour rechercher des séquences homologues.

CoCoNet [375] est entrainé sur un jeu publié auparavant, dans Pucci *et al.* [265] (en 2020). Les auteurs sélectionnent les structures de la PDB contenant de l'ARN libre et

monomérique, sans protéines ou ADN, de plus de 40 bases de long, et résolu par XRC à 3.64 Angströms ou moins. Pour tout groupe de séquences plus de 50% d'identité, la structure de meilleure résolution est conservée. Ils utilisent ensuite Infernal (`cmscan`) pour identifier une famille Rfam pour chacune, réalignent les séquences entre elles, et calculent les caractéristiques de DCA par pyDCA. Les méthodes sont donc très proches de celles de RNAnet, mais le jeu de données final contient seulement 69 structures, et n'est bien entendu pas construit de façon automatique.

Les jeux de données sérieusement concurrents de RNANet sont donc celui de Pucci et al, celui de RNAContact, et celui utilisé par SPOT-RNA 2, qui utilisent des structures 3D avec données d'homologie, mais sont tous d'assez petite taille. Dans le cas de RNAContact et SPOT-RNA 2, on s'approche probablement de la quantité maximum qu'on peut atteindre en appliquant les seuils d'identité de séquence choisis, les jeux sont donc qualitativement et quantitativement proche de RNANet (et sortis en même temps que RNANet). Néanmoins, l'approche BLASTN + *NCBI nucleotide archive* qu'ils utilisent nous parait moins pertinente qu'utiliser des programmes spécialisés pour l'ARN cherchant dans des bases de données d'ARN non codants. NCBI nucleotide archive est une base de génomes (d'ADN donc), et BLAST un algorithme de séquences qui ne modélise pas les covariations ni la structure secondaire (ce que font les outils basés sur les HMM et CM). C'est pour ces raisons que nous n'avions jamais considéré cette approche.

Avec 6 565 structures 3D (non filtrées) alignées à des familles Rfam, RNANet est donc le plus gros jeu de données 3D avec données d'homologie construit à ce jour. Après filtration, il deviendrait de taille similaire à certains jeux utilisés pour entrainer certains outils sortis en même temps. La structure secondaire des chaines 3D de RNANet peut être directement déduite de la structure 3D de façon reproductible et facile à sourcer (en citant le logiciel d'annotation, dans notre cas, DSSR). Le nombre de séquences alignées, dont la structure secondaire consensus peut être inférée par Infernal, se compte en millions. Enfin, RNANet est un code de qualité professionnelle, dont le développement et la maintenance est maintenant réalisée en équipe suivant des pratiques « DevOps ». A l'inverse, les scripts comme RNAContact produits pour prototyper rapidement une idée de recherche, ne sont pas directement utilisables par autrui. RNANet propose des résultats structurés et organisés en CSV ou SQL, et mis à jour mensuellement.

IV.2.5 Perspectives

De nombreuses idées d'ajouts et d'améliorations restent à explorer. D'une part, certains changements ou ajouts peuvent permettre d'améliorer la qualité ou la quantité de données disponibles. D'autre part, RNANet peut être utilisé dans de nouvelles tâches de minage de données ou d'apprentissage automatique. RNANet est actuellement toujours en cours de développement actif, et certaines de ces idées sont en cours d'exploration dans l'équipe.

IV.2.5.1 Idées d'amélioration de la qualité et quantité de données

Tout d'abord, la liste de classes d'équivalence du BGSU RNA Group utilisée comme liste de références des structures contenant de l'ARN n'est pas exhaustive. Son utilisation présente de nombreux avantages, mais on passe probablement à coté de structures intéressantes, notamment celles produites par RMN. Passer à un scan et extraction directe de toutes les structures contenant des ARN de la PDB n'est cependant pas simple : de nombreuses structures contiennent des ARN synthétiques (oligo-A, oligo-G, ou autres), des ARN codants, des hybrides ADN/ARN... Faut-il les considérer ? Si l'on considère les structures obtenues par RMN, qui proposent généralement une ou plusieurs dizaines de modèles de la structure par fichier 3D, comment gérer cette redondance ?

Même en se contentant des structures collectées par la liste de BGSU, nous pourrions concentrer les efforts sur l'identification d'homologues. En effet, la Figure 4.8 montre que près de 37% des chaines 3D disponibles sont ignorées car on n'a pas identifié de famille Rfam leur correspondant. On pourrait, premièrement, rechercher nous-mêmes les familles Rfam avec l'outil `cmscan` d'Infernal, plutôt que de se contenter des mappings proposés par Rfam dont on ne contrôle pas l'ancienneté (l'information de la date à laquelle ils ont été produits n'est pas disponible). Mieux on pourrait également générer nous-mêmes des alignements de séquences homologues indépendants des familles Rfam, en utilisant RNAlien ou RNACentral. L'obtention d'un alignement de séquences unique pour chaque chaine 3D permettrait de progresser vers une construction rationnelle de jeux d'entrainement et de test comme le faisait ProteinNet (Figure 4.5).

De nouveaux descripteurs pourraient être ajoutés à la base de données. On pense par exemple à lister la présence de modules connus dans chaque chaine. Un autre axe de développement serait l'intégration du contexte des chaines : en commençant par les contacts connus entre différentes chaines de RNANet, on pourrait lister les complexes d'ARN connus. Puis, on pourrait étendre le contexte d'interaction 3D aux autres macromolécules (ADN, protéines) et aux ligands et ions.

Enfin, une interface de visualisation et recherche de structures est en projet pour le site web EvryRNA. Une visualisation annotée des structures secondaires et tertiaires pourrait être proposée.

IV.2.5.2 Idées de valorisation

Le jeu de données a été imaginé pour permettre l'entrainement de réseaux de neurones et l'application de l'apprentissage profond supervisé à la prédiction de structures d'ARN. Cet objectif a été atteint. Cependant, la quantité de données étant encore faible, la performance de ces méthodes n'est pas garantie avant encore quelques années. En attendant, il est possible d'utiliser RNANet pour d'autres usages statistiques et de minage de données. La reproduction des "Pyle plots" (Figure 4.12) et l'estimation des distributions des paramètres géométriques (Figure 4.13) en est un exemple.

Un premier point d'intérêt serait l'entrainement et la mise à jour automatisée des potentiels statistiques à partir des mesures statistiques des paramètres géométriques. Un article paru en 2019 (Tan *et al.* [332]) compare les différentes façons d'établir une distribution de référence à laquelle se comparer pour construire un potentiel statistique. Il liste aussi certains outils utilisant déjà l'une de ces méthodes pour l'ARN [33,166,350]. Les paramètres des potentiels de ces outils n'ont jamais été mis à jour depuis leur sortie, ce qui pourrait être réalisé de façon automatique avec RNANet. On pourrait également intégrer la procédure proposée par IsRNA [377,378] pour éliminer la corrélation entre paramètres dans les distributions. Un stage est en cours dans l'équipe sur le sujet, et cette direction de recherche va être explorée.

Une seconde application serait la mise à jour automatique des jeux de données de modules. Le RNA 3D Motif Atlas est le seul actuellement mis à jour en continu. RNANet pourrait permettre de régénérer tous les mois les réseaux d'interactions récurrents de CaRNAval [273,318] ou VeRNAl [249], par exemple.

IV.3 Prédire la forme du squelette de la molécule avec un réseau récurrent : l'architecture RGN

Toute cette partie s'inspire du travail d'Al-Quraishi dans 3 articles [10–12] tous parus en 2019, où il applique les réseaux récurrents à la prédiction de structures protéiques en 3D. Al Quraishi a développé ProteinNet [12] comme jeu de données pour l'apprentissage profond. Il a également proposé une variante parallèle (et différentiable) de l'algorithme NERF [252] de Rosetta (appelée pNERF [11]), dans le but de transformer les coordonnées internes d'une structure (longueurs, angles, torsions) en coordonnées cartésiennes (X, Y, Z). Cette version parallèle est intégrable en tant que couche au sein d'un réseau de neurones, dont l'entrainement est réalisé sur GPU. Ces deux travaux sont appliqués ensemble à la prédiction de forme des squelettes protéiques, dans l'architecture RGN[10], qu'il propose très peu de temps après. La performance est similaire aux autres méthodes favorites de l'état de l'art, mais dans des temps de calcul quasi-instantanés, ce qui est un atout exceptionnel.
Ces travaux forment un tout très cohérent, les articles sont complets et pédagogiques, et leur publication dès 2019 (soit après le succès d'AlphaFold, mais avant l'article expliquant leurs méthodes), m'a fortement incité à tenter d'adapter la méthode à l'ARN et à m'initier à l'apprentissage profond. C'est cet objectif qui a motivé le développement de RNANet, déjà introduit au titre précédent.

Ce projet a été commencé en 2019, des difficultés ont été identifiées et sont présentées tout au long de cette partie. Du fait de la sortie de modèles concurrents plus simples et également performants ou meilleurs, notamment basés sur les CNN, et du fait de l'éloignement au sujet de la thèse, le RGN a été abandonné en Aout 2020 et laissé en l'état. Même si des pistes existent pour résoudre les problèmes identifiés, on les mentionnera sans donner de résultats, elles n'ont pas été testées en pratique.

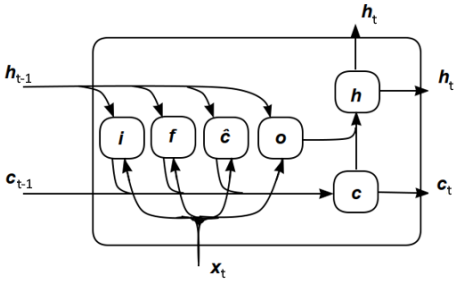
IV.3.1 Principe du réseau géométrique récurrent

Le réseau géométrique récurrent est un réseau « LSTM bi-directionnel » sur lequel une couche « géométrique » est rajoutée, et utilisant comme fonction de coût (*loss*) une mesure de comparaison de structures 3D appelée « dRMSD » pour *distance-based Root of Mean Squared Deviation*. Le réseau prend en entrée L caractéristiques par nucléotide : un encodage 1-hot de la séquence (0 ou 1 pour chaque valeur possible de résidu), les fréquences de chaque résidu à cette position observées dans des séquences homologues, et la position du résidu dans la séquence (position relative, entre 0 et 1). La ou les couches de LSTM bidirectionnels réalisent, pour chaque résidu, une prédiction des angles de torsion qui lui sont associés. La couche « géométrique » reconstruit une chaine en coordonnées cartésiennes à partir de la prédiction d'angles de torsion. Puis la loss compare la prédiction de chaine à la chaine réelle. Notons que ce modèle prédit la forme du squelette, et ne donne aucune information sur l'orientation des résidus.

IV.3.1.1 Les couches LSTM bidirectionnelles

Un réseau LSTM est un type de réseau récurrent (RNN) très répandu composé de nombreuses « cellules LSTM », elles-mêmes composées de plusieurs neurones chacune. Il y a deux façons de modéliser l'architecture : on peut considérer que des unités LSTM sont juxtaposées un grand nombre de fois, ou alors, considérer qu'il y a une seule unité LSTM dont la sortie est raccordée à l'entrée pour un grand nombre de cycles. Toutefois, la sortie de chaque étape peut être conservée par le réseau et pas seulement la sortie du dernier cycle. Un schéma de la cellule LSTM unique est proposé sur la Figure 4.14.

$$
\begin{aligned}
i_t &= \sigma(W_i \times x_t + U_i \times h_{t-1} + b_i) \\
f_t &= \sigma(W_f \times x_t + U_f \times h_{t-1} + b_f) \\
o_t &= \sigma(W_o \times x_t + U_o \times h_{t-1} + b_o) \\
\tilde{c}_t &= \tanh(W_{\tilde{c}} \times x_t + U_{\tilde{c}} \times h_{t-1} + b_{\tilde{c}}) \\
c_t &= i_t \cdot \tilde{c}_t + f_t \cdot c_{t-1} \\
h_t &= o_t \cdot \tanh(c_t)
\end{aligned}
$$



$$
\begin{aligned}
&x_t \in \{0,1\}^4 \times [0,1]^4 \times \mathbb{N}^+ \\
&i_t, f_t, \tilde{c}_t, o_t, c_t, h_t \in [0,1]^k \\
&W \in M_{k,9}, \qquad U \in M_{k,k} \\
&\qquad\qquad\qquad\quad b \in [0,1]^k
\end{aligned}
$$

Figure 4.14 – La cellule Long Short-Term Memory. *Une cellule LSTM est composée de 6 neurones, chacun possédant une fonction d'activation. À chaque cycle t, elle rend une valeur h_t et conserve un état interne c_t, calculés à partir d'entrées : la nouvelle donnée x_t, l'état interne précédent c_{t-1}, et le résultat du cycle précédent h_{t-1}. Dans notre cas, la donnée x_t correspond aux 9 caractéristiques du $t^{ième}$ nucléotide de la séquence. Grâce à son état interne, le réseau LSTM est capable de se « souvenir » à court terme du contexte précédant l'arrivée de la donnée x_t. Deux matrices de poids W et U ainsi qu'un vecteur de biais b sont entrainables et optimisés lors de l'apprentissage. Sur la figure, x_t est un vecteur de L=9 caractéristiques et h_t un vecteur de k valeurs, mais ces dimensions peuvent être modifiées sans changer le principe du LSTM.*

À chaque cycle, on donne à la cellule LSTM les données d'un nouveau nucléotide de la séquence. Pour une séquence de N nucléotides, on réalise donc N cycles. Le réseau LSTM simple est capable de propager l'information dans une seule direction (par exemple du début de la séquence vers la fin). Pour réaliser une prédiction d'angles en prenant en compte le contexte de séquence dans les deux directions autour d'un nucléotide, on utilise un LSTM « bidirectionnel », c'est-à-dire deux réseaux LSTM propageant l'information en sens contraire (l'un prendra la séquence du début vers la fin et l'autre de la fin vers le début). On obtient donc pour chaque position deux prédictions d'angles de torsion h_t^f et h_t^b (pour *forward* et *backward*).

Dans le modèle d'AlQuraishi, h_t est un vecteur « softmax » de taille k, c'est-à-dire un vecteur de probabilités d'appartenance des angles à prédire à k multiplets de valeurs possibles identifiés. On parle d'alphabet structural de k « lettres ». Le vecteur h_t, dont la somme des composantes vaut 1, indique quels sont les plus probables. Les multiplets en question, ou lettres de l'alphabet structural, sont des points de l'espace des angles choisis régulièrement espacés (pour les protéines, des zones du diagramme de Ramachandran).

Il est possible d'empiler plusieurs couches de LSTM en donnant les sorties h_t d'une couche à la suivante. Ceci permet, en théorie, de modéliser la molécule à des échelles plus globales et moins locales.

IV.3.1.2 L'angularisation

La dernière couche LSTM rend donc, pour chaque position dans la séquence, deux vecteurs softmax h_t^f et h_t^b donnant des probabilités d'appartenance des angles de torsions à des clusters bien définis. On empile donc par-dessus une couche « d'angularisation » pour déterminer une valeur numérique des angles. Le neurone d'angularisation réalise une moyenne pondérée par des poids entrainables des deux vecteurs h_t^f et h_t^b, on obtient donc un seul vecteur h_t. Puis il calcule un emplacement dans l'espace des angles de torsion avec une moyenne géométrique des k clusters pondérée par le vecteur softmax h_t obtenu. Les coordonnées de cet emplacement dans l'espace des angles donnent des valeurs numériques pour chaque angle de torsion. On passe alors à la couche suivante. Mathématiquement, le neurone d'angularisation effectue le calcul suivant :

$$h_t = \sigma\left(W \times h_t^f + W' \times h_t^b + b\right)$$
$$\omega_t = \arg\left(\sum_{j=1}^{k} h_{tj} \cos\omega_j + i \sum_{j=1}^{k} h_{tj} \sin\omega_j\right)$$

où W et W' sont deux matrices de poids entrainables, et b le vecteur de biais. Ensuite, chaque angle de torsion ω_t est calculé comme une moyenne pondérée de ses valeurs ω_j dans les k clusters connus pondérés par les poids h_{tj}. Pour rappel, la fonction $\arg(x)$ avec x un nombre complexe renvoie l'angle θ tel que $\cos\theta + i\sin\theta = x$. Donc $\arg(\cos\theta + i\sin\theta) = \theta$. Il s'agit donc bien ici d'une moyenne pondérée. On peut aussi l'exprimer en utilisant la fonction `atan2`, plus souvent utilisée en informatique :

$$\omega_t = \mathrm{atan2}\left(\sum_{j=1}^{k} h_{tj} \cos\omega_j\,, \sum_{j=1}^{k} h_{tj} \sin\omega_j\right)$$

Les deux formulations sont mathématiquement égales et équivalentes.

IV.3.1.3 La reconstruction de la chaine

Enfin, la couche « géométrique » applique l'algorithme pNERF aux $2N$ valeurs des angles de torsion prédites (2 angles par résidu). A partir d'un point de référence dans l'espace 3D, cet algorithme étend le squelette de la molécule atome par atome en calculant la nouvelle position de chaque nouvel atome pour qu'elle respecte les contraintes de longueur de liaison, d'angles, et de torsions souhaitées. La variante pNERF réalise ceci en parallèle en reconstruisant des fragments indépendamment, puis en assemblant ces fragments à la fin. On obtient à la fin un « serpent » en 3D, c'est-à-dire un enchainement linéaire d'atomes dans l'espace qui donne la forme prédite du squelette de la molécule.

La fonction de coût utilisée pour comparer la prédiction au squelette réel est appelée dRMSD, et s'affranchit de plusieurs défaut du RMSD classique. Avec un RMSD classique, on mesure les distances entre les positions de chaque atome prédit et la réalité. Pour pouvoir faire cette mesure, il faut pouvoir aligner, superposer la prédiction avec la vraie structure. L'alignement est un problème d'optimisation difficile à résoudre de façon optimale et coûteux en temps : il est impossible d'inclure ce calcul à l'intérieur d'un réseau de neurones entièrement différentiable. Le dRMSD compare donc les distances entre deux atomes de la même structure, dans la structure réelle et dans la prédiction. Cette métrique est donc invariante par translation, rotation, et symétrie, elle ne nécessite donc pas d'alignement. Elle rend également bien compte de la qualité locale des prédictions de structure : si un seul angle change de valeur complètement dans la prédiction, le RMSD sera grandement impacté, mais pas le dRMSD. Mathématiquement, on exprime le dRMSD comme :

$$dRMSD = \sqrt{\frac{2}{n(n-1)} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \left(d_{ij} - \hat{d}_{ij}\right)^2}$$

soit la racine de la moyenne des différences au carré entre les distances entre toute paire d'atomes parmi les n atomes de la chaine, mesurées une fois dans la prédiction et une fois dans le squelette réel. On note que la racine carrée n'apporte rien et peut éventuellement être omise pour faciliter la différentiation. La grande force du RGN vient du fait que cette fonction dRMSD est donc dérivable par rapport à tous les poids des couches précédentes, on peut donc entrainer le réseau de neurones directement.

IV.3.1.4 Adaptation du modèle à l'ARN

Pour les protéines, AlQuraishi utilisait 45 caractéristiques en entrée (22 pour l'encodage des 22 possibilités d'acides aminés, 22 autres pour leurs fréquences, et une pour la position relative). Il utilisait également 3 atomes par résidu (C_O, C_α et N) et les trois angles de torsions correspondants, Φ, ψ et ω (bien que ω soit quasiment constant). Pour N résidus, on avait donc n = 3N atomes dans le squelette à prédire. Les vecteurs h_t étaient de dimension 60, correspondant à des triplets de valeurs régulièrement espacées dans l'espace $[-\pi, \pi]^3$.

Pour l'adapter à l'ARN, j'ai donc proposé la formulation suivante : considérer désormais 11 caractéristiques en entrée par nucléotide (5 pour l'encodage des nucléotides possibles, A, C, G, U et 'autre', 5 autres pour leurs fréquences respectives, et une dernière pour la position relative). On utilisera cette fois le modèle de squelette utilisé par les outils de Pyle [173] et par VFold [51], à savoir considérer seulement les atomes de P et de C_1', et les angles de pseudo-torsion associés η' et θ'. On pourrait aussi utiliser les autres modèles illustrés sur la Figure 4.15B.

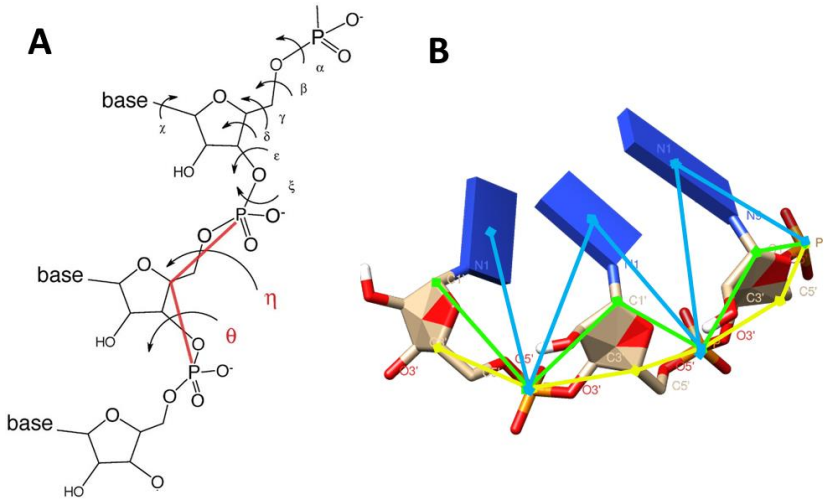


Figure 4.15 – Rappels des angles de torsion et pseudo-torsion. ***(A)*** *Rappel de la structure moléculaire du squelette. Il y a normalement 6 angles de torsion, représentés sur le nucléotide du haut. Mais en pratique, des systèmes à deux pseudo-torsions η/θ sont une bonne approximation.* ***(B)*** *Plusieurs modèles, selon qu'on utilise les atomes de P et C_4' (en jaune) et les angles associés η et θ, ou les atomes de P et C_1' (en vert) et les angles associés η' et θ', ou encore les atomes de P et les centres de gravité de la base (en bleu), et les angles associés η'' et θ''. Ces trois modèles pourraient être utilisés avec le RGN, dans la suite, on parlera simplement de η et θ.*

On a donc deux atomes par nucléotide, soit $n = 2N$. Pour les vecteurs h_t, on teste plusieurs idées différentes :

- Option 1 : Réaliser non pas un échantillonnage régulier de l'espace [-π, π]^2, mais plutôt définir l'alphabet avec des clusters identifiés sur les diagrammes de Pyle (voir Figure 4.12 dans les résultats de RNANet). Le fait que les valeurs fréquentes des couples d'angles (η', θ') soient rassemblées en clusters laisse penser que la prédiction d'un cluster est possible. Différents algorithmes de clustering sont testés (K-means, DBSCAN) pour essayer d'identifier des alphabets de k =10 à 20 clusters. Le vecteur h_t a donc la dimension du nombre de clusters identifiés. Pour rappel, le neurone d'angularisation calcule donc les valeurs prédites de η' et θ' par la formule :

$$\varphi_t = \begin{pmatrix}\eta_t \\ \theta_t\end{pmatrix} = \begin{pmatrix}\arg\left(\sum_{j=1}^{k} h_{tj}\cos\eta_j + i\sum_{j=1}^{k} h_{tj}\sin\eta_j\right) \\ \arg\left(\sum_{j=1}^{k} h_{tj}\cos\theta_j + i\sum_{j=1}^{k} h_{tj}\sin\theta_j\right)\end{pmatrix}.$$

- Option 2 : On essaie également de faire prédire directement aux couches LSTM les valeurs numériques des sinus et cosinus des angles, qui seront utilisées directement par le neurone d'angularisation. C'est alors un problème de régression (et non plus de classification multiple). Le vecteur h_t est alors de taille 4 (deux valeurs cos et sin pour chaque angle).

$$\varphi_t = \begin{pmatrix}\eta_t \\ \theta_t\end{pmatrix} = \begin{pmatrix}\arg(\cos\eta_t + i\sin\eta_t) \\ \arg(\cos\theta_t + i\sin\theta_t)\end{pmatrix}$$

- Option 3 : Enfin, on essaie aussi de faire prédire la valeur de chaque angle directement entre -π et π. On n'a alors plus besoin du neurone d'angularisation, on moyenne simplement les vecteurs h_t^f et h_t^b ensemble pour obtenir un vecteur h_t de taille 2.

- Enfin, on adapte l'algorithme pNERF pour qu'il fonctionne avec deux atomes par position, pour des valeurs de longueurs de liaisons et d'angles plans supposées constantes et estimées par une moyenne calculée sur les données de RNANet. L'architecture finale est illustrée sur la Figure 4.16.

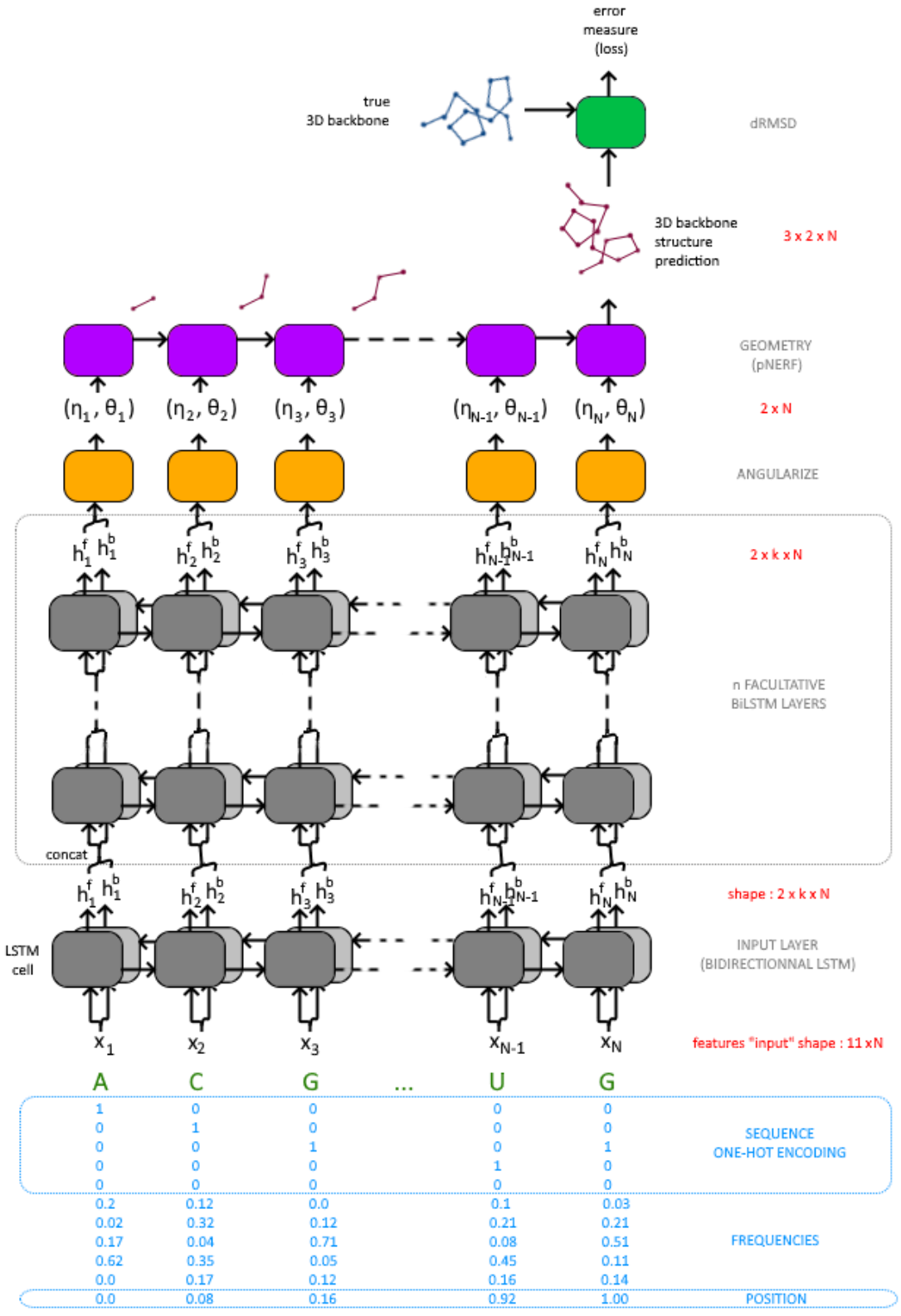


Figure 4.16 – Architecture RGN complète. *La longueur de la séquence donne le nombre de répétitions du réseau récurrent. Sinon, l'architecture RGN consiste en une superposition de couches de LSTM bidirectionnels, puis de la couche d'angularisation et de la couche « géométrique » pNERF, et enfin une comparaison à la structure réelle par dRMSD.*

IV.3.2 Gestion de la diversité de longueur

Un problème majeur avec le jeu de données d'ARN est la grande hétérogénéité de longueur des séquences. Une méthode courante en apprentissage profond, utilisée avec les images notamment, est de compléter les données trop courtes avec des données non-informatives (bordures blanches ou noires) jusqu'à ce qu'elles fassent la bonne taille. On appelle cela le *padding*. Dans notre cas, le réseau de neurones a besoin de connaitre à l'avance le nombre de cycles qu'il va devoir effectuer pour entrainer un batch. Or, ce nombre est égal aux nombre de nucléotide de chaque séquence en entrée, mais les séquences sont de taille différente.
Ici, utiliser du padding serait inapproprié : certains ARN comme les ARNt sont de longueur 72 environ, quand les ARNr peuvent atteindre plusieurs milliers de bases. Si on étendait les données des ARNt jusqu'à atteindre 5 000 bases avec du remplissage non informatif, le réseau serait entrainé majoritairement sur ce remplissage, et n'apprendrait jamais rien. En effet, tout ajout de padding détériore un peu le signal, il ne doit donc pas devenir majoritaire dans les données.

La solution imaginée à l'époque fut de trier les séquences par longueur, puis de les grouper par batches de taille similaire par intervalles d'une centaine de nucléotides. À l'intérieur d'un batch, le padding peut être appliqué pour compléter les séquences trop courtes. On pouvait ainsi informer le réseau du nombre de cycles à effectuer avant un batch, avant de l'entrainer sur le batch en question, toutes les séquences faisant la même taille à l'intérieur sans que cela ne nuise trop au rapport signal/bruit.

Malheureusement, la première implémentation complète du réseau comprenant cette idée, utilisant un alphabet de valeurs possibles pour les angles de torsion (algorithme original d'AlQuraishi), ne convergeait pas. La fonction de coût ne diminuait pas (du tout) d'une itération à l'autre au fur et à mesure de l'apprentissage, et les poids ne semblaient pas changer.
Pour tenter d'étudier le problème, on a décidé de prédire dans un premier temps les angles seuls au moyen des trois variantes proposées plus haut (classification multiple, régression des cosinus et sinus, et régression directe). On se passe alors de la couche pNERF et on n'obtient pas de reconstruction 3D du squelette, juste pour étudier dans un premier temps si la prédiction des pseudo-torsions se fait correctement.

IV.3.3 Test découplé de la prédiction des angles

Dans ce paragraphe, on ignore la reconstruction 3D et on essaie de prédire correctement les angles de torsion.

IV.3.3.1 Classification multiple

On doit commencer par construire un alphabet structural. Pour ce faire, il faut répondre à trois sous-questions : quelles données choisir pour sélectionner les conformations, quelle va être la taille de l'alphabet structural, et comment vont être déterminées les positions des clusters ? Pour les données, on utilise les chaines de RNANet de résolution inférieure ou égale à 4 Angströms. On obtient 4,78 millions de nucléotides, et on liste les valeurs observées des angles η' et θ'.
Ensuite, il s'agit d'un problème de clustering. De nombreux algorithmes de clustering existent, nous devons donc en choisir un qui soit adapté aux paramètres du problème. Nous avons des millions de points, répartis dans des dizaines de zones de densité différentes de plusieurs ordres de grandeur. Certains algorithmes de clustering nécessitent le choix en amont du nombre de clusters, d'autres non. K-Means et le clustering spectral sont éliminés d'office puisqu'ils sont prévus pour des clusters de taille similaires. On considère donc les modèles mixtes gaussiens (rapides à calculer) et le clustering DBSCAN (Density-Based Scan).

Les modèles mixtes Gaussiens (GMM) ont de bonnes propriétés pour résoudre ce problème, même si ce ne sont pas (en théorie) des algorithmes de clustering. Pour rappel, un GMM est une collection de k lois Normales (courbes de Gauss multi-dimensionnelles) dont la somme pondérée définit une distribution de probabilité maximisant la vraisemblance des données observées. On peut alors assimiler chaque loi à un cluster dont l'espérance est le centroïde. Pour chaque point, on peut déterminer à laquelle des k Gaussiennes il est plus probable qu'il appartienne, ce qui définit son appartenance à un cluster. Les clusters étant définis par des courbes de Gauss multi-dimensionnelles, ils ont une forme ovoïde souhaitée mais peuvent avoir des tailles et excentricités très différentes, ce qui modélise bien nos données. L'estimation du maximum de vraisemblance est par ailleurs rapide à calculer, même avec des millions de points. Le désavantage de la méthode est d'avoir à choisir le nombre de clusters à l'avance. Des résultats sont présentés sur la Figure 4.17 pour 10, 20, 50 et 100 lois Normales.

Le DBSCAN [299] est un algorithme cherchant à identifier des zones de forte densité séparées par des zones de faible densité. Il classe les points de données en deux types : les « échantillons centraux » et les autres. Un échantillon central est un point de données ayant au moins M voisins dans un rayon d autour de lui. À partir des échantillons centraux initialement identifiés, on étend la liste des échantillons centraux en regardant lesquels de leurs voisins sont également centraux. L'ensemble des échantillons centraux contigus entre eux forment un cluster. Il n'y a donc pas besoin de connaitre à l'avance le nombre de clusters, mais leur densité minimale souhaitée via les paramètres M et d. De plus, la métrique de distance entre points utilisée est arbitraire, on définit alors une mesure de distance non pas

dans un plan carré mais à la surface du 2-tore $[0, 2\pi]^2$. L'algorithme OPTICS [17] est de la même famille, mais propose en plus un moyen de détecter seul la distance d optimale proposant les meilleures « cassures » entre clusters. L'algorithme a également l'avantage d'une plus faible complexité mémoire, ce qui d'un point de vue pratique est utile pour ce gros jeu de données. Le principal désavantage de DBSCAN est qu'il ne permet en théorie pas d'extraire des « centroïdes » de clusters, ou des paramètres qui résumeraient sa position dans l'espace des angles. Ceci est dû au fait que les clusters se propagent par contact et ont donc des formes variées et non convexes. Cependant dans notre cas, on cherche des clusters ovoïdes : on se permettra donc quand même de calculer (manuellement) le centroïde des échantillons centraux. On cite aussi HDBSCAN [230], une implémentation parallèle similaire à OPTICS supposément prévue pour les larges jeux de données.

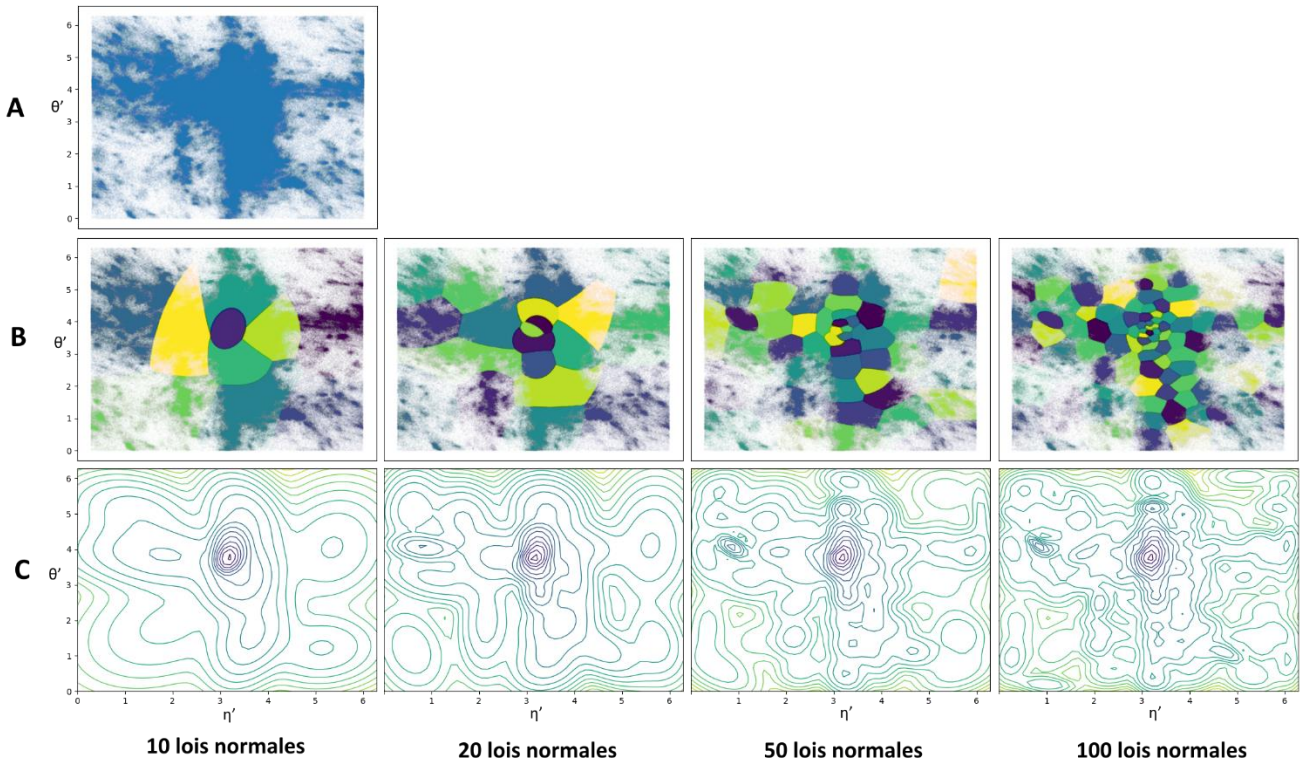


Figure 4.17 – Clustering des pseudo-torsions par modèle mixte Gaussien. ***(A)*** *Un nuage de points des torsions η′/θ′ de résolution 4.0 Angströms ou meilleure, similaire à ceux de la Figure 39. Note : La transparence des points est fortement augmentée ; il faut 100 points superposés pour obtenir un point bleu parfaitement opaque. Ceci aide un peu à mieux distinguer les zones de haute densité. La zone centrale (correspondant aux hélices) est toutefois encore plus dense de plusieurs ordres de grandeur.* ***(B)*** *Le même nuage de points, mais les points sont colorés selon quelle composante normale du modèle mixte a la plus grande probabilité de les avoir produits. Les zones délimitées correspondent donc à des pics de densité, mais on n'a aucune information à ce niveau sur la hauteur de ces pics. Cette représentation permet visuellement d'estimer la granularité du clustering, une valeur intermédiaire entre 50 et 100 lois pourrait être suffisante.* ***(C)*** *Courbes de niveau du modèle mixte global, estimant la densité de probabilité des valeurs. Elles donnent une idée de la densité du nuage de points, et bien entendu, plus le nombre de composantes normales utilisées augmente, plus cette estimation est précise.*

On essaie donc d'appliquer DBSCAN, HDBSCAN et OPTICS à ces données d'angles, en choisissant comme paramètres M = 500, d = 0.05, la métrique de distance torique, et un nombre minimum de points par cluster de 5 000. Malheureusement, aucun n'a pu terminer le calcul de la matrice de distances sur laquelle ils reposent dans des temps raisonnables. Réduire l'exigence de résolution (par exemple 2.0 Angströms ou meilleure) donne seulement 18456 nucléotides et ne permet pas de représenter la plupart des zones de haute densité qu'on voit à 4 Angströms.

On se contentera donc d'un alphabet structural obtenu par modèle mixte Gaussien avec 60 lois normales, soit le même chiffre qu'AlQuraishi. Il aurait été possible d'utiliser un clustering basé sur la densité aux prix d'efforts d'ingénierie logicielle pour accélérer le calcul de cette matrice de distances et réduire son impact mémoire, mais ils n'ont pas été jugés nécessaires. Il est évident que le problème de classification associé, devant déterminer un cluster parmi soixante, sera difficile. On définit donc comme fonction de coût une fonction d'entropie de Shannon ou entropie catégorielle croisée (en anglais *cross-entropy*). Une autre métrique de performance est définie, la *categorical accuracy*, c'est-à-dire le pourcentage de nucléotides correctement attribués au bon cluster. Des résultats d'entrainement sont présentés sur la Figure 4.18. On a réalisé plusieurs expériences, les trois premières en faisant varier la dimension de l'état interne des cellules LSTM (64, 128 et 256), et une dernière en utilisant trois couches de LSTM bidirectionnels au lieu de deux (avec 64 états internes). La performance et la vitesse de convergence augmentent toutes deux avec la complexité du modèle (la dimension de l'état interne). En comparaison, même si l'ajout d'une couche supplémentaire de LSTM augmente également la performance, la différence reste minime et le temps de calcul augmente sensiblement.

Malheureusement, même le meilleur modèle (courbes orange et jaune de la Figure 4.18) n'arrive pas à prédire les clusters correctement. L'exactitude (*accuracy*) plafonne à 60% de prédictions correctes. Le problème est d'autant plus difficile que les classes sont très déséquilibrées, certains clusters pouvant contenir vingt fois plus de nucléotides que d'autres. Il aurait fallu imaginer par exemple une fonction de coût contre-pondérée par les tailles des clusters, pour récompenser la prédiction de façon correcte de l'appartenance à un petit cluster. Cependant, la formulation mathématique d'un entropie de Shannon pondérée n'est pas triviale pour le cas multi-classe (*weighted softmax categorical cross-entropy*), ou alors il faudrait plutôt définir un poids pour chaque erreur de classification entre toute paire de classes. On s'est plutôt concentrés sur le passage à un problème de régression.

IV.3.3.2 Régression

Pour la régression des valeurs du cosinus et sinus de chaque angle, on fait prédire directement une valeur à la dernière couche de LSTM sans utiliser d'angularisation. Un neurone récupère les sorties des deux LSTM *forward* et *backward* et utilise une fonction d'activation « tangente hyperbolique » pour rendre des valeurs dans [-1,1]. On utilise une fonction de coût de type MSE (*Mean of Squared Error*) sur les valeurs des sinus et cosinus, en ignorant les valeurs non définies (`NaN`). Ces valeurs se présentent naturellement pour

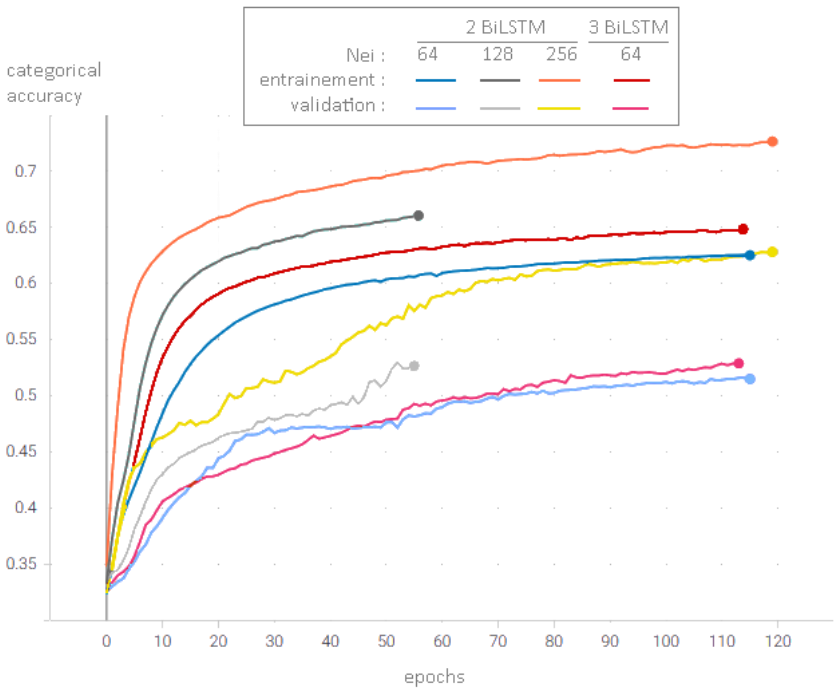


Figure 4.18 – Courbes d'apprentissage en mode classification multiple. *Chaque paire de courbes (entrainement + validation) correspond à une expérience. En abscisse le nombre d'epochs cumulés, en ordonnée la « categorical accuracy ». Dans chaque expérience, la performance augmente au cours du temps d'entrainement, ce qui confirme que l'apprentissage se passe bien.*

les nucléotides de début ou de fin de chaine, ou lorsque certains ne sont pas résolus en 3D. Ces nucléotides comptent pour une erreur de zéro. Malheureusement, cette formulation donne des gradients nuls, et le réseau n'apprend pas. La fonction de coût reste constante au cours des epochs, et on peut observer que les poids du réseau ne sont pas modifiés.

Pour la régression directe des valeurs, d'angles, on utilise comme en classification multiple un vecteur softmax en sortie des LSTM et une couche d'angularisation. On définit une fonction de coût de type MSE (moyenne des carrés des erreurs de prédiction), mais définie sur le 2-tore (une prédiction proche de 0 est une erreur faible si l'angle est proche de 2π). Pour tenir compte du déséquilibre de taille des clusters, l'erreur est pondérée pour pénaliser fortement la mauvaise prédiction des petits clusters et faiblement la mauvaise prédiction des gros. Malheureusement encore, avec ou sans pondération, les gradients sont nuls et le réseau n'apprend pas.

Donc, dans les différents essais de régression, le modèle échoue car les gradients sont nuls. Le problème n'a pas été étudié plus en détail pour l'instant. Les causes potentielles sont, premièrement, le « masquage » des valeurs non-numériques (`NaN`) et leur remplacement par la valeur zéro. Si trop de `NaN` sont produits pour des raisons indésirables (erreurs d'implémentation, divisions par zéro, etc…), les valeurs sont remplacées par zéro et les gradients sont nuls. Deuxièmement, l'utilisation des fonctions trigonométriques et de la fonction `atan2` (dont le gradient n'est pas défini en $(x, y) = (0,0)$) dans les formules de la fonction de coût rend peut-être le problème d'optimisation impossible à résoudre.

IV.3.4 Test découplé de la reconstruction des coordonnées cartésiennes

Malgré la performance médiocre des couches précédentes pour la prédiction des angles de torsion, on a voulu étudier la couche géométrique pour savoir si elle fonctionnait : en effet, si elle fonctionne, alors le problème est théoriquement soluble de cette façon (en comptant sur une bonne prédiction des angles). Si pNERF ne fonctionne pas, alors ce n'est pas la peine de continuer à s'intéresser à ce modèle. On prépare donc une expérience de reconstruction de structure 3D du squelette en partant des vraies pseudo-torsions η'/θ', comme si elles avaient été prédites parfaitement par les couches précédentes. On commence par exporter systématiquement les fragments reconstruits pour les visualiser, le résultat est présenté sur la Figure 4.19. Tout a l'air normal, on ne détecte pas d'anomalie à ce niveau, les structures sont plausibles.

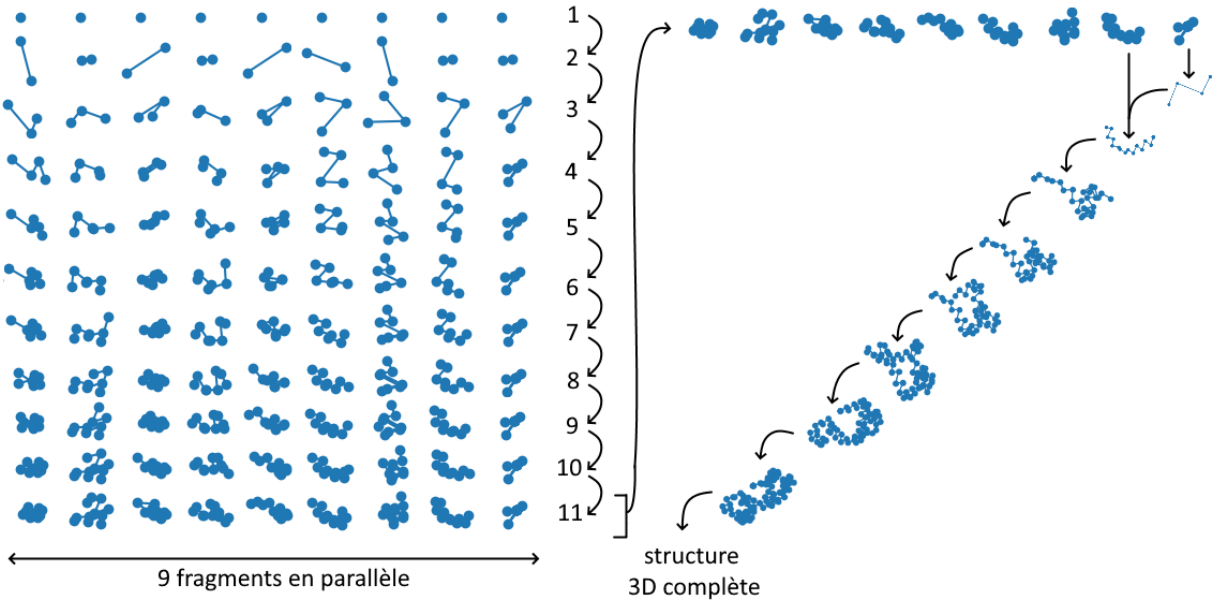


Figure 4.19 - Illustration graphique de l'algorithme pNERF. *Le processus de reconstruction est rendu parallèle sur GPU en découpant la chaine à reconstruire en fragments qui seront reconstruits en parallèle, puis une portion synchrone rassemble les fragments entre eux. Aucun neurone n'est entrainable dans la couche géométrique, le nombre de fragments influe donc sur la performance de calcul, mais pas sur la justesse du résultat. Un compromis est à trouver pour le nombre de fragments, en fonction de la machine GPU utilisée et de la longueur de l'ARN à reconstruire.*

On décide donc de comparer la structure du squelette reconstruite au squelette réel, au moyen de métriques précises comme le RMSD et le dRMSD. On applique la reconstruction par pNERF aux 6632 structures de RNANet. Pour chacune, on réaligne la prédiction de la forme du squelette à la forme réelle en calculant la matrice de rotation et le vecteur de translation qui minimisent le RMSD par décomposition en valeurs singulières (SVD). C'est un algorithme exact. On calcule ensuite le RMSD obtenu, et le dRMSD (qui ne nécessitait pas d'alignement). On en profite pour étudier s'il y a un effet de la longueur de la chaine d'ARN sur ces métriques. La Figure 4.20 montre les résultats obtenus en fonction de la longueur de la chaine. On a testé deux façons de calculer le RMSD et dRMSD : prendre la racine de la moyenne, ou prendre la moyenne des racines (des écarts au carré, dans les deux cas). Les formules sont rappelées ci-dessous :

$$dRMSD = \sqrt{\sum_{i,j} \left(d_{ij} - \hat{d}_{ij}\right)^2} \qquad dMD = \sum_{i,j} \sqrt{\left(d_{ij} - \hat{d}_{ij}\right)^2}$$

$$RMSD = \sqrt{\sum_{i} {d_i}^2} \qquad MD = \sum_{i} \sqrt{{d_i}^2}$$

Pour dRMSD et dMD, d_{ij} représente la distance entre deux atomes i et j à l'intérieur de la structure réelle, et $\hat{d}_{ij}$ la même distance dans la structure prédite. Pour RMSD et MD, d_i désigne la distance entre la position réelle d'un atome et sa position prédite lorsque les deux structures sont superposées. On constate plusieurs résultats :

- quelque soit la métrique utilisée, l'erreur entre la structure réelle et sa reconstruction par pNERF est non nulle, ou même très grande (plusieurs dizaines de nanomètres, c'est-à-dire plusieurs centaines d'Angströms),
- quelque soit la métrique utilisée, cette erreur est corrélée à la taille de la chaine d'atomes, linéairement ou sous-linéairement (logarithmiquement),
- pour les deux métriques, prendre la moyenne des racines donne des erreurs légèrement plus faibles que prendre la racine de la moyenne,
- l'erreur calculée par le RMSD est inférieure à celle calculée par le dRMSD si on les calcule par les racines des moyennes, et cette différence s'accroit avec la taille de la molécule d'ARN, mais cet effet n'est pas visible si on les calcule avec la moyenne des racines (Figure 4.20C).

On conclura donc de cette expérience qu'il vaut mieux calculer les RMSD et dRMSD en prenant la moyenne des racines (contrairement à ce qui est souvent utilisé dans la littérature, dont le RGN d'Al Quraishi), du fait de la non-corrélation de cette métrique par rapport à la longueur de la chaine. On conclura aussi que la reconstruction d'une chaine à partir des angles de torsion n'est pas possible (du moins pas sans une déviation conséquente et absolument pas négligeable). On examine une chaine d'ARN et sa reconstruction plus en détail pour comprendre les différences entre la prédiction et la structure réelle. L'exemple d'un ARNt est donné sur la Figure 4.21. On constate que la divergence entre réalité et prédiction est très rapide. En particulier, les angles plans diffèrent et ne sont pas constants : cette approximation n'est pas valable.

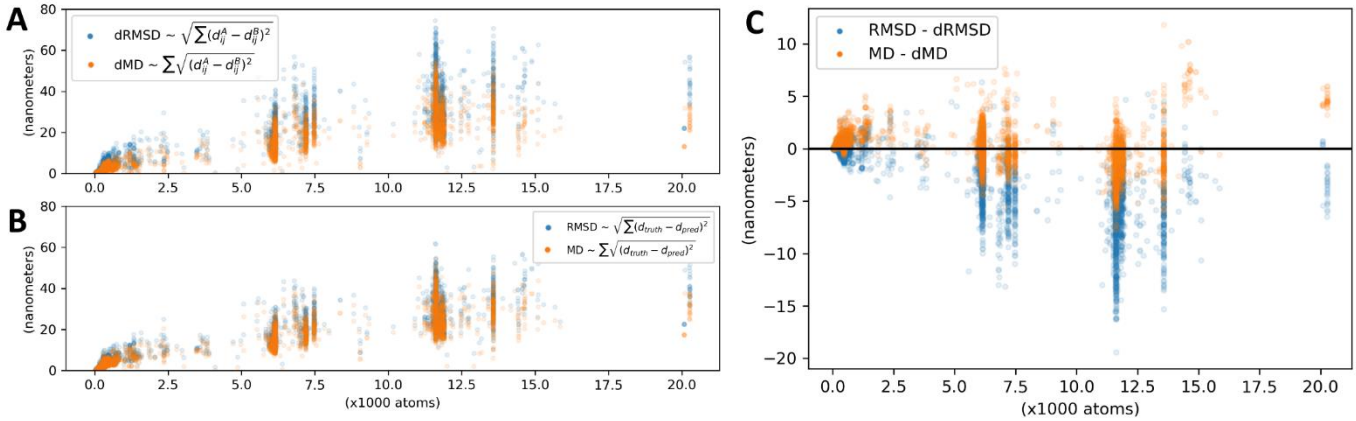


Figure 4.20 – Qualité des structures après reconstruction par pNERF. ***(A)*** *Le dRMSD obtenu (en nanomètres) en fonction de la longueur de chaine (en milliers d'atomes). Il est calculé de deux façons : en bleu, on calcule la racine de la moyenne, en orange, la moyenne des racines, ce qui aboutit à des valeurs légèrement plus faibles.* ***(B)*** *Le RMSD après superposition par SVD, par les deux mêmes méthodes de calcul.* ***(C)*** *Les différences entre RMSD et dRMSD, par les deux méthodes de calcul.*

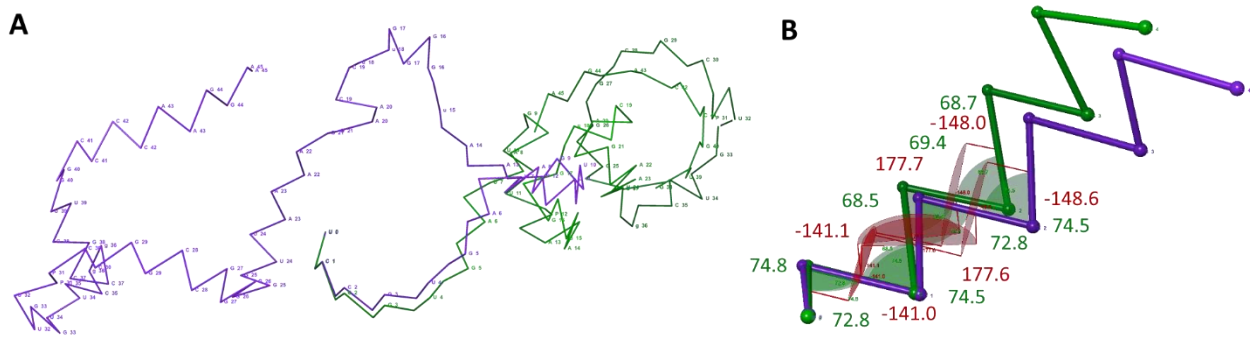


Figure 4.21 – Accumulation d'erreurs et divergence. ***(A)*** *Le squelette de la structure 1ASZ en vert, et la prédiction pNERF à partir des véritables angles de torsion en violet, disposées de façon à superposer leurs trois premiers atomes. Si la reconstruction était parfaite, les deux chaines seraient superposées. Or, on voit apparaitre la divergence très rapidement.* ***(B)*** *Détail des valeurs d'angles plans (en vert) et de torsion (en rouge). Les erreurs sur les angles de torsion sont faibles, toutefois, les erreurs sur les angles plans peuvent être élevées. En effet, pNERF suppose que les angles sont constants (72.8° pour P-C_1'-P et 74.5° pour C_1'-P- C_1').*

On devra donc renoncer à la prédiction de la structure du squelette par la simple prédiction des angles de torsion. Un modèle des angles plans est nécessaire. De plus, de petites erreurs de quelques dixièmes de degrés suffisent à ce que la reconstruction diverge de la structure initiale, il faudrait donc pouvoir prédire les torsions avec une très grande précision. Or pour l'instant, les modèles de régression proposés n'apprennent pas. Devant ces difficultés, et à la suite de la parution de modèles plus récents et performants en 2019 et 2020, le projet est abandonné en l'état, et on se recentrera sur des travaux plus pertinents.

IV.3.5 Autres modèles de prédiction d'angles

Les années 2019 et 2020 ont vu sortir plusieurs outils de prédiction d'angles pour les protéines basés sur les réseaux de convolution. Les deux plus connus font partie des outils meneurs dans les expériences CASP, à savoir RaptorX de l'équipe de Jinbo Xu (pionnière des méthodes d'apprentissage profond depuis CASP9) et AlphaFold de DeepMind (qui a poussé la performance des modèles jusqu'à celle des méthodes expérimentales lors des CASP 13 et 14).

RaptorX est un serveur de prédiction entrainé à prédire de nombreuses propriétés des protéines sur la base d'une valeur par résidu (contacts, angles, accessibilité au solvant, etc). On présentera ici la composante RaptorX-Angle [129], qui prédit les angles de torsion comme le RGN, mais avec une architecture différente. Le modèle s'appuie sur 30 clusters dans le diagramme de Ramachandran (clusters obtenus par K-Means) et résout le problème de classification multiple d'appartenance à un cluster par un réseau convolutif profond (CNN) résiduel (ResNet). Les caractéristiques d'entrée sont l'encodage one-hot de la séquence, des PSSM (ou fréquences nucléotidiques) obtenues par deux outils de recherche d'homologues et d'alignement (PSI-BLAST, HHpred), et des caractéristiques prédites par d'autres outils : l'accessibilité au solvant et la structure secondaire. L'angularisation se fait en pondérant les coordonnées des clusters par les probabilités d'appartenance au cluster. C'est donc un modèle très similaire au RGN, mais utilisant une architecture différente. La précision finale du modèle est de l'ordre de la dizaine de degrés (mais varie selon les acides aminés et les structures secondaires considérées).

Le modèle original AlphaFold 2018 utilise plusieurs réseaux interdépendants [303]. Il prédit notamment des matrices de distances entre paires de résidus, s'appuyant à la fois sur des matrices de distance mesurées et sur des caractéristiques dérivées d'alignements multiples, permettant d'enrichir largement les données 3D avec des données de séquence bien plus abondantes. Mais la partie qui génère réellement une structure 3D à partir de ces matrices de distance prédites réalise une descente de gradient dans un potentiel spécifique aux protéines, qui dépend notamment de leurs angles de torsion [302]. Les variables de décision du problème d'optimisation sont les angles de torsion. Un réseau génératif utilise des « LSTM convolutifs » pour proposer des conformations selon la distribution apprise des angles de torsion, et l'article de Senior 2019 [303] cite celui du RGN. Néanmoins, la prédiction n'est pas réalisée sur la seule base des angles de torsion.

Après la publication d'AlphaFold, on s'intéressera alors à l'adaptation de la méthode à l'ARN, toujours avec les données de RNANet.

IV.4 Apprendre la forme conservée d'ARN homologues avec un réseau de convolution

La performance d'AlphaFold à CASP13 repose sur plusieurs points, que nous allons essayer de considérer dans notre modèle pour l'ARN. On devra :

- réaliser la prédiction de matrices de distances au lieu de matrices de contacts (la principale innovation d'AlphaFold),
- utiliser des caractéristiques calculées sur des alignements multiples de séquences homologues à la séquence cible, pour capturer l'information contenue par la covariation des résidus entre séquences. Comme il y a plusieurs millions de séquences disponibles, contre seulement quelques centaines de structures, ce point est indispensable pour obtenir suffisamment de signal,
- optimiser une structure dans un potentiel statistique dépendant des prédictions de distances et d'angles (ou échantillonner des solutions dans le paysage énergétique de ce potentiel pour renvoyer plusieurs solutions),
- apprendre directement à partir de l'alignement multiple de séquences plutôt que calculer des caractéristiques intermédiaires à donner en entrée. Cette technique, qui laisse le réseau construire lui-même l'information dont il a besoin, a été proposée dans rawMSA [241] et utilisée dans le tout récent AlphaFold 2 [374], [375], [376].

IV.4.1 Calcul des matrices de distance et tests de normalité

On commence par calculer, pour chaque chaine d'ARN de RNANet, la matrice des distances euclidiennes entre paires de résidus résolus. On essaie deux points de référence comme approximation de la position d'un nucléotide : le carbone C'_1, qui est plutôt proche du squelette, et le centre de gravité de tous les atomes de la base, qui reflète plutôt bien son orientation dans l'espace.

On veut ensuite calculer les distributions de ces distances au sein d'ARN homologues. En première approximation, on les suppose normales, on les résumera donc à une espérance et une variance (estimées par la moyenne et l'écart-type). Dans RNANet, les ARN homologues sont ceux appartenant à la même famille Rfam. À partir des alignements de RNANet pour chaque famille, on récupère le re-mapping de chaque nucléotide par rapport au modèle de covariation utilisé avec `cmalign`. Pour chaque paire de positions du modèle de covariation, on calcule la moyenne et l'écart-type des distances entre ces positions. La matrice moyenne des distances par famille (et celle des écarts-type) sont ainsi obtenues, et elles ont la dimension du CM de la famille Rfam. Si un nucléotide n'est pas résolu en 3D ou manquant dans la séquence (gaps dans l'alignement), le calcul des distances le concernant est ignoré, et ce manque est comptabilisé dans le calcul de l'espérance et de la variance. Pour éviter d'avoir à stocker en mémoire de larges matrices de distances, la variance est calculée itérativement selon la formule de Huygens : $V(x) = E(x^2) - E(x)^2$. Ainsi, on somme itérativement les distances et leurs carrés en parcourant une seule fois chaque matrice de chaque ARN, puis on divise par le nombre de distances sommées pour obtenir des

estimations des espérances, et enfin on réalise la soustraction. Un exemple de résultat est présenté pour les ARNr 5S sur la Figure 4.22. C'est l'une des familles pour laquelle on possède le plus de structures 3D, et il n'y a pas de portions manquantes dans la matrice. Toutefois, pour de nombreuses familles moins bien décrites en 3D, certaines portions des matrices ne sont pas définies (`NaN`).

On veut aussi étudier l'hypothèse de normalité des distributions de ces distances. Ce travail a été réalisé par des étudiants du master 2 GENIOMHE de l'université, lors d'un projet transversal en bio-informatique que je leur ai confié dans le cadre de l'une de mes missions d'enseignement. Je remercie donc Nicolas Viart et Hosna Baniadam pour leur travail sur ce sujet et leurs conclusions que j'ai jugées suffisamment pertinentes pour les rapporter ici. Ils ont proposé d'appliquer un test statistique de Shapiro-Wilk (avec un risque de première espèce de 5%) à toutes les distances pour lesquelles on a assez de d'observations, soit 14006 distances dans les matrices des familles RF00001 (ARNr 5S) et RF00005 (ARNt). Les tests concluent à une loi normale dans seulement 171 cas, soit environ 1.2% des cas. Ils ont ensuite entrepris de comparer la distribution réelle à la distribution théorique (normale) à l'aide de graphiques quartile-quartile (Q-Q plots) ou certaines variantes (P-P plot, ou en comparant les fonctions de répartition). Un exemple est présenté sur la Figure 4.22.

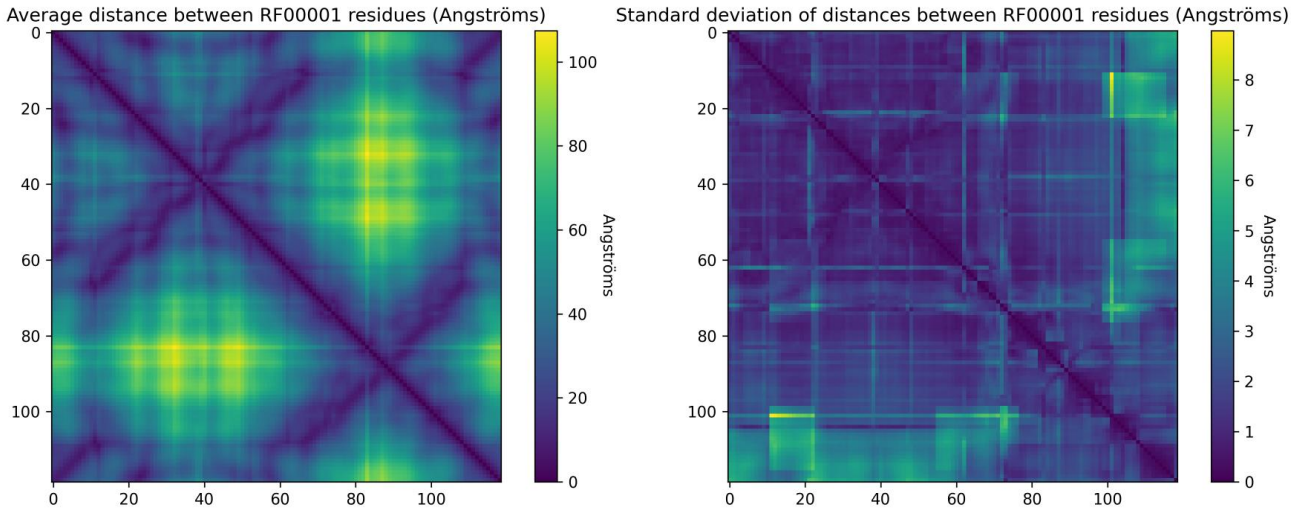


Figure 4.22 – Matrice moyenne et matrice d'écarts-type des distances entre résidus. *La famille utilisée pour construire cette figure est RF00001, les ARNr 5S, avec 1172 structures 3D connues. Les insertions présentes dans certaines chaines sont ignorées.*

On se rend compte que la majorité des distances ne suit pas une loi normale. Un nombre important suit d'ailleurs des distributions multimodales, comme l'exemple sur la Figure 4.23A. Même parmi les distributions monomodales, une majorité ne suit pas la loi normale. On ne parvient pas à identifier une loi statistique connue (beta, log-normale...) pour la plupart des distances.

On remarque également que les distributions multimodales sont plus fréquentes si l'on mesure la distance interbase à partir des centres de gravité des nucléotides, sensibles à l'orientation de la base, que si on les mesure à partir des carbones C'_1 dont la position est

contrainte par le squelette. Ceci laisse penser que la variabilité de l'orientation des bases est capturée par ce genre de mesures. Dans la suite, on travaillera avec les distances calculées à partir des carbones 1', ce qui rend l'hypothèse de normalité un peu moins mauvaise (mais tout de même fausse).

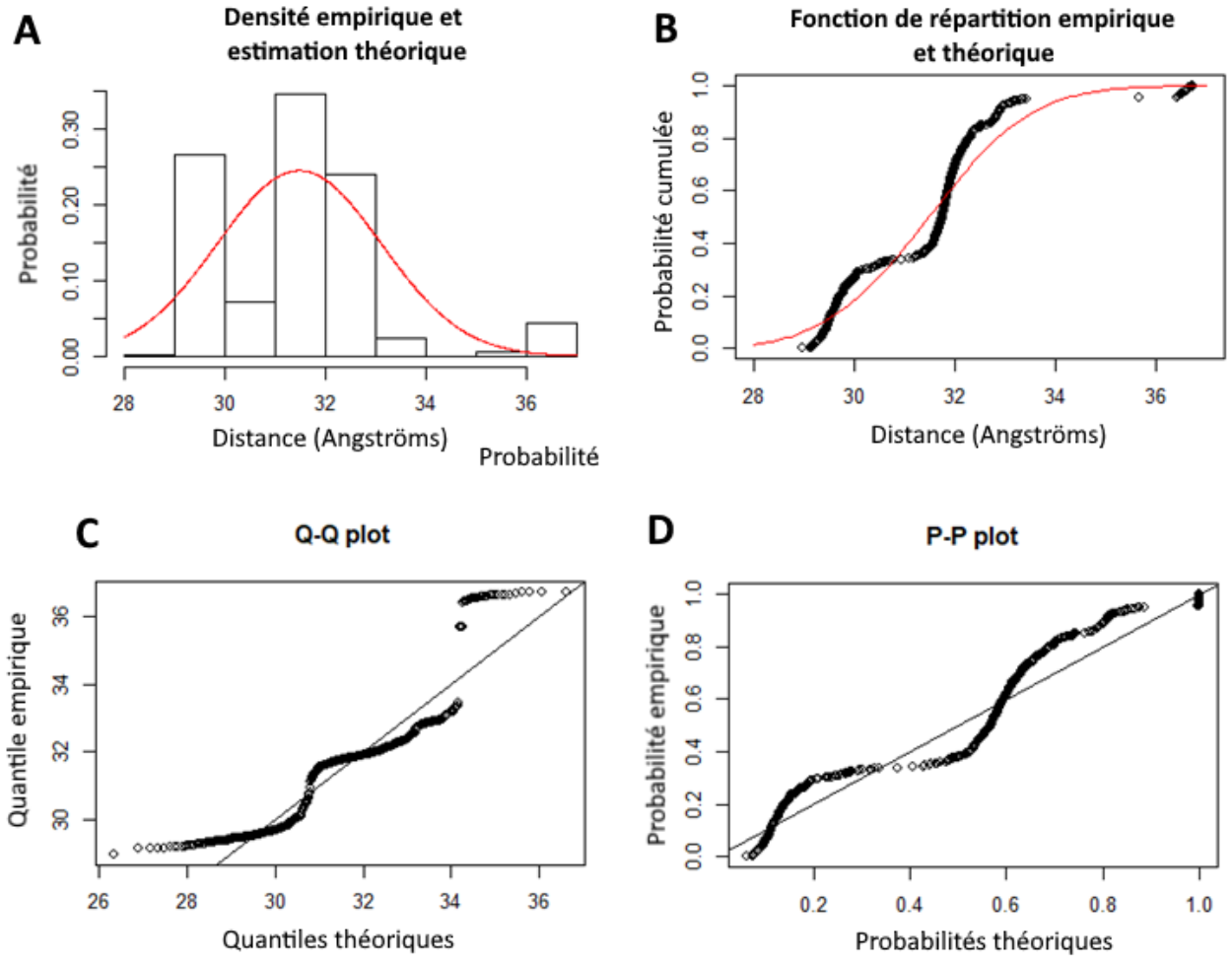


Figure 4.23 – Tests de normalité d'une distribution d'une distance. *Les 4 comparaisons proposées montrent que l'échantillon ne suit pas une distribution normale.* ***(A)*** *Un histogramme de la distance entre les positions 5 et 78 des ARNr 5S, sur lequel on superpose la densité théorique en rouge si elle était normale et de même espérance et variance.* ***(B)*** *Comparaison des fonctions de répartition de l'échantillon en noir et de la loi normale en rouge.* ***(C)*** *Graphique quartile à quartile : l'échantillon suit une distribution normale si les points sont correctement disposés sur la diagonale.* ***(D)*** *Graphique probabilité à probabilité : l'échantillon suit une loi normale si les points sont correctement disposés sur la diagonale.*

IV.4.2 Régression des matrices de distances par réseaux de convolution

Les travaux présentés dans cette partie sont les résultats du stage de fin d'études de Khodor Hannoush, étudiant du master 2 GENIOMHE, à qui j'ai proposé ce sujet et que j'ai ensuite co-encadré avec mes propres encadrants.

IV.4.2.1 Caractéristiques utilisées

On décide d'entrainer un réseau de convolution 2D à prédire les matrices de distances des ARN. On prépare en entrée des tenseurs 3D (de taille $L \times L \times N_c$ pour un ARN de longueur L), contenant les caractéristiques suivantes pour chaque position (i, j) :

- l'encodage one-hot des nucléotides aux positions i et j (2x6 caractéristiques, les six classes étant A, C, G, U, autre, ou gap) ;
- les fréquences des nucléotides dans l'alignement de séquences RNANet aux positions i et j, sans considérer les gaps (2x5 caractéristiques). En effet les gaps peuvent être majoritaires dans les insertions, leur fréquence n'a donc aucun sens ;
- la position relative des nucléotides i et j dans la séquence (i/L et j/L) et leur position dans l'alignement multiple de la famille (2x2 caractéristiques) ;
- les paramètres du modèle de Potts [111] issus de l'analyse des covariations (également utilisés dans AlphaFold). On les calcule via PyDCA [374], qui les obtient en maximisant une pseudo-vraisemblance (méthode plmDCA) à partir de l'alignement multiple. On utilise les 16 coefficients de la matrice J_{ij}, plus sa norme de Frobenius (norme L^2 matricielle) corrigée par le produit des moyennes de la ligne et de la colonne i et j (17 caractéristiques). À cela on ajoute le paramètre h_i du modèle de Potts pour chaque nucléotide possible (A, C, G, U) et pour i et j, ce qui rajoute 2x4 caractéristiques supplémentaires. En effet, l'implémentation de PyDCA ne renvoie pas de paramètres pour les nucléotides modifiés.

On a donc au total $N_c = 51$ caractéristiques pour chaque paire de positions (i, j).

IV.4.2.2 Premier essai : un modèle par famille d'ARN

À cause des difficultés rencontrées lors de l'entrainement du RGN dans la construction des batches de taille similaire, on commence par choisir d'entrainer un modèle par famille d'ARN. Au sein d'une famille d'ARN, les séquences sont de taille similaire, on peut donc se contenter de définir la taille du tenseur d'entrée comme étant $L \times L \times N_c$, où L est la largeur de l'alignement multiple des chaines 3D de la famille considérée.

L'architecture utilisée est un CNN simple, dont l'organigramme des couches est présenté sur la Figure 4.24A. C'est un problème de régression des distances, on utilise donc comme fonction de loss la MSE. Comme pour le RGN, le calcul de la MSE ne prend pas en compte les valeurs de distances non définies ou le remplissage. Les résultats pour cinq familles présentant suffisamment de chaines en 3D sont présentés dans la Table 1.

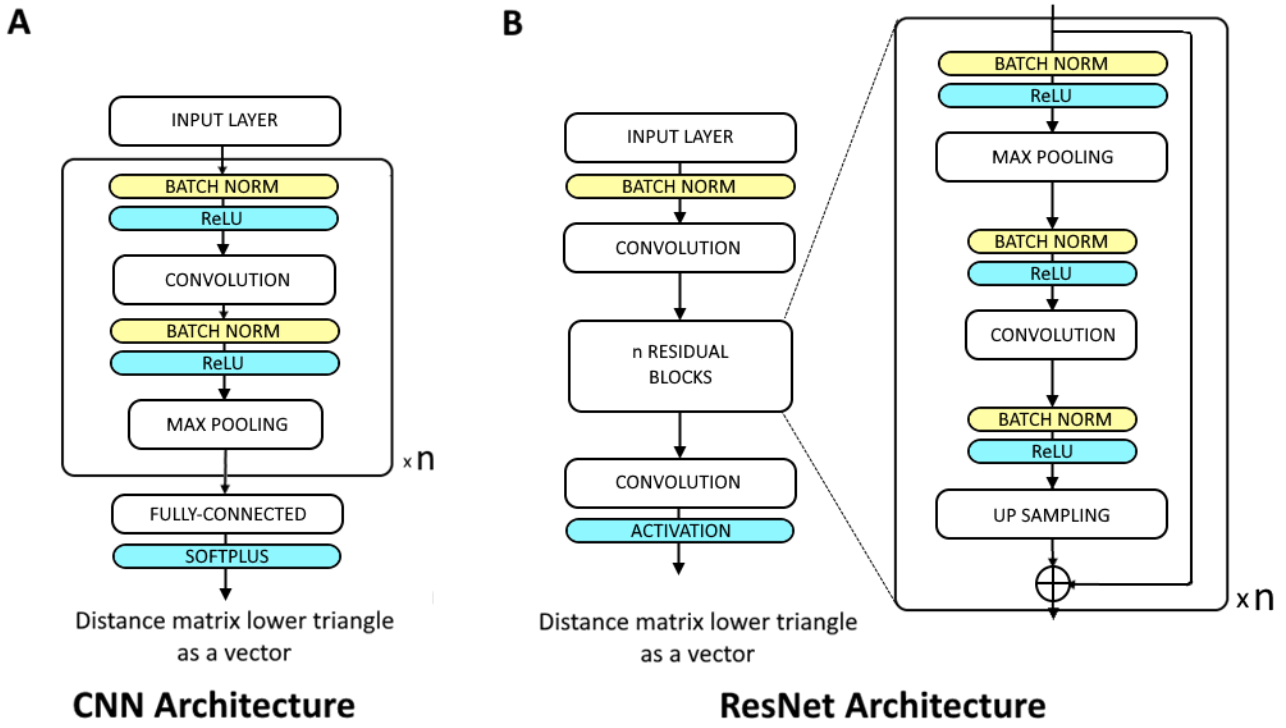


Figure 4.24 – Architectures des réseaux de convolution utilisés. ***(A)*** *Le réseau de convolution simple utilisé pour les modèles spécifiques d'une famille. On prend n=5 blocs, une taille de batch de 8 et un learning-rate spécifique de la famille.* ***(B)*** *Le réseau de convolution résiduel proposé pour le modèle généraliste. L'architecture du bloc résiduel est celle utilisée dans AlphaFold. La fonction d'activation finale utilisée sera softplus pour la régression, et softmax pour la classification multiple. On utilise n=5 blocs, et une taille de batch de 1 seulement, pour s'affranchir des problèmes théoriques de padding, et des problèmes pratiques de mémoire GPU. Ceci rend inutile les étapes de normalisation de batch, qui n'ont pas été implémentées en pratique.*

Famille d'ARN - Code Rfam (description)	MSE (validation) Modèle naïf	MSE (validation) Vrai modèle
RF00001 (ARNr 5S)	4 Å²	3 Å²
RF00002 (ARNr 5.8S)	2 Å²	2.5 Å²
RF00005 (ARNt)	6 Å²	4.5 Å²
RF01852 (ARNt-Sel)	32 Å²	320 Å²
RF01998 (Intron du Groupe II)	6 Å²	3 Å²

Table 1 – Performance des modèles après convergence pour 4 familles bien représentées.

Les valeurs reportées sont les valeurs obtenues pour le jeu de données de validation après convergence des poids du réseau de neurones, avant d'entrer dans la phase de surapprentissage (s'il y en a une). Dans le cas de RF01998, le réseau ne converge pas bien, on reporte la valeur moyenne obtenue selon les epochs. L'exemple des ARNt est également illustré sur la Figure 4.25. La Table 1 compare deux modèles :

- un modèle naïf, qu'on force à apprendre non pas la matrice de distances réelle de chaque ARN, mais la matrice moyenne des ARN de cette famille,
- un vrai modèle, à qui on essaie d'apprendre pour chaque ARN la matrice de distances correspondante.

Ceci permet de tester l'utilité du modèle d'apprentissage automatique : si le modèle (le vrai) ne permet pas d'obtenir une meilleure performance que simplement prendre la matrice moyenne de la famille, alors il n'est pas nécessaire d'entrainer un modèle d'apprentissage automatique pour cette tâche. Attribuer à toute chaine de la famille la matrice moyenne serait déjà un bon modèle en soi. On voit que c'est le cas avec les ARNr 5.8S et les ARNt-Sel, dont l'erreur au carré moyenne est faible après entrainement. On voit aussi que l'amélioration apportée par un vrai modèle pour les ARNt et ARNr 5S est relativement faible. Le cas RF01998 a du mal à converger, il est donc difficile à interpréter.

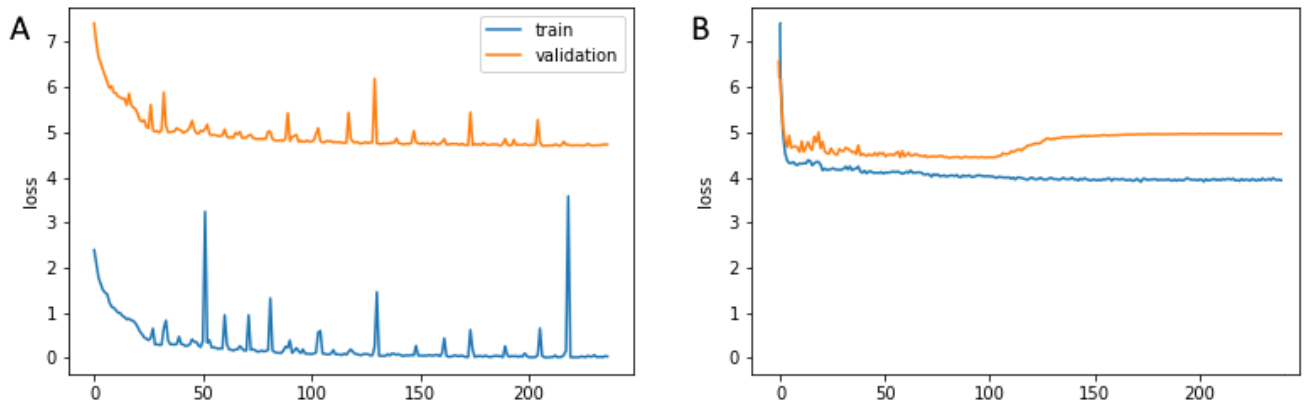


Figure 4.25 – Courbes d'apprentissage pour les ARNt. *Courbes de MSE pour les jeux de données d'entrainement et de validation du modèle des ARNt, en fonction du nombre d'epochs écoulés.* ***(A)*** *Modèle naïf pour les ARNt, à chaque point de données (chaine d'ARN) on associe la matrice moyenne de la famille.* ***(B)*** *Modèle réel pour les ARNt : à chaque ARN on associe sa matrice de distances réelle. La MSE de validation converge vers 4.5 Å² avant de commencer une phase de sur-apprentissage à partir de 100 epochs.*

En parallèle, on constate que les modèles pour lesquels un modèle fonctionne bien sont ceux dont la matrice d'écarts-type contient des valeurs faibles : la diversité au sein de la famille est faible. Il est donc inutile d'avoir un modèle prédictif pour ces familles dont la structure moyenne connue constitue un bon modèle pour chacun des membres. Plutôt qu'entrainer un modèle par famille, on se décide alors à tenter l'entrainement d'un modèle générique capable de prendre tout ARN en entrée.

IV.4.2.3 Second essai : un modèle généraliste, toutes familles confondues

Pour s'affranchir du problème de différence de longueur entre séquences de familles différentes, cette fois on essaie la stratégie des fenêtres glissantes. Cette stratégie consiste à choisir une fenêtre de taille W fixe qui va parcourir le tenseur d'entrée sur sa face $L \times L$, le long de la diagonale (voir Figure 4.26). Le réseau est donc prévu pour accueillir un tenseur d'entrée de taille $W \times W \times N_c$. On décale ensuite la fenêtre de 1 le long de la diagonale et on réalise une nouvelle prédiction (où on entraine à nouveau le réseau en considérant cette

nouvelle entrée comme un point de données à part entière). On répète l'opération jusqu'à avoir parcouru toute la longueur L, et on moyenne les prédictions sur les positions qui ont été prédites entre une et W fois. La taille de la fenêtre W doit être choisie suffisamment grande pour bien capturer les contacts longue distance entre nucléotides, mais suffisamment petite pour ne pas avoir besoin de faire du remplissage avec des données non informatives si une séquence est de longueur L inférieure à W. On calcule pour chaque ARN le nombre de contacts longue distance qui seraient manqués pour différentes tailles de fenêtres glissantes. Les résultats sont présentés sur la Figure 4.26. On choisira une fenêtre de taille 64, capable de capturer en moyenne 90% des contacts longue distance, et inférieure à la taille d'un ARNt (72 nucléotides).

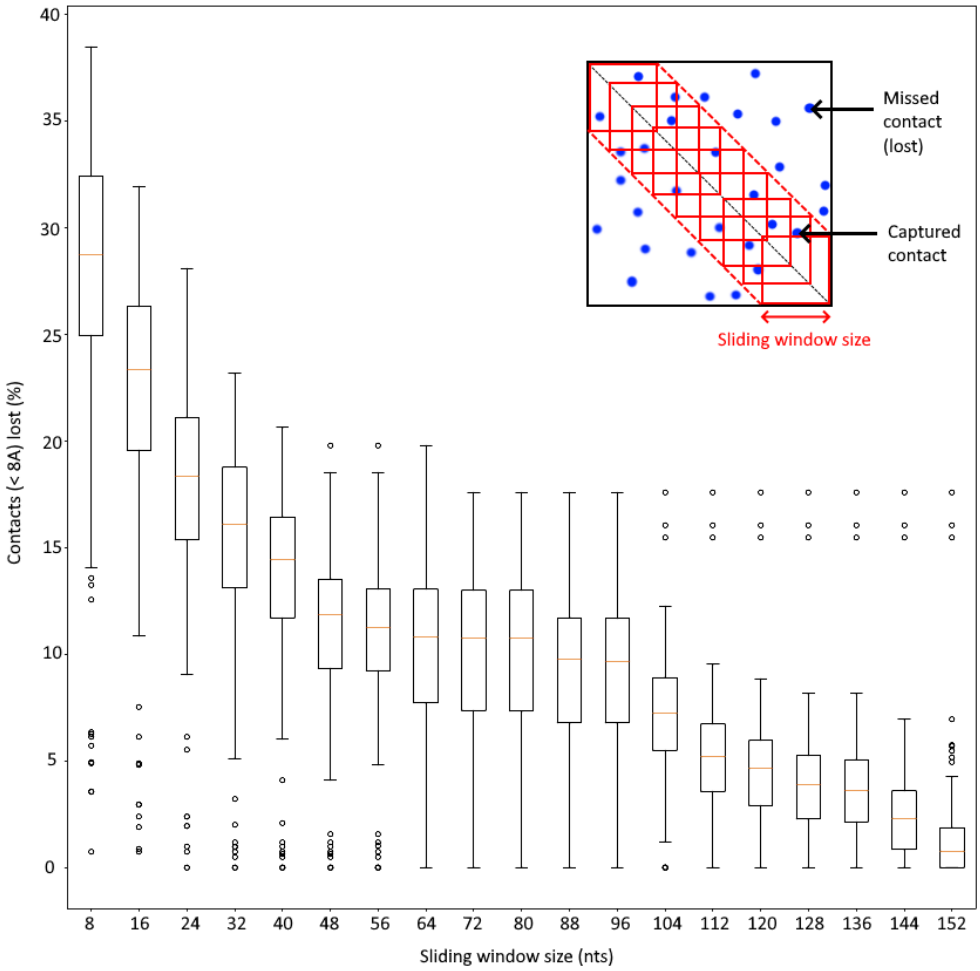


Figure 4.26 – Pourcentages de contacts non capturés par les fenêtres glissantes. *On définit deux nucléotides comme étant en contact si leurs carbones C'$_1$ sont à 8 Angströms de distance ou moins. Pour chaque ARN, on mesure combien de contacts seront manqués par une fenêtre de taille donnée. En prenant une fenêtre plus grande, le modèle capturera des contacts entre résidus plus éloignés dans la séquence. En contrepartie, les séquences qui ne sont pas assez longues pour remplir la fenêtre seront remplies avec davantage de données non informatives, pour égaliser les tailles.*

IV.4.2.4 Pertinence des caractéristiques d'homologie

On veut s'assurer que l'utilisation de caractéristiques dérivées de l'alignement multiple de séquences apporte bien un gain de performance au modèle. On entraine donc deux modèles : le premier avec seulement l'encodage one-hot, les pourcentages de gaps, et les positions relatives, le second avec toutes les caractéristiques (et notamment les fréquences nucléotidiques (PSSM) et les paramètres du modèle de Potts (DCA)).
Les valeurs de MSE obtenues après convergence et avant l'éventuel sur-apprentissage sont compilées dans la Table 2 pour plusieurs tailles de fenêtres glissantes.

Taille de la fenêtre W	**MSE sans PSSM/DCA (validation)**	**MSE avec PSSM/DCA (validation)**
48	45 Å²	20 Å²
64	65 Å²	26 Å²
96	70 Å²	38 Å²
104	80 Å²	38 Å²
128	120 Å²	56 Å²
152	150 Å²	85 Å²

Table 2 – Performance avec et sans données d'homologie.

On relève deux résultats de cette série d'expériences d'entrainement : premièrement, la performance du modèle diminue quand on augmente la taille de la fenêtre. Une explication probable est le remplissage nécessaire avec des données non informatives pour compléter les séquences de longueur inférieure à W. Deuxièmement, il apparait que les caractéristiques liées à l'homologie (fréquences et modèle de Potts) permettent bien une amélioration de la performance (d'un facteur 2 environ). On les conserve donc.

Néanmoins, la performance relativement correcte obtenue avec les petites fenêtres est probablement un artefact : en effet, une majorité des chaines d'ARN utilisées appartiennent à la famille des ARN de transfert (206 sur 484), dont on a vu au paragraphe 4.4.2.2. qu'elle possédait une très faible diversité et était donc très facile à apprendre. A l'inverse, l'utilisation de petites fenêtres rend inutile la prédiction des matrices de distances pour les sous-unités ribosomiques (SSU et LSU) dont la longueur se compte en milliers, et dont une majorité des contacts longue-distance ne seront pas capturés par une petite fenêtre. Par exemple avec $W = 100$, on obtient jusqu'à 78% de perte sur la structure 6az1-1 (une SSU, RF01960).

IV.4.2.5 Essai de réseau résiduel

Pour essayer d'augmenter la performance, on se propose d'essayer une architecture de réseau résiduel, c'est-à-dire en autorisant le saut de certains blocs par les connexions entre neurones. Ces connexions rapides permettent d'éviter la disparition du gradient lors de l'entrainement et rendent l'optimisation plus efficace sur des grands nombres de couches. Dans cette approche, on met de côté les fenêtres glissantes et on reconsidère les séquences

en entier, avec une taille de batch de 1 pour ne pas avoir de problèmes liés aux différences de longueur. On implémente l'architecture proposée sur la Figure 4.24B avec 5 blocs résiduels, car une exploration rapide montre qu'un réseau plus profond n'accélère pas la convergence (Figure 4.27). L'utilisation d'un plus grand nombre de blocs est difficile sur les GPU grand-publics à notre disposition. En effet, les plus longues séquences d'ARN demanderont l'allocation mémoire de plus de 12 Go de mémoire lors du calcul des gradients par TensorFlow.

Un première couche de convolution avant les blocs résiduels permet de transformer le tenseur d'entrée de dimension $L \times L \times N_c$ en tenseur de taille plus faible, $L \times L \times N_f$ où N_f est un nombre de « filtres », un hyperparamètre directement relié à la fois la performance, la complexité, et la difficulté d'entrainement du modèle. À l'intérieur du bloc, on retrouve une couche de convolution, précédée d'une couche de pooling réduisant la dimension du tenseur de $L \times L \times N_f$ à $L/2 \times L/2 \times N_f$, suivie d'une couche d'up-sampling rétablissant la dimension originale $L \times L \times N_f$, en dupliquant les valeurs. Ainsi, la convolution est faite sur un tenseur de plus faible dimension. Cela limite le sur-apprentissage en lissant le signal d'entrée, mais cela permet surtout une grande économie de ressources de calcul. L'up-sampling permet de retrouver un tenseur de dimension égale à celle attendue en entrée du bloc résiduel suivant. Pour pouvoir appliquer cette méthode, L doit être un nombre pair, on rajoute donc du padding de taille 1 à la fin des séquences de longueur impaire. En sortie, une couche de convolution avec un filtre de taille 1 permet de récupérer une seule valeur par couple de positions (i, j), à laquelle on applique la dernière fonction d'activation (softplus). Une dernière couche permet de contrôler la symétrie des deux triangles de la matrice de distance, en attribuant aux deux positions (i, j) et (j, i) la moyenne de leurs valeurs.

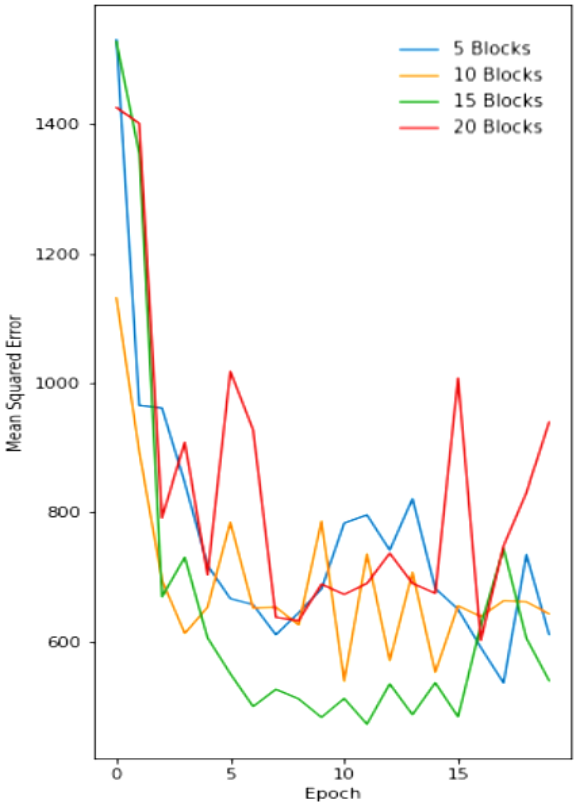


Figure 4.27 – Exploration du nombre de blocs résiduels. *La loss de validation pour des réseaux de différentes profondeurs est présentée (MSE en Angström carré). L'exploration rapide sur 20 epochs a pour but de déterminer la pente à l'origine des loss, pour voir si l'une des versions converge plus rapidement que les autres. Ici, le réseau à 15 blocs semble converger plus rapidement, mais la différence avec le modèle le plus simple (5 blocs) est faible.*

On entraine ce réseau résiduel en validation-croisée stratifiée à 4 stratifications, à partir des familles d'ARN. C'est-à-dire qu'on construit 4 divisions du jeu de données telles que toutes les familles d'ARN soient représentées à la fois dans le jeu d'entrainement et dans le jeu de validation, dans les mêmes proportions d'un jeu à l'autre, et que toute structure d'ARN appartienne au jeu de validation dans un (et un seul) des découpages. Ceci impose aux découpages de posséder toutes les familles d'ARN dans les mêmes proportions que dans le jeu de données complet original, et évite de se retrouver avec un découpage ou tous les ARN d'entrainement et de validation sont des ARNt.

Malgré tout, la performance moyenne observée est très décevante (plusieurs centaines d'Angströms carrés après convergence, données non présentées). On conclut que cette approche n'est pas appropriée. Ce type de réseau a besoin de gros jeux de données et est vraiment limité dans notre situation avec 500 points de données, encore subdivisés par la validation croisée. On se propose de changer le problème de régression en problème de classification pour simplifier le problème et faciliter l'apprentissage.

IV.4.3 Classification en « distogrammes » des matrices de distances par CNN-ResNet

AlphaFold a introduit (ou popularisé) la notion de « distogramme », c'est-à-dire un histogramme des distances. Au lieu de procéder à la régression d'une distance à une valeur numérique exacte, le réseau prédit son appartenance à l'une des classes de distances de l'histogramme. L'avantage est d'obtenir un problème de classification multiple au lieu d'un problème de régression, en conservant la majorité de l'information apportée par la prédiction. Les problèmes de classification sont considérés plus simples à apprendre que ceux de régression.

On se propose de classifier les distances entre paires de résidus dans 12 classes de distance. Ce nombre est choisi plus petit que celui d'AlphaFold (63 classes dans l'intervalle [0,22[et une classe « plus de 22 Angströms »), mais nous avons beaucoup moins de données, alors on l'a choisi plus faible pour que le problème soit plus simple. Le nombre 12 permet de choisir un découpage de façon que les classes soient d'ordre de grandeur similaire (hormis les 3 premières). Leurs fréquences respectives sont données dans la Table 3. On garde tout de même les premières classes, même déséquilibrées, puisque ce sont les classes les plus intéressantes : prédire les courtes distances permet de prédire les contacts et les structures locales. Ce sont donc ces classes qui sont réellement informatives.

Classe de distance (Angströms)	Fréquence dans le jeu de données	Poids w_k
[0, 5 [	0.0014	0.641
[5,10 [	0.0058	0.155
[10,15 [	0.0103	0.087
[15,20 [	0.0160	0.056
[20,40 [	0.1090	0.008
[40,55 [	0.1151	0.008
[55,70 [	0.1293	0.007
[70,85 [	0.1305	0.007
[85,100 [	0.1213	0.007
[100,120 [	0.1330	0.007
[120,140 [	0.0950	0.009
[140, +∞ [	0.1310	0.007

Table 3 – Classes de distance entre paires de résidus. *On choisit arbitrairement douze classes de distance définissant un histogramme. Les fréquences associées sont du même ordre de grandeur, sauf pour les trois premières classes. Pour prendre en compte le déséquilibre, on pondère l'erreur commise par le réseau de neurones lors d'une mauvaise prédiction par un poids inversement proportionnel à la taille de la classe à prédire.*

On modifie donc le réseau résiduel, en changeant le nombre de filtres de la dernière couche de convolution de 1 à 12, et en remplaçant la dernière fonction d'activation softplus par une fonction softmax : on obtient donc 12 valeurs entre 0 et 1 dont la somme vaut 1 pour chaque couple de positions (i,j). Ces valeurs sont interprétables comme des probabilités d'appartenance à chacune des 12 classes de distances. On utilisera comme loss l'entropie croisée catégorielle, couramment utilisée pour mesurer la performance dans les problèmes de classification multiple :

$$Loss = -\sum_{i=1}^{L-1}\sum_{j=i+1}^{L} y_{ij} \cdot \log \hat{y}_{ij},$$

où y_{ij} représente le vecteur one-hot indiquant la classe de distance à laquelle appartient le couple de résidu (i,j), et $\hat{y}_{ij}$ est le vecteur softmax obtenu en sortie du réseau.
Pour prendre en compte le déséquilibre des classes, car les trois premières sont à la fois les plus importantes à prédire et les moins représentées, on modifie la formule de l'entropie croisée pour pondérer chaque terme par un poids :

$$Loss = -\sum_{i=1}^{L-1}\sum_{j=i+1}^{L} \left(y_{ij} \cdot \log \hat{y}_{ij}\right) \times \left(y_{ij} \cdot w\right),$$

où le vecteur de poids w et défini par :

$$w_k = \frac{1/f_k}{\sum_l 1/f_l}$$

avec f_k la fréquence de la classe k observée dans le jeu de données, telle que reportée dans la Table 3. Ainsi, une mauvaise prédiction d'une courte distance sera fortement pénalisée, quand une mauvaise prédiction dans les autres classes sera moins pénalisante.

On pourra également définir pour chaque classe les métriques de performances s'appliquant habituellement aux problèmes de classification binaire : exactitude, sensibilité, valeur prédictive positive, et F1-score. L'exactitude mesure le pourcentage d'attributions correctes entre l'appartenance ou la non-appartenance à la classe. La sensibilité mesure le pourcentage des membres de la classe correctement détectés comme en faisant partie. La valeur prédictive positive (VPP, aussi appelée précision) mesure la probabilité qu'une distance prédite comme faisant partie de la classe en fasse effectivement partie. Le F1-score est une moyenne harmonique de la sensibilité et de la VPP, rendant compte de la performance générale de la classification. Deux séries de résultats sont présentés dans la Table 4, avec l'entropie croisée courante et avec l'entropie croisée pondérée.

Classe	**Sans pondération**				**Avec Pondération**			
	Exactitude	Sensibilité	VPP*	F1-score	Exactitude	Sensibilité	VPP	F1-score
[0,5 [	0.98	0.27	0.77	0.37	0.99	**0.87**	0.60	0.70
[5,10 [	0.95	0.17	0.75	0.26	0.95	0.43	0.71	0.52
[10,15 [	0.92	0.12	0.69	0.19	0.92	0.46	0.63	0.51
[15,20 [	0.88	0.11	0.57	0.16	0.87	0.40	0.49	0.41
[20,40 [	0.67	0.28	0.69	0.36	0.69	0.75	0.61	0.66
[40,55 [	0.68	0.36	0.20	0.21	0.84	0.29	0.37	0.30
[55,70 [	0.68	0.61	0.12	0.19	0.93	0.09	0.30	0.12
[70,85 [	0.91	0.095	0.03	0.037	0.97	0.004	0.05	0.004
[85,100 [	0.96	0.008	0.02	0.01	0.98	4e-4	0.01	7e-4
[100,120 [	0.93	0.009	0.009	0.005	0.99	4e-5	0.01	9e-5
[120,140 [	0.98	0.004	0.009	0.004	0.99	3e-5	0.01	6e-5
[140,+∞ [	0.95	0.005	0.02	0.002	0.90	0.01	0.01	0.01
Moyenne	**0.88**	**0.17**	**0.32**	**0.15**	**0.92**	**0.28**	**0.32**	**0.27**

Table 4 – Performances de prédiction avec 12 classes de distances. *On a réalisé deux séries d'expériences, la première utilisant l'entropie croisée courante, et la deuxième l'entropie croisée mais pondérée par un vecteur de poids comme expliqué plus haut. Pour chaque classe, on reporte les métriques de performance définies pour le problème de classification binaire entre cette classe et les autres (sans distinction des autres).*

On constate que la performance est très hétérogène selon les classes. Avec pondération, on obtient une très bonne performance de prédiction pour la première classe (qui avait un poids largement supérieur aux suivants) : 87% des paires de nucléotides à moins de 5 Angströms l'un de l'autre sont correctement détectées, et si une paire est prédite dans l'intervalle [0,5 [, cette prédiction est le plus souvent juste (60%). Ce résultat est l'effet direct du poids élevé pour cette classe : on ne l'observe pas sans pondération. Malheureusement,

cette performance s'estompe vite avec la distance (et les poids w_k décroissants), pour la classe de distance [5,10 [, on détecte moins d'une paire de nucléotides sur deux seulement. La performance pour les classes suivantes devient mauvaise, jusqu'à une absence quasi-totale de vrais positifs pour les classes 8 à 11 : le réseau ne fait presque aucune prédiction dans ces classes, leur poids est trop faible. Ces classes de distance sont peu importantes : à partir du moment où les nucléotides ne sont pas en contact, et qu'ils ne participent pas à une structure locale (SSE, ou pseudonoeud), il importe peu de savoir s'ils sont à 90 ou 120 Angströms de distance. Mais tout de même, on déplore les très faibles scores obtenus sur les longues distances.

On a ainsi obtenu un bon réseau de neurones capable de faire de la prédiction de contacts, avec une performance similaire aux autres outils sortis dans le même laps de temps pour cette tâche (CoCoNet [375], RNA-Contact [325], SPOT-RNA [312,313] notamment). L'objectif nouveau souhaité pour notre modèle par rapport à ces outils était la prédiction des autres classes de distance (une classification multiple au lieu d'une classification binaire), malheureusement, la performance est pour l'instant mauvaise sur les autres classes.

On peut également choisir de mélanger ensemble toutes les classes « distantes » puisque faire la distinction entre elles nous importe moins. Mécaniquement, le déséquilibre entre les classes s'accentue alors. Mais ceci permet d'augmenter la performance globale de classification. Par exemple, des résultats avec 4 classes au lieu de 12 sont présentés dans la Table 5.

Classe	**Fréq.**	**Sans pondération**				**Avec Pondération**			
		Exactitude	Sensibilité	VPP*	F1-score	Exactitude	Sensibilité	VPP*	F1-score
[0,5 [	0.0014	0.87	0.538	0.90	0.62	0.87	0.67	0.96	0.75
[5,10 [	0.0058	0.84	0.310	0.80	0.40	0.85	0.46	0.86	0.55
[10,15 [	0.0103	0.81	0.150	0.60	0.22	0.75	0.51	0.37	0.37
[15,+∞ [	0.9825	0.77	0.998	0.81	0.82	0.74	0.78	0.97	0.87
Moyenne		**0.82**	**0.50**	**0.78**	**0.51**	**0.80**	**0.60**	**0.79**	**0.63**

Table 5 – Performances de prédiction avec 4 classes de distance. *En mélangeant ensemble toutes les classes « longue distance », pour lesquelles on n'est pas réellement intéressés de faire la distinction, on obtient un jeu de données fortement déséquilibré entre les classes.*

Les performances observées sont plus équilibrées entre les classes, on est notamment capables de bien prédire les paires à courte distance et à très longue distance. La performance pour les classes centrales reste mauvaise.

IV.4.4 Prédiction des angles par CNN-ResNet

La prédiction des angles de torsion du squelette a été essayée (sans succès) avec l'architecture RGN en section 4.3. Dans le cadre du stage de Léa Boulos (étudiante du master 1 GENIOMHE), l'approche par réseau de convolution pour prédire ces mêmes angles est également testée. Contrairement à la prédiction des matrices de distances, on utilise ici de la convolution en une seule dimension, le long de la séquence, contre 2 dimensions auparavant. Les caractéristiques utilisées sont les mêmes que pour le RGN (présentées au paragraphe 4.3.1.4), à savoir : l'encodage one-hot des nucléotides de la séquence, les fréquences nucléotidiques à chaque position dans les ARN homologues, et la position relative du nucléotide dans la chaine. Le tenseur d'entrée est donc en deux dimensions, de taille Lx11. Les données utilisées et les alignements de séquences sous-jacents sont toujours ceux de RNANet. On explore de façon systématique :

- comme pour le RGN, les différents types de problèmes : régression de la valeur des angles, régression de la valeur des sinus et cosinus des angles, ou classification multiple attribuant chaque nucléotide à un cluster dans le plan η'/θ' du modèle de Pyle (ou modèle VFold),
- les clusters utilisés : soit ceux obtenus par modèles mixtes gaussiens selon la méthodologie présentée au paragraphe 4.3.3.1 et illustrés sur la Figure 4.17, soit une méthode plus proche de l'article original du RGN [10], où l'alphabet structural utilisé est une division égale du plan des angles de torsion en 60 points de référence, sans qu'il y ait eu de clustering. Dans les deux cas on explore le nombre de clusters à utiliser,
- le nombre de blocs résiduels,
- la taille du kernel utilisé pour la convolution,
- le nombre de filtres (la dimension des sorties des neurones des couches de convolution),
- le learning-rate,
- la pondération ou non de l'entropie croisée (loss) en fonction de la taille des clusters.

Les résultats ne sont pas bons, et surtout, pas meilleurs qu'avec le RGN. La meilleure méthode identifiée pour le problème de classification en 60 clusters présente une exactitude finale de prédiction d'environ 31% (10 blocks résiduels, kernel de taille 5, 11 filtres, learning-rate = 0.001). En diminuant le nombre de « lettres » dans l'alphabet structural, et en les choisissant régulièrement espacées, on peut obtenir jusqu'à 36% (25 clusters régulièrement espacés, entropie croisée pondérée, 2 blocs résiduels, kernel de taille 9, 32 filtres, learning-rate = 0.001).

De façon générale, la pondération de la loss permet d'augmenter la performance de prédiction, et l'augmentation de la taille du kernel aussi, jusqu'à un palier (kernel de taille 21). On observe peu d'effet du nombre de blocs résiduels, probablement parce qu'on manque de données pour entrainer correctement un réseau très profond, ce qui remet en question la pertinence de l'architecture ResNet pour ce problème.

IV.5 Conclusion et perspectives

Dans ce chapitre, on a imaginé comment adapter pour l'ARN les récentes avancées en deep-learning appliqué à la prédiction de structures de protéines.

On a notamment développé RNANet, un jeu de données standardisé et automatiquement mis à jour mensuellement, qui fournit aux chercheurs en informatique toutes les données nécessaires dans un format exploitable. Il regroupe des descripteurs des structures d'ARN 3D publiques à toutes les échelles, ainsi que des alignements de ces structures avec des familles et des modèles de covariance connus. Ce jeu est unique en son genre et constitue une contribution importante pour le champ de recherche. Les perspectives d'évolution liées à RNANet ont déjà été listées en section IV.2.5.

J'ai ensuite utilisé RNANet pour essayer de prédire les angles de pseudo-torsion du squelette des chaines sur la base de la séquence et des données d'homologie, sur la base de l'idée d'Al Quraishi et de son architecture RGN [10] basée sur les LSTM et l'algorithme pNERF [11] pour prédire les torsions des chaines protéiques. pNERF permet la reconstruction automatique de la chaine en 3D de façon entièrement différentiable et intégrée au réseau de neurones. Les principales difficultés étaient la gestion de la grande diversité de longueur des chaines, et la façon de représenter l'espace cyclique des angles pour le réseau de neurones. Différentes approches ont été envisagées : régression des angles ou de leurs cosinus et sinus, classification multiple avec un alphabet structural obtenu par clustering, et enfin la prédiction complète de la chaine avec pNERF. Malheureusement, le déséquilibre de classes rend le problème trop difficile pour le peu de données disponibles, à cause de la dominance écrasante des conformations en hélice. La performance finale est décevante. De plus, même si elle avait été parfaite, on a montré que la prédiction des angles de torsion seuls n'était pas suffisante. En effet et contrairement aux protéines, les angles plans ne peuvent pas être supposés constants dans l'ARN.

Deux sujets de stage ont ensuite été proposés sur ces thèmes à des étudiants du master GENIOMHE. Khodor Hannoush a travaillé sur la prédiction de « distogrammes », c'est-à-dire la prédiction des distances entre paires de résidus de la chaine d'ARN. Il a utilisé différentes architectures toutes basées sur la convolution. Il ressort de cette étude que ces modèles fonctionnent très bien à l'intérieur de certaines familles d'ARN, en classification comme en régression, mais ces familles ont déjà des structures moyennes bien connues et peu variables. Ces modèles prédictifs sont donc peu utiles. Des modèles inter-familles ont également été tentés : la régression n'est pas envisageable ici. La classification multiple en classe de distance fonctionne un peu, mais avec une performance mauvaise. L'état actuel des résultats laisse supposer que des améliorations techniques permettront d'augmenter la performance, avec de meilleures stratégies de pondération des classes par exemple (voir les perspectives plus bas). On a également réussi à entrainer des modèles de classification binaire pour prédire les résidus en contact, avec une performance comparable aux autres outils sortis durant la même période. Léa Boulos a aussi utilisé des réseaux de convolution pour essayer de prédire les angles de pseudo-torsion, en explorant différents alphabets

structuraux, mais sans faire mieux que le réseau récurrent RGN.

De nombreuses idées pour résoudre certains problèmes sont arrivées au fur et à mesure du stage de Khodor, sans que l'exploration systématique de chaque architecture ait été faite. On pourrait donc encore tester :

- L'utilisation de softplus à la place de ReLU à l'intérieur des blocs résiduels. Dans les problèmes de régression, softplus a permis de régler des problèmes de gradients nuls (le réseau n'apprend pas), dus à la présence de coefficients négatifs dans les sorties des couches de convolution.
- La pondération de l'erreur par des poids inversement proportionnels à la fréquence des familles d'ARN dans le jeu de données dans l'architecture CNN simple avec les fenêtres glissantes.
- L'utilisation de fenêtres glissantes dans le problème de classification en distogrammes. C'est ce que faisait AlphaFold, et l'utilisation d'une contrainte sur la distance maximale entre deux résidus considérés dans la séquence permettait de s'affranchir des très larges classes de distances qui n'apportent pas d'information. En effet, AlphaFold utilisait 64 classes de distances entre 0 et 22 Angströms seulement.
- L'utilisation de fenêtres glissantes dans le réseau résiduel, ce qui permettrait de réduire l'impact mémoire du réseau, et éventuellement d'utiliser des batches de taille supérieure pour accélérer la convergence, et donc de mieux entrainer le réseau en moins de temps.
- L'apprentissage direct à partir de l'alignement multiple, sans passer par le calcul intermédiaire de caractéristiques, n'a pas encore été essayé.
- La validation croisée stratifiée pourrait être testée dans tous les modèles pour des mesures de performance plus fiables.

CHAPITRE V : Un programme multi-objectif pour prédire la structure 3D

On a présenté en introduction de cette thèse les différentes méthodes d'évaluation d'une structure. Pour rappel, certaines sont basées sur des modèles physiques, d'autres sur la ressemblance à des données connues. Aucune n'étant objectivement meilleure que les autres dans le cas général, les outils qui les utilisent vont proposer des solutions différentes mais de qualité similaire, sans que l'utilisateur n'ait de réelle façon rationnelle de choisir entre les différentes propositions. Un élément clé, nécessaire pour faire ce choix, serait de connaître le score d'une solution obtenue par l'optimisation d'un modèle A dans un modèle B, et réciproquement. On pourrait alors comparer les solutions obtenues individuellement en connaissant leurs scores sur l'autre critère. Cependant, on manquerait de nombreuses solutions « moyennes » sur chaque critère pris individuellement, mais qui sont intéressantes si l'on considère les deux critères pris ensemble (par exemple, des solutions non-dominées du front de Pareto formé par les modèles A et B).

Dans ce chapitre, qui étudie l'hypothèse centrale de la thèse, on propose de considérer chaque méthode de prédiction comme un axe, et les structures d'ARN comme des points dont les coordonnées sont les scores qu'elles auraient dans le modèle de chaque axe. On résout ensuite le problème d'optimisation multicritère associé, et on renvoie une approximation du front de Pareto. Le front est organisé en une collection de solutions (des structures) réparties entre les points extrêmes, qui sont les meilleures solutions sur chaque axe.

L'utilisation de modèles à différentes échelles pourrait s'avérer utile, pour repérer des solutions de bonne qualité à toutes les échelles. On peut donc imaginer un outil d'optimisation multi-objectif qui proposerait des solutions non dominées pour un modèle possédant des critères sur la structure secondaire, d'autres sur la structure tertiaire, certains basés sur des modèles physiques, d'autres sur des modèles statistiques.
La notion d'échelle ne se limite pas à la structure secondaire contre la structure tertiaire : on peut par exemple vouloir comparer les méthodes d'assemblage de fragments. Certaines, comme FARFAR [85], assemblent des petits fragments monocaténaires et contenant tous les atomes. D'autres, comme MC-SYM [250], assemblent des motifs cycliques. D'autres comme VFold [51] ou RNAComposer [261] assemblent des éléments de structure secondaire entiers. Avec le pipeline RNA-MoIP/MC-SYM [272], certains peuvent être des modules.

Enfin, on a vu que l'ARN est dynamique, et qu'il y a souvent différentes structures à prédire correspondant à différents états métastables. Elles correspondent à des minima locaux du paysage énergétique de chaque critère (si les critères sont pertinents). En renvoyant un ensemble de Pareto, un outil d'optimisation multicritère se donne la capacité

d'utiliser plusieurs paysages énergétiques différents pour échantillonner obtenir ces solutions. On augmente donc les chances d'obtenir des solutions intéressantes dans l'échantillon de sortie, qui s'étend entre les minima globaux des critères utilisés dans l'espace des critères.

Pour des raisons de performance, on ne peut pas construire de procédure d'optimisation en faisant appel aux outils exécutables déjà existants. On aura nécessairement à réimplémenter les modèles qu'on voudra utiliser, ou intégrer leurs codes sources dans un unique exécutable.

Dans la première section du chapitre, on présentera les critères d'évaluation utilisés dans ce modèle, définissant ensemble un problème d'optimisation multi-objectif. Dans la section V.2, on essaiera de formaliser le problème d'optimisation. Ceci nous mènera au choix d'un algorithme d'optimisation générique, qu'on décrira dans cette section V.2. La section V.3 présente les résultats de simulation obtenus. On conclura en section V.5.

V.1 Approche et choix des critères d'évaluation

On souhaite, volontairement, utiliser une très large diversité de critères possibles pour tenter ensuite de comparer leurs utilités et performances respectives. On essaiera donc de trouver des formulations d'évaluation de la structure secondaire, de la structure tertiaire, des formulations qui tirent profit de toutes les informations disponibles (homologie, données de sondage chimique), en veillant aussi à la représentation des modèles physiques et statistiques.

V.1.1 Proximité à une prédiction de la forme du squelette

Ce critère utiliserait une prédiction de la forme du squelette produite par un des réseaux de neurones profonds présentés au chapitre précédent. Ce serait donc un modèle statistique. La fonction serait continue à variables continues : les angles de torsion du squelette (par exemple, η' et θ'). La fonction de distance pourrait être le dRMSD entre la prédiction et la structure à évaluer. Un inconvénient cependant : la prédiction par apprentissage profond sous-jacente nécessite d'avoir identifié une famille pour l'ARN à modéliser.

Cette idée permettrait d'identifier un « bassin énergétique » proche de la conformation prédite, dans lequel l'exploration pourrait se faire grâce aux autres critères. L'exploration n'est pas limitée à ce bassin, puisque le front de Pareto contiendra des points avec de très mauvais scores sur ce critère s'ils en ont des bons sur les autres. Cependant, l'exploration commencera par ce bassin, ce qui permet probablement d'accélérer la recherche.

Nous étions confiants dans le potentiel de ce critère, puisque la prédiction de ce squelette fonctionne très bien avec les protéines, et qu'on a montré que comme pour les clusters dans le diagramme de Ramachandran, il existe des clusters dans le diagramme η'/θ' (initialement montré par les articles de Pyle [342], puis reproduit avec les données de RNANet, voir section IV.2.3). Cependant, nous n'avons pas encore réussi à entrainer un réseau capable de faire ces prédictions d'angles. Le critère est donc pour l'instant en attente, et n'a pas été utilisé.

V.1.2 Compatibilité avec la forme des ARN de la même famille

En complémentarité du critère imposant une contrainte sur le squelette, il en faut un autre imposant une contrainte sur l'orientation des bases. On se propose d'utiliser un critère mesurant la différence entre les distances inter-résidus dans la structure à évaluer et dans la matrice moyenne des distances de la famille d'ARN (voir Figure 4.22). Ce critère exploite aussi les informations d'homologie, c'est-à-dire qu'il faut avoir identifié une famille d'ARN.

Une première façon de formaliser le problème serait d'utiliser des classes de distances, comme fait AlphaFold avec ses « distogrammes ». Le distogramme moyen p_F de

la distance entre chaque paire de bases d'une famille F peut être calculé (en le normalisant entre 0 et 1 pour ressembler à une probabilité). Si ce distogramme contient n_d classes, on peut exprimer la probabilité d'une structure s à évaluer sachant une famille F :

$$p(s|F) = \prod_{i=1}^{n-1} \prod_{j=i+1}^{n} \sum_{c=1}^{n_d} p_F\,[c] \times I(d_{ij} \in c)$$

où d_{ij} est la distance entre les bases i et j et $I(d_{ij} \in c)$ vaut 1 si d_{ij} appartient à la classe c et 0 sinon. Ainsi, on peut considérer que les d_{ij} sont les variables continues du problème, ou alors simplement les appartenances aux classes, des variables discrètes. Néanmoins cette formulation présente un problème : elle suppose l'indépendance des distances entre paires de résidus, ce qui est faux (d_{ij} contraint fortement d_{ij+1}). On décide donc de ne pas utiliser ce système de classes et de mesurer directement la différence à la moyenne avec une fonction de coût de type MSE (à minimiser). En notant $\bar{d}_{ij}$ le premier mode de l'histogramme des d_{ij} sur la famille F :

$$MSE(s|F) = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} (d_{ij} - \bar{d}_{ij})^2.$$

On note la similarité avec le dRMSD. Cependant, on se souvient que les distances d_{ij} en question suivent rarement une distribution normale, il est donc moins pertinent d'utiliser la MSE que la probabilité. Réfléchir à une meilleure formulation du problème avec les classes pourrait être une nouvelle piste de recherche.

Ce critère a l'avantage d'être indépendant de la séquence et des mutations et variations, il capture la forme des ARN de la famille. De plus, l'idée est innovante depuis qu'AlphaFold a démontré son utilité pour la prédiction de structures protéiques, aucune méthode ou outil ne l'ayant utilisée sur les ARN jusqu'à présent (à ma connaissance). Il est donc important de l'essayer.

V.1.3 Compatibilité avec des données expérimentales de sondage chimique

Les expériences de sondage chimique donnent une information sur la réactivité et la disponibilité d'un nucléotide, souvent interprétées après normalisation comme une probabilité d'être non-apparié. On a imaginé deux façons de modéliser une compatibilité avec ces données.

La première utiliserait des variables binaires (discrètes donc), définissant pour chaque paire de nucléotides leur état d'appariement. On pourrait alors formuler une crédence (compatibilité d'un modèle avec des données expérimentales) pour une solution s avec les données expérimentales D :

$$p(s|D) = \prod_{i=1}^{n-1} \prod_{j=i+1}^{n} \left[p\left(y_j^i|D\right).I\left(y_j^i \in s\right) + \left(1 - p\left(y_j^i|D\right)\right)\left(1 - I\left(y_j^i \in s\right)\right)\right]$$

où $I\left(y_j^i \in s\right)$ sont les variables de décision binaires du problème et valent 1 si i et j sont appariés dans s et 0 sinon, et $p\left(y_j^i|D\right)$ la vraisemblance de l'appariement (i,j) selon les données D. On peut calculer les valeurs de $p\left(y_j^i|D\right)$ d'une façon inspirée de Bindewald *et al.* [38] : soit D la donnée de sondage chimique à une position. Soit H_p l'hypothèse « le nucléotide est apparié » et H_{np} l'hypothèse contraire. Elles sont complémentaires, donc $p(H_p) = 1 - p(H_{np})$ et $p(H_p|D) = 1 - p(H_{np}|D)$. Par la formule de Bayes, on a :

$$\frac{p(H_p|D)}{p(H_{np}|D)} = \frac{p\left(D\middle|H_p\right).p(H_p)}{p\left(D\middle|H_{np}\right).p(H_{np})} = \frac{p\left(D\middle|H_p\right).p(H_p)}{p\left(D\middle|H_{np}\right).\left(1 - p(H_p)\right)}$$

En appelant $F(D)$ le rapport des vraisemblances $\frac{p(D|H_p)}{p(D|H_{np})}$ et f le rapport $\frac{p(H_p)}{1-p(H_p)}$, on a :

$$p\left(y_j^i|D\right) = p\left(H_p|D\right) = \frac{F(D).f}{1 + F(D).f}.$$

Les valeurs de $F(D)$ et f peuvent être calculées ou déterminées en fonction du protocole de sondage.

Une seconde façon de faire, décrite par Deigan *et al.* [90] et utilisée dans la thèse d'Afaf Saaidi [288], est de sommer des contributions pour chaque nucléotide apparié, la contribution étant une fonction de la réactivité du nucléotide concerné. Elle peut prendre des valeurs positives ou négatives. Par exemple, la formule proposée et que nous réutiliserons est la suivante :

$$F(s|D) = \sum_{i=1}^{n} [a.\ln(1 + D_i) + b] \times I(\exists k, y_k^i \in s), \qquad a > 0, b < 0$$

avec y_k^i un appariement faisant intervenir le nucléotide i. Ainsi, si un nucléotide n'est pas apparié, il ne contribue pas du tout. S'il est apparié et que sa réactivité D_i est nulle (cohérent), il apporte une contribution énergétique b choisie négative (favorable). Puis, si sa réactivité D_i n'est pas nulle, sa contribution énergétique se détériore et devient positive (pénalisante) si elle dépasse un certain seuil. Nous utiliserons ce critère.

Ce critère permet d'incorporer de la connaissance expérimentale dans le problème d'optimisation. Cette fonction est agnostique du type de protocole expérimental et du réactif utilisé, ou même de l'ordre de grandeur des mesures de réactivité fournies. Elle peut même être généralisée à tout type d'information sur la structure secondaire : D_i peut être un temps de séjour dans une conformation non-appariée obtenue par dynamique moléculaire, ou même une quantité fixe si la structure secondaire est connue. L'inconvénient du critère reste bien entendu la nécessité d'avoir des informations expérimentales disponibles. On note également qu'une publication récente [289] montre que l'utilisation

d'un seul réactif de sondage chimique n'informe pas toujours correctement sur les appariements (à cause des erreurs expérimentales), mais que l'utilisation de plusieurs protocoles de sondage avec des réactifs différents permet d'augmenter significativement le signal obtenu. On peut donc imaginer l'utilisation multiple de ce critère, de façon indépendante, en le définissant de façon séparée pour chaque protocole de sondage.

V.1.4 Nouveau modèle d'évaluation des structures secondaires

Plusieurs modèles d'estimation de l'énergie d'une structure existent. Les plus simples se contentent de sommer les contributions des empilements et/ou des appariements (c'est le cas de BiokoP [192] par exemple). Le modèle le plus élaboré et le plus utilisé, dit modèle du plus proche voisin, ou « Turner rules » [227,340], somme des contributions de différents éléments de structure (voir Figure 5.1A) lorsqu'ils sont observés :

- une pénalité associée à chaque appariement fermant une boucle (*closing basepair*) si ce n'est pas par un G-C (+0.45) ;
- des pénalités connues et tabulées, pour les empilements, boucles HL (*triloops, tetraloops, …*), dépendantes de leur séquence ;
- des pénalités pour l'initialisation (on dit aussi la « nucléation ») de chaque nouvelle boucle, en fonction de sa taille. Des valeurs tabulées existent pour les HL et IL de petite taille (<30 nucléotides). Pour des boucles de plus grande taille L, on prend la pénalité de nucléation la plus grande k et on lui ajoute une contre pénalité $k \times ln(L/30)$,
- une pénalité proportionnelle à l'asymétrie des boucles internes ;
- pour les boucles de taille 4 ou plus, une pénalité dite de « mismatch » dépendante de la séquence de l'appariement fermant la boucle et du premier couple de nucléotides ouvert (non appariés) à l'intérieur de la boucle suivant cet appariement. Des valeurs sont tabulées en fonction des séquences ;
- les ML sont modélisées par trois paramètres α_1, α_2, α_3 correspondant respectivement aux pénalités de nucléation (création du cycle dans le graphe), jonction (nombre d'hélices dans la ML) et nombre de nucléotides non appariés ;
- des pénalités d'empilement coaxial sont normalement utilisées mais pas dans notre modèle. En effet, elles représentent la stabilisation que la structure secondaire d'une boucle pourrait tirer des nucléotides non appariés mais empilés avec les hélices qui ferment la boucle. Il faut en général énumérer les configurations possibles et retenir celle de meilleure énergie pour connaitre la valeur de ce terme. Cependant, quand on a une vraie structure 3D, énumérer les configurations n'a pas de sens puisqu'on en a une et une seule, et la prise en considération de cette énergie d'empilement sera gardée pour le critère d'énergie en 3D.

Ces termes sont largement utilisés, dont les paramètres thermodynamiques sont proposés dans différentes publications (par exemple [16,227,347], voir le chapitre d'introduction pour une liste plus exhaustive). En plus de ces termes, le modèle de Nupack 3.2 [99] introduit de nouvelles pénalités $\beta_1, \beta_1^p, \beta_1^m, \beta_2$ et β_3 pour modéliser les pseudonoeuds

simples (de type H). Attention, Nupack a abandonné ce modèle à partir de la version 4.0. Ce modèle ne nous satisfait pas puisqu'il ne peut pas gérer tous les cas de pseudonoeuds, y compris de complexité élevée et plusieurs niveaux de récursion, qui pourraient être proposés lors de l'exploration de l'espace conformationnel. La plupart des modèles énergétiques des structures secondaires observés dans la littérature ne supportent pas les pseudonoeuds arbitraires. En effet, les outils correspondants cherchent à énumérer de façon exhaustive les structures secondaires possibles, par exemple à l'aide d'un schéma de programmation dynamique. Or, l'énumération exhaustive des structures avec pseudonoeuds arbitraires a été prouvée NP-difficile. Cependant, dans notre cas, on ne cherche pas à énumérer les structures secondaires, le programme d'optimisation se chargera de l'échantillonnage. Je propose donc une extension du modèle ci-dessous et illustré sur la Figure 5.1B.

- On conserve β_1 et β_1^m les pénalités d'initiation d'une nouvelle zone pseudo-nouée. Le terme β_1^m est utilisé quand le pseudonoeud est niché à l'intérieur d'une boucle existante, qui n'est alors plus modélisée comme la boucle qu'elle aurait été sans, mais comme un pseudonoeud elle aussi.
- Ces pénalités ne sont pas ajoutées une seule fois, mais une fois par niveau de pseudonoeud supplémentaire.
- On compte une pénalité β_2 par appariement fermant une boucle dans une zone pseudo-nouée. Ceci inclut les appariements extérieurs au pseudonoeud si celui-ci est niché dans une boucle (pour laquelle on ne comptabilisera plus les pénalités d'usage).
- On ne compte **pas** de pénalité β_3 par nucléotide non apparié dans un pseudonoeud, car l'intérieur et l'extérieur sont des concepts indéfinis pour un pseudonoeud arbitraire.

Ces choix sont assez subjectifs, ils reposent essentiellement sur la modification du modèle de Nupack pour pouvoir évaluer des structures à pseudonoeuds complexes.

On note que le problème de détermination de l'ordre d'un pseudonoeud (son nombre de niveaux) est NP-difficile [60,62,165,224]. On utilise donc une heuristique qui, en pratique, nous permet d'obtenir le nombre minimal de niveaux correct, dans la majorité des cas pour la minorité d'ARN présentant un pseudonoeud de grand ordre (typiquement, les pseudonoeuds observés n'ont jamais plus de 4 niveaux de pseudonoeuds). On procède de la façon suivante :

1. On commence avec un seul groupe d'appariements, vide.
2. On considère chaque nouvel appariement en partant du coté 5'-phosphate de la chaine. On vérifie s'il croise les appariements dans les groupes déjà existants.
3. S'il existe des groupes que cet appariement ne croise pas, on l'ajoute au groupe possédant déjà le plus grand nombre d'appariements. S'il y a égalité de taille, on tire au sort.
4. Sinon, on crée un nouveau groupe d'appariements et on l'ajoute dedans.
5. On retourne à l'étape 2 et on considère l'appariement suivant. S'il n'y a plus de nouveaux appariements, on compte le nombre de groupes d'appariements : c'est

l'ordre du pseudonoeud. On compte une nouvelle pénalité β_1 ou β_1^m pour chaque groupe supplémentaire en plus du premier groupe.

Une structure secondaire est annotée pour exemple sur la Figure 5.1B.

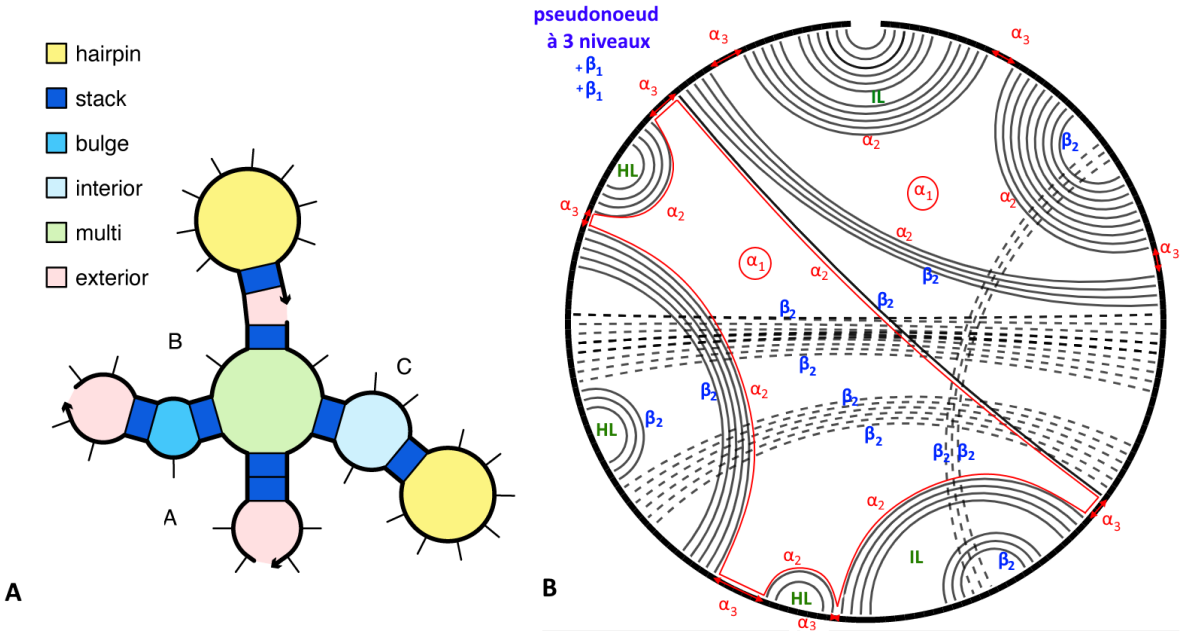


Figure 5.1 – Contributions à l'énergie d'une structure secondaire. *Seules les contributions des boucles sont représentées (il manque les énergies d'appariement, d'empilement, et d'autres pénalités).* ***(A)*** *Contributions classiques du modèle de Turner. L'exemple est un complexe d'ARN fictif.* ***(B)*** *Contributions dans notre modèle avec support des pseudonoeuds arbitraires, ici pour un intron du groupe I, issu de la structure 4P8Z-A. La structure secondaire est présentée sous forme de diagramme en cercle, les nucléotides étant disposés le long du trait noir, et les arêtes les reliant par l'intérieur s'ils sont appariés. La structure présente 3 niveaux de pseudonoeuds, car si l'on veut décomposer le graphe en groupes d'arêtes tels qu'aucune arête ne croise les autres au sein d'un même groupe, on a besoin de trois groupes. On note que l'appariement solitaire qui traverse la figure au centre est compté pour deux pénalités : α_2 pour fermer une boucle multiple à 5 jonctions (mise en valeur par un trait rouge), et β_2 pour fermer un pseudonoeud (de l'autre coté). Les α_1 entourés sont des pénalités qui ne portent pas sur un appariement en particulier, mais sur le cycle d'une boucle multiple.*

Figure A extraite de la documentation en ligne de Nupack 4.0 : https://piercelab-caltech.github.io/nupack-docs/definitions/, visité le 21/05/2021

On valide brièvement le modèle pour s'assurer que les modifications liées aux pseudonoeuds (ainsi que l'implémentation) ne s'écartent pas trop du modèle de Turner considéré standard. On compare les valeurs d'énergie retournées pour des ensembles de solutions générées par RNAsubopt [360] et par BiokoP [192]. On vérifie que les valeurs sont du même ordre de grandeur, et surtout que les structures secondaires sont classées dans le même ordre selon les deux modèles. Les résultats sont présentés sur la Figure 5.2, en A et B pour RNAsubopt et en C et D pour BiokoP. Sur les Figures 5.2A et 5.2C, on voit que dans la grande majorité des cas, les énergies sont corrélées. Elles sont parfois anti-corrélées, ceci

arrive quand l'ensemble des solutions contient deux solutions et qu'elles sont classées dans le mauvais ordre, on obtient un coefficient de corrélation proche de -1. Les Figures 5.2B et 5.2D confirment que le rang des solutions est fortement corrélé selon les deux modèles. On peut donc utiliser ce nouveau modèle dans une procédure de minimisation, les ordres de dominance seront préservés.

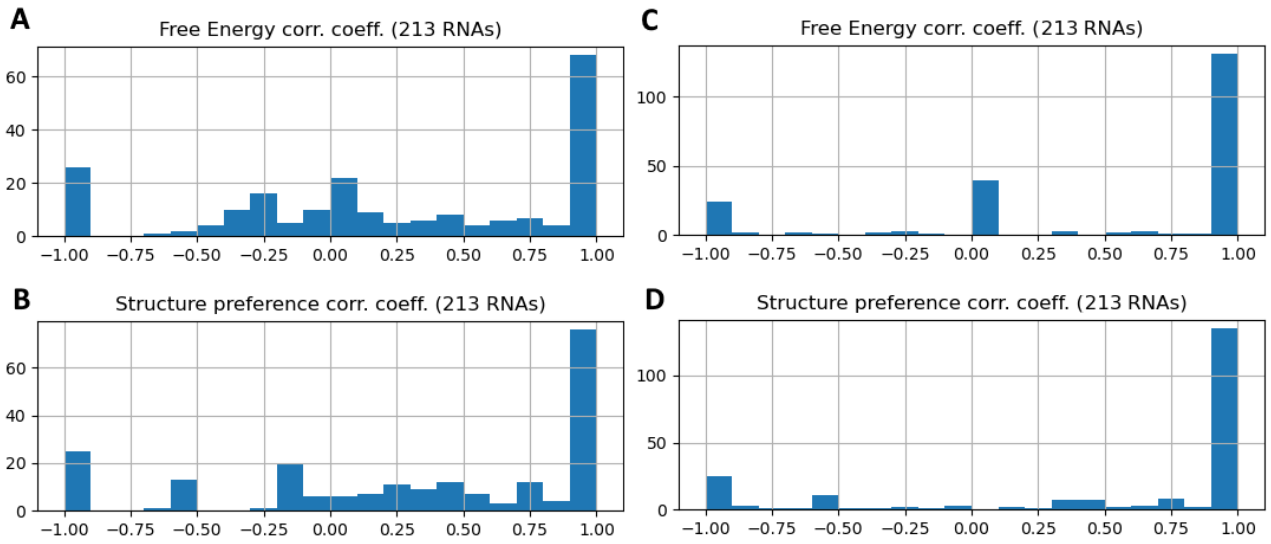


Figure 5.2 – Comparaison des modèles d'énergie de la structure secondaire. *Le benchmark est réalisé sur 384 structures secondaires vérifiables issues de RNAStrand 2.0, sans nucléotides modifiés et de longueur entre 20 et 1000 bases. Seuls 213 ARN parmi les 384 ont obtenu plusieurs solutions. Si dans les deux modèles, les structures proposées ont toutes la même énergie ou le même rang, le coefficient est fixé arbitrairement à 1. Si elles ont toutes la même énergie (ou le même rang) dans un seul des deux modèles, il est arbitrairement fixé à 0.* ***(A)*** *Distribution des coefficients de corrélation entre les valeurs d'énergie des structures sous-optimales d'un ARN prédites par RNAsubopt et les énergies de ces mêmes structures dans notre modèle.* ***(B)*** *Distribution des coefficients de corrélation entre les rangs des solutions de (A).* ***(C)*** *Distribution des coefficients de corrélation entre les valeurs d'énergie des structures proposées par le premier front de Pareto de BiokoP et leurs valeurs dans notre nouveau modèle.* ***(D)*** *Distribution des coefficients de corrélation entre les rangs des structures de (C).*

Ce critère utilise donc comme variables de décision les appariements (variables binaires, discrètes), mais la fonction peut être considérée comme continue. C'est aussi notre premier modèle physique, puisque sa paramétrisation repose sur des valeurs obtenues par des expériences de thermodynamique.

L'avantage de ce critère, c'est de permettre (théoriquement) l'accélération de la vitesse d'exploration du paysage énergétique par l'utilisation d'un move-set portant uniquement sur les appariements : en effet, une structure qui diffère d'un appariement a plus changé de conformation par rapport à l'originale qu'une structure dont une longueur ou un angle a changé. L'énergie de la structure secondaire est aussi un modèle robuste et largement utilisé par la plupart des autres méthodes. Les inconvénients du critère sont les

mêmes que dans les outils classiques de prédiction de structure secondaire : il ne tient pas compte des interactions non canoniques et a du mal à modéliser correctement l'entropie, l'exclusion du solvant, l'influence des ions, en particulier dans les structures contenant des pseudonoeuds, du fait d'une paramétrisation insuffisante. Néanmoins, on pense qu'il sera également très utile pour trier et rassembler les structures similaires à la fin de l'optimisation.

V.1.5 Modèle d'énergie 3D gros grains

Enfin, il nous faut une façon de discriminer finement les structures 3D entre elles pour renvoyer des solutions de qualité. Après avoir envisagé d'utiliser AMBER ou la fonction d'énergie de Rosetta, et devant la difficulté d'implémentation et leur complexité computationnelle, on décide de se tourner vers les modèles gros grains. Ils réduisent drastiquement le nombre de particules à simuler et le nombre de paramètres du modèle, en conservant les principaux degrés de liberté à modéliser. On se décide pour le modèle HiRE-RNA [72,73,228,253] (High-Resolution RNA), notamment sa formulation « v3 » [73]. C'est un modèle « haute résolution » avec 6 ou 7 billes par nucléotide (pour les pyrimidines et purines), ce qui lui permet de correctement modéliser les empilements et les interactions non-canoniques, avec jusqu'à trois interactions par base (une par coté). Un autre bon choix de modèle, suffisamment haute résolution et présentant les mêmes propriétés, aurait pu être le modèle « CG » de Xia *et al.* [362,363].

V.1.5.1 Présentation de HiRE-RNA

Le modèle HiRE-RNA résume les nucléotides par les atomes P, O_5, C'_5, C'_4, C'_1, plus un ou deux pseudo-atomes B_1 et B_2 (pour les purines seulement) placés au centre des cycles aromatiques. Pour les liaisons inter-nucléotidiques, l'atome de C'_4 est directement relié au P suivant. Les atomes de P et C'_1 étant toujours présents, on peut utiliser les pseudo-torsions eta' et theta' avec ce modèle. C'est un modèle qui utilise un solvant implicite, dont la présence influe sur le paramètre λ_D du potentiel de Debye. Pour pouvoir utiliser une valeur standard, on supposera que notre problème se pose en conditions « standard » de la biologie, à savoir une température de 37°C, des concentrations salines non nulles mais raisonnables, et une vitesse de déplacement du système faible ou nulle. La fonction d'énergie est une somme des termes suivants :

- des termes d'énergie harmoniques pour les longueurs des liaisons et les angles plans entre liaisons ;
- un terme pour la différence entre angles de torsion ;
- un terme exponentiel répulsif pour modéliser le volume exclu et empêcher l'interpénétration des atomes, remplacé par une sigmoïde décroissante dans les versions récentes (non publié) ;
- un potentiel de Debye-Hückel pour les interactions électrostatiques. Dans notre cas, on n'utilise pas d'ions ni de solvant explicitement, le potentiel est donc seulement utilisé pour modéliser la répulsion entre phosphates (entre atomes P) ;

- un terme stabilisant pour les nucléotides empilés, pondéré par trois facteurs pour prendre en compte la distance entre les nucléotides empilés, leur parallélisme, et leur déplacement latéral relatif ;
- un terme stabilisant pour les appariements, lui aussi pondéré par trois facteurs : le premier modélise la distance d'appariement, le second le parallélisme des bases appariées, le troisième leur alignement face à face. Chaque mode d'appariement possède des paramètres standard différents pour ces trois termes. Ainsi, la fonction d'énergie somme pour une paire de nucléotides donnés l'énergie qu'aurait la paire selon tous les modes d'appariement possibles. Puisque physiquement, les nucléotides n'interagissent selon qu'un seul mode (l'un des types d'interactions non-canoniques de Leontis-Westhof), un seul des termes de la somme aura une valeur conséquente, les autres seront très proches de zéro.

V.1.5.2 Détail de la fonction d'énergie

L'équation générale de la fonction d'énergie HiRE-RNA, ainsi que les différents types de paramètres qu'elle utilise, sont présentés ci-dessous :

$$E = \epsilon_b \sum_{distances\ d} k_b(d - d_0)^2 + \epsilon_a \sum_{angles\ \varphi} k_a(\varphi - \varphi_0)^2 + \epsilon_t \sum_{torsions\ \omega} k_t(1 + \cos(\omega - \omega_0))^2$$

$$+ \epsilon_{vol} \sum_{\substack{atom \\ pairs\ i,j}} b\left(1 - \frac{1}{1 + e^{-s\left(d_{ij} - d_{ij}^0\right)}}\right) + \epsilon_{el} \sum_{\substack{charged \\ pairs\ i,j}} \frac{q_i q_j e^{\frac{-d_{ij}}{\lambda_D}}}{4\pi\varepsilon_0\varepsilon_r d_{ij}}$$

$$- \epsilon_{emp} \sum_{\substack{nucleotide \\ pairs\ i,j}} \left[e^{-\frac{(r_{ij} - r_{emp})^2}{\sigma_{emp}}} \times \left(\vec{n}_i \cdot \vec{n}_j\right)^2 \times \left(1 - \left\|\vec{n}_i \times \vec{n}_{ij}\right\|^4\right) \times \left(1 - \left\|\vec{n}_j \times \vec{n}_{ij}\right\|^4\right) \right]$$

$$- \epsilon_H \sum_{\substack{nucleotide \\ pairs\ i,j}} \left[\varepsilon_{wc} k_H e^{-\frac{(\rho - \rho^0)^2}{\xi}} \times \cos(\alpha_i - \alpha_i^0)^4 \times \cos\left(\alpha_j - \alpha_j^0\right)^4 \right] \left[\varepsilon_{pl} e^{-\frac{\left(d_{Bi}^2 + d_{Bj}^2\right)}{\delta^2}} \right].$$

La notation d_{ij} renvoie à la distance entre les deux atomes i, j de la paire considérée, et les distances r_{ij}, ρ, d_B, les angles α et les vecteurs $\vec{n}$ sont définis graphiquement sur la Figure 5.3. Les paramètres en rouge sont dépendants de la séquence et/ou du type d'atomes ou de nucléotides considérés. Leurs valeurs sont tabulées. Ils sont donc estimés à partir de données, et font de HiRE-RNA un potentiel statistique. Les paramètres en vert ont été choisis à la main de façon à faire fonctionner le modèle, parfois avec une justification théorique, parfois par simple choix de modélisation. Les paramètres en bleu sont des constantes physiques. Enfin, les paramètres notés ϵ pondèrent les termes entre eux, et ont été fixés dans HiRE-RNA v3 par optimisation (par un algorithme génétique) [73].

L'équation commence avec les termes géométriques sur les longueurs, angles, et torsions. On ne considère ici que les longueurs, angles et torsions entre atomes reliés par des liaisons covalentes (contrairement aux termes suivants). L'énergie augmente de façon

quadratique par rapport à la distance à une valeur standard. Les « constantes de raideur » k_a, k_b et k_t permettent de pondérer les différentes distances et angles entre eux, pour imposer une contrainte plus forte sur certains par rapport à d'autres. L'utilisation d'un cosinus pour les torsions permet d'adoucir la forme de la courbe par rapport à une simple parabole.

Le terme de volume exclu n'est pas celui de l'article, où on trouvait une exponentielle décroissante. Récemment, l'équipe de Samuela Pasquali a préféré changer pour une sigmoïde décroissante, dont la hauteur initiale est b (la « barrière énergétique »), dont la raideur est fixée par s (*sharpness*) et dont l'emplacement du point d'inflexion est donné par d_{ij}^0.

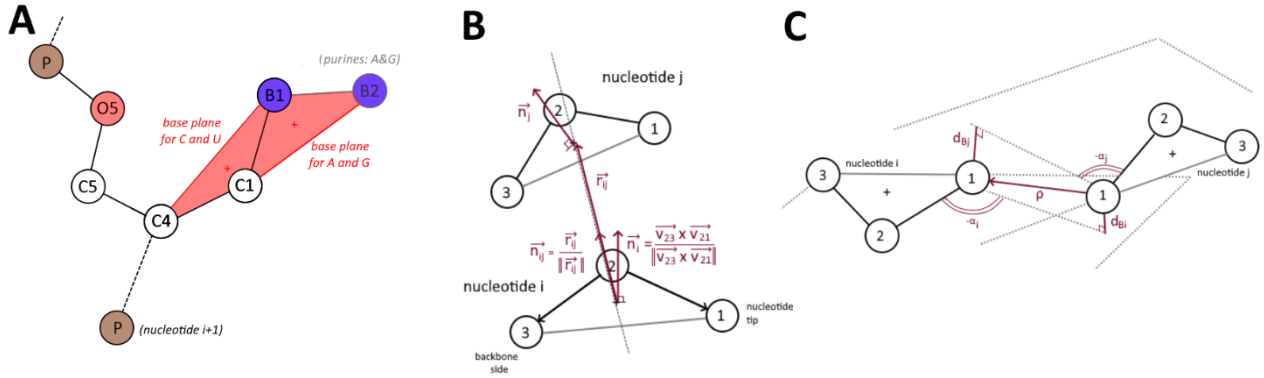


Figure 5.3 – Géométrie dans l'espace du modèle HiRE-RNA. ***(A)*** *Modèle gros-grains du nucléotide, résumé à 6 ou 7 billes. La chaine latérale forme un plan, qui peut être C4-C1-B1 pour les pyrimidines ou alors C1-B1-B2 pour les purines.* ***(B)*** *Vecteurs utilisés pour modéliser un empilement. Les billes des deux « plans » des nucléotides sont numérotés de 1 à 3 pour faire abstraction de la nature réelle des atomes, 3 étant le coté squelette et 1 l'apex de la chaine latérale. On définit les vecteurs unitaires normaux à ces plans $\vec{n}_i$ et $\vec{n}_j$, tout d'abord en calculant le vecteur normal par le produit vectoriel $\vec{v}_{2\to3} \times \vec{v}_{2\to1}$, puis en le normalisant, donc $\vec{n} = \frac{\vec{v}_{2\to3}\times\vec{v}_{2\to1}}{\|\vec{v}_{2\to3}\times\vec{v}_{2\to1}\|}$. On définit aussi le vecteur $\vec{r}_{ij}$ qui relie les centres de gravité des deux triangles, orienté de i vers j et de longueur r_{ij}, et sa version unitaire $\vec{n}_{ij} = \frac{\vec{r}_{ij}}{\|\vec{r}_{ij}\|}$.* ***(C)*** *Vecteurs et mesures pour modéliser un appariement. On utilise les mêmes plans des nucléotides, définis par les triangles précédemment présentés. On définit le vecteur $\vec{\rho}$ orienté de l'apex de la base j à l'apex de la base i, de longueur ρ. On note d_B la distance entre l'apex d'un nucléotide et sa projection orthogonale sur le plan du nucléotide partenaire. On définit encore α, l'angle entre $\vec{v}_{1\to2}$ et la projection orthogonale de $\vec{\rho}$ dans le plan du nucléotide. Cet angle est signé (ou encore compris entre 0 et 2π). Si l'angle de torsion 3_i-2_i-1_i-1_j est dans [-π/2, π/2], α_i sera dans [0, π], et inversement. Le même raisonnement s'applique au nucléotide j.*

Le terme de Debye-Hückel pénalise les phosphates de l'ARN qui seraient trop proches, selon un potentiel de Coulomb modifié par un terme faisant intervenir un paramètre appelé longueur de Debye (λ_D) qui englobe les propriétés diélectriques du solvant, notamment sa concentration en charges électriques. Grâce à ce paramètre, l'énergie devient une fonction

de la température et des concentrations ioniques du milieu, sans qu'on ait besoin de les représenter explicitement. On dit qu'on travaille en solvant et ions implicites. Je propose ci-après un récapitulatif personnel court de la théorie de Debye-Hückel, inspirée par le livre de Dill & Bromberg [365, chapitre 23] et Wikipédia [89], qui permet de comprendre d'où vient ce terme λ_D et surtout comment le modifier si besoin en fonction de la situation que l'on souhaite modéliser. Considérons les i espèces d'ions « implicites » de charge q_i et de concentration moyenne c_i^0 présents dans le solvant, par exemple le milieu intérieur d'une cellule vivante ou le plasma, ou tout autre milieu digne d'intérêt pour l'étude d'une molécule d'ARN. Leur concentration suit une distribution statistique de Boltzmann, donc à une température T donnée et à un endroit précis de coordonnées $\vec{x}$, la concentration locale $c_i(\vec{x}) = c_i^0 e^{-U_i(\vec{x})/k_B T}$, où $k_B = 1.38\times10^{-23}$ J/K est la constante de Boltzmann. Ensuite, l'énergie potentielle $U_i(\vec{x})$ de la particule chargée en $\vec{x}$ est donnée par le produit de la charge q_i de cette particule et du potentiel électrostatique local $V(\vec{x})$. On a donc $c_i(\vec{x}) = c_i^0 e^{-q_i V(\vec{x})/k_B T}$. Rajoutons maintenant au système une particule chargée A explicitement modélisée à la position $\vec{x}_A$, par exemple un atome de P dans une structure 3D d'ARN. Considérons alors l'équation de Poisson, qui permet d'exprimer le potentiel électrique $V(\vec{x})$ en fonction de la distribution spatiale $\rho(\vec{x})$ des charges. Dans notre cas, elle contiendra deux termes de densité des charges : le premier pour les concentrations locales des ions du solvant, et le second pour la particule A, dont la distribution est très localisée à l'aide d'un pic de Dirac $\delta_A(\vec{x})$ (qui vaut 1 en $\vec{x}_A$ et 0 ailleurs) :

$$\nabla^2 V(\vec{x}) + \frac{\rho(\vec{x})}{\varepsilon_0 \varepsilon_r} = 0$$

$$\nabla^2 V(\vec{x}) + \frac{\sum_i q_i c_i(\vec{x})}{\varepsilon_0 \varepsilon_r} + \frac{q_A \delta_A(\vec{x})}{\varepsilon_0 \varepsilon_r} = 0$$

avec ∇^2 l'opérateur Laplacien, ε_0 la permittivité diélectrique du vide, 8.85×10^{-12} F/m, et ε_r la permittivité diélectrique relative du solvant, 80.0 dans le cas de l'eau). En remplaçant les concentrations par leur expression, on obtient l'équation de Poisson-Boltzmann :

$$\nabla^2 V(\vec{x}) + \sum_i \frac{q_i c_i^0}{\varepsilon_0 \varepsilon_r} e^{\frac{-q_i V(\vec{x})}{k_B T}} + \frac{q_A \delta_A(\vec{x})}{\varepsilon_0 \varepsilon_r} = 0$$

Cette équation ne peut pas être résolue dans le cas général. Cependant, si le terme $q_i V(\vec{x})$ est très faible devant $k_B T$, une approximation par un développement de Taylor d'ordre 1 donne $e^{-q_i V(\vec{x})/k_B T} = \left(1 - \frac{q_i V(\vec{x})}{k_B T}\right)$. On peut donc écrire l'équation de Poisson-Boltzmann linéarisée, aussi appelée équation de Debye-Hückel :

$$\nabla^2 V(\vec{x}) + \sum_i \frac{q_i c_i^0}{\varepsilon_0 \varepsilon_r} - \sum_i \frac{{q_i}^2 c_i^0 V(\vec{x})}{\varepsilon_0 \varepsilon_r k_B T} + \frac{q_A \delta_A(\vec{x})}{\varepsilon_0 \varepsilon_r} = 0$$

La neutralité électrique globale de la solution suppose que $\sum_i q_i c_i^0 = 0$. On obtient donc :

$$\nabla^2 V(\vec{x}) - \left(\sum_i \frac{{q_i}^2 c_i^0}{\varepsilon_0 \varepsilon_r k_B T} \right) V(\vec{x}) + \frac{q_A \delta_A(\vec{x})}{\varepsilon_0 \varepsilon_r} = 0$$

Ce terme entre parenthèses est homogène à l'inverse d'une longueur au carré, la longueur de Debye, qu'on définit donc comme :

$$\lambda_D = \sqrt{\frac{\varepsilon_0 \varepsilon_r k_B T}{\sum_i {q_i}^2 c_i^0}}$$

On a alors $\nabla^2 V(\vec{x}) - \frac{1}{\lambda_D^2} V(\vec{x}) + \frac{q_A \delta_A(\vec{x})}{\varepsilon_0 \varepsilon_r} = 0$. Les solutions de cette équation ont la forme :

$$V(\vec{x}) = \frac{1}{4\pi \varepsilon_0 \varepsilon_r} \frac{q_A}{(\vec{x} - \vec{x}_A)} e^{\frac{-(\vec{x} - \vec{x}_A)}{\lambda_D}} + cte$$

Enfin, l'énergie potentielle d'une particule B de charge q_B à la position $\vec{x}_B$ (par exemple un autre atome de P) dans ce potentiel est toujours $U_B(\vec{x}_B) = q_B V(\vec{x}_B)$. Donc si on appelle d_{AB} la distance $\vec{x}_B - \vec{x}_A$, on trouve bien le terme de l'équation de HiRE-RNA :

$$U_B(\vec{x}_B) = \frac{q_A q_B}{4\pi \varepsilon_0 \varepsilon_r} \frac{e^{\frac{-d_{AB}}{\lambda_D}}}{d_{AB}}$$

Et en sommant ce terme pour toutes les paires d'atomes chargés A et B de la molécule, on obtient l'énergie potentielle électrostatique de la structure, tenant compte de la présence du solvant et de ses ions chargés. Une bonne valeur de λ_D à 37°C pour un milieu cellulaire (cytosolique) avec 0.15 mol/L d'ions potassium (K^+) et chlorure (Cl^-) est de l'ordre de 8 Angströms [93]. Enfin, puisque dans notre modèle de structure d'ARN, les seules charges explicitement modélisées sont les atomes de P (qui approximent la position des phosphates) portant une seule charge négative chacun, on aura toujours $q_i = q_j = -1$.

Le terme suivant modélise la contribution énergétique des empilements. On note le signe moins, les empilements étant stabilisants. C'est un produit de quatre facteurs. Le premier porte sur la distance entre les deux nucléotides, c'est une courbe de Gauss centrée sur la valeur standard r_{emp} et de « largeur » (d'écart-type) σ_{emp}. Le second mesure le parallélisme des plans des deux bases, et est égal au cosinus carré de l'angle entre les deux vecteurs normaux aux plans des deux bases. Enfin, les deux derniers termes modélisent le déplacement relatif d'un plan par rapport à l'autre. La fonction utilisée a une forme de « courbe en créneau » lissée, ce qui autorise un certain degré de liberté pour un plan de se déplacer en restant parallèle à l'autre plan.

Enfin le dernier terme modélise les appariements. C'est encore un produit de facteurs. Le paramètre ε_{pl} est redondant avec ϵ_H, et ε_{wc} redondant avec k_H, je les ai donc supprimés, et j'ai adapté la valeur de ϵ_H et k_H en conséquence. Le premier facteur modélise à nouveau une courbe de Gauss dont la variable est la distance entre les pointes des nucléotides appariés. La constante de couplage k_H donne la « force » de l'appariement. Historiquement, elle était reliée au nombre de liaisons hydrogènes présentes dans l'appariement. Le terme ε_{wc} était utilisé pour favoriser arbitrairement les appariement canoniques (Watson-Crick).

Chaque type d'appariement non-canonique a désormais une constante k_H spécifique, et dépendante de la nature des bases interagissant. Ensuite, les deux termes en cosinus à la puissance atténuent rapidement la force d'appariement si les nucléotides ont une mauvaise orientation. Enfin, le dernier facteur atténue la force d'appariement si les nucléotides ne sont pas bien alignés.

Dans un récent article, un terme supplémentaire a été ajouté pour prendre en compte des contraintes supplémentaires fournies par des données expérimentales SAXS [228], mais nous ne l'utiliserons pas. Nous intégrerons les données expérimentales par un autre objectif distinct du critère d'énergie de la structure 3D.

V.1.5.3 Difficultés et critique du modèle original

Dans l'état, et même après un échange avec S. Pasquali qui nous a gracieusement donné accès à un document technique traitant de détails d'implémentation ainsi qu'une portion conséquente de leur code source, plusieurs points restent incompris pour moi.

Tout d'abord, la comparaison des plans formés par les nucléotides sans distinction des purines et pyrimidines alors que les atomes qui les définissent ne sont pas les mêmes m'interroge. En pratique les plans des pyrimidines modélisent plus l'orientation du ribose que de la base azotée. Pour les empilements purine-pyrimidine, on modélise donc le parallélisme de deux parties du nucléotide qui ne correspondent pas. On aurait pu par exemple utiliser le plan défini par C4-C1-B1 partout, et compter sur la rotation libre des bases azotées pour se positionner correctement entre elles si les sucres sont bien alignés. Cette hypothèse est plausible, et le modèle comparerait des plans réellement comparables.

Ensuite, le terme $\epsilon_t k_t(1+\cos(\omega-\omega_0))$ pour modéliser l'énergie des torsions me paraît inadapté. En effet, sur l'intervalle [-π, π], le terme $1+\cos(x)$ admet un maximum en $x=0$, c'est-à-dire que l'énergie est maximale lorsque l'angle de torsion possède précisément la valeur souhaitée. La fonction pénalise donc les angles de torsion ayant la bonne valeur. En effet, les paramètres ϵ_t et k_t publiés sont bien positifs ; selon les angles, $\epsilon_t = 0.1, k_t \in [0,20]$ ([73]), l'expression est donc toujours positive ou nulle, et donc pénalisante. Dans la version originale d'HiRE-RNA (2010), le terme $\epsilon_t k_t$ était défini comme « *V_n/2, the energy barrier* » [253], et donc potentiellement une valeur négative, mais depuis que ce paramètre est obtenu par optimisation, il est positif. On pourrait plutôt préférer l'utilisation de la forme $\epsilon_t k_t(1-\cos(\omega-\omega_0))$, dont la courbe ressemble à un « puits d'énergie » classique. D'ailleurs, le choix de ce terme par rapport aux potentiels harmoniques utilisés pour les distances et angles n'est pas non plus défendu [253]. Le choix du calcul trigonométrique, computationnellement lent à exécuter, est-il justifié ?

Enfin et surtout, les choix de modélisation pour les termes de volume exclu ne sont pas entièrement compris. Dans la première version, aucun terme n'est présenté. Rien n'indique comment le modèle obtient des structures physiquement plausibles sans interpénétration atomique. À partir de la version 3 de 2015, un terme répulsif à courte

distance est introduit, de la forme $e^{-\kappa(d_{ij}-d_{ij}^0)}$, ce qui semble raisonnable. Cependant, la paramétrisation utilisée, avec $\kappa = 4.0$ et des d_{ij}^0 de 3.2 à 4.0 Angströms, perturbe les potentiels harmoniques pour les distances entre atomes, ce qui rend les structures toutes « étirées » et irréalistes. En effet, les distances standard des liaisons sont de l'ordre de 1.5 à 2.5 Angströms environ, et le potentiel de volume exclu pénalise exponentiellement toute distance inférieure à 4.0 Angströms. Ce phénomène est illustré sur les Figure 5.4A et 5.4D. Sur la Figure 5.4A, on devrait voir des paraboles dont le minimum a pour valeur d'énergie zéro, à l'abscisse indiquée dans la légende pour chaque courbe. Cependant, les paraboles sont perturbées par le terme énergétique de répulsion (Figure 5.4D) : toute liaison P-O_5 se voit donc attribuer une pénalité énergétique minimale de 220 kcal/mol, par exemple. Samuela Pasquali nous a ensuite indiqué que son équipe utilise maintenant une sigmoïde pour ce terme, selon l'équation présentée plus haut. Mais l'utilisation d'une sigmoïde ne change pas le problème si le point d'inflexion est maintenu entre 3.2 et 4.0 Angströms : les minima des paraboles de la Figure 5.4A auront la valeur b quand ils devraient avoir la valeur zéro.

De façon plus générale, je ne comprends pas la paramétrisation. Les ordres de grandeur utilisés me semblent disproportionnés. Les paramètres tels que publiés dans l'article de 2015 [73] sont illustrés sur la Figure 5.4. Les termes d'énergie sont représentés pondérés par les divers poids ϵ. On remarque que les contributions stabilisantes, notamment les empilements et appariements (Figures 5.4E et 5.4F), sont très faibles devant les variations d'angles et de longueurs. Les torsions et la répulsion électrostatique n'ont quasiment aucun effet. Pourtant, ce sont les empilements et appariements qui dirigent la stabilité d'une structure 3D : il est tout à fait plausible d'observer une ou plusieurs déviations d'angles de 20° par rapport à la valeur moyenne si cela permet de former un appariement et/ou un empilement. La paramétrisation actuelle ne le permet pas du tout. Les paramètres utilisés dans les fichiers transmis par S. Pasquali avaient évolué depuis 2015, ils sont illustrés sur la Figure 5.5. Cela donne une légère amélioration, mais les mêmes constats peuvent toujours être faits.

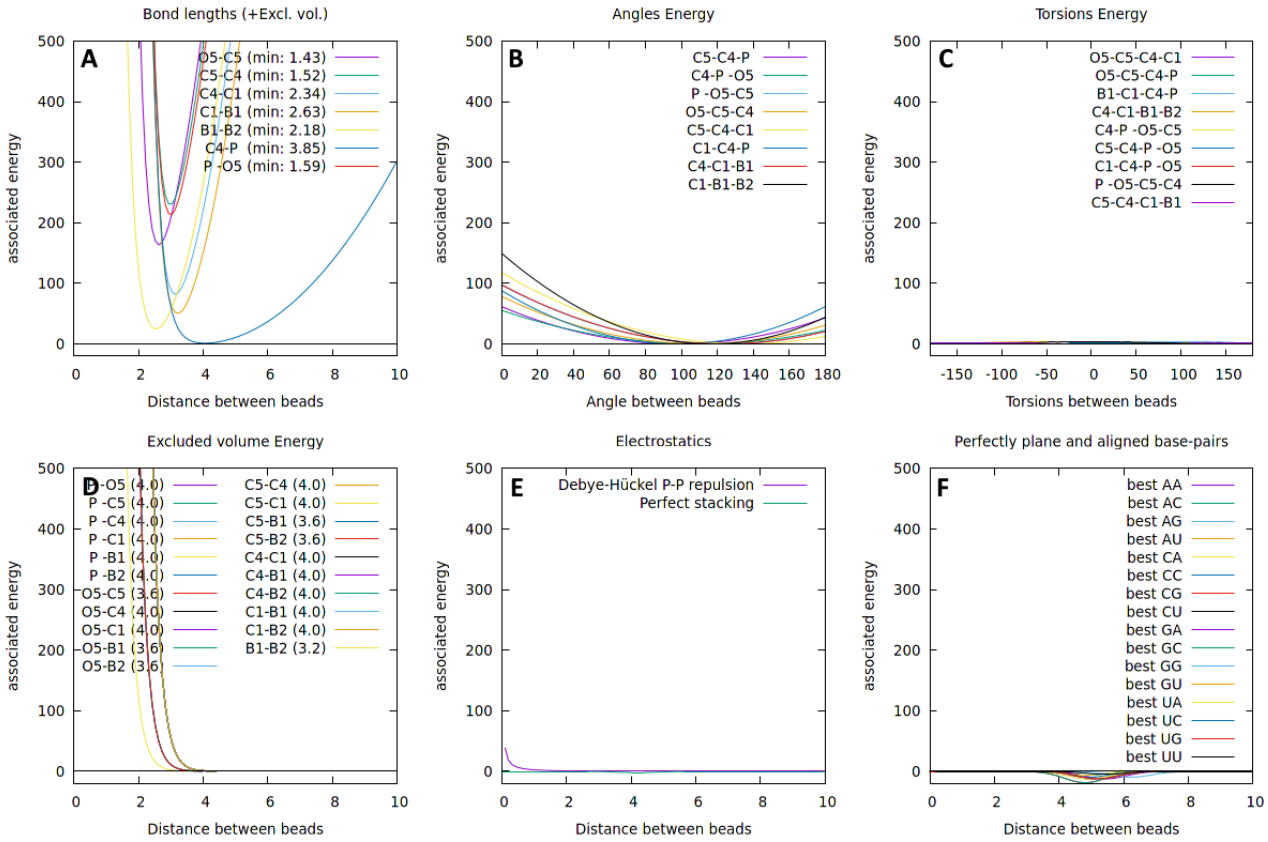


Figure 5.4 – Paramètres énergétiques originaux publiés de HiRE-RNAv3. *Energie associée* ***: (A)*** *à chaque longueur de liaison,* ***(B)*** *à chaque angle plan,* ***(C)*** *à chaque angle de torsion,* ***(D)*** *à la distance entre paires d'atomes quelconques (non reliés), selon le terme publié dans l'article de 2015* [73]*, une exponentielle décroissante,* ***(E)*** *à la répulsion électrostatique et aux empilements (considérés parfaitement alignés).* ***(F)*** *aux appariements considérés parfaitement alignés (meilleure énergie possible parmi les modes d'appariement).*

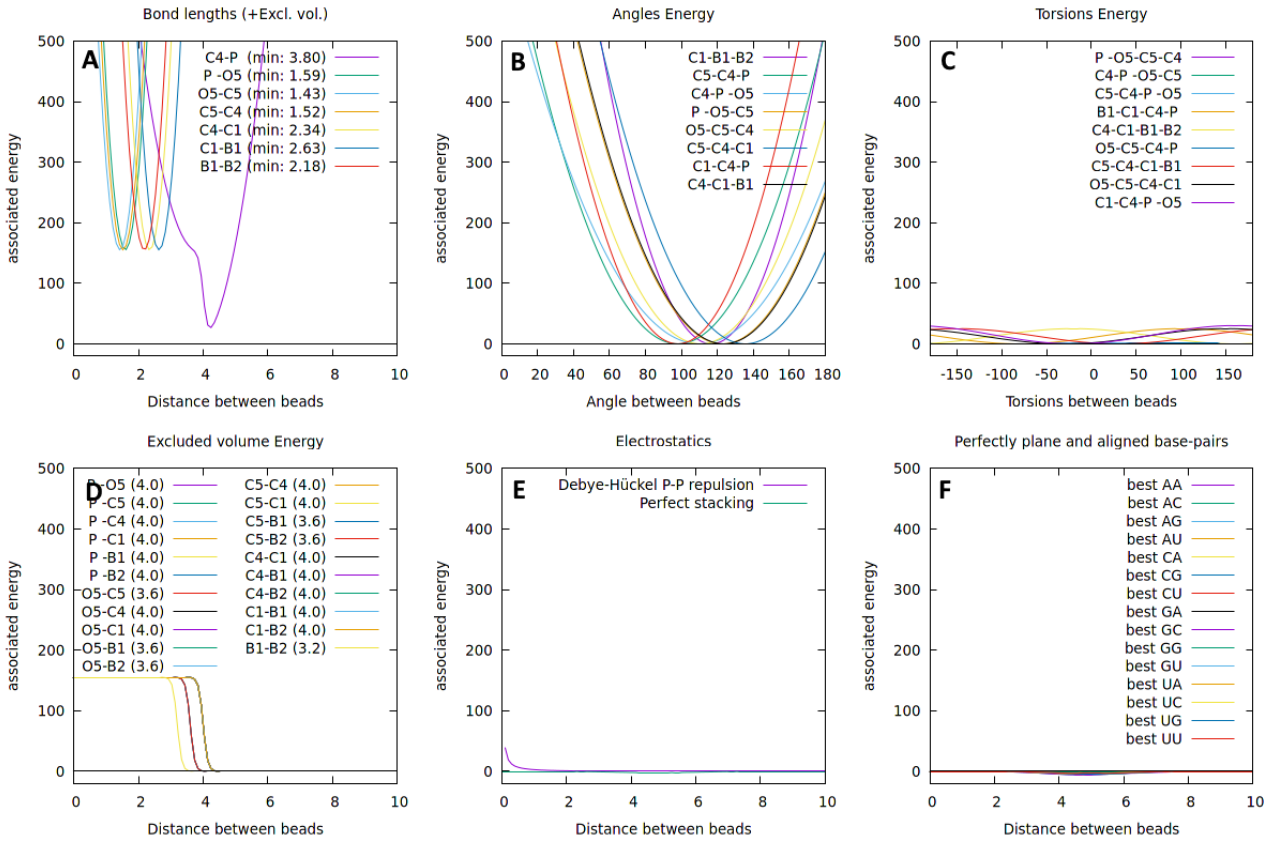


Figure 5.5 – Paramètres énergétiques actuels de HiRE-RNA. ***(A,B,C,D,E,F)*** *Mêmes termes énergétiques que sur la Figure 5.4, à l'échelle. On note cette fois l'utilisation d'un terme sigmoïdal pour l'exclusion de volume en (D), ce qui change la forme des courbes en (A), mais ne résout pas le problème. La paramétrisation présentée ici vient du code source de HiRE-RNA transmis en interne et n'est pas publique ni publiée.*

D'un point de vue purement pratique, on regrette l'absence d'un service qui aurait permis de comparer mon implémentation à l'implémentation originale. HiRE-RNA ne dispose malheureusement pas d'un exécutable public, ni d'un code source complet et compilable, ni d'un webservice permettant de soumettre une séquence d'ARN pour simulation. Il ne m'est donc pas possible de comparer les modèles obtenus par mon implémentation avec l'implémentation originale, ce qui aurait permis de savoir si les défauts identifiés viennent de mon code, ou d'un problème plus général de modélisation ou paramétrisation. A ce stade, je l'ignore toujours, des discussions sur ce sujet sont prévues avec S. Pasquali pour essayer d'identifier le(s) problème(s).

Mon implémentation de HiRE-RNA ne donnant pas de solutions physiquement réalistes (voir Figure 5.6), j'ai donc modifié le modèle officiel (et publié) jusqu'à obtention de structures plausibles avec mon implémentation. Ce faisant, j'ai abandonné malheureusement la fiabilité apportée par les publications passées de HiRE-RNA, sur laquelle j'aurais aimé pouvoir compter.

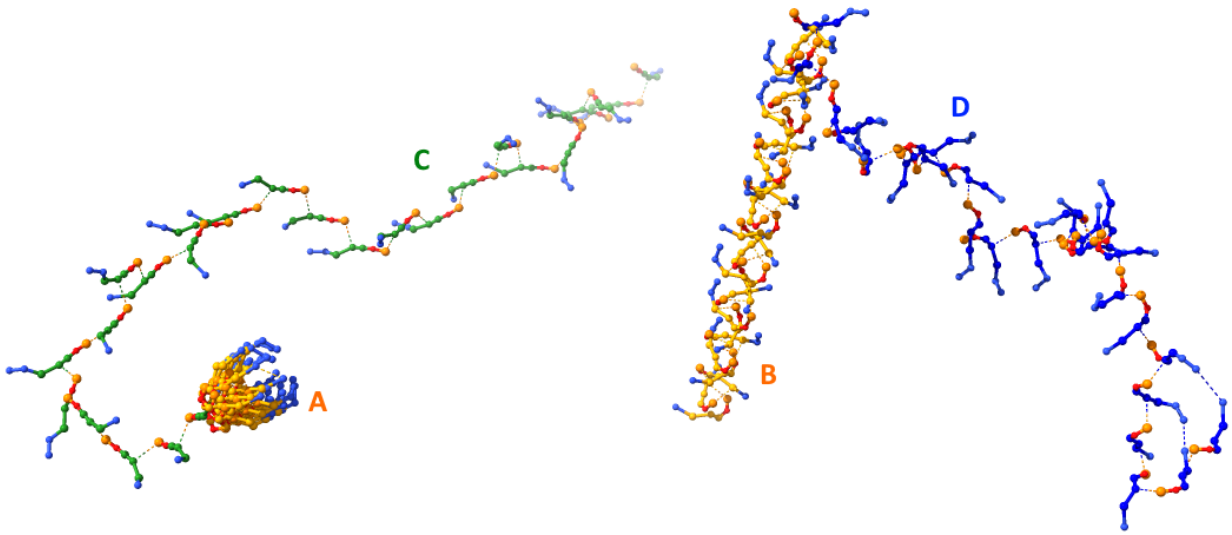


Figure 5.6 – Chaines obtenues avec les modèles HiRE-RNA originaux. *La séquence d'ARN simulée est fictive et aléatoire. Les 6 ou 7 billes du modèle HiRE-RNA sont représentées reliées par des liaisons covalentes, et les liaisons internucléotides entre C_4 et P par des pointillés. Les chaines A et B sont des chaines obtenues par génération artificielle en utilisant les angles et longueurs standard, et servant de point de départ à la simulation. Elles ne sont évidemment pas physiquement plausibles (c'est normal). Les chaines C et D sont des résultats de simulation obtenus à partir de A et B après 10^4 itérations du programme d'optimisation. Le programme est introduit en section 5.3, mais ces résultats sont présentés ici pour comparer les effets des paramètres sur la structure obtenue. Les chaines A et C (en vert) sont obtenues avec le modèle publié de l'article de 2015 : les atomes sont soumis à une forte contrainte par le terme de volume exclu et cherchent à s'éloigner au maximum les uns des autres : on obtient une structure effilée peu réaliste. Les chaines B et D utilisent les paramètres actuels de HiRE-RNA non publiés. Le résultat est physiquement plus plausible, mais même après un long temps de simulation, on n'obtient pas d'appariements ou empilements, car leurs contributions énergétiques sont trop faibles. La solution D présente une petite boucle fermée par deux appariements en bout de chaine, mais cette structure a été retenue aussi grâce au critère d'énergie de la structure secondaire, d'où sa présence. Elle n'aurait pas forcément été conservée dans un programme d'optimisation mono-objectif. Ces aspects seront rediscutés plus loin dans la section suivante.*

V.1.5.4 Re-paramétrisation du modèle

Je propose donc une paramétrisation personnelle du modèle, portant attention aux points suivants :

1. Un rééquilibrage des poids ϵ permettant une meilleure cohérence des termes entre eux, et notamment un effet réel des empilements et appariements,
2. La restauration de l'exponentielle décroissante pour le volume exclu, mais en utilisant des distances limites bien plus faibles, de façon à ne pas perturber l'emplacement des minima des paraboles pour le terme sur les longueurs de liaisons,
3. Des valeurs standard $d_0, \varphi_0, \omega_0, \rho^0, \alpha^0$ estimées statistiquement à partir de RNANet.

Les nouveaux poids ϵ ont été choisis à la main de façon arbitraire (appréciation subjective). On note aussi la modification du terme des torsions en $\epsilon_t k_t(1-\cos(\omega-\omega_0))$, de façon à appliquer une pénalité (contribution positive) lorsque l'angle s'éloigne de la valeur standard. Les profondeurs des puits d'énergie des empilements et appariements ont été choisies itérativement, pour voir apparaitre des structures secondaires intéressantes relativement vite dans la simulation que nous présenterons en section V.3 (au bout de quelques milliers d'itérations). La longueur de Debye utilisée est de 7.90 Angströms. C'est le seul paramètre dont la valeur est obtenue par le calcul, et malgré tout, on voit qu'encore une fois son action sera minime puisque la pénalité de volume exclu reste active plus rapidement et à plus longue distance que la répulsion électrostatique. Cette nouvelle paramétrisation est dont encore fortement imparfaite et mériterait une conception plus rationnelle. La nouvelle paramétrisation est illustrée sur la Figure 5.7.

Le paramètre ξ représente la variance de la courbe de Gauss modélisant la distance entre les nucléotides appariés. L'utilisation de la valeur par défaut publiée, 0.404, ne nous permettait pas de retrouver de façon fiable les appariements canoniques tels que détectés par DSSR [208] à partir d'une même structure 3D. En utilisant l'annotation DSSR comme référence, on explore la valeur du paramètre jusqu'à obtenir une détection fiable et satisfaisante. Les performances sont récapitulées dans la Table 6. On se décide donc pour la valeur $\xi = 0.7$.

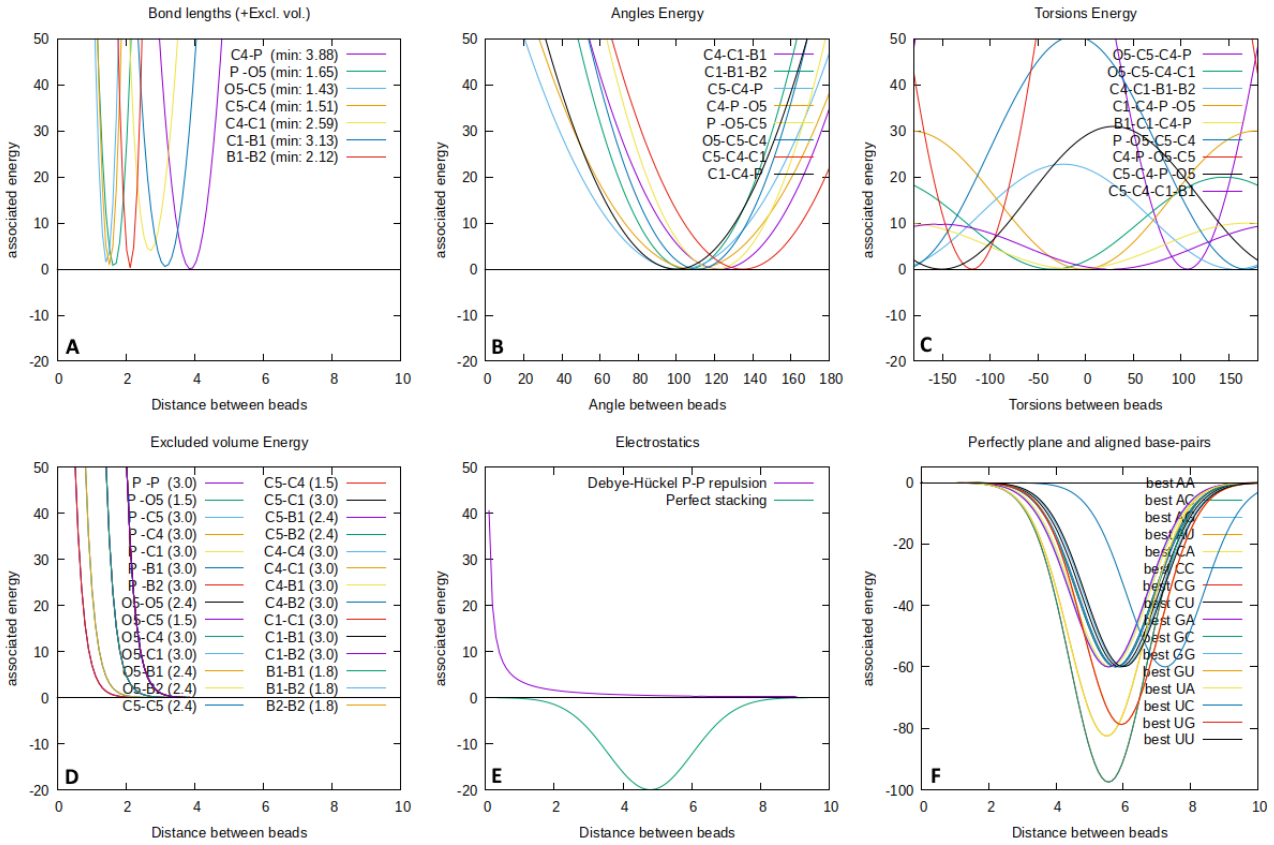


Figure 5.7 – Révision personnelle des paramètres du modèle HiRE-RNA. *(A,B,C,D,E,F)* *Mêmes termes énergétiques que dans les Figures 5.4 et 5.5, mais l'échelle n'est plus la même. On note l'inversion du terme (1+cos²) en (1-cos²) en (C), et le rétablissement de l'exponentielle décroissante en (D).*

ξ	Exactitude	Sensibilité	VPP	F1-score	MCC
0.404 (originale)	0.97	0.65	0.91	0.75	0.75
0.5	0.99	0.90	0.93	0.92	0.91
0.6	0.99	0.94	0.94	0.94	0.93
0.7 à 1.5	1.0	1.0	0.94	0.97	0.97

Table 6 - Performance de détection des appariements canoniques. *Métriques de performance obtenues pour différentes valeurs de ξ en comparant la liste des appariements obtenus par mon implémentation comparée à la liste de référence obtenue par DSSR, pour l'annotation d'une même structure 3D (4l81, chaine A). L'expérience est répétée (à la main) pour plusieurs valeurs de ξ jusqu'à obtention d'une performance satisfaisante. On voit qu'à partir d'une certaine valeur $\xi = 0.7$, la performance n'augmente plus.*

L'ensemble de paramètres obtenu permet d'observer des structures physiquement plausibles, comme il en sera présenté dans la section V.3.4. Je n'ai pas implémenté d'algorithme de dynamique moléculaire (ou même d'algorithme d'optimisation monocritère) utilisant la fonction d'énergie 3D seule pour valider ce modèle. On évaluera la performance de la fonction au sein du modèle multicritère, dont elle est un élément clé, en section V.3.5. avec un petit benchmark sur des structures de RNA-Puzzles [238].

V.2 Algorithme d'optimisation multicritère

Maintenant que les critères sont définis, il convient de formuler le problème sous une forme « standard », c'est-à-dire de définir quelles sont les variables, quels sont leurs domaines possibles à explorer, par quelles contraintes d'égalité et d'inégalité sont-elles reliées, etc. On s'attaque à un problème difficile, en effet, certains des critères s'expriment en fonction de variables réelles (des distances et des angles) quand d'autres utilisent des variables discrètes (et même binaires : les appariements). On considère donc un problème d'optimisation multicritère, non-linéaire, mixte.

V.2.1 Formalisation du problème

On a deux principales stratégies candidates :

- soit on utilise toutes les variables des fonctions objectifs comme variables de décision, c'est-à-dire d'une part les coordonnées internes de la structure, des variables continues : longueurs, angles, torsions (6n-6 variables pour n atomes). D'autre part, les appariements, des variables binaires (jusqu'à N/2 variables pour N nucléotides). On formule ensuite des contraintes limitant les valeurs des coordonnées internes en fonction de la valeur des appariements pour que le tout soit cohérent. On calcule, en plus, des variables dérivées : les distances entre résidus et entre atomes non reliés par des liaisons covalentes ;
- soit on n'utilise uniquement que les coordonnées internes continues comme variables de décision, et on considère les appariements non plus comme des variables de décision mais comme des « conséquences » des coordonnées internes, c'est-à-dire des variables dérivées comme les longueurs inter-résidus. Cette approche empêche l'exploration rapide des conformations par des mouvements directs dans l'espace des structures secondaires, agissant sur la structure globale. En revanche, le problème d'optimisation n'est plus mixte : il est seulement à variables continues.

Pour choisir, tentons d'abord d'expliciter la forme des contraintes reliant les appariements et les coordonnées internes.

Rappelons les noms des variables utilisées. On appelle d_i les distances entre atomes reliés par des liaisons covalentes (pour n atomes, il y a $n-1$ variables d_i). On appelle φ_i les angles plans entre triplets d'atomes reliés par des liaisons covalentes, et ω_i les angles de torsions entre quadruplets d'atomes reliés ($n-2$ variables φ_i et $n-3$ variables ω_i). On note d_{ij} la distance entre deux nucléotides i et j, plus précisément entre les atomes C_1' de ces nucléotides. On note y_j^i la variable binaire indiquant si les nucléotides i et j sont appariés.

Définissons y_j^i en fonction de la position des atomes des nucléotides i et j concernés. Pour que y_j^i puisse prendre la valeur 1 (pour qu'il y ait appariement), il faut :

- Que la séquence des nucléotides concernés corresponde à un appariement canonique, c'est-à-dire qu'on ait GC, AU ou GU.

- Que le mode d'appariement corresponde à l'appariement cis-Watson-Watson (cWW). Ceci implique des valeurs de distance et angles de référence pour ρ^0, α_i^0 et α_j^0 . Il faut donc que l'énergie de l'appariement soit suffisamment grande (suffisamment négative) en utilisant ces valeurs standard pour l'appariement cWW. On choisit un seuil d'énergie de -10^{-10} unités (~kcal/mol) pour le cas général, en étant plus exigeant si les nucléotides sont espacés de moins de 10 positions ($j < i + 10$), jusqu'à un seuil de -10^{-5} pour les nucléotides espacés de seulement 2 positions. L'objectif est de fixer un seuil très faible à grande distance pour arriver à retenir les solutions avec un appariement longue distance en formation.

Les contraintes sur la séquence sont assez faciles à gérer : il suffit de ne définir les variables y_j^i que pour les positions (i, j) compatibles. On peut alors définir les contraintes :

1. $\forall (i,j) \in [\![1, N]\!]^2,\ j - i \in [2,10]$ tel que $\{s[i], s[j]\} \in \big\{\{G,C\},\{G,U\},\{A,U\}\big\}$:

$$\left[\frac{-1}{10^{-(5/8(j-i)+15/4)}} \times \varepsilon_{wc}\varepsilon_{pl}k_H e^{-\frac{(\rho-\rho^0)^2}{\xi}} \cos(\alpha_i - \alpha_i^0)^4 \cos\big(\alpha_j - \alpha_j^0\big)^4 e^{-\frac{\left(d_{Bi}^2 + d_{Bj}^2\right)}{\delta^2}} - 1\right] y_j^i \geq 0$$

2. $\forall (i,j) \in [\![1, N]\!]^2,\ j - i > 10$ tel que $\{s[i], s[j]\} \in \big\{\{G,C\},\{G,U\},\{A,U\}\big\}$:

$$\left[-10^{10}\varepsilon_{wc}\varepsilon_{pl}k_H e^{-\frac{(\rho-\rho^0)^2}{\xi}} \cos(\alpha_i - \alpha_i^0)^4 \cos\big(\alpha_j - \alpha_j^0\big)^4 e^{-\frac{\left(d_{Bi}^2 + d_{Bj}^2\right)}{\delta^2}} - 1\right] y_j^i \geq 0$$

Cette formulation est simple, mais elle ne s'exprime pas encore uniquement en fonction des variables d_i, α_i et ω_i. Il nous faut réexprimer les variables d_{Bi}, d_{Bj}, α_i et ρ en fonction des d_i, φ_i et ω_i. Définissons, pour tout nucléotide i, $\vec{v}_{i1}, \vec{v}_{i2}$ et $\vec{v}_{i3}$ les trois vecteurs de coordonnées cartésiennes (x, y, z) des trois dernières billes du nucléotide i (en partant de l'extrémité de la base). Définissons $\vec{v}_{i32} = \vec{v}_{i3} - \vec{v}_{i2}$ et $\vec{v}_{i12} \equiv \vec{v}_{i1} - \vec{v}_{i2}$, puis $\vec{n}_i = \frac{\vec{v}_{i32} \times \vec{v}_{i12}}{\|\vec{v}_{i32} \times \vec{v}_{i12}\|}$, on a alors les expressions suivantes (illustrées précédemment sur la Figure 5.3C) :

- $\vec{\rho} = \vec{v}_{i1} - \vec{v}_{j1}$ et $\rho = \|\vec{\rho}\|$,
- $d_{Bi} = -\vec{\rho} \cdot \vec{n}_i$ et $d_{Bj} = -\vec{\rho} \cdot \vec{n}_j$,
- $\vec{\rho}_i = -\vec{\rho} + d_{Bi} \cdot \vec{n}_i$ et $\vec{\rho}_j = \vec{\rho} + d_{Bj} \cdot \vec{n}_j$,
- enfin, $\alpha_i = \cos^{-1}\left[\frac{\vec{v}_{i32}}{\|\vec{v}_{i32}\|}\frac{\vec{\rho}_i}{\|\vec{\rho}_i\|}\right]$ et $\alpha_j = \cos^{-1}\left[\frac{\vec{v}_{j32}}{\|\vec{v}_{j32}\|}\frac{\vec{\rho}_j}{\|\vec{\rho}_j\|}\right]$.

En remplaçant chaque variable par sa valeur dans la liste ci-dessus, on peut donc bien exprimer la contrainte en fonction des coordonnées cartésiennes des nucléotides impliqués, $\vec{v}_{i1}, \vec{v}_{i2}, \vec{v}_{i3}, \vec{v}_{j1}, \vec{v}_{j2}$, et $\vec{v}_{j3}$. Maintenant, montrons que ces coordonnées peuvent être exprimés en fonction des coordonnées internes d_i, φ_i et ω_i. En effet, pour toute bille b de coordonnées cartésiennes $\vec{v}_b$, on a :

$$\vec{v}_b = \vec{v}_{b-1} + \begin{bmatrix} \vec{m}_b & \vec{n}_b \times \vec{m}_b & \vec{n}_b \end{bmatrix} \cdot \begin{pmatrix} -d_b \cos\varphi_b \\ d_b \sin\varphi_b \cos\omega_b \\ -d_b \sin\varphi_b \sin\omega_b \end{pmatrix}$$

avec $\vec{m}_b = \vec{v}_{b-1} - \vec{v}_{b-2}$ et $\vec{n}_b = \vec{m}_{b-1} \times \vec{m}_b$. Ainsi, en calculant cette expression récursive jusqu'à avoir une fonction de $\vec{v}_0 = (0,0,0)$, il est mathématiquement possible d'exprimer

$y_j^i = f(d, \varphi, \omega)$. En revanche, cette expression fait appel à toutes les coordonnées internes des atomes des nucléotides j et précédents. Si la décomposition de la relation en plusieurs étapes comme nous l'avons fait est possible, il est extrêmement difficile de formuler une seule inéquation pour chaque appariement à la main. Il n'est pas exclu qu'un ordinateur puisse le faire, mais nous n'avons pas creusé cette piste. De plus, la fonction réciproque $d, \varphi, \omega = f^{-1}(y_j^i)$, qui permettrait de contraindre les coordonnées internes lorsque l'on modifierait un appariement, n'a pas de formulation.

On n'est donc pas en mesure d'utiliser un solveur automatique, ces solveurs nécessitant une formulation propre des contraintes à l'aide d'une fonction. Pour commencer à étudier le problème, on se propose de considérer le cas simple :

- on considèrera uniquement des variables continues comme variables de décision, les coordonnées internes, les autres seront des variables « intermédiaires » dérivées,
- on utilisera un algorithme simple d'implémentation, un algorithme génétique, facilement adaptable aux problèmes non linéaires à contraintes non linéaires.

V.2.2 Algorithme d'optimisation

On a donc décidé d'utiliser un algorithme génétique. Pour rappel, un algorithme génétique utilise une population de solutions, une façon d'effectuer des « croisements » entre ces solutions pour en créer de nouvelles, et une façon de les faire évoluer par « mutation ». La population est soumise à chaque génération à un processus de sélection (mimant la sélection naturelle), qui conduira à la mort de certains individus et à la survie des plus adaptés et/ou des plus chanceux selon la procédure, l'adaptation étant mesurée par la valeur de la fonction de coût (ou son opposé).

Dans le cas multicritère avec plus de deux objectifs, l'un des algorithmes de référence est NSGA-III (*Non-dominated Sorting Genetic Algorithm III*) [87]. C'est un algorithme dont la sélection se fait sur la base du rang de Pareto, c'est-à-dire que les solutions les moins dominées sont jugées les plus efficaces, et qu'au sein d'un front de Pareto de même rang, toutes les solutions sont aussi efficaces. Ceci n'est pas vrai pour le dernier front considéré (de plus haut rang) : NSGA-III favorise la diversité des solutions en augmentant l'efficacité des solutions proches de points de référence dans des régions faiblement peuplées en solutions. Ces points de référence sont échantillonnés régulièrement espacés dans la CHIM normalisée (voir détails plus bas). Le choix de NSGA-III plutôt qu'un autre algorithme vient initialement de la performance des méthodes NBI [82] (et autres variantes basées sur la CHIM) observée par Burachik *et al.* [46] pour les problèmes mixtes. À la recherche d'un algorithme génétique dans lequel la diversité des solutions serait encouragée grâce à des points de référence échantillonnés dans la CHIM, j'ai trouvé NSGA-III. Cependant, puisque l'on n'utilise plus de variables discrètes comme variables de décision directement, on aurait pu utiliser d'autres algorithmes évolutionnaires multi-objectifs. NSGA-III reste cependant l'une des références connues et citées.

Chaque individu de la population est une structure 3D, définie par les valeurs de ses coordonnées internes d, φ, ω, à partir desquels ont peut calculer les y_j^i. La population initiale de taille N (hyperparamètre modifiable par l'utilisateur) est composée de structures identiques, linéaires, générées avec les valeurs moyennes des longueurs, angles, et torsions pour le modèle HiRE-RNA. La mutation permet une modification d'une ou plusieurs de ces coordonnées internes à des positions aléatoires, d'un pas également aléatoire tiré entre -1.0 et +1.0 Angströms pour les longueurs, et entre -0.5 et +0.5 radians pour les angles et les torsions. Une translation ou rotation rigide des nucléotides (ou groupes d'atomes) suivants est appliquée à la structure pour chaque modification. Le croisement entre deux solutions parentes est réalisé en tirant une position aléatoire k au sort, puis en coupant et échangeant les fragments $[k, n]$ des deux chaines parentes dans leurs conformations respectives, selon les vecteurs C_4^{k-1}-P^k. La solution composée du fragment $[1, k\text{-}1]$ de la première structure parente et du fragment $[k, n]$ de la seconde est renvoyée comme solution fille. À chaque itération du programme, on réalise N croisements, chaque structure de la population initiale étant utilisée deux fois comme parent, ce qui aboutit à une population de taille $2N$.

On trie ensuite les solutions parentes et enfantes obtenues par rang de dominance. Pour ce faire, on choisit l'algorithme de tri T-ENS [382], identifié comme étant le plus efficace computationnellement [337] quand on a entre 3 et 5 critères et quelques centaines de points En effet, avec M le nombre de critères et N la taille de la population, sa complexité en temps est de $O(MN^2)$ dans le pire cas, comme de nombreux autres algorithmes de tri selon la dominance, mais sa complexité dans le meilleur cas est seulement de $O(MN\frac{\ln N}{ln\ M})$ ce qui lui donne l'avantage en pratique [337]. Il repose sur la construction d'un arbre de dominance entre solutions, ce qui permet d'éviter de trop nombreuses comparaisons.

On applique ensuite la sélection : les solutions qui peuplent le (ou les) premier(s) front(s) de Pareto sont immédiatement conservées, tant que le nombre de solutions ne dépasse pas N. Si le premier front de Pareto contient déjà plus de solutions qu'on ne peut en stocker dans la population de taille N, alors on agrandit automatiquement la taille de la population à 1,5 fois la taille du premier front de Pareto. Ceci ne fait pas partie de l'algorithme NSGA-III original, c'est une adaptation personnelle. En effet, l'algorithme original aurait considéré seulement N solutions du premier front de Pareto. Cependant, ceci conduit à la perte de solutions intéressantes, ce que l'on préfère éviter. Plus grave encore, cela conduit parfois à la perte d'un point extrême, faussant ainsi l'estimation du point idéal. On a donc opté pour la croissance automatique de la taille de la population.

S'il manque encore quelques solutions pour compléter la population de taille N, mais qu'on ne peut pas y stocker le front de Pareto d'ordre suivant en entier, on sélectionne des solutions de façon à maximiser la diversité, à l'aide de directions de référence. Il faut, tout d'abord, construire ces directions de référence. On commence par estimer le point idéal à l'aide des minima individuels observés jusqu'ici sur chaque critère. On normalise ensuite le front de Pareto et l'enveloppe convexe des minima individuels (CHIM) pour qu'ils soient inclus dans l'hypercube $[0,1]^M$. La procédure de normalisation utilisée en pratique est plus complexe, pour gérer plusieurs cas particuliers, elle est discutée en détail dans Blank *et al.*

2019 [39]. On échantillonne ensuite des points de référence régulièrement espacés dans la CHIM, comme dans la méthode NBI et ses variantes. En reliant le point idéal à chaque point de référence, on obtient une droite dite « direction de référence ». Chaque solution des fronts précédents est associée à la direction de référence la plus proche, en mesurant les distances perpendiculaires entre chaque point et chaque droite. On considère ensuite les solutions du front de Pareto d'ordre le plus faible que l'on n'a pas encore intégré à la nouvelle population, et on les attribue de la même façon à la direction de référence la plus proche. On sélectionne ensuite les solutions associées aux directions les moins représentées dans les fronts précédents pour être ajoutées à leur tour à la nouvelle population, jusqu'à atteindre N solutions pile. Lorsque plusieurs choix sont possibles avec le même nombre de solutions autour, on tire au sort. Mon implémentation de cette assignation de niches (ou *niching procedure*) diffère de l'algorithme original de Deb & Jain [87], que je trouve inefficace. Deb et Jain considèrent tous les points de référence un par un, les classent du moins représenté au plus représenté dans la nouvelle population, en choisissent un parmi les moins représentés aléatoirement, et s'il est représenté dans le front de Pareto suivant considéré, alors le point le plus proche est inclus dans la nouvelle population. Les comptes sont alors mis à jour et la procédure recommence jusqu'à atteindre exactement N solutions. Je trouve la considération explicite de tous les points lente, en particulier quand le nombre de critères augmente (et avec lui la dimension de l'espace de critères et donc le nombre de directions de référence). Mon implémentation liste les directions associées à chaque point dans le front de Pareto suivant, ainsi que leurs comptes dans la nouvelle population et dans le front à partager. Les points des directions les moins représentées sont ensuite incorporés à la nouvelle population en ne tirant au sort que si nécessaire, plusieurs d'un coup quand cela est possible.

L'algorithme s'arrête lorsqu'on a atteint un nombre prédéfini d'itérations, ou lorsque la diversité de la population devient trop faible (selon une métrique et un seuil dépendants du problème). On pourrait aussi imaginer l'arrêter lorsque l'ensemble de Pareto atteint une taille souhaitée.

L'implémentation finale est donc un algorithme NSGA-III fortement modifié (amélioré). Les algorithmes 1 (reproduction et tri) et 3 (attribution des directions de référence) de l'article original de Deb et Jain [87] ont été implémentés en parallèle. L'algorithme 2 (normalisation) a été remplacé par celui proposé dans Blank *et al.* [39], et l'algorithme 4 (niching) a été modifié par moi-même (sans que le principe change). De plus, la structure de données utilisée n'est pas un simple set de solutions, mais un arbre ordonné par dominance produit par T-ENS. La condition d'arrêt (provisoire) utilisée arrête la simulation si la diversité de la population sur le critère d'énergie en 3D devient inférieure à 10^4 (différence entre les valeurs minimale et maximale observées), ou si on atteint 10^5 itérations de l'algorithme.

V.2.3 Reconstruction all-atom et raffinement

Le programme d'optimisation renvoie toutes les solutions identifiées dans le front de Pareto. Après plusieurs milliers d'itérations avec trois ou quatre critères, le nombre de

solutions se compte en milliers. On échantillonne des structures dans le front de Pareto, régulièrement espacées selon les directions de référence. On reconstruit ensuite des structures contenant tous les atomes à partir des solutions « gros-grains » et des géométries de références proposées par la NDB [65] :

- On replace un ribose entier en conformation 3'-endo, en superposant les plans formés par les atomes C_1'-C_4'-C_5', puis en superposant les droites passant par C_4' et C_5' (en une seule rotation).
- On replace la base azotée. Pour les purines (à deux cycles), on superpose les droites B_1-B_2, c'est-à-dire les droites passant par les centres des cycles aromatiques. Pour les pyrimidines (à un seul cycle), on superpose la droite passant par B_1 et C_1' dans le cas du modèle gros grains avec celle passant par le centre du cycle aromatique et l'atome N_1 ou N_9 qui relie la base azotée au sucre. En effet, les atomes C_1', N_1 ou N_9 et le centre du cycle sont alignés.
- On replace le phosphate en superposant les droites passant par l'atome de phosphore et l'atome d'oxygène O_5'.

Toutes les superpositions sont faites par le calcul d'une matrice de rotation et d'un vecteur de translation pour superposer les vecteurs unitaires décrivant les droites citées.
Les longueurs et angles pouvant varier entre la solution et la géométrie de référence, les points de référence pour les superpositions sont les atomes de P, C_4', et le centre du premier cycle aromatique.

Une fois les structures reconstruites, il reste de nombreux défauts des structures liées à la modélisation gros grains imprécise : l'orientation des bases azotées autour de leur axe est arbitraire, les empilements ne sont donc pas visibles, et donc les hélices non plus. Il est nécessaire de raffiner les structures dans une fonction d'énergie haute définition, capable de détailler les contributions de chaque empilement de cycle aromatique et de chaque liaison hydrogène. De plus, certains atomes s'interpénètrent à cause de la reconstruction atomique qui ne prend pas garde à la position de voisins. On note aussi d'autres imperfections mineures (les atomes d'oxygène des phosphates forment des angles improbables par exemple). On décide donc d'utiliser QRNAS [321], un outil spécifiquement prévu pour ce problème : il raffine des structures obtenues par modélisation gros-grains dans le champ de force AMBER. C'est une simple implémentation de la fonction d'énergie ainsi que d'un algorithme de minimisation (algorithme du gradient conjugué). Elle rend des solutions plus plausibles, avec peu d'atomes s'interpénétrant, et des orientations des bases plus cohérentes. Les atomes d'hydrogène auront aussi été rajoutés.

V.2.4 Paramètres du modèle

Le modèle est maintenant entièrement défini, on dresse cependant la liste des paramètres à explorer pour obtenir une modélisation de bonne qualité :

- La densité des directions de référence. Elle a probablement un impact sur la vitesse d'obtention d'une bonne approximation du front de Pareto, mais ne change pas la

qualité des solutions à la fin si l'on réalise un grand nombre d'itérations. En effet selon quelques expériences rapides, la surface de Pareto obtenue à la fin est de densité uniforme et similaire quelque soit la densité des directions de référence initialement choisie.

- Les paramètres des fonctions d'énergie des structures secondaires et tertiaires choisies. La fonction d'énergie de la structure secondaire s'inspire largement du modèle du plus proche voisin, les paramètres sont donc des mesures issues d'expériences réelles et ne sont pas supposés être modifiés arbitrairement. Cependant, les paramètres β_1, β_2 et β_1^m ont été choisis arbitrairement par les auteurs de Nupack [99] et par moi-même, ils pourraient mériter une petite exploration des valeurs possibles. Les paramètres standard utilisés pour le modèle HiRE-RNA, dont nous avons déjà discuté en section 5.2.5.4, ont été mesurés en réalisant des moyennes sur les données de RNANet. Cependant, on a aussi montré que leurs distributions n'étaient pas monomodales (en section 4.2.3.4, et sur la Figure 4.13). On pourrait donc réfléchir à une meilleure façon de définir la valeur standard pour une distribution multimodale, en prenant le premier mode plutôt que la moyenne par exemple.

- Le nombre d'itérations de l'algorithme génétique, ou une condition d'arrêt (dépendante de la qualité du front de Pareto, et/ou de l'évolution du système),
- le nombre d'itérations du programme de raffinement QRNAS,
- la température, la salinité, la longueur de Debye utilisée,
- le seuil d'énergie définissant si deux nucléotides sont appariés ou non, ou la fonction qui définit ce seuil en fonction de la distance entre atomes. C'est un problème classique : chaque outil d'annotation des structures 3D d'ARN utilise son propre modèle (RNA-View [367], MC-Annotate [132], DSSR [208], …). Il n'y a pas de consensus clair sur comment faire. Ainsi, la façon dont ce modèle le fait n'est pas toujours satisfaisante. On obtient parfois des nucléotides appariés ayant pourtant une géométrie peu propice, et à l'inverse le programme manque parfois un appariement pourtant évident en 3D. L'ajustement de ces seuils doit évidemment être fait de façon conjointe à l'ajustement du modèle énergétique de l'appariement.

V.3 Résultats

V.3.1 Le programme MOARNA

On a implémenté l'algorithme décrit ci-dessus en langage Go 1.16, choisi pour l'excellent compromis simplicité/performance qu'il propose, et la facilité avec laquelle il est possible de paralléliser les procédures en peu de temps de développement. On a implémenté aussi les fonctions de coût décrites plus haut. Le programme, nommé MOARNA pour « Multicriteria Optimisation Applied to RNA », supporte plusieurs types d'entrées :

- la séquence directement passée en argument,
- ET/OU une structure 3D de référence au format mmCIF, dont la séquence sera extraite, et la structure uniquement utilisée à des fins de comparaison,
- ET/OU des données de sondage chimique contenant l'information de séquence au format RDAT (le format officiel proposé par RMDB [67]).

Le programme est capable de tronquer automatiquement les séquences présentes dans les différents fichiers, mais pas de les réaligner s'il y a des insertions, substitutions ou délétions, l'utilisateur doit donc s'assurer que les fichiers d'entrée sont cohérents entre eux. Le programme lance automatiquement Infernal (`cmscan`) pour détecter l'éventuelle appartenance de la séquence test à une famille Rfam connue. En fonction des informations disponibles, deux, trois, ou quatre critères sont utilisés pour l'optimisation. Il est possible d'interdire explicitement l'usage d'un critère avec un argument de ligne de commande.

Pendant le déroulement du programme, il est possible de suivre sur des graphiques en temps réel la distribution des énergies sur chaque critère, la forme du front de Pareto, et la convergence des différentes fonctions de coût. La meilleure structure secondaire identifiée d'un point de vue énergétique, ainsi que celle correspondant le mieux aux données de sondage chimique, sont reportées en temps réel, comme illustré ci-dessous sur la Figure 5.8.

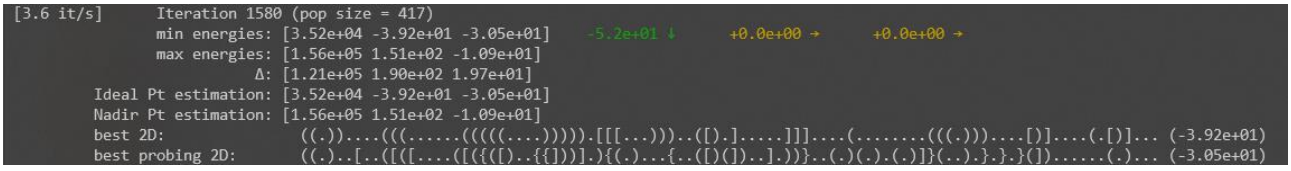


Figure 5.8 – Sortie console de MOARNA. *Exemple de sortie console pour une simulation de la séquence du RNA-Puzzle 8. Toutes les 100 itérations, le programme récapitule les changements d'énergies observés et les meilleures structures secondaires obtenues selon les critères d'énergie de la structure secondaire et de compatibilité avec les données de sondage chimique.*

Par défaut, le programme s'arrête si l'on atteint 100 000 itérations, ou si la diversité d'énergie de la structure 3D devient faible (la différence entre l'énergie minimale et maximale dans le front de Pareto est inférieure à 10^4).

En sortie, le programme génère deux dossiers contenant les solutions du front de Pareto en 3D, au format mmCIF : l'un contient les modèles gros-grains, l'autre les reconstructions all-atom. Les fichiers gros-grains contiennent l'information de structure secondaire explicitement encodée, ce qui permet de mettre en valeur les appariements canoniques dans les logiciels de visualisation. Par défaut, MOARNA lance QRNAS sur les solutions les plus proches de chaque direction de référence. Le nombre de directions de référence (et donc de solutions raffinées) dépend du nombre de critères utilisés, qui dépend des informations disponibles, ainsi que d'un paramètre de « densité des directions de référence » fourni en entrée. Un troisième dossier contient donc ces solutions au format PDB, avant et après raffinement. Les structures pouvant être considérées comme les solutions finales de l'outil sont donc ces structures PDB raffinées, néanmoins, elles ne représentent pas le front de Pareto complet. Le programme peut aussi enregistrer les graphiques en tant qu'image.

On réalise un petit benchmark sur les structures du benchmark communautaire RNA-Puzzles [238] pour lesquelles on dispose de données de sondage chimique, trouvées sur RMDB dans la section « RNA-Puzzles ». Le choix des structures de RNA-Puzzles permet d'utiliser les valeurs publiques de performance des autres outils de l'état de l'art pour s'y comparer. On récupère donc les structures des Puzzles 7, 8, 9, 10 et 18, dont les séquences contenues dans les fichiers RDAT de RMDB matchent bien celles des structures 3D disponibles. Ces structures sont présentées dans la Table 7. Pour le puzzle 8, un SAM riboswitch, une famille Rfam est identifiable (RF01725). Pour chacun, on teste plusieurs combinaisons de critères utilisables.

Puzzle	PDB	Type	Longueur	Famille Rfam
7	4r4v A	Satellite ribozyme	185	x
8	4l81 A	SAM riboswitch	96	RF01725
9	5kpy A	Aptamer	71	x
10	4lck C	T-Box riboswitch	102	x
18	5tpy A	7SK	71	x

Table 7 – Sous-jeu de données RNA Puzzles. *Voir rnapuzzles.org et rmdb.stanford.edu.*

V.3.2 Fronts de Pareto obtenus

Selon le nombre de critères utilisés, le front de Pareto sera en deux, trois, ou quatre dimensions (le dernier cas nécessitant de choisir 3 des critères sur quatre pour être représentés sur un graphique). Un exemple de la distribution de la population dans l'espace des critères à la fin de la simulation est présenté sur la Figure 5.9, en utilisant les énergies des structures secondaires et tertiaires plus la compatibilité avec les données de sondage. Chaque point sur la figure correspond à une structure 3D d'ARN. Les points sont représentés différemment selon leur rang de dominance de Pareto. Le premier front de Pareto contient l'ensemble des solutions renvoyées à la fin par MOARNA, et celles les plus proches des directions de référence (les flèches grises) seront raffinées et sauvées au format PDB.

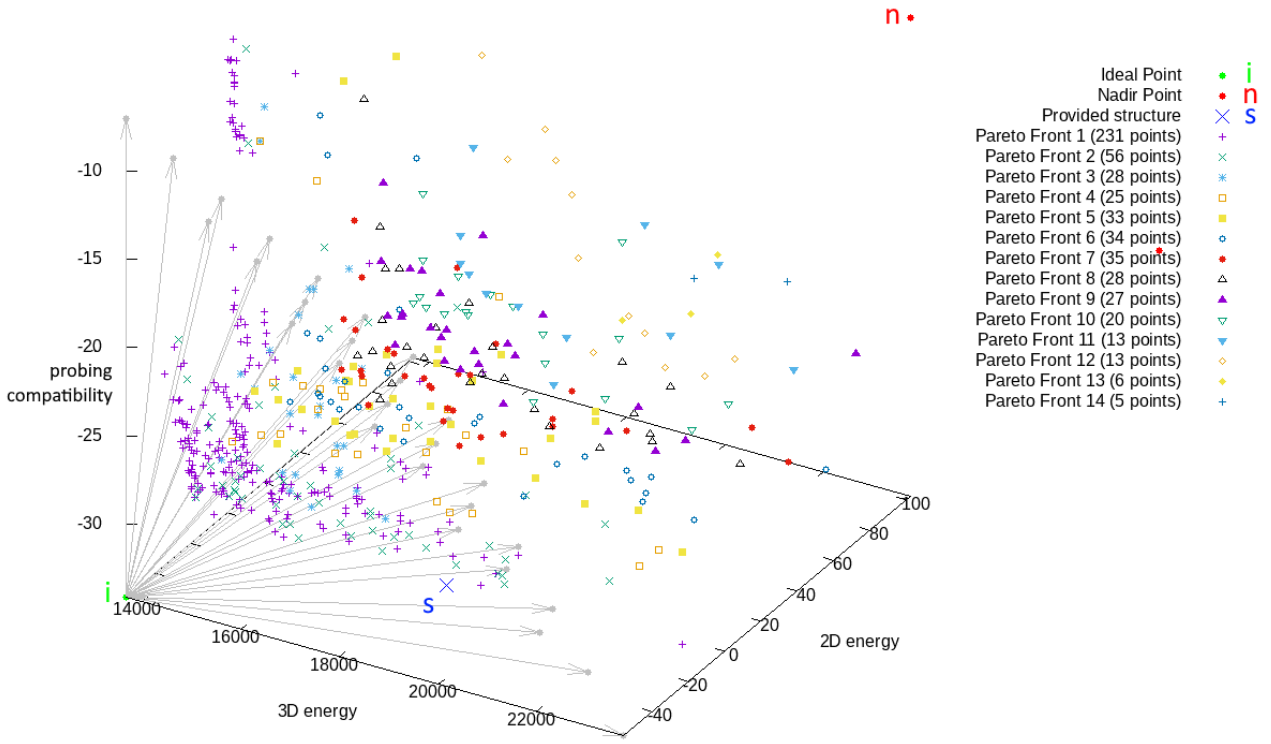


Figure 5.9 – Exemple de front de Pareto de MOARNA à trois critères. *Ce résultat est obtenu après 5 070 itérations de NSGA-III à partir de la séquence du RNA-Puzzle 8 (SAM riboswitch, de longueur 96 bases). La structure 3D connue correspondante (4L81) est mise en valeur avec une croix bleue marquée « S ». Le front de Pareto, composé des 231 points identifiés par une petite croix violette, est compris dans le cube délimité par le point idéal (« i » en vert) et le point nadir (« n » en rouge). Les directions de référence de la méthode NBI, également utilisées dans NSGA-III, sont représentées par des flèches grises à partir du point idéal.*

On observe que la structure réelle notée S est proche du front de Pareto, elle semble avoir un bon score sur le critère d'énergie de la structure secondaire et sur la compatibilité avec les données de sondage (et un score moyen sur le critère d'énergie 3D). A la simple visualisation du front de Pareto, c'est plutôt encourageant : cela laisse supposer que l'une des structures du front de Pareto sera proche de la structure réelle. Malheureusement, ce n'est pas le cas : l'espace des critères et l'espace des variables sont différents, et la proximité dans l'espace des critères n'implique pas du tout la proximité dans l'espace des variables. De toutes façons, il est courant que le front de Pareto obtenu ne soit pas proche de la structure réelle, cette dernière étant située en dehors de l'hypercube considéré, souvent à cause d'une valeur de dRMSD ou d'énergie de la structure secondaire plus élevée ou plus basse que celles observées dans les solutions de la simulation.

Globalement, sur les 5 cas testés, et quelque soit la combinaison de critères utilisée, on observe des fronts de Pareto bien étalés. Ceci indique que les critères ne sont pas corrélés, auquel cas on aurait un très faible nombre de solutions dominant toutes les autres. On aurait pourtant pu s'attendre à la corrélation des critères de même échelle entre eux : l'énergie de

la structure secondaire et la compatibilité avec les données de sondage d'une part, l'énergie de la structure 3D et la compatibilité avec les distances interbases de la famille identifiée de l'autre. On n'observe pas ces corrélations.

On observe aussi que le nombre de solutions retrouvées dans le front de Pareto augmente exponentiellement avec le nombre de critères utilisés : quelques centaines à 2 critères, quelques milliers à 3 critères, quelques dizaines de milliers à 4 critères, comme illustré sur la Figure 5.10.

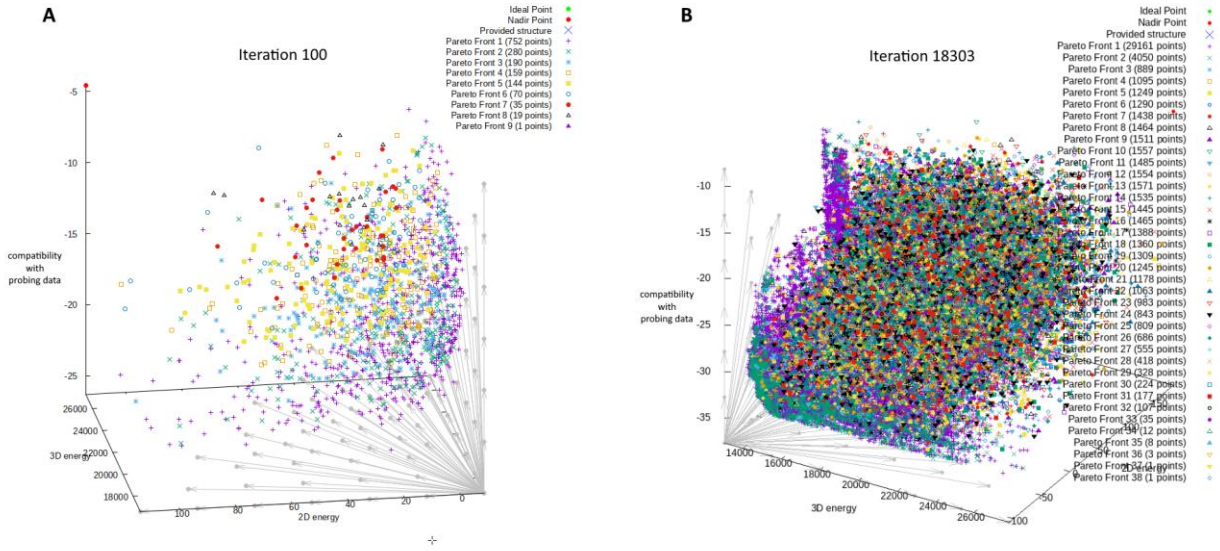


Figure 5.10 – Fronts de Pareto à 4 critères en début et fin de simulation. *Les points et les directions de référence sont dans un espace à quatre dimensions projeté sur 3 axes. Le critère de compatibilité avec la matrice de distances de la famille n'est pas représenté.* ***(A)*** *Une population de 1 400 points environ autour de l'itération 100 (au début).* ***(B)*** *Trois jours de temps de calcul plus tard, l'état des fronts de Pareto : la population a augmenté à presque 50 000 structures (29 161 dans le premier front de Pareto).*

V.3.3 Convergence et interactions entre critères

Le programme affiche en temps réel les courbes de progression des valeurs de chaque critère au fur et à mesure des itérations. On peut donc s'intéresser à la vitesse de convergence. Les courbes de suivi de la progression sont représentées sur les Figures 5.11, 5.12 et 5.13 pour différentes combinaisons de 2, 3 et 4 critères avec les séquences du RNA Puzzle 8. L'objectif est d'étudier les profils de convergence. La tendance générale, illustrée par les trois expériences de la Figure 5.11, est la suivante : L'énergie de la structure secondaire converge par « paliers », à la suite de l'obtention rare d'une nouvelle structure : on n'obtient aucune « innovation » pendant des milliers d'itérations jusqu'à l'apparition soudaine d'un repliement de qualité supérieure, donnant une courbe en forme d'escalier. Ce n'est pas surprenant : notre modèle d'échantillonnage repose sur des variations d'angles, il est nécessaire d'en accumuler plusieurs et dans des gammes de valeurs très spécifiques pour observer un changement plausible de structure secondaire. Le critère de compatibilité avec les données de sondage chimique converge souvent plus lentement que les autres, mais de

façon continue et progressive. On peut donc conclure sur son utilité : il permet de guider la structure secondaire de la molécule en étant plus facile à optimiser que le modèle d'énergie discret. Les données de sondage sont donc précieuses pour simuler le repliement de la chaine. Les critères « 3D » (l'énergie HiRE-RNA et le dRMSD aux distances moyennes dans la famille d'ARN connue) convergent très rapidement. Ces observations sont également vraies pour la simulation d'autres ARN (données non présentées).

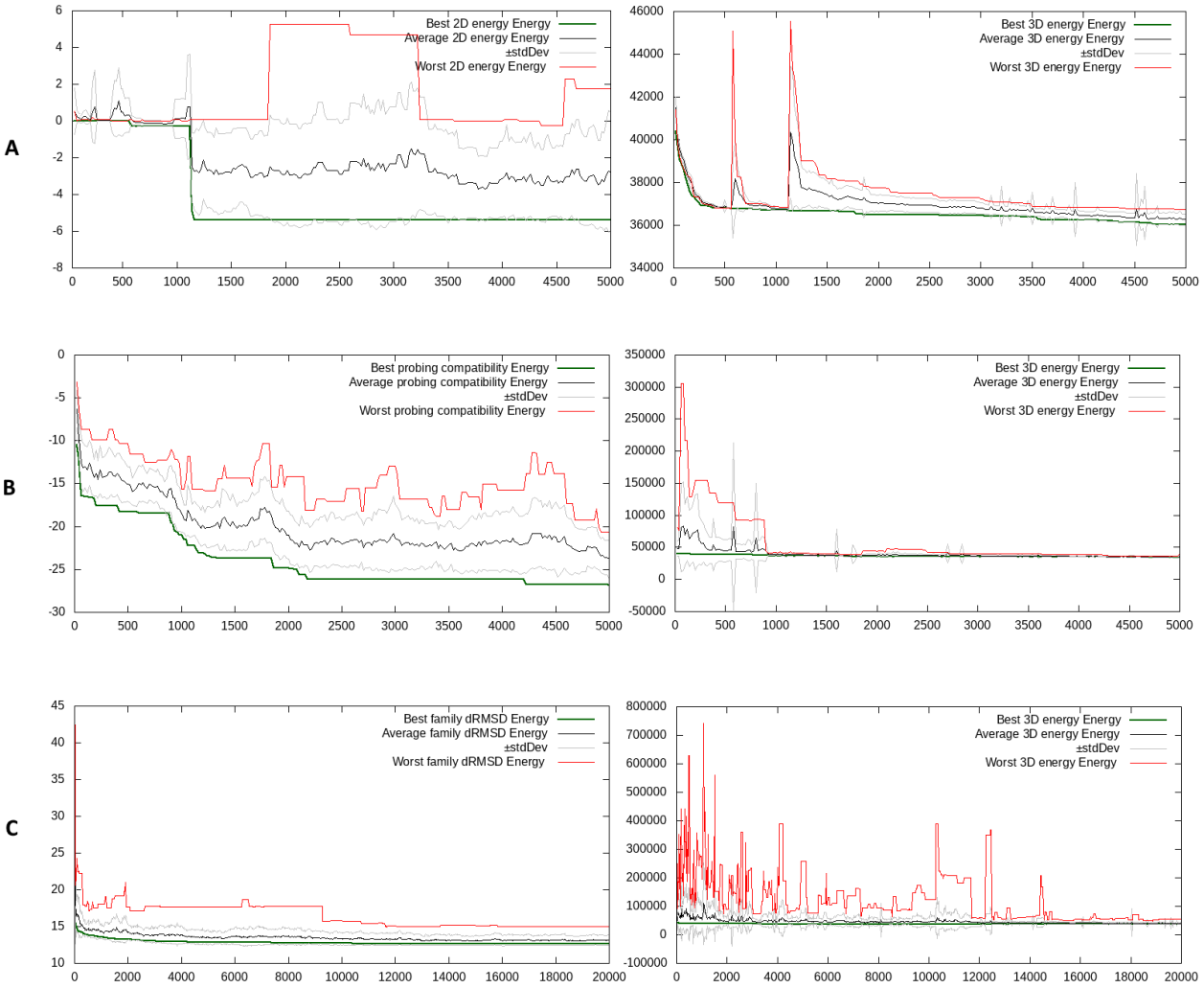


Figure 5.11 – **Convergence des critères (avec deux critères).** *Trois expériences sont réalisées avec la séquence du puzzle 8 (4l81), chacune avec une combinaison de critères différente. Les axes en abscisse comptent les itérations de NSGA-III, et les axes en ordonnée les valeurs du critère considéré. Les deux premières ont duré 5 000 itérations et la troisième 20 000. Elles permettent de comparer la convergence des 3 critères (à gauche), l'énergie 3D étant toujours considérée pour optimiser finement la position des résidus (et représentée à droite).* ***(A)*** *L'énergie de la structure secondaire.* ***(B)*** *La compatibilité avec les données de sondage.* ***(C)*** *Le dRMSD avec la matrice des distances de la famille d'ARN identifiée.*

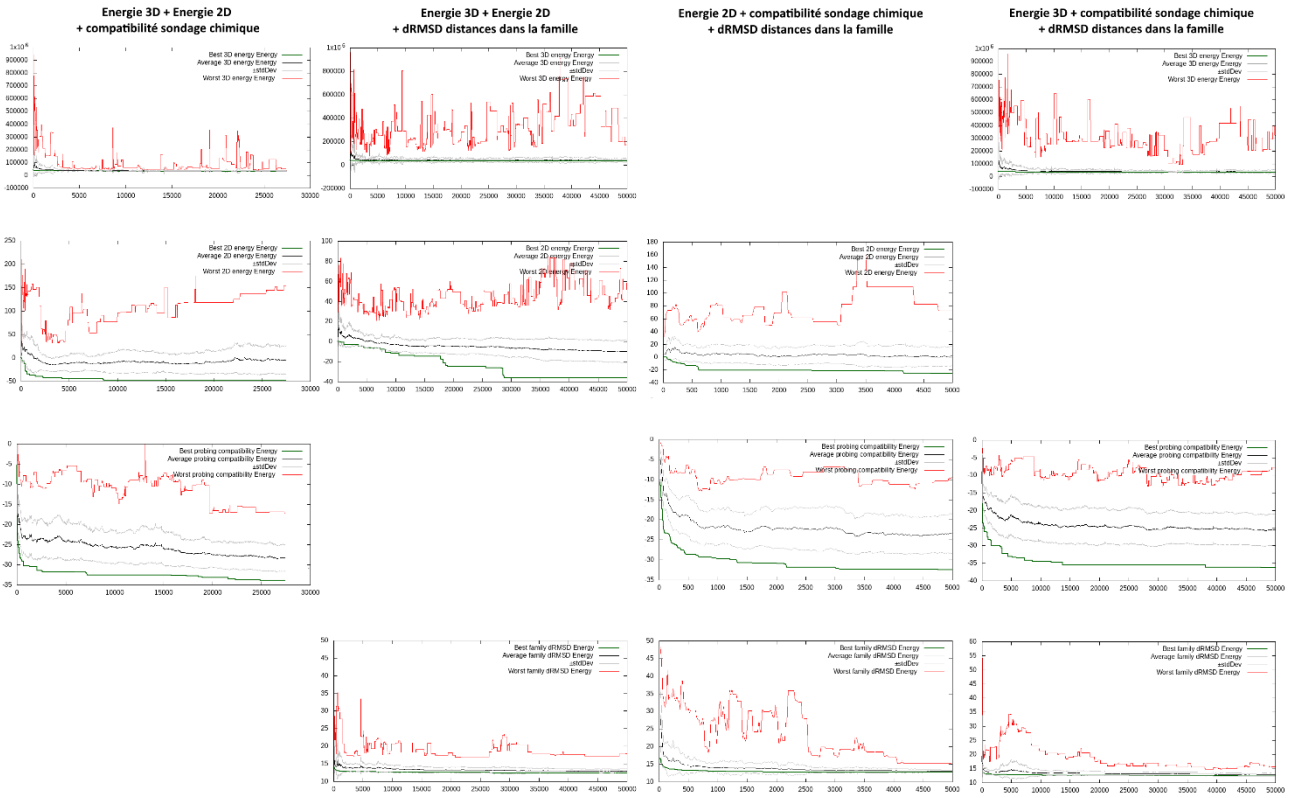


Figure 5.12 – **Convergence des critères (avec trois critères).** *Quatre expériences utilisant trois critères sont présentées, avec les durées respectives de 27 500, 50 000, 5 000 et 50 000 itérations avant convergence, toujours avec la séquence du puzzle 8 (4l81).*

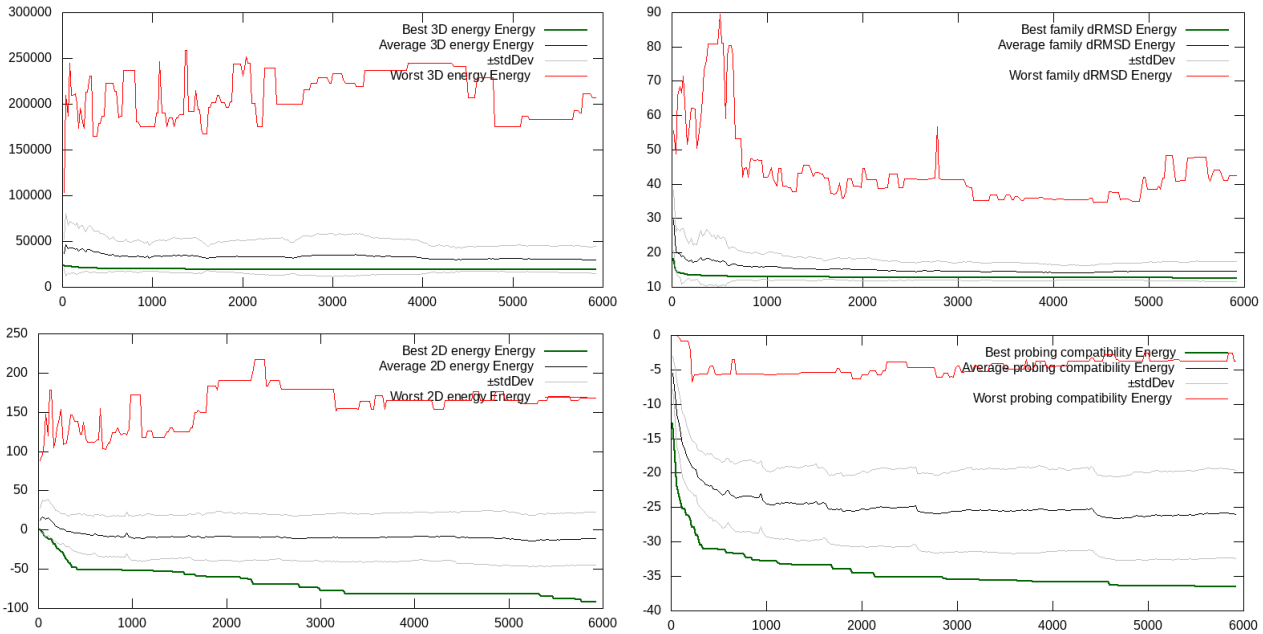


Figure 5.13 – **Convergence des critères (avec quatre critères).** *Les 4 graphiques correspondent à une seule expérience où l'on optimise tous les critères ensemble pour la séquence du puzzle 8 (4l81). La simulation a été tronquée artificiellement à 6 000 itérations : en effet, avec 4 critères, la condition d'arrêt n'est pas atteinte, et la simulation peut durer plusieurs centaines de milliers d'itérations en continuant d'optimiser l'énergie de la structure 3D seulement, les autres critères ne variant plus.*

À propos de la vitesse de convergence respective des 4 critères pris individuellement, on voit que les critères « 3D » (l'énergie HiRE-RNA et le dRMSD aux distances moyennes dans la famille d'ARN connue) convergent très rapidement. L'énergie de la structure secondaire converge plutôt lentement : on n'obtient aucune « innovation » pendant des milliers d'itérations jusqu'à l'apparition soudaine d'un repliement de qualité supérieure, donnant une courbe en forme d'escalier. Ce n'est pas surprenant : notre modèle d'échantillonnage repose sur des variations d'angles, il est nécessaire d'en accumuler plusieurs et dans des gammes de valeurs très spécifiques pour observer un changement plausible de structure secondaire. Enfin, le critère de compatibilité avec les données de sondage chimique converge souvent plus lentement que les autres, mais de façon continue et progressive. On peut donc conclure sur son utilité : il permet de guider la structure secondaire de la molécule en étant plus facile à optimiser que le modèle d'énergie discret. Les données de sondage sont donc précieuses pour simuler le repliement de la chaine.

De façon générale, on observe que d'avantage de critères permettent d'obtenir de meilleures valeurs individuelles sur chaque critère. On note sur la structure 5.13 une meilleure structure secondaire finale d'énergie proche de -100 unités, une performance nettement supérieure à ce qui a été obtenue avec deux ou trois critères seulement. A l'inverse sur la Figure 5.11, les valeurs d'énergie de la structure secondaire obtenues ne sont pas aussi bonnes, même en ne considérant que les 5 000 premières itérations, pour une comparaison équitable. Il y a donc un réel bénéfice à utiliser plus de critères.

V.3.4 Forme et aspect des solutions

On regarde maintenant l'aspect des solutions obtenues. Un exemple de structure est présenté sur la Figure 5.12. Sans critère portant sur la structure secondaire, les solutions obtenues ne contiennent pas d'hélices ou boucles, ce qui les rend peu plausibles. Sans le critère d'énergie 3D, les structures ne sont pas plausibles en 3D du tout, notamment à cause des atomes s'interpénétrant qui ne sont jamais pénalisés. Avec au moins un critère d'énergie 2D et 3D, il est souvent possible d'observer des éléments de structure secondaire comme des hélices se former, ce qui est souhaitable (comme c'est le cas sur la Figure 5.14).

Pour chaque direction de référence (les flèches grises sur le front de Pareto de la Figure 5.14), on repère la structure du premier front de Pareto la plus proche, et on la soumet à minimisation dans QRNAS [321] (une implémentation d'AMBER). La Figure 5.15 présente une structure obtenue par simulation dans MOARNA lors du processus de reconstruction. Le raffinement ne fait que très peu varier la structure globale, mais il permet de réajuster la position des cycles aromatiques ou des sucres, et d'éviter les clashs atomiques, ce qui donne souvent un modèle de meilleure qualité.

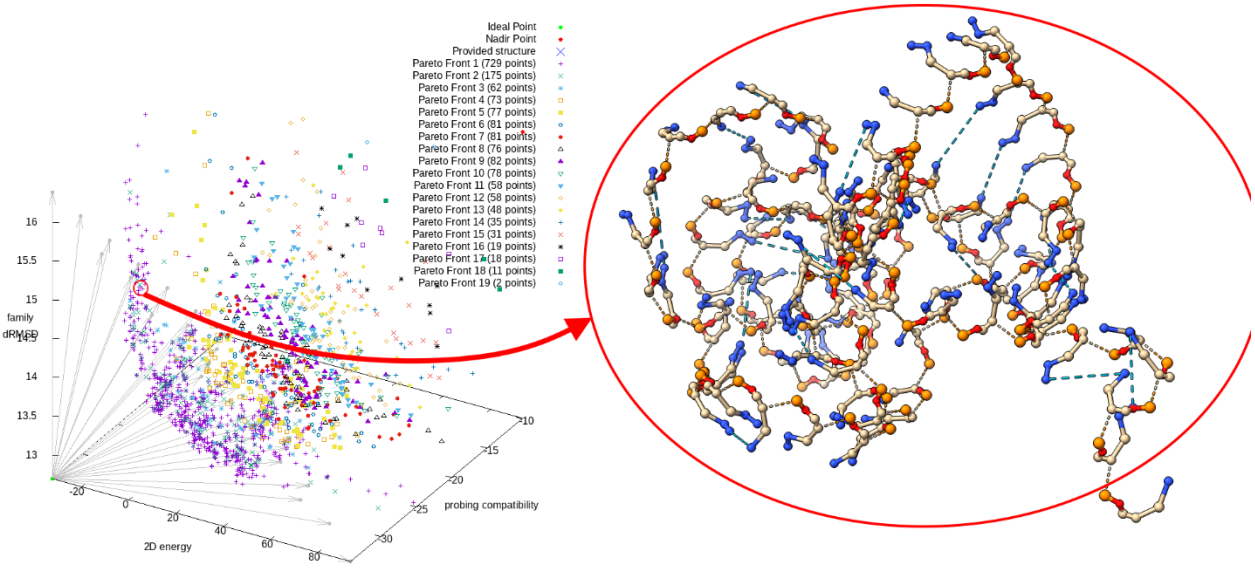


Figure 5.12 – Reconstruction des modèles 3D. *À chaque solution du front de Pareto correspond une structure 3D gros grains. MOARNA sauvegarde ces structures au format mmCIF. Ici une solution obtenue dans une simulation de la structure 4l81 avec 3 critères, les énergies en 2D et 3D, et le dRMSD avec la matrice moyenne de distances de la famille d'ARN.*

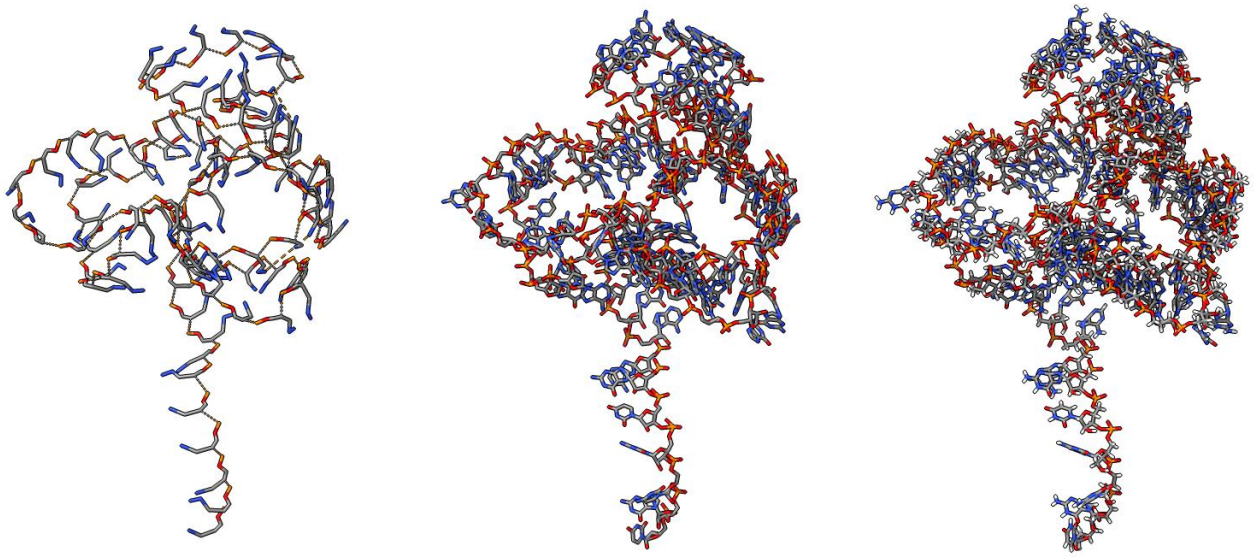


Figure 5.15 – Modèles 3D reconstitués à partir des solutions. *À gauche, le modèle gros grains HiRE-RNA. Au centre, la reconstruction all-atom. À droite, la reconstruction all-atom après minimisation dans QRNAS. La solution représentée ici est issue de la simulation de 4l81 avec les deux critères 3D mais sans critère basé sur la structure secondaire. On n'observe d'ailleurs pas d'élément de structure secondaire.*

V.3.5 Performance des prédictions

V.3.5.1 Comparaison aux structures natives

Pour chaque solution du front de Pareto (avant raffinement donc), on calcule le TM-Score [137] associé à la structure réelle, sur la base des distances entre carbones C_1'. La superposition des structures est réalisée par décomposition en valeurs singulières pour calculer une rotation et translation entre les deux nuages de points de coordonnées.
Les résultats sont présentés dans la Table 8. Les scores sont extrêmement mauvais, on peut directement conclure que la méthode n'est pas du tout suffisante pour proposer des structures natives, et ce quelque soit la combinaison de critères. Pour rappel, le TM-score est défini comme :

$$TM - score = \frac{1}{L}\sum_{i=1}^{L} \frac{1}{1 + (d_i/d_0)^2}$$

avec d_i les distances entre carbone C'_1 réel et carbone C'_1 prédit (après superposition) et $d_0 = 0.6\sqrt{L - 0.5} - 2.5$ un facteur d'échelle permettant d'affranchir la métrique du biais lié à la longueur de la structure.

PDB	Longueur	Énergie 2D	Énergie 3D	Compat. sondage	dRMSD	Meilleur TM-score	Pire TM-score
4lck	102 (T-Box)		✔	✔		0.03	0.01
		✔	✔			0.02	0.01
		✔	✔	✔		**0.04**	0.01
4r4v	185 (satellite ribozyme)		✔	✔		0.03	0.01
		✔	✔			0.02	0.01
		✔	✔	✔		**0.07**	0.01
4l81	96 (SAM riboswitch)	✔	✔			0.01	0.01
			✔	✔		0.04	0.03
			✔		✔	0.06	0.04
		✔	✔		✔	**0.09**	0.02
		✔	✔	✔		0.06	0.02
			✔	✔	✔	**0.09**	0.03
		✔	✔	✔	✔	**0.10**	0.01
		✔		✔	✔	**0.09**	0.03
5kpy	71 (Aptamère)		✔	✔		0.03	0.02
		✔	✔			0.02	0.02
		✔	✔	✔		**0.05**	0.01
5tpy	71 (7SK)		✔	✔		0.03	0.02
		✔	✔			0.03	0.03
		✔	✔	✔		**0.04**	0.01

Table 8 – Résultats de benchmark sur les Puzzles 7, 8, 9, 10, et 18. *Pour chaque Puzzle et pour chaque combinaison de critères testée, on reporte les meilleurs et pires TM-scores observés dans le front de Pareto obtenu. Les TM-scores sont tous très faibles, ce qui indique*

une absence de corrélation entre les formes prédites par le modèle et les formes natives.

V.3.5.2 Utilité des critères

Dans chaque puzzle, l'utilisation du critère de compatibilité avec les données de sondage donne une performance légèrement supérieure à l'énergie de la structure secondaire, probablement parce qu'une plus grande diversité de scores est possible sur ce critère.

L'objectif initial d'avoir un ou plusieurs critères basés sur la structure secondaire était de retenir des solutions plus diverses en utilisant un critère de modélisation à une échelle différente. Ceci est particulièrement vrai dans la combinaison de critères utilisant les deux énergies en 2D et en 3D, où la diversité d'énergies en 3D diminue très rapidement. Néanmoins, on a observé que l'utilisation de n'importe quel autre critère suffit à maintenir la diversité. Il n'y a donc pas forcément besoin de conserver un critère portant sur la structure secondaire. C'est pour cette raison qu'on a expressément testé la combinaison { énergie 3D, dRMSD } avec 4L81 (puzzle 8). Sans critère basé sur la structure secondaire, la meilleure performance obtenue est de 0.06. Avec un critère basé sur la structure secondaire, que ce soit l'énergie, ou la compatibilité avec les données de sondage, la performance est 50% plus élevée (0.09), si l'on ose comparer ces chiffres de toutes façons très bas. On peut quand même reconnaitre l'utilité d'utiliser un critère à chaque échelle.

L'utilisation de la matrice des distances moyennes peut être une alternative à l'utilisation de l'énergie en 3D, et peut même s'avérer légèrement plus performante. Cependant, les défauts de cette fonction ne sont pas capturés par le TM-score : la fonction optimise le repliement global de la chaine, ce qui est capturé par le TM-score (position des carbones C_1'), mais pas la position précise des bases et leur orientation (ce qui n'est pas observé par le TM-score).

Enfin, l'utilisation combinée de l'énergie de la structure secondaire et de la compatibilité avec les données améliore la performance par rapport à l'utilisation d'un seul de ces deux critères.

V.4 Conclusion et perspectives

V.4.1 État du modèle

Pour répondre à la question principale posée dans cette thèse, on a implémenté un outil optimisant conjointement plusieurs modèles d'évaluation des structures, suivant une approche de Pareto, et retournant en tant que solutions l'ensemble de Pareto (ou un sous-ensemble de solutions régulièrement échantillonnées sur le front de Pareto).

Les modèles choisis sont volontairement divers, pour essayer d'utiliser le maximum d'informations disponibles. On a ainsi implémenté :

- une fonction d'évaluation de l'énergie de la structure secondaire, basée sur les règles de Turner mais supportant les pseudonoeuds de complexité arbitraire, ceux-ci pouvant être proposés par le programme d'optimisation,
- une fonction d'évaluation de l'énergie de la structure 3D gros grains, fortement inspirée du modèle HiRE-RNAv3 [73], mais avec une nouvelle paramétrisation essentiellement basée sur RNANet,
- une fonction d'évaluation de la compatibilité entre une structure proposée et des données de sondage chimique fournies en entrée,
- une fonction d'évaluation de la forme d'une structure, la comparant à une forme moyenne connue dans la famille d'ARN de la molécule cible, si une telle famille a pu être identifiée (par `cmscan` [246] de la suite Infernal).

L'utilisation sélective d'un ou plusieurs de ces critères seulement est bien sûr possible, tout comme la future implémentation de nouveaux critères d'évaluation.

L'algorithme d'optimisation utilisé est un algorithme génétique multi-objectif appelé NSGA-III [39,87]. Son étape de sélection est majoritairement basée sur la dominance de Pareto. Cet algorithme semble peu efficace pour explorer l'espace de conformations, et reste à améliorer ou remplacer (voir paragraphes V.4.2 et V.4.3 suivants).

Le programme MOARNA enregistre les solutions du front de Pareto obtenues au format mmCIF, en modèle gros-grains et après réinsertion de tous les atomes. Enfin, des solutions sont échantillonnées régulièrement dans le front de Pareto et soumises au raffinement all-atom par le programme QRNAS [321], qui les rend au format PDB.

On a mis en évidence plusieurs interactions entre critères visibles sur les profils de convergence. On retiendra :

1. Que l'apparition d'une nouvelle structure secondaire peut soudainement améliorer la meilleure énergie de structure secondaire, et en même temps détériorer temporairement la pire énergie 3D observée dans la population. Cependant, cette perturbation est éliminée en quelques centaines d'itérations, et la nouvelle structure secondaire est conservée.
2. Utiliser plus de critères est synonyme d'un nombre exponentiellement plus grand de solutions dans les fronts de Pareto (et donc dans la population au total), et avec ce

nombre, un plus grand temps de calcul de chaque itération.

3. Utiliser plus de critère est aussi corrélé à une augmentation de la diversité des valeurs des différents critères dans la population, car certaines solutions ayant un très bon score sur un axe en auront un très mauvais sur un ou plusieurs autres. Le nombre de solutions plus élevé participe aussi mécaniquement à l'augmentation de la diversité.
4. Enfin, utiliser plus de critères permet d'obtenir des améliorations d'un des critères plus souvent. Le programme converge donc plus vite vers des solutions de bonne qualité sur chaque axe. Certaines n'auraient parfois pas été atteintes avec moins de critères. En parallèle, la grande diversité des valeurs d'énergie 3D, ralentit l'arrêt du programme qui optimise le système pendant plus longtemps. La convergence est donc plus rapide et le programme fait plus d'itérations, ce qui mène à de meilleures solutions.

Néanmoins, les solutions jugées de bonne qualité par les critères de MOARNA ne sont pas particulièrement proches de structures natives. Un benchmark a été réalisé sur quelques structures des RNA-Puzzles [237] pour lesquelles des données de sondage chimique sont disponibles dans la base RMDB [67]. Les performances sont mauvaises, on n'arrive pas à des conformations proches (ni même ressemblantes) aux structures réelles, que ce soit visuellement, ou à l'aide d'une mesure de TM-scores. Ceci est très probablement dû à une exploration insuffisante de l'espace des conformations. L'approche entière n'est pas encore condamnable pour autant, d'autant plus que de nombreux réglages techniques n'ont pas encore été explorés. Dans les paragraphes suivants, on liste les points de travail les plus importants à améliorer.

V.4.2 Définir des move-sets plus intelligents

Comme de nombreuses méthodes existantes (mais en pire), MOARNA utilise une méthode d'échantillonnage trop peu efficace pour faire apparaitre des solutions intéressantes en un temps raisonnable. En effet, de nombreuses solutions très peu plausibles sont générées à chaque génération de NSGA-III puis aussitôt éliminées : on perd du temps.

L'échantillonnage actuel, basé sur les croisements entre deux structures parentes, et sur la « mutation » d'angles, n'a aucune raison de favoriser la création d'appariements ou empilements, ce qui rend très lente l'exploration de l'espace des structures secondaires. La génération de nouvelles structures secondaires se fait uniquement par hasard, et on pourrait imaginer plus efficace.

À ma connaissance, plusieurs move-sets ont déjà été proposés pour définir plus intelligemment ce que pourrait être le « voisinage » d'une structure d'ARN : on peut citer le move-set SWM [354] (Step-Wise Monte-Carlo) maintenant utilisé dans FARFAR 2 [355], qui explore automatiquement les conformations possibles des boucles HL et IL. On cite aussi les mouvements de chaine rigides inspirés de la robotique et présentés dans le chapitre 3 de la thèse d'Amélie Héliou [149], dont je ne connais pas encore d'implémentation dans un outil.

Ces pistes ont l'avantage de ne pas nécessiter d'autres changements dans l'algorithme actuel que l'étape de mutation, et pourraient accélérer grandement

l'exploration de l'espace conformationnel par des changements de structure secondaire obtenus en une seule itération.

V.4.3 Éliminer plus tôt les solutions inutiles

De nombreuses solutions sont actuellement conservées et consomment de la puissance de calcul même si elles n'apportent pas grand-chose à la population. Cela peut être parce qu'elles ont des valeurs catastrophiques sur l'un des critères, par exemples les structures présentant des clashs atomiques. On pourrait par exemple refuser par défaut toute structure fille qui se trouverait au-delà de l'estimation du point Nadir sur l'un des axes.

Certaines autres solutions sont de bonne qualité, mais n'apportent pas non plus beaucoup à la population parce qu'elles se situent dans des zones denses du front. On pourrait implémenter un mécanisme de filtration limitant la densité maximale de solutions dans chaque zone du front, comme ce qui est fait dans l'algorithme SPEA [176] par exemple.

V.4.4 Utiliser un meilleur critère d'arrêt

Actuellement, le critère d'arrêt utilisé surveille simplement que la diversité des énergies des structures 3D des solutions reste supérieure à 10 000. On n'a pas encore étudié réellement le temps de simulation nécessaire à l'obtention de solutions intéressantes : l'algorithme peut probablement les trouver, mais comme l'échantillonnage est naïf, l'exploration est lente. Le temps de simulation nécessaire est d'ailleurs probablement dépendant de la combinaison de critères utilisée. On pourrait imaginer un critère surveillant que l'obtention d'une diversité de solutions « suffisante » ait été obtenue, plutôt qu'un critère qui mesure la perte de diversité.

CHAPITRE VI : Discussion critique et concluante

Cette section concluante dresse un bilan de l'état actuel du champ de recherche, la modélisation informatique des structures d'ARN. Dans cette thèse, on a déjà évoqué certaines des différences théoriques entre ARN et protéines, qui ont un impact sur la façon dont ces macromolécules biologiques sont modélisées. On rappelle les plus courantes en section VI.1. Je commente sur la pertinence de l'optimisation multi-objectif en section VI.2. Enfin, la section VI.3. résume les contributions de la thèse en quelques pages.

VI.1 Différences entre ARN et protéines en modélisation

VI.1.1 Différences d'ordre théorique

À propos des séquences, les ARN sont des polymères d'un alphabet de seulement 4 résidus, contre 21 pour les protéines. Ainsi, les motifs de séquence ont toujours plus de chances d'arriver par hasard dans un ARN que dans une protéine. Les possibilités de mutation sont également moindres : toute mutation est non-sens dans l'ARN non codant, sauf si le nucléotide qui interagissait avec le nucléotide muté mute lui aussi pour maintenir l'interaction. On dénombre en revanche une plus haute fréquence de résidus modifiés dans l'ARN que dans les protéines, tant en quantité (fréquence d'occurrence) qu'en diversité (nombre de modifications possibles).

On a également discuté des différences structurales. La structure secondaire d'une protéine dépend des interactions faites par son squelette, les chaines latérales des résidus s'adaptant et pointant dans des directions compatibles. A l'inverse, la structure secondaire d'un ARN dépend des interactions faites par les chaines latérales des résidus (les bases azotées qui s'empilent et s'apparient). Le squelette adapte sa position en conséquence. Le squelette de l'ARN possède également plus de degrés de liberté que celui des protéines, avec six angles de torsion au lieu de trois dont un constant. Néanmoins, le modèle de Pyle permet de résumer la configuration de n'importe quel squelette d'ARN par deux angles de pseudo-torsion. Le ribose ne possède que deux conformations majoritaires (3'-endo et 2'-endo), donnant lieu à deux formes d'hélices (nommées A et Z, également utilisées pour décrire l'ADN). Alors que les chaines latérales des acides-aminés sont très diverses en taille, flexibilité, et propriétés physico-chimiques, les bases azotées sont planes, permettant de former des empilements de chaque côté par interactions entre cycles aromatiques. Elles peuvent aussi toutes chélater un cation. Enfin, elles ont trois cotés majoritaires pour former des appariements entre elles (plus deux « angles »). Certains appariements sont isostériques entre eux, c'est-à-dire qu'ils occupent exactement le même volume, si bien que l'on peut les substituer les uns aux autres sans perturber la structure. C'est le cas des appariements dits

« canoniques », qu'ils partagent avec l'ADN. On note enfin la distribution régulière des charges négatives portées par les phosphates le long du squelette des molécules d'ARN à pH neutre, charges qui ont besoin d'être neutralisées par des ions positifs.

Le faible nombre de bases, ainsi que l'isostéricité des appariements, permet de définir des modules (motifs, réseaux d'interactions…), c'est-à-dire de petites sous-structures locales fréquemment rencontrées dans des ARN même phylogénétiquement ou fonctionnellement éloignés. Ces motifs constituent une ressource importante dans la prédiction des structures. Même si on pense d'abord à les utiliser comme patrons pour la structure 3D, ce que d'autres ont fait (par exemple avec le pipeline RNA-MoIP+MC-Sym [272]), nous avons proposé dans le chapitre III de les utiliser pour la prédiction de structure secondaire, comme une étape pour appréhender la structure 3D.

Malgré les différences théoriques citées précédemment, les méthodes couramment utilisés en bioinformatique structurale pour les protéines ont été facilement adaptés pour l'ARN. Les champs de forces comme AMBER, CHARMM ou même HiRE-RNA possèdent des termes spécifiques modélisant les empilements et appariements, ainsi que l'effet répulsif des phosphates entre eux et la couche d'hydratation qu'ils génèrent autour de la molécule. La différence de nombre d'angles de torsion a été facilement contournée à l'aide du modèle de Pyle et des pseudo-torsions eta-theta. On a ainsi facilement pu au chapitre IV, théoriquement, adapter l'architecture du réseau géométrique récurrent (RGN) pour prédire les angles du squelette des ARN. De même, pour les autres architectures de réseaux de neurones que nous avons essayées, nous avons pu utiliser les mêmes caractéristiques issues du modèle de Potts pour extraire du signal des alignements multiples de séquences. Les modèles de covariance d'Infernal et Rfam ressemblent de par leur conception aux modèles de Markov cachés utilisés pour les familles de protéines, et permettent tout autant de classer les ARN en familles ou de détecter l'appartenance à l'une de ces familles.

VI.1.2 Différences d'ordre pratique

Les difficultés de modélisation de l'ARN par rapport aux protéines sont donc plutôt non pas de l'ordre théorique, mais pratique. On constate la jeunesse du champ de recherche entre autres par le manque de données structurales : on possède peu de structures 3D d'ARN. Les ARN sont souvent difficiles à cristalliser, et trop longs pour être étudiés par RMN. Parmi les quelques structures disponibles, on constate une très grande redondance avec une sur-représentation des ARN ribosomaux et de transfert. Au sein de ces familles, les structures sont très similaires et présentent peu de diversité. Heureusement, la cryo-microscopie électronique s'annonce prometteuse pour la résolution des structures d'ARN [83]. De plus, de nombreuses classes d'ARN non codants n'ont été découvertes que récemment, et de nouvelles continuent d'être découvertes. Du coté des structures secondaires, la plupart sont issues d'annotation de structures 3D existantes, d'études comparatives à partir d'alignements multiples de séquences, ou d'expériences de FRET ou de sondage chimique. On déplore, en 2D comme en 3D, la faible qualité des bases de données dédiées à l'ARN. Jusqu'à ces dernières années (2018-2021) il n'existait pas d'archive exhaustive des structures

d'ARN 3D publiques existantes annotées (la NDB se contente de référencer la PDB et de lister des résultats d'annotation de logiciels externes, mais ne propose pas de navigation intégrée comme la PDB). Pour les structures secondaires, il n'existait pas d'archive unique des structures vérifiées expérimentalement (RNA Strand ou Pseudobase par exemple ne donnent pas les sources de chaque séquence qu'elles contiennent). Elles ne sont d'ailleurs plus mises à jour depuis leur sortie. On peut faire le même reproche aux bases de données de motifs, ou même la NNDB [339] pour les paramètres thermodynamiques. Les différentes bases de données et outils intégrés du groupe de BGSU [65,251], bpRNA [78], ou encore notre jeu de données RNANet [27] présenté au chapitre IV participent à l'amélioration progressive et à l'accessibilité des données d'ARN non codants.

Les logiciels dédiés à l'ARN sont également souvent moins développés et moins robustes, à quelques exceptions près (Infernal, ViennaRNA, DSSR, …). Parfois, il n'existe pas d'équivalent (on pense à jackHHMER, HHBlits, MMseqs2, …), non pas parce qu'une raison théorique l'empêche, mais parce que personne n'a eu l'occasion de réaliser une implémentation. De nombreux outils basés sur les données disponibles, ou dont les paramètres en sont dérivés, seraient encore pertinents s'ils étaient fréquemment ré-entrainés avec les données les plus récentes. Trop souvent, leurs auteurs sont passés à d'autres sujets d'études sans même avoir donné de code source ou rédigé de documentation pour leurs outils. Documenter un outil bioinformatique, le rendre efficace computationnellement et robuste selon les situations nécessite une équipe de développement sur plusieurs mois ou années. De telles conditions de développement logiciel n'existent pas encore dans de nombreux laboratoires académiques travaillant sur l'ARN, où les outils sont développés au cours d'une thèse ou d'un contrat post-doctoral de quelques mois puis plus maintenus. La capitalisation des expertises ne se fait pas toujours, en particulier d'un laboratoire à l'autre. En conséquence, l'écosystème des outils spécialisés est redondant car aucun outil de référence ne s'impose. Les différences entre deux outils ne sont pas toujours très claires, et les benchmarks rarement à jour. Ce sont ces défauts qui retardent les avancées théoriques dans le domaine.

VI.1.3 Conséquences sur cette thèse

Cette importance de l'ingénierie logicielle est évidente dans cette thèse ; en effet, malgré les nombreuses idées théoriques proposées, le développement d'outils de prédiction performants n'a pas été possible. Le développement d'un jeu de données fiable et robuste, RNANet, a consommé la majorité du temps de recherche. Il n'était plus possible ensuite, en quelques mois, de développer correctement les réseaux de neurones profonds que nous avons proposés et d'explorer leur paramétrisation pour obtenir la meilleure performance. Ni pour le modèle HiRE-RNA, que j'ai naïvement pensé pouvoir reparamétrer en quelques semaines quand ses auteurs y travaillent depuis 2010. Nous avons alors choisi dans l'équipe de faire recours à des stages pour constituer ce qui devrait être une équipe de développement, parfois avec succès, mais sans avancer réellement plus vite. J'ai donc le sentiment que les conclusions préalablement présentées sur la faisabilité ou la performance des méthodes proposées aux chapitres III, IV et V, actuellement faible ou nulle, ne sont pas

encore tranchées tant qu'une exploration systématique (ou une conception rationnelle et documentée) des paramètres n'aura pas été faite, et qu'un test systématique du code de chaque outil pour y chercher d'éventuelles erreurs n'aura pas été réalisé. De très nombreuses idées théoriques en réserve permettront de poursuivre ces recherches ou d'étendre les modèles proposés ici. Cependant, l'obtention préalable d'outils robustes (au lieu des prototypes proposés dans cette thèse) par un travail de développement logiciel et de documentation sérieux, par des personnes formées aux bonnes pratiques de programmation, me parait être une condition nécessaire à toute progression. Ce travail d'ingénierie est malheureusement trop peu valorisé dans les critères d'évaluation de la recherche académique, ce qui n'incite pas les chercheurs à tous niveaux à y investir du temps et des ressources.

Heureusement encore, l'intérêt grandissant pour l'ARN en tant que nouvelle cible ou principe thérapeutique devrait imposer une pression pour faire avancer l'ingénierie dans le domaine. Aussi, les tendances récentes à ouvrir les codes des logiciels et à généraliser les pratiques « DevOps » (l'intégration continue, le test systématique et les benchmarks automatiques) devraient améliorer dans les années qui viennent la qualité de l'écosystème d'outils et de bases de données spécialisés dans l'ARN.

VI.2 L'optimisation multicritère pallie le manque de standardisation

Dans le contexte dépeint au paragraphe précédent, l'optimisation multicritère a été proposée dans cette thèse comme une piste pour intégrer des modèles différents.

VI.2.1 Multiplicité des outils et modèles et tentatives de bancs d'essai

Comme présentés en section 1.4 et sur la Figure 1.17, le retard du champ de recherche sur les ARN par rapport aux protéines n'a pas empêché le développement de très nombreux modèles et outils au cours des 5 décennies passées. Le défaut du champ de recherche est l'absence de consensus sur quel outil et modèle utiliser pour réaliser une prédiction, même si le « meilleur » outil dépend du contexte. Il n'y a pour l'instant pas de règles, même avec le contexte. Il est difficile, pour le bioinformaticien non-expert, de connaitre à la fois l'étendue des possibilités d'outils disponibles, et surtout les différences entre ces outils, leurs performances respectives, leur meilleure adaptation à certaines situations particulières. En pratique, c'est souvent le référencement Google qui décide de quel outil sera utilisé, plutôt qu'un choix rationnel en connaissance de cause.

Le premier objectif serait d'arriver à établir des standards communautaires faisant consensus. Ceci passe par deux points d'action : d'abord, combiner les outils similaires en un seul, au moins au sein d'une même institution de recherche, en ajoutant une option ou un fichier de configuration pour utiliser les variantes d'algorithme ou les ensembles de paramètres différents. Ensuite, procéder systématiquement à des benchmarks pour situer un outil par rapport aux précédents et aux concurrents, et rendre les résultats publics et accessibles. Ceci peut être fait à l'échelle du groupe de recherche, comme le fait très bien le groupe de Vienne sur la page de téléchargement de ses outils (https://www.tbi.univie.ac.at/RNA/#performance), ou dans des méta-analyses publiées de façon indépendante. Par exemple, l'expérience communautaire RNA-Puzzles [74,237–239] a pour objectif de comparer la performance des méthodes de prédiction de structures 3D à l'aveugle, sur de nouvelles structures tenues secrètes avant publication, sur le modèle de l'expérience CASP pour les protéines. Saluant la démarche de RNA-Puzzles, on constate cependant que peu d'auteurs d'outils y participent, souvent les mêmes, et que les performances stagnent d'année en année. Un autre défaut de RNA-Puzzles est l'éparpillement des puzzles dans le temps, une nouvelle structure est proposée tous les quelques mois, mais cela n'a pas la valeur d'un benchmark systématique de grande envergure. Un tel benchmark avait été proposé par le groupe de Bujnicki pour la prédiction de structure secondaire sur le serveur CompaRNA [266], une très bonne initiative, malheureusement jamais mise à jour avec les outils récents et sur les données de test récentes. Il n'existe pas encore de benchmark automatique des méthodes de prédiction de structure 3D, notamment parce qu'il n'existait pas, avant RNANet, de jeu de données 3D facilement exploitable. C'est l'une des applications potentielles de RNANet.

VI.2.2 L'optimisation multicritère est aussi un outil de comparaison

L'hypothèse centrale à l'origine de cette thèse suppose que lorsqu'on ne sait pas ou ne veut pas choisir entre deux ou plusieurs modèles, quelle qu'en soit la raison, on peut chercher à obtenir les solutions du front de Pareto du problème d'optimisation multi-objectif formé par ces modèles. Les solutions du front, organisées dans l'espace entre plusieurs axes, permettent de visualiser un peu de la diversité des structures possibles mais en ayant sélectionné les meilleures selon les critères choisis, et surtout de comparer les scores des meilleures solutions sur un axe aux scores des meilleures solutions sur un autre.

Nous avons appliqué cette méthode avec succès, dans un programme bi-objectif de prédiction des structures secondaires maximisant l'exactitude attendue (MEA) et un critère d'insertions de modules 3D connus. Puis, nous l'avons appliquée en plus grande dimension, en optimisant conjointement un modèle multi-échelle à 3 ou 4 critères (l'énergie de la structure selon un modèle 2D et un modèle 3D, et la compatibilité avec la forme des ARN d'une famille connue, ou la compatibilité avec des données expérimentales de sondage chimique). On peut tirer plusieurs sortes de conclusion de ces analyses : en commentant la forme du front, on peut estimer la corrélation des critères comparés. Les critères sont très corrélés si un faible nombre de solutions domine tout l'espace, le front de Pareto est très réduit ou très rectangulaire, avec une courbe en forme de « genou » marqué. A l'inverse, si le front s'étale selon une courbe faiblement convexe, les critères ne sont pas équivalents. Ensuite, lorsqu'une structure est connue, on peut repérer sa position par rapport au front. Si elle est proche d'une extrémité du front, seul un sous-ensemble des critères est réellement utile à sa prédiction. Si elle ne fait pas partie du front, en particulier si elle semble dominée par le front, alors les critères choisis ne sont pas pertinents pour faire de la prédiction de structures (ou alors la structure n'est pas native, ou pas cohérente avec la paramétrisation des critères choisie).

L'optimisation multicritère est donc d'abord un outil permettant d'étudier la relation entre modèles de prédiction, plus qu'un outil permettant d'augmenter la performance de prédiction des modèles qu'elle combine. La performance n'augmente que dans le cas où, premièrement les critères sont tous pertinents et non corrélés, deuxièmement la quantité de solutions dans le front de Pareto obtenu est suffisamment faible pour considérer que la méthode donne un bon enrichissement.

VI.3 Résumé des contributions et perspectives

VI.3.1 Résumé de la thèse en quatre points

BiORSEO et l'insertion de modules dans les structures secondaires : La première contribution de cette thèse porte sur l'utilisation de modules (motifs 3D, réseaux d'interactions, plusieurs modèles existent) pour faire de la prédiction de structure secondaire. En s'inspirant des outils RNA-MoIP et BiokoP, tous deux basés sur la programmation linéaire en nombres entiers, on a développé un nouvel outil BiORSEO qui résout un problème d'optimisation bi-objectif pour prédire des structures secondaires en maximisant la probabilité d'existence des appariements et en y insérant des modules connus. Plusieurs fonctions objectifs ont été proposées et testées pour l'insertion de modules, ainsi que plusieurs façons de détecter les modules dans les séquences, et plusieurs modèles de modules. Un benchmark de grande ampleur permet de conclure que la prédiction de structure par repérage des sites d'insertion de modules n'améliore pas la performance par rapport aux outils existants. La performance est correcte cependant, on pourrait plutôt chercher à valoriser BiORSEO comme outil d'insertion de modules en soi (bien qu'il fasse appel à BayesPairing ou Jar3d). Des applications pour la prise en compte de contraintes pour la prédiction de structures 3D sont possibles (comme le faisait le pipeline RNA-MoIP+MC-Sym), ou tout simplement à la recherche de sites de contact avec des protéines, ces sites pouvant être définis comme des modules avec une suite de nucléotides adoptant une certaine structure secondaire.

RNANet, un jeu de données de référence à grand potentiel : La thèse s'est déroulée dans un contexte de grand succès des méthodes d'apprentissage profond dans le domaine des protéines, alors que rien n'existait encore pour l'ARN. Je me suis intéressé à l'entrainement de réseaux de neurones pour les prédiction de caractéristiques de l'ARN et de structures 3D complètes. Cependant, j'ai très vite remarqué le manque de données, leur difficulté d'accessibilité et/ou d'intégration, et leur manque de standardisation. J'ai donc investi plusieurs mois et le travail d'étudiants en stage au développement d'un jeu de données automatiquement construit et mis à jour à partir des bases de données publiques. Les données sont organisées et classées selon leur famille et la résolution des structures. On peut utiliser les fichiers texte prêts à l'emploi directement avec un outil d'apprentissage, ou utiliser la base de données SQLite pour exécuter des requêtes complexes. Les chaines d'ARN ont été isolées et renumérotées de façon standard. Les coordonnées de chaque position dans les alignements multiples de séquence ou dans les modèles de covariance Rfam correspondants sont fournis et directement exploitables, ainsi que les annotations fournies par DSSR. RNANet remplace le besoin d'intégrer ensemble des bases de données de séquences, la Protein Data Bank, et les outils d'alignement et d'annotation, faisant économiser de longues heures de calcul à l'informaticien qui l'utilise. De plus, des statistiques utiles sont calculées sur le jeu de données final et sont mises à disposition, comme les clusters de pseudo-torsions, les matrices moyennes de distance entre bases au sein d'une famille, ou les distributions des distances et angles entre atomes. Ces mesures aussi sont donc en permanence à jour.

Les applications possibles de RNANet sont donc évidemment l'entrainement de modèles d'apprentissage supervisé, mais aussi le minage de données en général, pour entrainer des potentiels statistiques dans les champs de forces, pour créer des bases de données de modules, ou encore pour mettre en place des plateformes de benchmark automatiques d'outils bioinformatiques. Il est probable que la recherche en apprentissage profond reste fortement active dans la décennie à venir, RNANet a donc un gros potentiel pour la communauté. D'autres outils ont déjà utilisé des jeux de données similaires, soit de moins bonne qualité, soit tout simplement construits de façon non automatique et donc non mis à jour. Sachant que la quantité de données d'ARN non codants disponibles en 3D va probablement augmenter au fur et à mesure que la cryo-microscopie électronique se démocratise, on s'attend à voir augmenter la qualité du jeu de données, tant en quantité des points de données qu'en diversité. Il est donc important de garder un jeu de données à jour dans les années à venir pour profiter des dernières structures, potentiellement uniques en leur genre, au fur et à mesure de leur découverte.

De nombreuses idées de développement et d'amélioration sont encore en liste d'attente dans l'équipe, il est probable que le développement de RNANet continue à l'issue de cette thèse, notamment pour atteindre l'exhaustivité (de nombreuses structures 3D sont encore inexploitées car absentes de la liste non-redondante de BGSU ou non affiliées à une famille Rfam).

L'apprentissage profond pour la prédiction de structures d'ARN : Dans la suite du chapitre IV, on a proposé plusieurs projets de modèles d'apprentissage profond pour la prédiction de structures. L'adaptation du réseau géométrique récurrent à l'ARN d'abord, sur laquelle j'ai travaillé moi-même et assez tôt, qui a motivé le développement de RNANet inspiré par ProteinNet. Puis plus tard, des modèles à étudier ont été proposés en stage à deux étudiants du master GENIOMHE, basés sur les réseaux de convolution, pour prédire les matrices de distances ou les angles de torsions des chaines. Ces différents modèles ont pour l'instant échoué, soit par manque de temps accordé à l'ingénierie (débogage, optimisation, exploration et réglage des hyperparamètres), soit parce que le problème n'est pas soluble en tant que tel avec les données et les caractéristiques choisies. Ces essais restent les premiers à ma connaissance s'intéressant à la structure 3D des ARN. Ils pourront être repris et améliorés au fil du temps, maintenant que les principales difficultés techniques ont été identifiées et documentées. Une implémentation est déjà disponible, ce qui facilitera leur redéploiement.

MOARNA et la combinaison arbitraire de modèles différents à co-optimiser : Dans le chapitre V, on a développé l'idée centrale de la thèse : utiliser l'optimisation multicritère pour obtenir des structures 3D sur le front de Pareto d'un problème d'optimisation combinant plusieurs modèles, à plusieurs échelles, paramétrés de façon différente, et exploitant un maximum d'informations. La première qualité de ce projet fut pédagogique : il a nécessité l'exploration de la littérature dans de nombreux aspects de la modélisation des ARN (énergie des structures secondaires, construction des champs de forces, méthode expérimentales de sondage chimique) ainsi que la mobilisation de compétences variées comme l'algèbre et la géométrie euclidienne dans l'espace,

l'apprentissage d'un nouveau langage de programmation bien adapté au projet, le Go (pour sa facilité d'utilisation en prototypage d'idées et en mise en place de procédures de calcul parallèle), ou l'apprentissage des formats de fichiers standards utilisés en bioinformatique (mmCIF, PDB, RDAT en particulier). La contribution en elle-même, l'outil MOARNA, reste un prototype pour l'instant. Les solutions renvoyées ne rivalisent pas encore avec celles des outils concurrents. Il est également encore trop tôt pour une publication dans un journal à comité de lecture, qui suivra probablement lorsque la paramétrisation aura été faite de manière plus rigoureuse et documentée, ainsi que l'exploration des hyperparamètres, suivie de tests rigoureux. Cependant, on a déjà pu montrer l'influence qu'ont certains critères sur les autres, et leur corrélation. L'outil est donc plutôt un outil de comparaison des critères entre eux qu'un outil de prédiction de structures. On regrette également le manque d'une contribution théorique au champ de recherche de l'optimisation multicritères, qui avait été souhaitée en début de thèse, notamment pour les problèmes mixtes. L'algorithme génétique multicritère utilisé, NSGA-III, est simple mais pas très efficace, des recherches théoriques sur l'algorithme seraient utiles à l'avenir.

VI.3.2 Conclusion générale

Dans l'ensemble, cette thèse a permis une familiarisation très large avec la modélisation des structures d'ARN, les modèles et outils utilisés dans ce champ de recherche. On a proposé deux problèmes d'optimisation multicritères pour la prédiction de structures, l'un bi-objectif (avec l'outil BiORSEO), l'autre pouvant utiliser jusqu'à quatre critères en fonction des données disponibles pour l'instant (avec l'outil MOARNA), et extensible sans limite théorique à de nouveaux critères.

On s'est également intéressés à l'apprentissage profond, méthodologie incontournable en bioinformatique structurale. On a notamment conçu le premier jeu de données standardisé de structures d'ARN multi-échelle construit et mis à jour automatiquement (RNANet), qu'on supposera utile à la communauté à l'avenir.

La thèse a donné lieu à deux publications dans le *Bioinformatics* (Oxford Press) en janvier et novembre 2020, ainsi que des présentations dans des conférences et séminaires scientifiques nationaux et internationaux. Une publication des travaux du chapitre IV est envisagée prochainement. Ceux du chapitre V ne sont pas encore matures pour publication mais pourraient l'être dans un futur proche.

Remerciements

Cette thèse est le fruit de la collaboration avec de nombreuses personnes qu'il me faut remercier pour les riches expériences humaines vécues ces trois dernières années. Je commencerai naturellement par remercier mes encadrants de thèse, Fariza Tahi et Eric Angel, pour la grande confiance qu'ils m'ont vite accordée dans la gestion de mon avancement. Ils m'ont également fait confiance dans le choix de mes directions de recherche, et de celles des stagiaires que nous avons co-encadrés. Fariza a été une directrice très arrangeante, flexible, toujours disponible et toujours réactive très rapidement, ce qui a été très apprécié. Éric a su s'intéresser pour moi à la bioinformatique structurale, qui n'est pas son domaine d'origine. Il a surveillé la bibliographie et a toujours été capable de suggérer une bonne publication lorsqu'un problème technique nous bloquait. Je mentionne aussi la qualité de la relation que nous avons eu dans les activités d'enseignement à l'université.

Je remercie ensuite les étudiants qui se sont montrés intéressés par nos sujets de stage sur ces thématiques. Lénaïc Durand (en 3e année à l'ENSIIE) et Nathalie Bernard (en master AMIS de l'Université de Versailles) ont travaillé sur BiORSEO pour implémenter quelques améliorations souhaitables, et ont vaillamment su comprendre et utiliser le code cryptique de cet outil. Aglaé Tabot (en 2e année à l'école Centrale de Nantes), pour son investissement sur le code de RNANet, lui aussi extrêmement long et complexe, ayant détecté et corrigé de nombreux bogues et permis d'automatiser le calcul de mesures géométriques sur les données. Léa Boulos (en master GENIOMHE), qui s'est spontanément proposée pour un stage dans l'équipe avant même que nous ayons proposé un sujet, et qui est maintenant déterminée à réussir à prédire ces angles de torsion qui nous résistent. Enfin, Khodor Hannoush (en master GENIOMHE), qui a réellement su comprendre les difficultés pour appliquer l'apprentissage profond aux structures d'ARN. Khodor a été à l'initiative de nombreuses idées, et a également contribué à l'amélioration de RNANet et sa gestion des alignements multiples de séquences. Il fut un interlocuteur de qualité pendant toute la dernière année de thèse sur ces sujets. J'adresse enfin un remerciement général aux autres étudiants du master GENIOMHE et de l'ENSIIE que j'ai fait travailler sur ces sujets dans des projets transversaux en bioinformatique, et dont certains ont fourni des résultats novateurs dans ce cadre.

Pour les échanges scientifiques, pour leurs conseils ponctuels et pour le temps qu'ils m'ont accordé, je voudrais également remercier Roman Sarrazin-Gendron (Univ. McGill), Samuela Pasquali (Univ. Paris Descartes), Craig Zirbel (BGSU), Xiang-Jun Lu (Univ. Columbia), François Major (IRIC Montréal), Vladimir Reinharz (Univ. McGill), Yann Ponty (LIX), Blaise Hanczar (IBISC), Isaure Chauvot de Beauchêne (LORIA), Christophe Blanchet (IFB), Ioanna Kalvari (EMBL), Farida Zehraoui (IBISC), Christian Hoener zu Siederdissen (TBI Vienna), Florian Eggenhofer (Univ. Freiburg), Savandara Besse (Univ. Montréal), Audrey Legendre (IBISC), Clément Bertrand (IBISC), Tina Issa (IBISC), Victoria Bourgeais (IBISC), Thang Nguyen (IBISC), Miriam López-Ramos (GLPG), Arthur Zalevsky (IBCh Moscou), et Diego Gallego Perez (IRB Barcelone).

Parmi les collègues d'IBISC, je remercie une deuxième fois Blaise Hanczar pour avoir spontanément accepté de co-superviser nos stagiaires en apprentissage profond. Il est également animateur du groupe de lecture interne « deep-learning », très instructif. Je remercie vivement Ludovic Ishiomin pour son support technique et la mise en place de machines de calcul indispensables à cette thèse et au travail des stagiaires. J'ai réellement acquis des compétences pratiques grâce à nos discussions. Je remercie aussi Sergiu Ivanov et Jean-Christophe Janodet pour nos nombreuses discussions, d'intérêt scientifique ou non, mais toujours passionnantes. Merci encore à Tina pour avoir constitué avec moi un groupe de lecture de l'intégrale des publications de Jean-Antoine Désidéri (dont aucune n'est finalement citée dans cette thèse).

Je remercie enfin tous ceux qui ont participé à la bonne qualité de ma vie sociale à IBISC et en dehors, élément indispensable à la réussite d'une thèse dans de bonnes conditions psychologiques. Murielle, pour ta surveillance active de cet aspect. Victoria, pour tes nombreux engagements administratifs et associatifs que tu tiens. Clément et Johan, pour vos propositions de jeux et d'activités sociales au sein du laboratoire. Jérémie, Mahmoud, Emmanuel, Najett et Mickael pour nos activités associatives sur le développement de drones et nos expériences fortes aux concours DroneLoad. Célia, Junkai, Ying Li, Naziha, Cristel, Mohamed, Ahmed, Sophie, Paul et Savandara depuis l'iGEM 2016, et encore une fois mon inénarrable collègue de bureau Tina : merci pour votre amitié.

Bibliographie

[1] Aarts E & Korst J. *Simulated Annealing and Boltzmann Machines: A Stochastic Approach to Combinatorial Optimization and Neural Computing*. John Wiley & Sons, Inc., USA, 1989.

[2] Abu Almakarem AS, Petrov AI, Stombaugh J, Zirbel CL & Leontis NB. Comprehensive survey and geometric classification of base triples in RNA structures. *Nucleic Acids Research* (2012); **40**:1407–23, DOI: 10.1093/nar/gkr810.

[3] Aduri R, Psciuk BT, Saro P, Taniga H, Schlegel HB & SantaLucia J. AMBER Force Field Parameters for the Naturally Occurring Modified Nucleosides in RNA. *Journal of Chemical Theory and Computation* (2007); **3**:1464–75, DOI: 10.1021/ct600329w.

[4] Akiyama M, Sakakibara Y & Sato K. NeuraFold: Direct inference of base-pairing probabilities with neural networks improves RNA secondary structure prediction with pseudoknots. *bioRxiv* (2018):303172, DOI: 10.1101/303172.

[5] Akiyama M, Sato K & Sakakibara Y. MXFold: A max-margin training of RNA secondary structure prediction integrated with the thermodynamic model. *Journal of Bioinformatics and Computational Biology* (2018); **16**:1840025, DOI: 10.1142/S0219720018400255.

[6] Akutsu T. Dynamic programming algorithms for RNA secondary structure prediction with pseudoknots. *Discrete Applied Mathematics* (2000); **104**:45–62, DOI: 10.1016/S0166-218X(00)00186-4.

[7] Alford RF *et al.* The Rosetta All-Atom Energy Function for Macromolecular Modeling and Design. *Journal of Chemical Theory and Computation* (2017); **13**:3031–48, DOI: 10.1021/acs.jctc.7b00125.

[8] AlQuraishi M. AlphaFold at CASP13. *Bioinformatics*, DOI: 10.1093/bioinformatics/btz422. Accessed 9 September 2019, http://academic.oup.com/bioinformatics/advance-article/doi/10.1093/bioinformatics/btz422/5497249.

[9] AlQuraishi M. AlphaFold2 @ CASP14: “It feels like one’s child has left home.”AlphaFold2 @ CASP14: “It feels like one’s child has left home.,” https://moalquraishi.wordpress.com/2020/12/08/alphafold2-casp14-it-feels-like-ones-child-has-left-home/, accessed 2021/05/18

[10] AlQuraishi M. End-to-End Differentiable Learning of Protein Structure. *Cell Systems* (2019); **8**:292-301.e3, DOI: 10.1016/j.cels.2019.03.006.

[11] AlQuraishi M. Parallelized Natural Extension Reference Frame: Parallelized Conversion from Internal to Cartesian Coordinates. *Journal of Computational Chemistry* (2019); **40**:885–92, DOI: 10.1002/jcc.25772.

[12] AlQuraishi M. ProteinNet: a standardized data set for machine learning of protein structure. *BMC Bioinformatics* (2019); **20**:311, DOI: 10.1186/s12859-019-2932-0.

[13] Altschul SF, Gish W, Miller W, Myers EW & Lipman DJ. Basic local alignment search tool. *Journal of Molecular Biology* (1990); **215**:403–10, DOI: 10.1016/S0022-2836(05)80360-2.

[14] Andronescu M, Bereg V, Hoos HH & Condon A. RNA STRAND: The RNA Secondary

Structure and Statistical Analysis Database. *BMC Bioinformatics* (2008); **9**:340, DOI: 10.1186/1471-2105-9-340.

[15] Andronescu M, Condon A, Hoos HH, Mathews DH & Murphy KP. Computational approaches for RNA energy parameter estimation. *RNA* (2010); **16**:2304–18, DOI: 10.1261/rna.1950510.

[16] Andronescu M, Condon A, Hoos HH, Mathews DH & Murphy KP. Efficient parameter estimation for RNA secondary structure prediction. *Bioinformatics* (2007); **23**:i19–28, DOI: 10.1093/bioinformatics/btm223.

[17] Ankerst M, Breunig MM, Kriegel H-P & Sander J. OPTICS: ordering points to identify the clustering structure. *Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data*. Association for Computing Machinery, New York, NY, USA, 1999, 49–60, , DOI: 10.1145/304182.304187.

[18] Antczak M, Zablocki M, Zok T, Rybarczyk A, Blazewicz J & Szachniuk M. RNAvista: a webserver to assess RNA secondary structures with non-canonical base pairs. *Bioinformatics* (2019); **35**:152–5, DOI: 10.1093/bioinformatics/bty609.

[19] Antczak M, Zok T, Osowiecki M, Popenda M, Adamiak RW & Szachniuk M. RNAfitme: a webserver for modeling nucleobase and nucleoside residue conformation in fixed-backbone RNA structures. *BMC Bioinformatics* (2018); **19**:304, DOI: 10.1186/s12859-018-2317-9.

[20] Antczak M, Zok T, Popenda M, Lukasiak P, Adamiak RW, Blazewicz J & Szachniuk M. RNApdbee—a webserver to derive secondary structures from pdb files of knotted and unknotted RNAs. *Nucleic Acids Research* (2014); **42**:W368–72, DOI: 10.1093/nar/gku330.

[21] Apostolico A, Ciriello G, Guerra C, Heitsch CE, Hsiao C & Williams LD. Finding 3D motifs in ribosomal RNA structures. *Nucleic Acids Research* (2009); **37**:e29–e29, DOI: 10.1093/nar/gkn1044.

[22] Aytenfisu AH, Spasic A, Grossfield A, Stern HA & Mathews DH. Revised RNA Dihedral Parameters for the Amber Force Field Improve RNA Molecular Dynamics. *Journal of Chemical Theory and Computation* (2017); **13**:900–15, DOI: 10.1021/acs.jctc.6b00870.

[23] Ban N. The Complete Atomic Structure of the Large Ribosomal Subunit at 2.4 A Resolution. *Science* (2000); **289**:905–20, DOI: 10.1126/science.289.5481.905.

[24] van Batenburg FH, Gultyaev AP & Pleij CW. An APL-programmed genetic algorithm for the prediction of RNA secondary structure. *Journal of Theoretical Biology* (1995); **174**:269–80, DOI: 10.1006/jtbi.1995.0098.

[25] Bayrak CS, Kim N & Schlick T. RAGTOP: Using sequence signatures and kink-turn motifs in knowledge-based statistical potentials for RNA structure prediction. *Nucleic Acids Research* (2017); **45**:5414–22, DOI: 10.1093/nar/gkx045.

[26] Becquey L, Angel E & Tahi F. BiORSEO: a bi-objective method to predict RNA secondary structures with pseudoknots using RNA 3D modules. *Bioinformatics* (2020); **36**:2451–7, DOI: 10.1093/bioinformatics/btz962.

[27] Becquey L, Angel E & Tahi F. RNANet: an automatically built dual-source dataset integrating homologous sequences and RNA structures. *Bioinformatics* (2020), DOI: 10.1093/bioinformatics/btaa944. Publié le 2 November 2020, Accessed 16 February 2021, https://doi.org/10.1093/bioinformatics/btaa944.

[28] Bell DR, Cheng SY, Salazar H & Ren P. RACER: Capturing RNA Folding Free Energy with Coarse-Grained Molecular Dynamics Simulations. *Scientific Reports* (2017); **7**:45812, DOI: 10.1038/srep45812.

[29] Bellaousov S & Mathews DH. ProbKnot: Fast prediction of RNA secondary structure including pseudoknots. *RNA* (2010); **16**:1870–80, DOI: 10.1261/rna.2125310.

[30] Bergonzo C, Henriksen NM, Roe DR & Cheatham TE. Highly sampled tetranucleotide and tetraloop motifs enable evaluation of common RNA force fields. *RNA* (2015); **21**:1578–90, DOI: 10.1261/rna.051102.115.

[31] Berman HM, Olson WK, Beveridge DL, Westbrook J, Gelbin A, Demeny T, Hsieh SH, Srinivasan AR & Schneider B. The nucleic acid database. A comprehensive relational database of three-dimensional structures of nucleic acids. *Biophysical Journal* (1992); **63**:751–9.

[32] Berman HM, Westbrook J, Feng Z, Gilliland G, Bhat TN, Weissig H, Shindyalov IN & Bourne PE. The Protein Data Bank. *Nucleic Acids Research* (2000); **28**:235–42, DOI: 10.1093/nar/28.1.235.

[33] Bernauer J, Huang X, Sim AYL & Levitt M. RNAkb: Fully differentiable coarse-grained and all-atom knowledge-based potentials for RNA structure evaluation. *RNA* (2011); **17**:1066–75, DOI: 10.1261/rna.2543711.

[34] Bernhart SH, Hofacker IL & Stadler PF. Local RNA base pairing probabilities in large sequences. *Bioinformatics* (2006); **22**:614–5, DOI: 10.1093/bioinformatics/btk014.

[35] Bernhart SH, Hofacker IL, Will S, Gruber AR & Stadler PF. RNAalifold: improved consensus structure prediction for RNA alignments. *BMC Bioinformatics* (2008); **9**:474, DOI: 10.1186/1471-2105-9-474.

[36] Bernhart SH, Tafer H, Mückstein U, Flamm C, Stadler PF & Hofacker IL. RNAcofold: Partition function and base pairing probabilities of RNA heterodimers. *Algorithms for Molecular Biology* (2006); **1**:3, DOI: 10.1186/1748-7188-1-3.

[37] Bida JP & Maher LJ. RSim: Improved prediction of RNA tertiary structure with insights into native state dynamics. *RNA* (2012); **18**:385–93, DOI: 10.1261/rna.027201.111.

[38] Bindewald E, Wendeler M, Legiewicz M, Bona MK, Wang Y, Pritt MJ, Grice SFJL & Shapiro BA. Correlating SHAPE signatures with three-dimensional RNA structures. *RNA* (2011); **17**:1688–96, DOI: 10.1261/rna.2640111.

[39] Blank J, Deb K & Roy PC. Investigating the Normalization Procedure of NSGA-III. In: Deb K, Goodman E, Coello Coello CA, Klamroth K, Miettinen K, Mostaghim S, Reed P (eds.). *Evolutionary Multi-Criterion Optimization*. Springer International Publishing, Cham, 2019, 229–40, , DOI: 10.1007/978-3-030-12598-1_19.

[40] Boccaletto P *et al.* RNArchitecture: a database and a classification system of RNA families, with a focus on structural information. *Nucleic Acids Research* (2018); **46**:D202–5, DOI: 10.1093/nar/gkx966.

[41] Boniecki MJ, Lach G, Dawson WK, Tomala K, Lukasz P, Soltysinski T, Rother KM & Bujnicki JM. SimRNA: a coarse-grained method for RNA folding simulations and 3D structure prediction. *Nucleic Acids Research* (2016); **44**:e63–e63, DOI: 10.1093/nar/gkv1479.

[42] Bottaro S, Di Palma F & Bussi G. eSCORE: The role of nucleobase interactions in RNA structure and dynamics. *Nucleic Acids Research* (2014); **42**:13306–14, DOI: 10.1093/nar/gku972.

[43] Bottaro S, Palma FD & Bussi G. ESCORE: Towards de novo RNA 3D Structure Prediction. *RNA & DISEASE* (2015); **2**, DOI: 10.14800/rd.544. Publié le 27 January 2015, Accessed 4 June 2018, http://www.smartscitech.com/index.php/RD/article/view/544.

[44] Boudard M, Barth D, Bernauer J, Denise A & Cohen J. GARN2: coarse-grained prediction of 3D structure of large RNA molecules by regret minimization. *Bioinformatics* (2017); **33**:2479–86, DOI: 10.1093/bioinformatics/btx175.

[45] Boudard M, Bernauer J, Barth D, Cohen J & Denise A. GARN: Sampling RNA 3D Structure Space with Game Theory and Knowledge-Based Scoring Strategies. *PLOS ONE* (2015); **10**:e0136444, DOI: 10.1371/journal.pone.0136444.

[46] Burachik RS, Kaya CY & Rizvi MM. Algorithms for Generating Pareto Fronts of Multi-objective Integer and Mixed-Integer Programming Problems. *arXiv:190307041 [math]* (2019). Publié le 17 March 2019, Accessed 14 October 2020, http://arxiv.org/abs/1903.07041.

[47] Cannone JJ *et al.* The Comparative RNA Web (CRW) Site: an online database of comparative sequence and structure information for ribosomal, intron, and other RNAs. *BMC Bioinformatics* (2002); **3**:2, DOI: 10.1186/1471-2105-3-2.

[48] Cannone JJ, Sweeney BA, Petrov AI, Gutell RR, Zirbel CL & Leontis N. R3D-2-MSA: the RNA 3D structure-to-multiple sequence alignment server. *Nucleic Acids Research* (2015); **43**:W15–23, DOI: 10.1093/nar/gkv543.

[49] Cao S & Chen S-J. Predicting RNA folding thermodynamics with a reduced chain representation model. *RNA* (2005); **11**:1884–97, DOI: 10.1261/rna.2109105.

[50] Cao S & Chen S-J. Predicting RNA pseudoknot folding thermodynamics. *Nucleic Acids Research* (2006); **34**:2634–52, DOI: 10.1093/nar/gkl346.

[51] Cao S & Chen S-J. Vfold2D+3D: Physics-based de novo prediction of RNA 3D structures. *The journal of physical chemistry B* (2011); **115**:4216–26, DOI: 10.1021/jp112059y.

[52] Capriotti E, Norambuena T, Marti-Renom MA & Melo F. RASP: All-atom knowledge-based potential for RNA structure prediction and assessment. *Bioinformatics* (2011); **27**:1086–93, DOI: 10.1093/bioinformatics/btr093.

[53] Chakraborty D, Collepardo-Guevara R & Wales DJ. Energy Landscapes, Folding Mechanisms, and Kinetics of RNA Tetraloop Hairpins. *Journal of the American Chemical Society* (2014); **136**:18052–61, DOI: 10.1021/ja5100756.

[54] Chen AA & García AE. High-resolution reversible folding of hyperstable RNA tetraloops using molecular dynamics simulations. *Proceedings of the National Academy of Sciences* (2013); **110**:16820–5, DOI: 10.1073/pnas.1309392110.

[55] Chen JL, Dishler AL, Kennedy SD, Yildirim I, Liu B, Turner DH & Serra MJ. Testing the Nearest Neighbor Model for Canonical RNA Base Pairs: Revision of GU Parameters. *Biochemistry* (2012); **51**:3508–22, DOI: 10.1021/bi3002709.

[56] Chen X, Li Y, Umarov R, Gao X & Song L. E2EFold: RNA Secondary Structure Prediction By Learning Unrolled Algorithms. *arXiv:200205810 [cs, stat]* (2020). Publié le 13 February 2020,

Accessed 29 July 2020, http://arxiv.org/abs/2002.05810.

[57] Chojnowski G, Waleń T & Bujnicki JM. RNA Bricks—a database of RNA 3D motifs and their interactions. *Nucleic Acids Research* (2014); **42**:D123–31, DOI: 10.1093/nar/gkt1084.

[58] Chojnowski G, Zaborowski R, Magnus M & Bujnicki JM. RNAMasonry: RNA fragment assembly with experimental restraints. *bioRxiv* (2021):2021.02.08.430198, DOI: 10.1101/2021.02.08.430198.

[59] Chou F-C, Kladwang W, Kappel K & Das R. Blind tests of RNA nearest-neighbor energy prediction. *Proceedings of the National Academy of Sciences* (2016); **113**:8430–5, DOI: 10.1073/pnas.1523335113.

[60] Chung FRK, Leighton FT & Rosenberg AL. Embedding Graphs in Books: A Layout Problem with Applications to VLSI Design. *SIAM Journal on Algebraic Discrete Methods* (1987); **8**:33–58, DOI: 10.1137/0608002.

[61] Chung J, Gulcehre C, Cho K & Bengio Y. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *arXiv:14123555 [cs]* (2014). Publié le 11 December 2014, Accessed 5 May 2021, http://arxiv.org/abs/1412.3555.

[62] Clote P, Dobrev S, Dotu I, Kranakis E, Krizanc D & Urrutia J. On the page number of RNA secondary structures with pseudoknots. *Journal of Mathematical Biology* (2012); **65**:1337–57, DOI: 10.1007/s00285-011-0493-6.

[63] Cock PJA *et al.* Biopython: freely available Python tools for computational molecular biology and bioinformatics. *Bioinformatics* (2009); **25**:1422–3, DOI: 10.1093/bioinformatics/btp163.

[64] COELLO CAC & CHRISTIANSEN AD. Two New Ga-Based Methods for Multiobjective Optimization. *Civil Engineering and Environmental Systems* (1998); **15**:207–43, DOI: 10.1080/02630259808970240.

[65] Coimbatore Narayanan B, Westbrook J, Ghosh S, Petrov AI, Sweeney B, Zirbel CL, Leontis NB & Berman HM. The Nucleic Acid Database: new features and capabilities. *Nucleic Acids Research* (2014); **42**:D114–22, DOI: 10.1093/nar/gkt980.

[66] Collette Y & Siarry P. *Optimisation multiobjectif: Algorithmes*. Editions Eyrolles, 2002.

[67] Cordero P, Lucks JB & Das R. An RNA Mapping DataBase for curating RNA structure mapping experiments. *Bioinformatics* (2012); **28**:3006–8, DOI: 10.1093/bioinformatics/bts554.

[68] Corne DW, Jerram NR, Knowles JD & Oates MJ. PESA-II: region-based selection in evolutionary multiobjective optimization. *Proceedings of the 3rd Annual Conference on Genetic and Evolutionary Computation*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2001, 283–90.

[69] Corne DW, Knowles JD & Oates MJ. The Pareto Envelope-Based Selection Algorithm for Multiobjective Optimization. In: Schoenauer M, Deb K, Rudolph G, Yao X, Lutton E, Merelo JJ, Schwefel H-P (eds.). *Parallel Problem Solving from Nature PPSN VI*. Springer, Berlin, Heidelberg, 2000, 839–48, , DOI: 10.1007/3-540-45356-3_82.

[70] Cornell WD *et al.* AMBER ff94: A Second Generation Force Field for the Simulation of Proteins, Nucleic Acids, and Organic Molecules. *Journal of the American Chemical Society* (1995); **117**:5179–97, DOI: 10.1021/ja00124a002.

[71] Costales MG, Matsumoto Y, Velagapudi SP & Disney MD. Small Molecule Targeted Recruitment of a Nuclease to RNA. *Journal of the American Chemical Society* (2018); **140**:6741–4, DOI: 10.1021/jacs.8b01233.

[72] Cragnolini T, Derreumaux P & Pasquali S. HiRE-RNA: Coarse-Grained Simulations of RNA and DNA Duplexes. *The Journal of Physical Chemistry B* (2013); **117**:8047–60, DOI: 10.1021/jp400786b.

[73] Cragnolini T, Laurin Y, Derreumaux P & Pasquali S. HiRE-RNA: Coarse-Grained HiRE-RNA Model for ab Initio RNA Folding beyond Simple Molecules, Including Noncanonical and Multiple Base Pairings. *Journal of Chemical Theory and Computation* (2015); **11**:3510–22, DOI: 10.1021/acs.jctc.5b00200.

[74] Cruz JA *et al.* RNA-Puzzles: A CASP-like evaluation of RNA three-dimensional structure prediction. *RNA* (2012); **18**:610–25, DOI: 10.1261/rna.031054.111.

[75] Cruz JA & Westhof E. Sequence-based identification of 3D structural modules in RNA with RMDetect. *Nature Methods* (2011); **8**:513–21, DOI: 10.1038/nmeth.1603.

[76] Dallaire P. Thèse de Doctorat : Une signature du polymorphisme structural d'acides ribonucléiques non-codants permettant de comparer leurs niveaux d'activités biochimiques.Université de Montréal (2015)

[77] Dallaire P & Major F. Exploring Alternative RNA Structure Sets Using MC-Flashfold and db2cm. In: Turner DH, Mathews DH (eds.). *RNA Structure Determination: Methods and Protocols*. Springer, New York, NY, 2016, 237–51, , DOI: 10.1007/978-1-4939-6433-8_15.

[78] Danaee P, Rouches M, Wiley M, Deng D, Huang L & Hendrix D. bpRNA: large-scale automated annotation and analysis of RNA secondary structure. *Nucleic Acids Research* (2018); **46**:5381–94, DOI: 10.1093/nar/gky285.

[79] Dans PD, Gallego D, Balaceanu A, Darré L, Gómez H & Orozco M. Modeling, Simulations, and Bioinformatics at the Service of RNA Structure. *Chem* (2019); **5**:51–73, DOI: 10.1016/j.chempr.2018.09.015.

[80] Dantzig G. *Linear Programming and Extensions*., 1998.

[81] Darty K, Denise A & Ponty Y. VARNA: Interactive drawing and editing of the RNA secondary structure. *Bioinformatics* (2009); **25**:1974–5, DOI: 10.1093/bioinformatics/btp250.

[82] Das I & Dennis JE. Normal-Boundary Intersection: A New Method for Generating the Pareto Surface in Nonlinear Multicriteria Optimization Problems. *SIAM Journal on Optimization* (1998); **8**:631–57, DOI: 10.1137/S1052623496307510.

[83] Das R. RNA structure: a renaissance begins? *Nature Methods* (2021); **18**:439–439, DOI: 10.1038/s41592-021-01132-4.

[84] Das R & Baker D. FARNA: Automated de novo prediction of native-like RNA tertiary structures. *Proceedings of the National Academy of Sciences* (2007); **104**:14664–9, DOI: 10.1073/pnas.0703836104.

[85] Das R, Karanicolas J & Baker D. FARFAR: Atomic accuracy in predicting and designing noncanonical RNA structure. *Nature Methods* (2010); **7**:291–4, DOI: 10.1038/nmeth.1433.

[86] Dawson WK, Maciejczyk M, Jankowska EJ & Bujnicki JM. Coarse-grained modeling of RNA 3D structure. *Methods* (2016); **103**:138–56, DOI: 10.1016/j.ymeth.2016.04.026.

[87] Deb K & Jain H. An Evolutionary Many-Objective Optimization Algorithm Using Reference-Point-Based Nondominated Sorting Approach, Part I: Solving Problems With Box Constraints. *IEEE Transactions on Evolutionary Computation* (2014); **18**:577–601, DOI: 10.1109/TEVC.2013.2281535.

[88] Deb K, Pratap A, Agarwal S & Meyarivan T. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation* (2002); **6**:182–97, DOI: 10.1109/4235.996017.

[89] Debye length. *Wikipedia* (2021). Publié le 9 April 2021, Accessed 25 May 2021, https://en.wikipedia.org/w/index.php?title=Debye_length&oldid=1016924779.

[90] Deigan KE, Li TW, Mathews DH & Weeks KM. Accurate SHAPE-directed RNA structure determination. *Proceedings of the National Academy of Sciences* (2009); **106**:97–102, DOI: 10.1073/pnas.0806929106.

[91] Denesyuk NA & Thirumalai D. TIS: Coarse-Grained Model for Predicting RNA Folding Thermodynamics. *The Journal of Physical Chemistry B* (2013); **117**:4901–11, DOI: 10.1021/jp401087x.

[92] Denning EJ, Priyakumar UD, Nilsson L & Mackerell AD. CHARMM36 All Atom: Impact of 2′-hydroxyl sampling on the conformational properties of RNA: Update of the CHARMM all-atom additive force field for RNA. *Journal of Computational Chemistry* (2011); **32**:1929–43, DOI: https://doi.org/10.1002/jcc.21777.

[93] Dill KA & Bromberg S. *Molecular Driving Forces: Statistical Thermodynamics in Biology, Chemistry, Physics, and Nanoscience*. 2nd ed. Garland Science, London ; New York, 2011.

[94] Dima RI, Hyeon C & Thirumalai D. Extracting Stacking Interaction Parameters for RNA from the Data Set of Native Structures. *Journal of Molecular Biology* (2005); **347**:53–69, DOI: 10.1016/j.jmb.2004.12.012.

[95] Ding F, Sharma S, Chalasani P, Demidov VV, Broude NE & Dokholyan NV. DMD: Ab initio RNA folding by discrete molecular dynamics: From structure prediction to folding mechanisms. *RNA* (2008); **14**:1164–73, DOI: 10.1261/rna.894608.

[96] Ding Y, Chan CY & Lawrence CE. RNA secondary structure prediction by centroids in a Boltzmann weighted ensemble. *RNA* (2005); **11**:1157–66, DOI: 10.1261/rna.2500605.

[97] Ding Y & Lawrence CE. Sfold: A statistical sampling algorithm for RNA secondary structure prediction. *Nucleic Acids Research* (2003); **31**:7280–301, DOI: 10.1093/nar/gkg938.

[98] Dirks RM, Bois JS, Schaeffer JM, Winfree E & Pierce NA. Thermodynamic Analysis of Interacting Nucleic Acid Strands. *SIAM Review* (2007); **49**:65–88, DOI: 10.1137/060651100.

[99] Dirks RM & Pierce NA. A partition function algorithm for nucleic acid secondary structure including pseudoknots. *Journal of Computational Chemistry* (2003); **24**:1664–77, DOI: https://doi.org/10.1002/jcc.10296.

[100] Dirks RM & Pierce NA. An algorithm for computing nucleic acid base-pairing probabilities including pseudoknots. *Journal of Computational Chemistry* (2004); **25**:1295–304, DOI:

10.1002/jcc.20057.

[101] Djelloul M & Denise A. Automated motif extraction and classification in RNA tertiary structures. *RNA* (2008); **14**:2489–97, DOI: 10.1261/rna.1061108.

[102] Do CB, Woods DA & Batzoglou S. CONTRAfold: RNA secondary structure prediction without physics-based models. *Bioinformatics* (2006); **22**:e90–8, DOI: 10.1093/bioinformatics/btl246.

[103] Dotu I, Mechery V & Clote P. Energy Parameters and Novel Algorithms for an Extended Nearest Neighbor Energy Model of RNA. *PLOS ONE* (2014); **9**:e85412, DOI: 10.1371/journal.pone.0085412.

[104] Dowell RD & Eddy SR. Evaluation of several lightweight stochastic context-free grammars for RNA secondary structure prediction. *BMC Bioinformatics* (2004); **5**:71, DOI: 10.1186/1471-2105-5-71.

[105] Dror O, Nussinov R & Wolfson H. ARTS: alignment of RNA tertiary structures. *Bioinformatics* (2005); **21**:ii47–53, DOI: 10.1093/bioinformatics/bti1108.

[106] Duarte CM & Pyle AM. Stepping through an RNA structure: a novel approach to conformational analysis (Edited by D. Draper). *Journal of Molecular Biology* (1998); **284**:1465–78, DOI: 10.1006/jmbi.1998.2233.

[107] Duarte CM, Wadley LM & Pyle AM. PRIMOS: RNA structure comparison, motif search and discovery using a reduced representation of RNA conformational space. *Nucleic Acids Research* (2003); **31**:4755–61, DOI: 10.1093/nar/gkg682.

[108] Eddy SR. A memory-efficient dynamic programming algorithm for optimal alignment of a sequence to an RNA secondary structure. *BMC Bioinformatics* (2002); **3**:18, DOI: 10.1186/1471-2105-3-18.

[109] Eddy SR & Durbin R. RNA sequence analysis using covariance models. *Nucleic Acids Research* (1994); **22**:2079–88, DOI: 10.1093/nar/22.11.2079.

[110] Eggenhofer F, Hofacker IL & Höner zu Siederdissen C. RNAlien – Unsupervised RNA family model construction. *Nucleic Acids Research* (2016); **44**:8433–41, DOI: 10.1093/nar/gkw558.

[111] Ekeberg M, Lövkvist C, Lan Y, Weigt M & Aurell E. Improved contact prediction in proteins: Using pseudolikelihoods to infer Potts models. *Physical Review E* (2013); **87**:012707, DOI: 10.1103/PhysRevE.87.012707.

[112] Elman JL. Finding Structure in Time. *Cognitive Science* (1990); **14**:179–211, DOI: https://doi.org/10.1207/s15516709cog1402_1.

[113] Emmerich M, Beume N & Naujoks B. An EMO algorithm using the hypervolume measure as selection criterion. *Proceedings of the Third International Conference on Evolutionary Multi-Criterion Optimization*. Springer-Verlag, Berlin, Heidelberg, 2005, 62–76, , DOI: 10.1007/978-3-540-31880-4_5.

[114] Emmerich MTM & Deutz AH. A tutorial on multiobjective optimization: fundamentals and evolutionary methods. *Natural Computing* (2018); **17**:585–609, DOI: 10.1007/s11047-018-9685-y.

[115] Engelen S & Tahi F. Tfold: efficient in silico prediction of non-coding RNA secondary

structures. *Nucleic Acids Research* (2010); **38**:2453–66, DOI: 10.1093/nar/gkp1067.

[116] Engrand P. A multi-objective optimization approach based on simulated annealing and its application to nuclear fuel management. (1998). Publié le 31 December 1998, Accessed 23 May 2021, https://www.osti.gov/etdeweb/biblio/316961.

[117] Entzian G & Raden M. pourRNA—a time- and memory-efficient approach for the guided exploration of RNA energy landscapes. *Bioinformatics* (2020); **36**:462–9, DOI: 10.1093/bioinformatics/btz583.

[118] Erfani T & Utyuzhnikov SV. DSD: Directed search domain: a method for even generation of the Pareto frontier in multiobjective optimization. *Engineering Optimization* (2011); **43**:467–84, DOI: 10.1080/0305215X.2010.497185.

[119] Flamm C, Fontana W, Hofacker IL & Schuster P. Kinfold: RNA folding at elementary step resolution. *RNA* (2000); **6**:325–38, DOI: 10.1017/S1355838200992161.

[120] Flamm C, Hofacker IL, Stadler PF & Wolfinger MT. Barriers: Barrier Trees of Degenerate Landscapes. (2002); **216**:155, DOI: 10.1524/zpch.2002.216.2.155.

[121] Flores SC, Sherman M, Bruns CM, Eastman P & Altman RB. RNABuilder: Fast Flexible Modeling of RNA Structure Using Internal Coordinates. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* (2011); **8**:1247–57, DOI: 10.1109/TCBB.2010.104.

[122] Foloppe N & Jr ADM. CHARMM27: All-atom empirical force field for nucleic acids: I. Parameter optimization based on small molecule and condensed phase macromolecular target data. *Journal of Computational Chemistry* (2000); **21**:86–104, DOI: https://doi.org/10.1002/(SICI)1096-987X(20000130)21:2<86::AID-JCC2>3.0.CO;2-G.

[123] Fonseca CM & Fleming PJ. Genetic Algorithms for Multiobjective Optimization: FormulationDiscussion and Generalization. *Proceedings of the 5th International Conference on Genetic Algorithms*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993, 416–23.

[124] Freier SM, Kierzek R, Jaeger JA, Sugimoto N, Caruthers MH, Neilson T & Turner DH. Improved free-energy parameters for predictions of RNA duplex stability. *Proceedings of the National Academy of Sciences* (1986); **83**:9373–7, DOI: 10.1073/pnas.83.24.9373.

[125] Frellsen J, Moltke I, Thiim M, Mardia KV, Ferkinghoff-Borg J & Hamelryck T. BARNACLE: A Probabilistic Model of RNA Conformational Space. *PLOS Computational Biology* (2009); **5**:e1000406, DOI: 10.1371/journal.pcbi.1000406.

[126] Fu L, Cao Y, Wu J, Peng Q, Nie Q & Xie X. UFold: Fast and Accurate RNA Secondary Structure Prediction with Deep Learning. *bioRxiv* (2021):2020.08.17.254896, DOI: 10.1101/2020.08.17.254896.

[127] Gan HH, Fera D, Zorn J, Shiffeldrim N, Tang M, Laserson U, Kim N & Schlick T. RAG: RNA-As-Graphs database—concepts, analysis, and features. *Bioinformatics* (2004); **20**:1285–91, DOI: 10.1093/bioinformatics/bth084.

[128] Gan HH, Pasquali S & Schlick T. Exploring the repertoire of RNA secondary motifs using graph theory; implications for RNA design. *Nucleic Acids Research* (2003); **31**:2926–43, DOI: 10.1093/nar/gkg365.

[129] Gao Y, Wang S, Deng M & Xu J. RaptorX-Angle: real-value prediction of protein backbone

dihedral angles through a hybrid method of clustering and deep learning. *BMC Bioinformatics* (2018); **19**:100, DOI: 10.1186/s12859-018-2065-x.

[130] Ge P, Islam S, Zhong C & Zhang S. De novo discovery of structural motifs in RNA 3D structures through clustering. *Nucleic Acids Research* (2018); **46**:4783–93, DOI: 10.1093/nar/gky139.

[131] Geman S & Geman D. Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (1984); **PAMI-6**:721–41, DOI: 10.1109/TPAMI.1984.4767596.

[132] Gendron P, Lemieux S & Major F. MC-Annotate: Quantitative analysis of nucleic acid three-dimensional structures. *Journal of Molecular Biology* (2001); **308**:919–36, DOI: 10.1006/jmbi.2001.4626.

[133] Ghane-Kanafi A & Khorram E. mCHIM: A new scalarization method for finding the efficient frontier in non-convex multi-objective problems. *Applied Mathematical Modelling* (2015); **39**:7483–98, DOI: 10.1016/j.apm.2015.03.022.

[134] Gilbert J-C. *Fragments d'Optimisation Différentiable — Théories et Algorithmes. Syllabus de cours à l'ENSTA Paris.*, 2020.

[135] Gilbert JC. Selected Topics on Continuous Optimization.Université Paris Saclay (2019)

[136] Glover F. Future paths for integer programming and links to artificial intelligence. *Computers & Operations Research* (1986); **13**:533–49, DOI: 10.1016/0305-0548(86)90048-1.

[137] Gong S, Zhang C & Zhang Y. RNA-align: quick and accurate alignment of RNA 3D structures based on size-independent TM-scoreRNA. *Bioinformatics* (2019); **35**:4459–61, DOI: 10.1093/bioinformatics/btz282.

[138] Griffiths-Jones S, Bateman A, Marshall M, Khanna A & Eddy SR. Rfam: an RNA family database. *Nucleic Acids Research* (2003); **31**:439–41, DOI: 10.1093/nar/gkg006.

[139] Gruber AR, Findeiß S, Washietl S, Hofacker IL & Stadler PF. RNAz 2.0: Improved Noncoding RNA Detection. *Biocomputing 2010*. World Scientific, 2009, 69–79, , DOI: 10.1142/9789814295291_0009.

[140] Hamada M, Kiryu H, Sato K, Mituyama T & Asai K. Prediction of RNA secondary structure using generalized centroid estimators. *Bioinformatics* (2009); **25**:465–73, DOI: 10.1093/bioinformatics/btn601.

[141] Hanson J, Paliwal K, Litfin T, Yang Y & Zhou Y. Accurate prediction of protein contact maps by coupling residual two-dimensional bidirectional long short-term memory with convolutional neural networks. *Bioinformatics* (2018); **34**:4039–45, DOI: 10.1093/bioinformatics/bty481.

[142] Harmanci AO, Sharma G & Mathews DH. TurboFold: Iterative probabilistic estimation of secondary structures for multiple RNA sequences. *BMC Bioinformatics* (2011); **12**:108, DOI: 10.1186/1471-2105-12-108.

[143] Harrison A-M, South DR, Willett P & Artymiuk PJ. NASSAM: Representation, searching and discovery of patterns of bases in complex RNA structures. *Journal of Computer-Aided Molecular Design* (2003); **17**:537–49, DOI: 10.1023/B:JCAM.0000004603.15856.32.

[144] Hastings WK. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika* (1970); **57**:97–109, DOI: 10.1093/biomet/57.1.97.

[145] He K, Zhang X, Ren S & Sun J. Deep Residual Learning for Image Recognition. *arXiv:151203385 [cs]* (2015). Publié le 10 December 2015, Accessed 5 May 2021, http://arxiv.org/abs/1512.03385.

[146] He X, Li S, Ou X, Wang J & Xiao Y. Inference of RNA structural contacts by direct coupling analysis. *Communications in Information and Systems* (2019); **19**:279–97, DOI: 10.4310/CIS.2019.v19.n3.a3.

[147] He Y, Liwo A & Scheraga HA. Optimization of a Nucleic Acids united-RESidue 2-Point model (NARES-2P) with a maximum-likelihood approach. *The Journal of Chemical Physics* (2015); **143**:243111, DOI: 10.1063/1.4932082.

[148] He Y, Maciejczyk M, Ołdziej S, Scheraga HA & Liwo A. NARES-2P: Mean-Field Interactions between Nucleic-Acid-Base Dipoles can Drive the Formation of a Double Helix. *Physical Review Letters* (2013); **110**:098101, DOI: 10.1103/PhysRevLett.110.098101.

[149] Héliou A. Thèse de Doctorat : Molecular conformations and game theory.Université Paris-Saclay (ComUE) (2017)

[150] Hochreiter S & Schmidhuber J. Long Short-Term Memory. *Neural Computation* (1997); **9**:1735–80, DOI: 10.1162/neco.1997.9.8.1735.

[151] Hoeser T & Kuenzer C. Object Detection and Image Segmentation with Deep Learning on Earth Observation Data: A Review-Part I: Evolution and Recent Trends. *Remote Sensing* (2020); **12**:1667, DOI: 10.3390/rs12101667.

[152] Hofacker IL, Fekete M & Stadler PF. RNAalifold: Secondary Structure Prediction for Aligned RNA Sequences. *Journal of Molecular Biology* (2002); **319**:1059–66, DOI: 10.1016/S0022-2836(02)00308-X.

[153] Hofacker IL, Fontana W, Stadler PF, Bonhoeffer LS, Tacker M & Schuster P. Fast folding and comparison of RNA secondary structures. *Monatshefte für Chemie / Chemical Monthly* (1994); **125**:167–88, DOI: 10.1007/BF00818163.

[154] Hofacker IL, Priwitzer B & Stadler PF. RNALfold: Prediction of locally stable RNA secondary structures for genome-wide surveys. *Bioinformatics* (2004); **20**:186–90, DOI: 10.1093/bioinformatics/btg388.

[155] Hopfinger MC, Kirkpatrick CC & Znosko BM. Predictions and analyses of RNA nearest neighbor parameters for modified nucleotides. *Nucleic Acids Research* (2020); **48**:8901–13, DOI: 10.1093/nar/gkaa654.

[156] Horn J, Nafpliotis N & Goldberg DE. A niched Pareto genetic algorithm for multiobjective optimization. *Proceedings of the First IEEE Conference on Evolutionary Computation. IEEE World Congress on Computational Intelligence*. 1994, 82–7 vol.1, , DOI: 10.1109/ICEC.1994.350037.

[157] Huang L, Zhang H, Deng D, Zhao K, Liu K, Hendrix DA & Mathews DH. LinearFold: linear-time approximate RNA folding by 5’-to-3’ dynamic programming and beam search. *Bioinformatics* (2019); **35**:i295–304, DOI: 10.1093/bioinformatics/btz375.

[158] Humphris-Narayanan E & Pyle AM. Discrete RNA Libraries from Pseudo-Torsional Space.

Journal of Molecular Biology (2012); **421**:6–26, DOI: 10.1016/j.jmb.2012.03.002.

[159] Islam S, Rahaman MM & Zhang S. RNAMotifContrast: a method to discover and visualize RNA structural motif subfamilies. *Nucleic Acids Research* (2021), DOI: 10.1093/nar/gkab131. Publié le 8 March 2021, Accessed 14 April 2021, https://doi.org/10.1093/nar/gkab131.

[160] Jaderberg M, Simonyan K, Zisserman A & Kavukcuoglu K. Spatial transformer networks. *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*. MIT Press, Cambridge, MA, USA, 2015, 2017–25.

[161] Jain S, Laederach A, Ramos SBV & Schlick T. A pipeline for computational design of novel RNA-like topologies. *Nucleic Acids Research* (2018); **46**:7040–51, DOI: 10.1093/nar/gky524.

[162] Jain S & Schlick T. F-RAG: Generating Atomic Coordinates from RNA Graphs by Fragment Assembly. *Journal of Molecular Biology* (2017); **429**:3587–605, DOI: 10.1016/j.jmb.2017.09.017.

[163] Janssen S & Giegerich R. The RNA shapes studio. *Bioinformatics* (2015); **31**:423–5, DOI: 10.1093/bioinformatics/btu649.

[164] Jian Y, Wang X, Qiu J, Wang H, Liu Z, Zhao Y & Zeng C. DIRECT: RNA contact predictions by integrating structural patterns. *BMC Bioinformatics* (2019); **20**:497, DOI: 10.1186/s12859-019-3099-4.

[165] Jiang M. Approximation Algorithms for Predicting RNA Secondary Structures with Arbitrary Pseudoknots. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* (2010); **7**:323–32, DOI: 10.1109/TCBB.2008.109.

[166] Jonikas MA, Radmer RJ, Laederach A, Das R, Pearlman S, Herschlag D & Altman RB. NAST: Coarse-grained modeling of large RNA molecules with knowledge-based potentials and structural filters. *RNA* (2009); **15**:189–99, DOI: 10.1261/rna.1270809.

[167] Jost D & Everaers R. Prediction of RNA multiloop and pseudoknot conformations from a lattice-based, coarse-grain tertiary structure model. *The Journal of Chemical Physics* (2010); **132**:095101, DOI: 10.1063/1.3330906.

[168] Jumper J. AlphaFold 2 : Presentation at CASP14.AlphaFold 2 : Presentation at CASP14, https://predictioncenter.org/casp14/doc/presentations/2020_12_01_TS_predictor_AlphaFold2.pdf, accessed 2021/05/18

[169] Jumper J *et al.* Highly accurate protein structure prediction with AlphaFold. *Nature* (2021):1–11, DOI: 10.1038/s41586-021-03819-2.

[170] Kalé L *et al.* NAMD2: Greater Scalability for Parallel Molecular Dynamics. *Journal of Computational Physics* (1999); **151**:283–312, DOI: 10.1006/jcph.1999.6201.

[171] Kalvari I, Argasinska J, Quinones-Olvera N, Nawrocki EP, Rivas E, Eddy SR, Bateman A, Finn RD & Petrov AI. Rfam 13.0: shifting to a genome-centric resource for non-coding RNA families. *Nucleic Acids Research* (2018); **46**:D335–42, DOI: 10.1093/nar/gkx1038.

[172] Kalvari I, Nawrocki EP, Argasinska J, Quinones-Olvera N, Finn RD, Bateman A & Petrov AI. Non-Coding RNA Analysis Using the Rfam Database. *Current Protocols in Bioinformatics* (2018); **62**:e51, DOI: https://doi.org/10.1002/cpbi.51.

[173] Keating KS, Humphris EL & Pyle AM. A new way to see RNA. *Quarterly reviews of*

biophysics (2011); **44**:433–66, DOI: 10.1017/S0033583511000059.

[174] Kerpedjiev P, Hammer S & Hofacker IL. Forna (force-directed RNA): Simple and effective online RNA secondary structure diagrams. *Bioinformatics* (2015); **31**:3377–9, DOI: 10.1093/bioinformatics/btv372.

[175] Kerpedjiev P, Siederdissen CH zu & Hofacker IL. ERNWIN: Predicting RNA 3D structure using a coarse-grain helix-centered model. *RNA* (2015); **21**:1110–21, DOI: 10.1261/rna.047522.114.

[176] Kim M, Hiroyasu T, Miki M & Watanabe S. SPEA2+: Improving the Performance of the Strength Pareto Evolutionary Algorithm 2. In: Yao X et al. (eds.). *Parallel Problem Solving from Nature - PPSN VIII*. Springer, Berlin, Heidelberg, 2004, 742–51, , DOI: 10.1007/978-3-540-30217-9_75.

[177] Kim N, Laing C, Elmetwaly S, Jung S, Curuksu J & Schlick T. RNAJAG: Graph-based sampling for approximating global helical topologies of RNA. *Proceedings of the National Academy of Sciences* (2014); **111**:4079–84, DOI: 10.1073/pnas.1318893111.

[178] Kladwang W, VanLang CC, Cordero P & Das R. Understanding the Errors of SHAPE-Directed RNA Structure Modeling. *Biochemistry* (2011); **50**:8049–56, DOI: 10.1021/bi200524n.

[179] Klostermeier D & Millar DP. RNA Conformation and Folding Studied with Fluorescence Resonance Energy Transfer. *Methods* (2001); **23**:240–54, DOI: 10.1006/meth.2000.1135.

[180] Knowles JD & Corne DW. Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy. *Evolutionary Computation* (2000); **8**:149–72, DOI: 10.1162/106365600568167.

[181] Knowles JD & Corne DW. M-PAES: a memetic algorithm for multiobjective optimization. *Proceedings of the 2000 Congress on Evolutionary Computation. CEC00 (Cat. No.00TH8512)*. Vol 1. 2000, 325–32 vol.1, , DOI: 10.1109/CEC.2000.870313.

[182] Knudsen B & Hein J. Pfold: RNA secondary structure prediction using stochastic context-free grammars. *Nucleic Acids Research* (2003); **31**:3423–8, DOI: 10.1093/nar/gkg614.

[183] Knudsen B & Hein J. RNA secondary structure prediction using stochastic context-free grammars and evolutionary history. *Bioinformatics* (1999); **15**:446–54, DOI: 10.1093/bioinformatics/15.6.446.

[184] Krokhotin A, Houlihan K & Dokholyan NV. iFoldRNA v2: folding RNA with constraints. *Bioinformatics* (2015); **31**:2891–3, DOI: 10.1093/bioinformatics/btv221.

[185] Kucharík M, Hofacker IL, Stadler PF & Qin J. Pseudoknots in RNA folding landscapes. *Bioinformatics* (2016); **32**:187–94, DOI: 10.1093/bioinformatics/btv572.

[186] Kucharík M, Hofacker IL, Stadler PF & Qin J. RNAlocmin: Basin Hopping Graph: a computational framework to characterize RNA folding landscapes. *Bioinformatics* (2014); **30**:2009–17, DOI: 10.1093/bioinformatics/btu156.

[187] Kührová P, Mlýnský V, Zgarbová M, Krepl M, Bussi G, Best RB, Otyepka M, Šponer J & Banáš P. Improving the Performance of the Amber RNA Force Field by Tuning the Hydrogen-Bonding Interactions. *Journal of Chemical Theory and Computation* (2019); **15**:3288–305, DOI: 10.1021/acs.jctc.8b00955.

[188] Lai C-E, Tsai M-Y, Liu Y-C, Wang C-W, Chen K-T & Lu CL. FASTR3D: a fast and accurate search tool for similar RNA 3D structures. *Nucleic Acids Research* (2009); **37**:W287–95, DOI: 10.1093/nar/gkp330.

[189] Laing C, Jung S, Kim N, Elmetwaly S, Zahran M & Schlick T. RNAJAG: Predicting Helical Topologies in RNA Junctions as Tree Graphs. *PLOS ONE* (2013); **8**:e71947, DOI: 10.1371/journal.pone.0071947.

[190] Laing C, Wen D, Wang JTL & Schlick T. RNA Junction Explorer: Predicting coaxial helical stacking in RNA junctions. *Nucleic Acids Research* (2012); **40**:487–98, DOI: 10.1093/nar/gkr629.

[191] Lecun Y, Bottou L, Bengio Y & Haffner P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE* (1998); **86**:2278–324, DOI: 10.1109/5.726791.

[192] Legendre A, Angel E & Tahi F. Bi-objective integer programming for RNA secondary structure prediction with pseudoknots. *BMC Bioinformatics* (2018); **19**:13, DOI: 10.1186/s12859-018-2007-7.

[193] Lei G, Dou Y, Wan W, Xia F, Li R, Ma M & Zou D. CPU-GPU hybrid accelerating the Zuker algorithm for RNA secondary structure prediction applications. *BMC Genomics* (2012); **13**:S14, DOI: 10.1186/1471-2164-13-S1-S14.

[194] Lemieux S & Major F. Automated extraction and classification of RNA tertiary structure cyclic motifs. *Nucleic Acids Research* (2006); **34**:2340–6, DOI: 10.1093/nar/gkl120.

[195] Lemieux S & Major F. RNA canonical and non-canonical base pairing types: a recognition method and complete repertoire. *Nucleic Acids Research* (2002); **30**:4250–63, DOI: 10.1093/nar/gkf540.

[196] Leontis NB, Stombaugh J & Westhof E. The non-Watson–Crick base pairs and their associated isostericity matrices. *Nucleic Acids Research* (2002); **30**:3497–531, DOI: 10.1093/nar/gkf481.

[197] Leontis NB & Westhof E. Geometric nomenclature and classification of RNA base pairs. *RNA* (2001); **7**:499–512.

[198] Leontis NB & Zirbel CL. Nonredundant 3D Structure Datasets for RNA Knowledge Extraction and Benchmarking. In: Leontis N, Westhof E (eds.). *RNA 3D Structure Analysis and Prediction*. Springer, Berlin, Heidelberg, 2012, 281–98, , DOI: 10.1007/978-3-642-25740-7_13.

[199] Li S, Zhang H, Zhang L, Liu K, Liu B, Mathews DH & Huang L. LinearTurboFold: Linear-Time RNA Structural Alignment and Conserved Structure Prediction with Applications to Coronaviruses. *bioRxiv* (2020):2020.11.23.393488, DOI: 10.1101/2020.11.23.393488.

[200] Liu L & Chen S-J. Computing the conformational entropy for RNA folds. *The Journal of Chemical Physics* (2010); **132**:235104, DOI: 10.1063/1.3447385.

[201] Liu Y-C, Yang C-H, Chen K-T, Wang J-R, Cheng M-L, Chung J-C, Chiu H-T & Lu CL. R3D-BLAST: a search tool for similar RNA 3D substructures. *Nucleic Acids Research* (2011); **39**:W45–9, DOI: 10.1093/nar/gkr379.

[202] Lorenz R, Bernhart SH, Höner zu Siederdissen C, Tafer H, Flamm C, Stadler PF & Hofacker IL. ViennaRNA Package 2.0. *Algorithms for Molecular Biology* (2011); **6**:26, DOI: 10.1186/1748-7188-6-26.

[203] Lorenz R, Flamm C & Hofacker IL. *RNA2DFold: 2D Projections of RNA Folding Landscapes*. Gesellschaft für Informatik e.V., 2009.

[204] Lorenz R, Hofacker IL & Stadler PF. RNA folding with hard and soft constraints. *Algorithms for Molecular Biology* (2016); **11**:8, DOI: 10.1186/s13015-016-0070-z.

[205] Lorenz R, Wolfinger MT, Tanzer A & Hofacker IL. Predicting RNA secondary structures from sequence and probing data. *Methods* (2016); **103**:86–98, DOI: 10.1016/j.ymeth.2016.04.004.

[206] Lu W, Tang Y, Wu H, Huang H, Fu Q, Qiu J & Li H. Predicting RNA secondary structure via adaptive deep recurrent neural networks with energy-based filter. *BMC Bioinformatics* (2019); **20**:684, DOI: 10.1186/s12859-019-3258-7.

[207] Lu X & Olson WK. 3DNA: a software package for the analysis, rebuilding and visualization of three-dimensional nucleic acid structures. *Nucleic Acids Research* (2003); **31**:5108–21, DOI: 10.1093/nar/gkg680.

[208] Lu X-J, Bussemaker HJ & Olson WK. DSSR: an integrated software tool for dissecting the spatial structure of RNA. *Nucleic Acids Research* (2015); **43**:e142–e142, DOI: 10.1093/nar/gkv716.

[209] Lu ZJ, Gloor JW & Mathews DH. Improved RNA secondary structure prediction by maximizing expected pair accuracy. *RNA* (2009); **15**:1805–13, DOI: 10.1261/rna.1643609.

[210] Lukasiak P, Antczak M, Ratajczak T, Bujnicki JM, Szachniuk M, Adamiak RW, Popenda M & Blazewicz J. RNAlyzer—novel approach for quality analysis of RNA structural models. *Nucleic Acids Research* (2013); **41**:5978–90, DOI: 10.1093/nar/gkt318.

[211] Lukasiak P, Antczak M, Ratajczak T, Szachniuk M, Popenda M, Adamiak RW & Blazewicz J. RNAssess—a web server for quality assessment of RNA 3D structures. *Nucleic Acids Research* (2015); **43**:W502–6, DOI: 10.1093/nar/gkv557.

[212] Lyngsø RB. Complexity of Pseudoknot Prediction in Simple Models. In: Díaz J, Karhumäki J, Lepistö A, Sannella D (eds.). *Automata, Languages and Programming*. Springer, Berlin, Heidelberg, 2004, 919–31, , DOI: 10.1007/978-3-540-27836-8_77.

[213] Macke TJ & Case DA. NAB: Modeling Unusual Nucleic Acid Structures. *Molecular Modeling of Nucleic Acids*. Vol 682. American Chemical Society, 1997, 379–93, , DOI: 10.1021/bk-1998-0682.ch024.

[214] MacKerell AD, Wiorkiewicz-Kuczera J & Karplus M. CHARMM22: An all-atom empirical energy function for the simulation of nucleic acids. *Journal of the American Chemical Society* (1995); **117**:11946–75, DOI: 10.1021/ja00153a017.

[215] Magnus M, Kappel K, Das R & Bujnicki JM. EvoClustRNA: RNA 3D structure prediction guided by independent folding of homologous sequences. *BMC Bioinformatics* (2019); **20**:512, DOI: 10.1186/s12859-019-3120-y.

[216] Major F. MC-Sym: Building three-dimensional ribonucleic acid structures. *Computing in Science Engineering* (2003); **5**:44–53, DOI: 10.1109/MCISE.2003.1225860.

[217] Major F, Turcotte M, Gautheret D, Lapalme G, Fillion E & Cedergren R. MC-Sym: The combination of symbolic and numerical computation for three-dimensional modeling of RNA. *Science* (1991); **253**:1255–60, DOI: 10.1126/science.1716375.

[218] Malhotra A, Tan RK & Harvey SC. YAMMP: Modeling large RNAs and ribonucleoprotein particles using molecular mechanics techniques. *Biophysical Journal* (1994); **66**:1777–95, DOI: 10.1016/S0006-3495(94)80972-5.

[219] Mao K, Wang J & Xiao Y. 2dRNA: Prediction of RNA secondary structure with pseudoknots using coupled deep neural networks. *Biophysics Reports* (2020); **6**:146–54, DOI: 10.1007/s41048-020-00114-x.

[220] Marchand L. Les qualifications des contraintes en optimisation à contraintes non linéaires. *CaMUS (Cahiers Mathématiques de l'Université de Sherbrooke)* (2014); **5**:23–36, DOI: http://hdl.handle.net/11143/16140.

[221] Markham NR & Zuker M. UNAFold. In: Keith JM (ed.). *Bioinformatics: Structure, Function and Applications*. Humana Press, Totowa, NJ, 2008, 3–31, , DOI: 10.1007/978-1-60327-429-6_1.

[222] Marler RT & Arora JS. Survey of multi-objective optimization methods for engineering. *Structural and Multidisciplinary Optimization* (2004); **26**:369–95, DOI: 10.1007/s00158-003-0368-6.

[223] Martinez HM, Jr JVM & Shapiro BA. RNA2D3D: A program for Generating, Viewing, and Comparing 3-Dimensional Models of RNA. *Journal of Biomolecular Structure and Dynamics* (2008); **25**:669–83, DOI: 10.1080/07391102.2008.10531240.

[224] Masuda S, Nakajima K, Kashiwabara T & Fujisawa T. Crossing minimization in linear embeddings of graphs. *IEEE Transactions on Computers* (1990); **39**:124–7, DOI: 10.1109/12.46286.

[225] Mathews DH, Andre TC, Kim J, Turner DH & Zuker M. An Updated Recursive Algorithm for RNA Secondary Structure Prediction with Improved Thermodynamic Parameters. *Molecular Modeling of Nucleic Acids*. Vol 682. American Chemical Society, 1997, 246–57, , DOI: 10.1021/bk-1998-0682.ch015.

[226] Mathews DH, Disney MD, Childs JL, Schroeder SJ, Zuker M & Turner DH. Incorporating chemical modification constraints into a dynamic programming algorithm for prediction of RNA secondary structure. *Proceedings of the National Academy of Sciences of the United States of America* (2004); **101**:7287–92, DOI: 10.1073/pnas.0401799101.

[227] Mathews DH, Sabina J, Zuker M & Turner DH. Expanded sequence dependence of thermodynamic parameters improves prediction of RNA secondary structure (Edited by I. Tinoco). *Journal of Molecular Biology* (1999); **288**:911–40, DOI: 10.1006/jmbi.1999.2700.

[228] Mazzanti L, Alferkh L, Frezza E & Pasquali S. HiRE-RNA: Biasing RNA coarse-grained folding simulations with Small–Angle X–ray Scattering (SAXS) data. *bioRxiv* (2021):2021.03.29.437449, DOI: 10.1101/2021.03.29.437449.

[229] McCaskill JS. The equilibrium partition function and base pair binding probabilities for RNA secondary structure. *Biopolymers* (1990); **29**:1105–19, DOI: 10.1002/bip.360290621.

[230] McInnes L, Healy J & Astels S. hdbscan: Hierarchical density based clustering. *Journal of Open Source Software* (2017); **2**:205, DOI: 10.21105/joss.00205.

[231] Merino EJ, Wilkinson KA, Coughlan JL & Weeks KM. RNA Structure Analysis at Single Nucleotide Resolution by Selective 2'-Hydroxyl Acylation and Primer Extension (SHAPE). *Journal of the American Chemical Society* (2005); **127**:4223–31, DOI: 10.1021/ja043822v.

[232] Messac A. Physical programming - Effective optimization for computational design. *AIAA Journal* (1996); **34**:149–58, DOI: 10.2514/3.13035.

[233] Messac A, Ismail-Yahaya A & Mattson CA. The normalized normal constraint method for generating the Pareto frontier. *Structural and Multidisciplinary Optimization* (2003); **25**:86–98, DOI: 10.1007/s00158-002-0276-1.

[234] Messac A & Mattson CA. Generating Well-Distributed Sets of Pareto Points for Engineering Design Using Physical Programming. *Optimization and Engineering* (2002); **3**:431–50, DOI: 10.1023/A:1021179727569.

[235] Messac A, Sukam CP & Melachrinoudis E. Mathematical and Pragmatic Perspectives of Physical Programming. *AIAA Journal* (2001); **39**:885–93, DOI: 10.2514/2.1392.

[236] Metropolis N, Rosenbluth AW, Rosenbluth MN, Teller AH & Teller E. Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics* (1953); **21**:1087–92, DOI: 10.1063/1.1699114.

[237] Miao Z *et al.* RNA-Puzzles Round II: assessment of RNA structure prediction programs applied to three large RNA structures. *RNA* (2015); **21**:1066–84, DOI: 10.1261/rna.049502.114.

[238] Miao Z *et al.* RNA-Puzzles Round III: 3D RNA structure prediction of five riboswitches and one ribozyme. *RNA* (2017); **23**:655–72, DOI: 10.1261/rna.060368.116.

[239] Miao Z *et al.* RNA-Puzzles Round IV: 3D structure predictions of four ribozymes and two aptamers. *RNA* (2020):rna.075341.120, DOI: 10.1261/rna.075341.120.

[240] Miettinen K. *Nonlinear Multiobjective Optimization*. Springer US, 1998, DOI: 10.1007/978-1-4615-5563-6.

[241] Mirabello C & Wallner B. rawMSA: End-to-end Deep Learning using raw Multiple Sequence Alignments. *PLOS ONE* (2019); **14**:e0220182, DOI: 10.1371/journal.pone.0220182.

[242] Mlýnský V, Kührová P, Kühr T, Otyepka M, Bussi G, Banáš P & Šponer J. Fine-Tuning of the AMBER RNA Force Field with a New Term Adjusting Interactions of Terminal Nucleotides. *Journal of Chemical Theory and Computation* (2020); **16**:3936–46, DOI: 10.1021/acs.jctc.0c00228.

[243] Mortimer SA & Weeks KM. A Fast-Acting Reagent for Accurate Analysis of RNA Secondary and Tertiary Structure by SHAPE Chemistry. *Journal of the American Chemical Society* (2007); **129**:4144–5, DOI: 10.1021/ja0704028.

[244] Mueller-Gritschneder D, Graeb H & Schlichtmann U. SPO: A Successive Approach to Compute the Bounded Pareto Front of Practical Multiobjective Optimization Problems. *SIAM Journal on Optimization* (2009); **20**:915–34, DOI: 10.1137/080729013.

[245] Mustoe AM, Al-Hashimi HM & Brooks CL. TOPRNA: Coarse Grained Models Reveal Essential Contributions of Topological Constraints to the Conformational Free Energy of RNA Bulges. *The Journal of Physical Chemistry B* (2014); **118**:2615–27, DOI: 10.1021/jp411478x.

[246] Nawrocki EP & Eddy SR. Infernal 1.1: 100-fold faster RNA homology searches. *Bioinformatics* (2013); **29**:2933–5, DOI: 10.1093/bioinformatics/btt509.

[247] Nelson MT, Humphrey W, Gursoy A, Dalke A, Kalé LV, Skeel RD & Schulten K. NAMD: a Parallel, Object-Oriented Molecular Dynamics Program. *The International Journal of*

Supercomputer Applications and High Performance Computing (1996); **10**:251–68, DOI: 10.1177/109434209601000401.

[248] Nussinov R & Jacobson AB. Fast algorithm for predicting the secondary structure of single-stranded RNA. *Proceedings of the National Academy of Sciences* (1980); **77**:6309–13, DOI: 10.1073/pnas.77.11.6309.

[249] Oliver C, Mallet V, Philippopoulos P, Hamilton WL & Waldispuhl J. VeRNAl: Mining RNA Structures for Fuzzy Base Pairing Network Motifs. *arXiv:200900664 [cs, q-bio]* (2020). Publié le 19 December 2020, Accessed 21 April 2021, http://arxiv.org/abs/2009.00664.

[250] Parisien M & Major F. The MC-Fold and MC-Sym pipeline infers RNA structure from sequence data. *Nature* (2008); **452**:51–5, DOI: 10.1038/nature06684.

[251] Parlea LG, Sweeney BA, Hosseini-Asanjan M, Zirbel CL & Leontis NB. The RNA 3D Motif Atlas: Computational methods for extraction, organization and evaluation of RNA motifs. *Methods* (2016); **103**:99–119, DOI: 10.1016/j.ymeth.2016.04.025.

[252] Parsons J, Holmes JB, Rojas JM, Tsai J & Strauss CEM. Practical conversion from torsion space to Cartesian space for in silico protein synthesis. *Journal of Computational Chemistry* (2005); **26**:1063–8, DOI: https://doi.org/10.1002/jcc.20237.

[253] Pasquali S & Derreumaux P. HiRE-RNA: A High Resolution Coarse-Grained Energy Model for RNA. *The Journal of Physical Chemistry B* (2010); **114**:11957–66, DOI: 10.1021/jp102497y.

[254] Pérez A, Marchán I, Svozil D, Sponer J, Cheatham TE, Laughton CA & Orozco M. Refinement of the AMBER Force Field for Nucleic Acids: Improving the Description of α/γ Conformers. *Biophysical Journal* (2007); **92**:3817–29, DOI: 10.1529/biophysj.106.097782.

[255] Petrov AI, Zirbel CL & Leontis NB. Automated classification of RNA 3D motifs and the RNA 3D Motif Atlas. *RNA* (2013); **19**:1327–40, DOI: 10.1261/rna.039438.113.

[256] Piątkowski P, Jabłońska J, Żyła A, Niedziałek D, Matelska D, Jankowska E, Waleń T, Dawson WK & Bujnicki JM. SupeRNAlign: a new tool for flexible superposition of homologous RNA structures and inference of accurate structure-based sequence alignments. *Nucleic Acids Research* (2017); **45**:e150–e150, DOI: 10.1093/nar/gkx631.

[257] Poblete S, Bottaro S & Bussi G. SPQR: A nucleobase-centered coarse-grained representation for structure prediction of RNA motifs. *Nucleic Acids Research* (2018); **46**:1674–83, DOI: 10.1093/nar/gkx1269.

[258] Ponder JW & Richards FM. Tinker: An efficient newton-like method for molecular mechanics energy minimization of large molecules. *Journal of Computational Chemistry* (1987); **8**:1016–24, DOI: https://doi.org/10.1002/jcc.540080710.

[259] Poolsap U, Kato Y & Akutsu T. Prediction of RNA secondary structure with pseudoknots using integer programming. *BMC Bioinformatics* (2009); **10**:S38, DOI: 10.1186/1471-2105-10-S1-S38.

[260] Popenda M, Błażewicz M, Szachniuk M & Adamiak RW. RNA FRABASE version 1.0: an engine with a database to search for the three-dimensional fragments within RNA structures. *Nucleic Acids Research* (2008); **36**:D386–91, DOI: 10.1093/nar/gkm786.

[261] Popenda M, Szachniuk M, Antczak M, Purzycka KJ, Lukasiak P, Bartol N, Blazewicz J &

Adamiak RW. RNAComposer: Automated 3D structure composition for large RNAs. *Nucleic Acids Research* (2012); **40**:e112–e112, DOI: 10.1093/nar/gks339.

[262] Popenda M, Szachniuk M, Blazewicz M, Wasik S, Burke EK, Blazewicz J & Adamiak RW. RNA FRABASE 2.0: an advanced web-accessible database with the capacity to search the three-dimensional fragments within RNA structures. *BMC Bioinformatics* (2010); **11**:231, DOI: 10.1186/1471-2105-11-231.

[263] Pruesse E, Peplies J & Glöckner FO. SINA: Accurate high-throughput multiple sequence alignment of ribosomal RNA genes. *Bioinformatics* (2012); **28**:1823–9, DOI: 10.1093/bioinformatics/bts252.

[264] Pruesse E, Quast C, Knittel K, Fuchs BM, Ludwig W, Peplies J & Glöckner FO. SILVA: a comprehensive online resource for quality checked and aligned ribosomal RNA sequence data compatible with ARB. *Nucleic Acids Research* (2007); **35**:7188–96, DOI: 10.1093/nar/gkm864.

[265] Pucci F, Zerihun MB, Peter EK & Schug A. Evaluating DCA-based method performances for RNA contact prediction by a well-curated data set. *RNA (New York, NY)* (2020); **26**:794–802, DOI: 10.1261/rna.073809.119.

[266] Puton T, Kozlowski LP, Rother KM & Bujnicki JM. CompaRNA: a server for continuous benchmarking of automated methods for RNA secondary structure prediction. *Nucleic Acids Research* (2013); **41**:4307–23, DOI: 10.1093/nar/gkt101.

[267] Rackers JA, Wang Z, Lu C, Laury ML, Lagardère L, Schnieders MJ, Piquemal J-P, Ren P & Ponder JW. Tinker 8: Software Tools for Molecular Design. *Journal of Chemical Theory and Computation* (2018); **14**:5273–89, DOI: 10.1021/acs.jctc.8b00529.

[268] Rahrig RR, Leontis NB & Zirbel CL. R3D Align: global pairwise alignment of RNA 3D structures using local superpositions. *Bioinformatics* (2010); **26**:2689–97, DOI: 10.1093/bioinformatics/btq506.

[269] Rangan R *et al.* De novo 3D models of SARS-CoV-2 RNA elements from consensus experimental secondary structures. *Nucleic Acids Research* (2021); **49**:3092–108, DOI: 10.1093/nar/gkab119.

[270] Rangan R, Zheludev IN, Hagey RJ, Pham EA, Wayment-Steele HK, Glenn JS & Das R. RNA genome conservation and secondary structure in SARS-CoV-2 and SARS-related viruses: a first look. *RNA (New York, NY)* (2020); **26**:937–59, DOI: 10.1261/rna.076141.120.

[271] Reeder J & Giegerich R. pknotsRG: Design, implementation and evaluation of a practical pseudoknot folding algorithm based on thermodynamics. *BMC Bioinformatics* (2004); **5**:104, DOI: 10.1186/1471-2105-5-104.

[272] Reinharz V, Major F & Waldispühl J. Towards 3D structure prediction of large RNA molecules: an integer programming framework to insert local 3D motifs in RNA secondary structure. *Bioinformatics* (2012); **28**:i207–14, DOI: 10.1093/bioinformatics/bts226.

[273] Reinharz V, Soulé A, Westhof E, Waldispühl J & Denise A. Mining for recurrent long-range interactions in RNA structures reveals embedded hierarchies in network families. *Nucleic Acids Research* (2018); **46**:3841–51, DOI: 10.1093/nar/gky197.

[274] Remmert M, Biegert A, Hauser A & Söding J. HHblits: lightning-fast iterative protein sequence searching by HMM-HMM alignment. *Nature Methods* (2012); **9**:173–5, DOI:

10.1038/nmeth.1818.

[275] Ren J, Rastegari B, Condon A & Hoos HH. HotKnots: Heuristic prediction of RNA secondary structures including pseudoknots. *RNA* (2005); **11**:1494–504, DOI: 10.1261/rna.7284905.

[276] Ren P, Wu C & Ponder JW. Polarizable Atomic Multipole-Based Molecular Mechanics for Organic Molecules. *Journal of Chemical Theory and Computation* (2011); **7**:3143–61, DOI: 10.1021/ct200304d.

[277] Richardson JS *et al.* RNA backbone: Consensus all-angle conformers and modular string nomenclature (an RNA Ontology Consortium contribution). *RNA* (2008); **14**:465–81, DOI: 10.1261/rna.657708.

[278] Rivas E. The four ingredients of single-sequence RNA secondary structure prediction. A unifying perspective. *RNA Biology* (2013); **10**:1185–96, DOI: 10.4161/rna.24971.

[279] Rivas E & Eddy SR. pknotsRE: A dynamic programming algorithm for RNA structure prediction including pseudoknots. *Journal of Molecular Biology* (1999); **285**:2053–68, DOI: 10.1006/jmbi.1998.2436.

[280] Rivas E, Lang R & Eddy SR. A range of complex probabilistic models for RNA secondary structure prediction that includes the nearest-neighbor model and more. *RNA* (2012); **18**:193–212, DOI: 10.1261/rna.030049.111.

[281] RNAcentral Consortium. RNAcentral 2021: secondary structure integration, improved sequence search and new member databases. *Nucleic Acids Research* (2021); **49**:D212–20, DOI: 10.1093/nar/gkaa921.

[282] Roberts GO & Tweedie RL. Exponential Convergence of Langevin Distributions and Their Discrete Approximations. *Bernoulli* (1996); **2**:341–63, DOI: 10.2307/3318418.

[283] Robertson MJ, Tirado-Rives J & Jorgensen WL. Improved treatment of nucleosides and nucleotides in the OPLS-AA force field. *Chemical Physics Letters* (2017); **683**:276–80, DOI: 10.1016/j.cplett.2017.02.049.

[284] Rosenblad MA, Gorodkin J, Knudsen B, Zwieb C & Samuelsson T. SRPDB: Signal Recognition Particle Database. *Nucleic Acids Research* (2003); **31**:363–4, DOI: 10.1093/nar/gkg107.

[285] Rother M, Rother K, Puton T & Bujnicki JM. ModeRNA: a tool for comparative modeling of RNA 3D structure. *Nucleic Acids Research* (2011); **39**:4007–22, DOI: 10.1093/nar/gkq1320.

[286] Ruan J, Stormo GD & Zhang W. ILM: An Iterated loop matching approach to the prediction of RNA secondary structures with pseudoknots. *Bioinformatics* (2004); **20**:58–66, DOI: 10.1093/bioinformatics/btg373.

[287] Ruder S. An overview of gradient descent optimization algorithms. *arXiv:160904747 [cs]* (2016). Publié le 15 September 2016, Accessed 10 September 2019, http://arxiv.org/abs/1609.04747.

[288] Saaidi A. Thèse de Doctorat : Multi-dimensional probing for RNA secondary structure(s) prediction.Université Paris-Saclay (ComUE) (2018)

[289] Saaidi A, Allouche D, Regnier M, Sargueil B & Ponty Y. IPANEMAP: integrative probing analysis of nucleic acids empowered by multiple accessibility profiles. *Nucleic Acids Research* (2020); **48**:8276–89, DOI: 10.1093/nar/gkaa607.

[290] Sakuraba S, Iwakiri J, Hamada M, Kameda T, Tsuji G, Kimura Y, Abe H & Asai K. Free-Energy Calculation of Ribonucleic Inosines and Its Application to Nearest-Neighbor Parameters. *Journal of Chemical Theory and Computation* (2020); **16**:5923–35, DOI: 10.1021/acs.jctc.0c00270.

[291] Salser W. Globin mRNA Sequences: Analysis of Base Pairing and Evolutionary Implications. *Cold Spring Harbor Symposia on Quantitative Biology* (1978); **42**:985–1002, DOI: 10.1101/SQB.1978.042.01.099.

[292] Sargsyan K & Lim C. Arrangement of 3D structural motifs in ribosomal RNA. *Nucleic Acids Research* (2010); **38**:3512–22, DOI: 10.1093/nar/gkq074.

[293] Sarrazin-Gendron R, Reinharz V, Oliver CG, Moitessier N & Waldispühl J. Automated, customizable and efficient identification of 3D base pair modules with BayesPairing. *Nucleic Acids Research* (2019), DOI: 10.1093/nar/gkz102. Publié le January 2019, Accessed 14 March 2019, https://academic.oup.com/nar/advance-article/doi/10.1093/nar/gkz102/5369007.

[294] Sarrazin-Gendron R, Yao H-T, Reinharz V, Oliver CG, Ponty Y & Waldispühl J. Stochastic Sampling of Structural Contexts Improves the Scalability and Accuracy of RNA 3D Modules Identification. *bioRxiv* (2019):834762, DOI: 10.1101/834762.

[295] Sarver M, Zirbel CL, Stombaugh J, Mokdad A & Leontis NB. FR3D: finding local and composite recurrent structural motifs in RNA 3D structures. *Journal of Mathematical Biology* (2008); **56**:215–52, DOI: 10.1007/s00285-007-0110-x.

[296] Sato K, Akiyama M & Sakakibara Y. MXFold2: RNA secondary structure prediction using deep learning with thermodynamic integration. *Nature Communications* (2021); **12**:941, DOI: 10.1038/s41467-021-21194-4.

[297] Sato K, Kato Y, Hamada M, Akutsu T & Asai K. IPknot: fast and accurate prediction of RNA secondary structures with pseudoknots using integer programming. *Bioinformatics* (2011); **27**:i85–93, DOI: 10.1093/bioinformatics/btr215.

[298] Schlick T. Adventures with RNA graphs. *Methods* (2018); **143**:16–33, DOI: 10.1016/j.ymeth.2018.03.009.

[299] Schubert E, Sander J, Ester M, Kriegel HP & Xu X. DBSCAN Revisited, Revisited: Why and How You Should (Still) Use DBSCAN. *ACM Transactions on Database Systems* (2017); **42**:19:1-19:21, DOI: 10.1145/3068335.

[300] Seetin MG & Mathews DH. Automated RNA tertiary structure prediction from secondary structure and low-resolution restraints. *Journal of Computational Chemistry* (2011); **32**:2232–44, DOI: 10.1002/jcc.21806.

[301] Seetin MG & Mathews DH. TurboKnot: rapid prediction of conserved RNA secondary structures including pseudoknots. *Bioinformatics* (2012); **28**:792–8, DOI: 10.1093/bioinformatics/bts044.

[302] Senior AW *et al.* AlphaFold: Improved protein structure prediction using potentials from deep learning. *Nature* (2020); **577**:706–10, DOI: 10.1038/s41586-019-1923-7.

[303] Senior AW *et al.* AlphaFold: Protein structure prediction using multiple deep neural networks in the 13th Critical Assessment of Protein Structure Prediction (CASP13). *Proteins: Structure, Function, and Bioinformatics* (2019); **87**:1141–8, DOI: https://doi.org/10.1002/prot.25834.

[304] Serra MJ & Turner DH. Predicting thermodynamic properties of RNA. *Methods in Enzymology*. Vol 259. Academic Press, 1995, 242–61, , DOI: 10.1016/0076-6879(95)59047-1.

[305] Shareghi P, Wang Y, Malmberg R & Cai L. Simultaneous prediction of RNA secondary structure and helix coaxial stacking. *BMC Genomics* (2012); **13**:S7, DOI: 10.1186/1471-2164-13-S3-S7.

[306] Sharma P, Mitra A, Sharma S, Singh H & Bhattacharyya D. Quantum Chemical Studies of Structures and Binding in Noncanonical RNA Base pairs: The Trans Watson-Crick:Watson-Crick Family. *Journal of Biomolecular Structure and Dynamics* (2008); **25**:709–32, DOI: 10.1080/07391102.2008.10507216.

[307] Sharma S, Ding F & Dokholyan NV. iFoldRNA: three-dimensional RNA structure prediction and folding. *Bioinformatics* (2008); **24**:1951–2, DOI: 10.1093/bioinformatics/btn328.

[308] Sheikh S, Backofen R & Ponty Y. Impact of the Energy Model on the Complexity of RNA Folding with Pseudoknots. In: Kärkkäinen J, Stoye J (eds.). *Combinatorial Pattern Matching*. Vol 7354. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, 321–33, , DOI: 10.1007/978-3-642-31265-6_26.

[309] zu Siederdissen CH, Bernhart SH, Stadler PF & Hofacker IL. A folding algorithm for extended RNA secondary structures. *Bioinformatics* (2011); **27**:i129–36, DOI: 10.1093/bioinformatics/btr220.

[310] Siegfried NA, Busan S, Rice GM, Nelson JAE & Weeks KM. RNA motif discovery by SHAPE and mutational profiling (SHAPE-MaP). *Nature Methods* (2014); **11**:959–65, DOI: 10.1038/nmeth.3029.

[311] Sieradzan AK, Golon Ł & Liwo A. NARES-CSA: Prediction of DNA and RNA structure with the NARES-2P force field and conformational space annealing. *Physical Chemistry Chemical Physics* (2018); **20**:19656–63, DOI: 10.1039/C8CP03018A.

[312] Singh J, Hanson J, Paliwal K & Zhou Y. SPOT-RNA: RNA secondary structure prediction using an ensemble of two-dimensional deep neural networks and transfer learning. *Nature Communications* (2019); **10**:1–13, DOI: 10.1038/s41467-019-13395-9.

[313] Singh J, Paliwal K, Zhang T, Singh J, Litfin T & Zhou Y. SPOT-RNA2: Improved RNA secondary structure and tertiary base-pairing prediction using evolutionary profile, mutational coupling and two-dimensional transfer learning. *Bioinformatics* (2021), DOI: 10.1093/bioinformatics/btab165. Publié le 11 March 2021, Accessed 6 May 2021, https://doi.org/10.1093/bioinformatics/btab165.

[314] Sloma MF & Mathews DH. CycleFold: Base pair probability estimates improve the prediction accuracy of RNA non-canonical base pairs. *PLoS Computational Biology* (2017); **13**, DOI: 10.1371/journal.pcbi.1005827. Publié le 6 November 2017, Accessed 8 January 2019, https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5690697/.

[315] Sloma MF & Mathews DH. ProbScan: Exact calculation of loop formation probability identifies folding motifs in RNA secondary structures. *RNA* (2016); **22**:1808–18, DOI: 10.1261/rna.053694.115.

[316] Smith LG, Zhao J, Mathews DH & Turner DH. Physics-based all-atom modeling of RNA energetics and structure. *WIREs RNA* (2017); **8**:e1422, DOI: https://doi.org/10.1002/wrna.1422.

[317] Soares TA, Hünenberger PH, Kastenholz MA, Kräutler V, Lenz T, Lins RD, Oostenbrink C & Gunsteren WF van. An improved nucleic acid parameter set for the GROMOS force field. *Journal of Computational Chemistry* (2005); **26**:725–37, DOI: https://doi.org/10.1002/jcc.20193.

[318] Soulé A, Reinharz V, Sarrazin-Gendron R, Denise A & Waldispühl J. Finding recurrent RNA structural networks with fast maximal common subgraphs of edge-colored graphs. *PLOS Computational Biology* (2021); **17**:e1008990, DOI: 10.1371/journal.pcbi.1008990.

[319] Spasic A, Assmann SM, Bevilacqua PC & Mathews DH. Modeling RNA secondary structure folding ensembles using SHAPE mapping data. *Nucleic Acids Research* (2018); **46**:314–23, DOI: 10.1093/nar/gkx1057.

[320] Srinivas N & Deb K. Muiltiobjective Optimization Using Nondominated Sorting in Genetic Algorithms. *Evolutionary Computation* (1994); **2**:221–48, DOI: 10.1162/evco.1994.2.3.221.

[321] Stasiewicz J, Mukherjee S, Nithin C & Bujnicki JM. QRNAS: software tool for refinement of nucleic acid structures. *BMC Structural Biology* (2019); **19**:5, DOI: 10.1186/s12900-019-0103-1.

[322] Steinbrecher T, Latzer J & Case DA. Revised AMBER Parameters for Bioorganic Phosphates. *Journal of Chemical Theory and Computation* (2012); **8**:4405–12, DOI: 10.1021/ct300613v.

[323] Stephenson JD, Kenyon JC, Symmons MF & Lever AML. Characterizing 3D RNA structure by single molecule FRET. *Methods* (2016); **103**:57–67, DOI: 10.1016/j.ymeth.2016.02.004.

[324] Šulc P, Romano F, Ouldridge TE, Doye JPK & Louis AA. oxRNA: A nucleotide-level coarse-grained model of RNA. *The Journal of Chemical Physics* (2014); **140**:235102, DOI: 10.1063/1.4881424.

[325] Sun S, Wang W, Peng Z & Yang J. RNAContact: RNA inter-nucleotide 3D closeness prediction by deep residual neural networks. *Bioinformatics* (2020), DOI: 10.1093/bioinformatics/btaa932. Publié le 2 November 2020, Accessed 4 May 2021, https://doi.org/10.1093/bioinformatics/btaa932.

[326] Svergun DI & Koch MHJ. Small-angle scattering studies of biological macromolecules in solution. *Reports on Progress in Physics* (2003); **66**:1735–82, DOI: 10.1088/0034-4885/66/10/R05.

[327] Tahi F, Engelen S & Regnier M. P-DCFold: A fast algorithm for RNA secondary structure prediction including pseudoknots. *Third IEEE Symposium on Bioinformatics and Bioengineering, 2003. Proceedings.* 2003, 11–7, , DOI: 10.1109/BIBE.2003.1188924.

[328] Tahi F, Gouy M & Régnier M. DCFold: Automatic RNA secondary structure prediction with a comparative approach. *Computers & Chemistry* (2002); **26**:521–30, DOI: 10.1016/S0097-8485(02)00012-8.

[329] Tan D, Piana S, Dirks RM & Shaw DE. RNA force field with accuracy comparable to state-of-the-art protein force fields. *Proceedings of the National Academy of Sciences* (2018); **115**:E1346–55, DOI: 10.1073/pnas.1713027115.

[330] Tan RK-Z & Harvey SC. Yammp: Development of a molecular mechanics program using the modular programming method. *Journal of Computational Chemistry* (1993); **14**:455–70, DOI:

https://doi.org/10.1002/jcc.540140410.

[331] Tan RKZ, Petrov AS & Harvey SC. YUP: A Molecular Simulation Program for Coarse-Grained and Multiscaled Models. *Journal of Chemical Theory and Computation* (2006); **2**:529–40, DOI: 10.1021/ct050323r.

[332] Tan Y-L, Feng C-J, Jin L, Shi Y-Z, Zhang W & Tan Z-J. What is the best reference state for building statistical potentials in RNA 3D structure evaluation? *RNA (New York, NY)* (2019); **25**:793–812, DOI: 10.1261/rna.069872.118.

[333] Tan Z, Fu Y, Sharma G & Mathews DH. TurboFold II: RNA structural alignment and secondary structure prediction informed by multiple homologs. *Nucleic Acids Research* (2017); **45**:11570–81, DOI: 10.1093/nar/gkx815.

[334] Taufer M, Licon A, Araiza R, Mireles D, van Batenburg FHD, Gultyaev AP & Leung M-Y. PseudoBase++: an extension of PseudoBase for easy searching, formatting and visualization of pseudoknots. *Nucleic Acids Research* (2009); **37**:D127–35, DOI: 10.1093/nar/gkn806.

[335] Theis C, Höner zu Siederdissen C, Hofacker IL & Gorodkin J. Automated identification of RNA 3D modules with discriminative power in RNA structural alignments. *Nucleic Acids Research* (2013); **41**:9999–10009, DOI: 10.1093/nar/gkt795.

[336] Theis C, Zirbel CL, Siederdissen CH zu, Anthon C, Hofacker IL, Nielsen H & Gorodkin J. RNA 3D Modules in Genome-Wide Predictions of RNA 2D Structure. *PLOS ONE* (2015); **10**:e0139900, DOI: 10.1371/journal.pone.0139900.

[337] Tian Y, Wang H, Zhang X & Jin Y. Effectiveness and efficiency of non-dominated sorting for evolutionary multi- and many-objective optimization. *Complex & Intelligent Systems* (2017); **3**:247–63, DOI: 10.1007/s40747-017-0057-5.

[338] Tijerina P, Mohr S & Russell R. DMS footprinting of structured RNAs and RNA–protein complexes. *Nature Protocols* (2007); **2**:2608–23, DOI: 10.1038/nprot.2007.380.

[339] Turner DH & Mathews DH. NNDB: the nearest neighbor parameter database for predicting stability of nucleic acid secondary structure. *Nucleic Acids Research* (2010); **38**:D280–2, DOI: 10.1093/nar/gkp892.

[340] Turner DH, Sugimoto N & Freier SM. RNA Structure Prediction. *Annual Review of Biophysics and Biophysical Chemistry* (1988); **17**:167–92, DOI: 10.1146/annurev.bb.17.060188.001123.

[341] Ulungu EL, Teghem J, Fortemps PH & Tuyttens D. MOSA method: a tool for solving multiobjective combinatorial optimization problems. *Journal of Multi-Criteria Decision Analysis* (1999); **8**:221–36, DOI: https://doi.org/10.1002/(SICI)1099-1360(199907)8:4<221::AID-MCDA247>3.0.CO;2-O.

[342] Wadley LM, Keating KS, Duarte CM & Pyle AM. Evaluating and Learning from RNA Pseudotorsional Space: Quantitative Validation of a Reduced Representation for RNA Structure. *Journal of Molecular Biology* (2007); **372**:942–57, DOI: 10.1016/j.jmb.2007.06.058.

[343] Wadley LM & Pyle AM. The identification of novel RNA structural motifs using COMPADRES: an automated approach to structural discovery. *Nucleic Acids Research* (2004); **32**:6650–9, DOI: 10.1093/nar/gkh1002.

[344] Waleń T, Chojnowski G, Gierski P & Bujnicki JM. ClaRNA: a classifier of contacts in RNA 3D structures based on a comparative analysis of various classification schemes. *Nucleic Acids Research* (2014); **42**:e151–e151, DOI: 10.1093/nar/gku765.

[345] Wales DJ & Doye JPK. Global Optimization by Basin-Hopping and the Lowest Energy Structures of Lennard-Jones Clusters Containing up to 110 Atoms. *The Journal of Physical Chemistry A* (1997); **101**:5111–6, DOI: 10.1021/jp970984n.

[346] Wales DJ & Yildirim I. Improving Computational Predictions of Single-Stranded RNA Tetramers with Revised α/γ Torsional Parameters for the Amber Force Field. *The Journal of Physical Chemistry B* (2017); **121**:2989–99, DOI: 10.1021/acs.jpcb.7b00819.

[347] Walter AE, Turner DH, Kim J, Lyttle MH, Müller P, Mathews DH & Zuker M. Coaxial stacking of helixes enhances binding of oligoribonucleotides and improves predictions of RNA folding. *Proceedings of the National Academy of Sciences* (1994); **91**:9218–22, DOI: 10.1073/pnas.91.20.9218.

[348] Wang J, Cieplak P & Kollman PA. AMBER ff99: How well does a restrained electrostatic potential (RESP) model perform in calculating conformational energies of organic and biological molecules? *Journal of Computational Chemistry* (2000); **21**:1049–74, DOI: https://doi.org/10.1002/1096-987X(200009)21:12<1049::AID-JCC3>3.0.CO;2-F.

[349] Wang J, Mao K, Zhao Y, Zeng C, Xiang J, Zhang Y & Xiao Y. 3dRNA 2.0: Optimization of RNA 3D structure prediction using evolutionary restraints of nucleotide–nucleotide interactions from direct coupling analysis. *Nucleic Acids Research* (2017); **45**:6299–309, DOI: 10.1093/nar/gkx386.

[350] Wang J, Zhao Y, Zhu C & Xiao Y. 3dRNAscore: a distance and torsion angle dependent evaluation function of 3D RNA structures. *Nucleic Acids Research* (2015); **43**:e63–e63, DOI: 10.1093/nar/gkv141.

[351] Wang L, Liu Y, Zhong X, Liu H, Lu C, Li C & Zhang H. DMfold: A Novel Method to Predict RNA Secondary Structure With Pseudoknots Based on Deep Learning and Improved Base Pair Maximization Principle. *Frontiers in Genetics* (2019); **10**, DOI: 10.3389/fgene.2019.00143. Publié le 2019, Accessed 16 July 2020, https://www.frontiersin.org/articles/10.3389/fgene.2019.00143/full.

[352] Wang S, Sun S, Li Z, Zhang R & Xu J. Accurate De Novo Prediction of Protein Contact Map by Ultra-Deep Learning Model. *PLOS Computational Biology* (2017); **13**:e1005324, DOI: 10.1371/journal.pcbi.1005324.

[353] Washietl S, Hofacker IL & Stadler PF. RNAz: Fast and reliable prediction of noncoding RNAs. *Proceedings of the National Academy of Sciences* (2005); **102**:2454–9, DOI: 10.1073/pnas.0409169102.

[354] Watkins AM, Geniesse C, Kladwang W, Zakrevsky P, Jaeger L & Das R. Blind prediction of noncanonical RNA structure at atomic accuracy. *Science Advances* (2018); **4**:eaar5316, DOI: 10.1126/sciadv.aar5316.

[355] Watkins AM, Rangan R & Das R. FARFAR2: Improved De Novo Rosetta Prediction of Complex Global RNA Folds. *Structure* (2020); **28**:963-976.e6, DOI: 10.1016/j.str.2020.05.011.

[356] Weinberg CE, Weinberg Z & Hammann C. Novel ribozymes: discovery, catalytic mechanisms, and the quest to understand biological function. *Nucleic Acids Research* (2019);

47:9480–94, DOI: 10.1093/nar/gkz737.

[357] Weinreb C, Riesselman AJ, Ingraham JB, Gross T, Sander C & Marks DS. 3D RNA and Functional Interactions from Evolutionary Couplings. *Cell* (2016); **165**:963–75, DOI: 10.1016/j.cell.2016.03.030.

[358] Willmott D, Murrugarra D & Ye Q. Improving RNA secondary structure prediction via state inference with deep recurrent neural networks. *Computational and Mathematical Biophysics* (2020); **8**:36–50, DOI: 10.1515/cmb-2020-0002.

[359] Wirecki TK, Merdas K, Bernat A, Boniecki MJ, Bujnicki JM & Stefaniak F. RNAProbe: a web server for normalization and analysis of RNA structure probing data. *Nucleic Acids Research* (2020); **48**:W292–9, DOI: 10.1093/nar/gkaa396.

[360] Wuchty S, Fontana W, Hofacker IL & Schuster P. Complete suboptimal folding of RNA and the stability of secondary structures. *Biopolymers* (1999); **49**:145–65, DOI: https://doi.org/10.1002/(SICI)1097-0282(199902)49:2<145::AID-BIP4>3.0.CO;2-G.

[361] Xayaphoummine A, Bucher T, Thalmann F & Isambert H. KineFold: Prediction and statistics of pseudoknots in RNA structures using exactly clustered stochastic simulations. *Proceedings of the National Academy of Sciences* (2003); **100**:15310–5, DOI: 10.1073/pnas.2536430100.

[362] Xia Z, Bell DR, Shi Y & Ren P. RNA 3D Structure Prediction by Using a Coarse-Grained Model and Experimental Data. *The Journal of Physical Chemistry B* (2013); **117**:3135–44, DOI: 10.1021/jp400751w.

[363] Xia Z, Gardner DP, Gutell RR & Ren P. CG: Coarse-Grained Model for Simulation of RNA Three-Dimensional Structures. *The Journal of Physical Chemistry B* (2010); **114**:13497–506, DOI: 10.1021/jp104926t.

[364] Xu J. Distance-based protein folding powered by deep learning. *Proceedings of the National Academy of Sciences* (2019); **116**:16856–65, DOI: 10.1073/pnas.1821309116.

[365] Xu X & Chen S-J. VfoldLA: Hierarchical Assembly of RNA Three-Dimensional Structures Based on Loop Templates. *The Journal of Physical Chemistry B* (2018); **122**:5327–35, DOI: 10.1021/acs.jpcb.7b10102.

[366] Yang C, Lim M, Kim E & Pak Y. Predicting RNA Structures via a Simple van der Waals Correction to an All-Atom Force Field. *Journal of Chemical Theory and Computation* (2017); **13**:395–9, DOI: 10.1021/acs.jctc.6b00808.

[367] Yang H, Jossinet F, Leontis N, Chen L, Westbrook J, Berman H & Westhof E. RNAView: Tools for the automatic identification and classification of RNA base pairs. *Nucleic Acids Research* (2003); **31**:3450–60, DOI: 10.1093/nar/gkg529.

[368] Yen C-Y, Lin J-C, Chen K-T & Lu CL. R3D-BLAST2: an improved search tool for similar RNA 3D substructures. *BMC Bioinformatics* (2017); **18**:574, DOI: 10.1186/s12859-017-1956-6.

[369] Yesselman JD *et al.* Computational design of three-dimensional RNA structure and function. *Nature Nanotechnology* (2019); **14**:866–73, DOI: 10.1038/s41565-019-0517-8.

[370] Yildirim I, Kennedy SD, Stern HA, Hart JM, Kierzek R & Turner DH. Revision of AMBER Torsional Parameters for RNA Improves Free Energy Predictions for Tetramer Duplexes with GC and iGiC Base Pairs. *Journal of Chemical Theory and Computation* (2012); **8**:172–81, DOI:

10.1021/ct200557r.

[371] Yildirim I, Stern HA, Kennedy SD, Tubbs JD & Turner DH. Reparameterization of RNA χ Torsion Parameters for the AMBER Force Field and Comparison to NMR Spectra for Cytidine and Uridine. *Journal of Chemical Theory and Computation* (2010); **6**:1520–31, DOI: 10.1021/ct900604a.

[372] Zahran M, Sevim Bayrak C, Elmetwaly S & Schlick T. RAG-3D: a search tool for RNA 3D substructures. *Nucleic Acids Research* (2015); **43**:9474–88, DOI: 10.1093/nar/gkv823.

[373] Zakov S, Goldberg Y, Elhadad M & Ziv-ukelson M. ContextFold: Rich Parameterization Improves RNA Structure Prediction. *Journal of Computational Biology* (2011); **18**:1525–42, DOI: 10.1089/cmb.2011.0184.

[374] Zerihun MB, Pucci F, Peter EK & Schug A. pydca v1.0: a comprehensive software for direct coupling analysis of RNA and protein sequences. *Bioinformatics* (2020); **36**:2264–5, DOI: 10.1093/bioinformatics/btz892.

[375] Zerihun MB, Pucci F & Schug A. CoCoNet: Boosting RNA contact prediction by convolutional neural networks. *bioRxiv* (2020):2020.07.30.229484, DOI: 10.1101/2020.07.30.229484.

[376] Zgarbová M, Otyepka M, Šponer J, Mládek A, Banáš P, Cheatham TE & Jurečka P. Refinement of the Cornell et al. Nucleic Acids Force Field Based on Reference Quantum Chemical Calculations of Glycosidic Torsion Profiles. *Journal of Chemical Theory and Computation* (2011); **7**:2886–902, DOI: 10.1021/ct200162x.

[377] Zhang D & Chen S-J. IsRNA: An Iterative Simulated Reference State Approach to Modeling Correlated Interactions in RNA Folding. *Journal of Chemical Theory and Computation* (2018); **14**:2230–9, DOI: 10.1021/acs.jctc.7b01228.

[378] Zhang D, Li J & Chen S-J. IsRNA1: De Novo Prediction and Blind Screening of RNA 3D Structures. *Journal of Chemical Theory and Computation* (2021); **17**:1842–57, DOI: 10.1021/acs.jctc.0c01148.

[379] Zhang H, Zhang C, Li Z, Li C, Wei X, Zhang B & Liu Y. CDPFold: A New Method of RNA Secondary Structure Prediction Based on Convolutional Neural Network and Dynamic Programming. *Frontiers in Genetics* (2019); **10**, DOI: 10.3389/fgene.2019.00467. Publié le 2019, Accessed 29 July 2020, https://www.frontiersin.org/articles/10.3389/fgene.2019.00467/full.

[380] Zhang H, Zhang L, Mathews DH & Huang L. LinearPartition: linear-time approximation of RNA folding partition function and base-pairing probabilities. *Bioinformatics* (2020); **36**:i258–67, DOI: 10.1093/bioinformatics/btaa460.

[381] Zhang Q & Li H. MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition. *IEEE Transactions on Evolutionary Computation* (2007); **11**:712–31, DOI: 10.1109/TEVC.2007.892759.

[382] Zhang X, Tian Y, Cheng R & Jin Y. A Decision Variable Clustering-Based Evolutionary Algorithm for Large-Scale Many-Objective Optimization. *IEEE Transactions on Evolutionary Computation* (2018); **22**:97–112, DOI: 10.1109/TEVC.2016.2600642.

[383] Zhao Y, Huang Y, Gong Z, Wang Y, Man J & Xiao Y. 3dRNA: Automated and fast building of three-dimensional RNA structures. *Scientific Reports* (2012); **2**:734, DOI: 10.1038/srep00734.

[384] Zhong C, Tang H & Zhang S. RNAMotifScan: automatic identification of RNA structural motifs using secondary structural alignment. *Nucleic Acids Research* (2010); **38**:e176–e176, DOI: 10.1093/nar/gkq672.

[385] Zhong C & Zhang S. RNAMotifScanX: a graph alignment approach for RNA structural motif identification. *RNA* (2015); **21**:333–46, DOI: 10.1261/rna.044891.114.

[386] Zhong C & Zhang S. RNAMSC: Clustering RNA structural motifs in ribosomal RNAs using secondary structural alignment. *Nucleic Acids Research* (2012); **40**:1307–17, DOI: 10.1093/nar/gkr804.

[387] Zhou H, Luo F, Luo Z, Li D, Liu C & Li X. Programming Conventional Electron Microscopes for Solving Ultrahigh-Resolution Structures of Small and Macro-Molecules. *Analytical Chemistry* (2019); **91**:10996–1003, DOI: 10.1021/acs.analchem.9b01162.

[388] Zirbel CL, Roll J, Sweeney BA, Petrov AI, Pirrung M & Leontis NB. Identifying novel sequence variants of RNA 3D motifs. *Nucleic Acids Research* (2015); **43**:7504–20, DOI: 10.1093/nar/gkv651.

[389] Zitzler E, Laumanns M & Thiele L. SPEA2: Improving the strength pareto evolutionary algorithm. *TIK-Report* (2001); **103**, DOI: 10.3929/ethz-a-004284029. Publié le 2001, Accessed 4 May 2021, https://www.research-collection.ethz.ch/handle/20.500.11850/145755.

[390] Zok T, Antczak M, Zurkowski M, Popenda M, Blazewicz J, Adamiak RW & Szachniuk M. RNApdbee 2.0: multifunctional tool for RNA structure annotation. *Nucleic Acids Research* (2018); **46**:W30–5, DOI: 10.1093/nar/gky314.

[391] Zok T, Badura J, Swat S, Figurski K, Popenda M & Antczak M. New models and algorithms for RNA pseudoknot order assignment. *International Journal of Applied Mathematics and Computer Science* (2020); **30**:315–24, DOI: 10.34768/amcs-2020-0024.

[392] Zuber J, Cabral BJ, McFadyen I, Mauger DM & Mathews DH. Analysis of RNA nearest neighbor parameters reveals interdependencies and quantifies the uncertainty in RNA secondary structure prediction. *RNA* (2018); **24**:1568–82, DOI: 10.1261/rna.065102.117.

[393] Zubradt M, Gupta P, Persad S, Lambowitz AM, Weissman JS & Rouskin S. DMS-MaPseq for genome-wide or targeted RNA structure probing in vivo. *Nature Methods* (2017); **14**:75–82, DOI: 10.1038/nmeth.4057.

[394] Zuker M. On finding all suboptimal foldings of an RNA molecule. *Science* (1989); **244**:48–52, DOI: 10.1126/science.2468181.

[395] Zuker M & Sankoff D. RNA secondary structures and their prediction. *Bulletin of Mathematical Biology* (1984); **46**:591–621, DOI: 10.1007/BF02459506.

[396] Zuker M & Stiegler P. Optimal computer folding of large RNA sequences using thermodynamics and auxiliary information. *Nucleic Acids Research* (1981); **9**:133–48, DOI: 10.1093/nar/9.1.133.

Annexe

1. Disponibilité des données et logiciels

Les outils et modèles présentés sont disponibles au téléchargement sur la plateforme EvryRNA sur le site du laboratoire IBISC, aux adresses :

https://evryrna.ibisc.univ-evry.fr/evryrna/biorseo (BiORSEO)
https://evryrna.ibisc.univ-evry.fr/evryrna/rnanet/rnanet_home (RNANet)

Le code source des logiciels est disponible sur la Forge GitLab d'IBISC :

https://forge.ibisc.univ-evry.fr/lbecquey/biorseo (BiORSEO)
https://forge.ibisc.univ-evry.fr/lbecquey/RNANet (RNANet)
https://forge.ibisc.univ-evry.fr/lbecquey/RGN (Réseau Géométrique Récurrent)
https://forge.ibisc.univ-evry.fr/lbecquey/rna-multi-optimizer (MOARNA)

Les codes sources de MOARNA et du RGN ne sont actuellement pas ouverts au public à l'extérieur d'IBISC, mais le seront probablement après leur publication dans un journal à comité de lecture.

2. Regroupement des outils de modélisation par groupe de recherche

Je propose une liste (toujours non exhaustive) des outils de l'état de l'art, organisés cette fois non plus par ordre chronologique, mais par groupe de recherche, ce qui permet de repérer plus vite les liens entre outils et modèles. Cette annexe est à utiliser en complément de la chronologie présentée en section I.4.2.

D'autres laboratoires d'intérêt existent, non inclus de façon détaillée dans la liste. On cite :

- Clote group (Etats Unis, Boston) : Travaux de biomathématiques aboutissant sur de nombreux outils. Pas toujours utiles en pratique, mais intéressants néanmoins, ou conduisant à des conclusions théoriques sur l'ARN intéressantes.
- Denise group : Développement de VARNA, des bases de données de modules Rna3Dmotifs, CaRNAval, CaRNAval 2. Développement des outils GARN et GARN2.
- Yann Ponty, qui collabore souvent avec d'autres groupes et auteurs, et dont de nombreuses publications non citées ici pourraient être d'intérêt : travaux d'algorithmique, travaux sur le design de séquences, l'échantillonnage de structures secondaires, (co)développeur d'IPANEMAP et VARNA.

Adamiak group (Pologne, RNApolis.pl)

2006 : 3DRNApredict + CYANA : Un pipeline de prédiction de structure 3D. Mfold est utilisé pour obtenir une structure secondaire. RNAview pour annoter des structures et préparer des fragments. Ceci prépare un fichier d'input pour CYANA contenant la séquence, une liste d'appariements, des angles de torsion initiaux, et d'autres informations. CYANA est un programme de « Torsion angle dynamics » qui va produire une structure 3D. La structure peut être raffinée par le programme XPLOR. Tout ce pipeline reste très peu documenté.
2008 : RNA FRABASE : Base de données de fragments et de SSE, recherche des fragments sur la base d'une séquence.
2010 : RNA FRABASE 2.0 inclut maintenant des annotations de 3DRNA et la possibilité de rechercher des modules ou propriétés 3D.
2010 (publié en 2012) : RNA Composer : Assemble des fragments venant de FRABASE 2.0 pré-minimisés individuellement dans CHARMM pour proposer un modèle 3D. Les fragments manquants sont générés par NAB et CYANA. Le raffinement final est fait par CYANA pour les angles de torsion et XPLOR (CHARMM) pour les coordonnées cartésiennes en général. Il n'y a pas d'échantillonnage de structures ni d'évaluation.
2013 : RNAlyzer : Compare des modèles 3D entre eux (typiquement utile pour comparer un vrai modèle à une prédiction).
2014 : RNApdbee : Webserver, annote et visualise des structures à l'aide d'outils externes.
2015 : RNAssess : Succède à RNAlyzer.
2018 : RNApdbee 2.0 : Supporte les interactions non canoniques dans les sorties. Supporte un nouvel algorithme pour annoter les pseudonoeuds.
2018 : RNAfitme : Remodélise les bases d'un ARN, à squelette fixé. Cela permet aussi de

générer un modèle all-atom à partir d'un modèle du squelette et d'une séquence au format FASTA. L'algorithme utilise des jeux de templates au choix, et finit par un raffinement dans CHARMM (par NAMD).
2019 : RNAvista : Prédit les interactions non-canoniques dans une structure secondaire à partir d'une structure secondaire canonique. Six outils de prédiction de structure secondaire sont proposés pour une première prédiction de structure canonique. Ensuite, RNA Composer prédit une structure 3D, ce qui a pour effet d'incorporer des interactions non-canoniques. Enfin, RNApdbee réannote la structure 3D produite en structure secondaire étendue.
2020 : Modèles de détermination de l'ordre d'un pseudonoeud.

Bujnicki group (Pologne, genesilico.pl)

2011 : ModeRNA : Permet de faire de la modélisation comparative de structure 3D. À partir d'un template 3D et d'un alignement des séquences du template et de l'ARN à modéliser, un modèle 3D peut être proposé en remplaçant les bases.
2013 : CompaRNA (obsolète) : Benchmark automatisé des méthodes de prédiction de structure secondaire.
2014 : ClaRNA : Annote des structures 3D utilisant les annotations de FR3D, MC-Annotate et RNA-View en interne et proposant un consensus.
2014 : RNA Bricks : Base de données de motifs sur base géométrique, incluant le contexte du motif s'il y a interaction (ions, protéines, ligands, autres ARN). Elle est continuellement mise à jour depuis.
2016 : SimRNA : Prédit des structures 3D par échantillonnage de Monte-Carlo de modèles gros grains évalués par une fonction entrainée sur un jeu de données. Un raffinement final est effectué.
2017 : SupeRNAlign : Superpose des structures d'ARN en 3D pour obtenir des alignements de séquences basés sur la structure 3D.
2018 : RNArchitecture : Propose des familles d'ARN sur la base de la forme des structures 3D.
2019 : QRNAS : Outil de raffinement des modèles effectuant une minimisation d'AMBER par descente de gradient, supportant les ions, ligands, structures hybrides, etc…
2019 : EvoClustRNA (x Das group) : Réalise des modèles 3D pour chaque séquence d'un alignement par SimRNA et retient la forme consensus.
2020 : RNAprobe : Normalise et analyse les données de sondage chimique.
2021 : RNAmasonry : Assemble des fragments 3D de SSE issus de RNA Bricks, et les score avec la fonction de SimRNA. L'outil supporte les contraintes externes (sondage chimique, contacts connus)

Das group (Etats-Unis, Stanford, CA)

2007 FARNA : Échantillonne des modèles 3D all-atom par une procédure de Monte-Carlo qui remplace successivement des fragments de longueur 3, guidé par une fonction gros-grains prenant en compte le rayon global de la molécule, les clashs atomiques, les

empilements et les appariements (paramètres knowledge-based). Le move-set consiste à choisir un intervalle de 3 nucléotides et à remplacer les torsions par celles observées dans un fragment de la structure 1FFK.
2010 FARFAR : Rajoute une minimisation finale à FARNA par la fonction d'énergie de Rosetta. La fonction gros-grains de FARNA possède maintenant des termes pour les interactions base-squelette et squelette-squelette. La procédure intègre maintenant 3 étapes de FARNA avec des fragments de taille 3, puis 2, puis 1 nucléotides avant le raffinement final.
2011 : Modélisation « pas à pas » de la structure des boucles seules.
2012 : Mise en place de la base de données RMDB pour les résultats d'expériences de sondage chimique.
2017 : Mise à jour de RMDB.
2017 : La fonction d'énergie de Rosetta subit une grosse mise à jour, elle mélange vraiment des termes physiques et des termes entrainés.
2018 : Proposition du move-set « appariement pas à pas », qui consiste à modéliser les boucles en tentant de forcer des appariements non canoniques pour « assembler » pas à pas la boucle.
2019 : EvoClustRNA (x Bujnicki lab) : Réalise des modèles 3D pour chaque séquence d'un alignement par FARFAR et retient la forme consensus.
2020 : FARFAR 2 : Mise à jour de la base de données de fragments sur la base des NR Lists de BGSU, et intégration du move-set pas à pas à l'algorithme Monte-Carlo.

Hofacker group (Autriche, ViennaRNA Group)

1994 : RNAfold (une ré-implémentation parallélisée et rapide de l'algorithme de Zuker-Stiegler), RNAinverse et RNAdistance forment le package ViennaRNA.
1999 : RNAsubopt : Énumère des structures sous-optimales à une certaine distance de la MFE.
2000 : Kinfold : Calcule des trajectoires de formation des structures secondaires par une simulation de type Monte-Carlo
2002 : RNAalifold : Prédit une structure secondaire consensus à partir d'un ensemble de séquences. Prend en compte la stabilité thermodynamique et la covariation.
2002 : Barriers : Calcule les minima locaux et les barrières énergétiques à partir d'une liste de conformations (structures secondaires) pour passer des unes aux autres. Produit des images du paysage énergétique sous forme d'arbre.
2004 : RNAduplex : Calcule une structure secondaire à partir de deux ARN, uniquement avec des appariements inter-ARN.
2004 : RNALfold : Prédit de structures secondaires locales dans les longs ARN.
2005 : RNAz : Prédit de structures à partir d'alignements en prenant en compte à la fois l'énergie et la covariation, mais en retournant un score séparé pour chaque (différent de RNAalifold).
2006 : RNAplfold : Prédit des probabilités d'appariements locaux dans les très longs ARN.
2006 : RNAcofold : Prédit de structures pour dimères d'ARN (appariements intra et inter moléculaires).

2008 : ViennaRNA devient un web-service.
2008 : RNAplex : Prédit l'interaction entre brins d'ARN sur la base d'une prédiction de structure secondaire entre ces deux brins, en autorisant seulement des appariements inter-moléculaires.
2008 : RNAalifold mis à jour, et RNALalifold : Prédit une structure consensus à partir de plusieurs séquences. Améliore la gestion des gaps et introduit des matrices de substitution.
2009 : RNA2DFold : Décrit le paysage énergétique et les structures secondaires voisines ainsi que leurs énergies, autour de structures secondaires de référence passées en entrée. Des visualisations 2D du paysage sont proposées.
2010 : RNAz 2.0 : Amélioration algorithmique et technique de RNAz.
2010 : RNAsnoop et RNAfoldz sont des programmes spécialisés.
2010 : RNAPKplex : Prédit des structures avec pseudonoeuds. Il considère une structure comme un « dimère interne », ce qui permet la formation de pseudonoeuds.
2011 : MC-Fold-DP : Réimplémente le modèle MC-Fold de Major sous forme de SCFG et avec un algorithme de programmation dynamique.
2011 : RNAwolf : Énumère des structures secondaires étendues à toutes les non-canoniques, avec support des appariements triples.
2011 : ViennaRNA 2.0 : Propose un ensemble complet d'outils rapides et parallèles (OpenMP), une documentation, une bibliothèque C appelée RNAlib, et des API Perl d'abord puis Python. Les outils proposent des ré-implémentations de la plupart des algorithmes déjà publiés (RNAFold, `RNAFold -p` = probabilités d'appariement, RNAsubopt, `RNAsubopt -p` = Sfold, `RNAsubopt -z` = Mfold). En pratique, ViennaRNA devient l'outil le plus fiable et robuste pour l'étude des structures secondaires. Le succès de ViennaRNA, et sa capacité à s'imposer comme outil de référence parmi de nombreux concurrents, sont principalement dus à sa facilité d'installation et aux talents d'ingénierie logicielle de Ronny Lorentz. Cette « success story » démontre la nécessité absolue pour un laboratoire de bio-informatique voulant gagner en réputation d'investir massivement dans le travail d'ingénierie et de documentation, qui fait défaut à de nombreux groupes, dont le mien. Quelques autres exemples (Rosetta à Stanford, ou la très bonne l'intégration entre eux des outils du BGSU group) mènent aussi à cette conclusion.
2014 : RNAlocmin : Effectue une descente de gradient à partir d'une structure secondaire pour l'optimiser. BHGBuilder permet de visualiser ce graphe.
2015 : Logiciel de visualisation FORNA.
2015 : Ernwin : Construit des structures 3D gros grains en assemblant des templates de SSE par une procédure de Monte-Carlo. Une fonction knowledge based les évalue.
2016 : BHGBuilder : Supporte maintenant les pseudonoeuds.
2016 : Les outils de ViennaRNA se dotent d'une « couche de support des contraintes » qui permettent de leur rajouter à tous des contraintes de compatibilité avec des résultats de probing chimique par exemple.
2016 : RNAlien : Pipeline qui construit automatiquement des familles d'ARN homologues et éventuellement d'alignements à partir d'une seule séquence. Il repose sur les outils d'Infernal, RNAz, et d'autres.
2017 : RNABlueprint : Permet de faire du design de séquences.
2019 : Forgi 2.0 : Permet la manipulation des SSE en Python (bibliothèque Python).

2020 : pourRNA : Remplace barriers (plus rapide).

Leontis & Zirbel group (Etats-Unis, BGSU, OH)

2008 : FR3D : Annote les structures 3D, et permet également de faire de la recherche de motifs (symbolique, géométrique, ou les deux).
2010 : R3D-Align : Superpose des structures 3D d'ARN sur la base d'alignements locaux.
2011 : WebFR3D : Web-service de recherche de modules en ligne dans les structures publiques.
2012 : Listes d'ARN non redondants.
2013 : R3D Align : Devient un web-service.
2013 : Le RNA 3D Motif Atlas est proposé.
2015 : R3D-2-MSA : R3D-Align peut proposer un alignement de séquences sur la base d'un alignement structural.
2015 : JAR3D : Modélise les distributions de probabilité des séquences pour chaque module du 3D Motif Atlas. Ceci permet de détecter, à partir de la séquence de chaque boucle d'un ARN, les modules les plus probables qu'elle pourrait former.
2015 : (x Hofacker lab) Recherche de modules dans des séquences génomiques entières (par metaRNAmodules et JAR3D) pour essayer d'aider à prédire la structure secondaire.
2016 : JAR3D : Devient un web-service
2020 : Le RNA Structure Atlas devient une base de données en ligne intégrant tous les outils du groupe et permettant la visualisation, annotation, et navigation facile.

Major group (Canada, Udem/IRIC Montréal, QC)

1991 : MC-SYM : Génère des structures 3D (all-atom) à partir de la séquence et d'un ensemble de contraintes sur la structure. Partant d'une dizaine de configurations « templates » de nucléotides (valeurs discrètes des angles de torsion) et un algorithme de satisfaction de contraintes de type branch-and-cut énumère les structures plausibles séquentiellement. Plusieurs branches sont maintenues en parallèle tant qu'elles ne violent pas les contraintes, et tous les templates sont essayés. Une minimisation est ensuite effectuée dans CHARMM, c'est elle qui évaluera les structures.
2001 : MC-Annotate : Annote des structures 3D.
2002 : Mise à jour de MC-Annotate avec un algorithme probabiliste.
2002 : MC-Search : Recherche des NCM dans des structures. Développé mais jamais publié.
2003 : MC-Sym : Améliore la génération avec une plus large collection de templates obtenus après annotation par le nouveau MC-Annotate. Il supporte l'entrée d'une structure secondaire étendue en input. Cette structure est découpée en cycles (NCM) qui sont modélisés individuellement par l'algorithme de satisfaction de contraintes, puis ré-assemblés. Un back-tracking de type Las-Vegas est implémenté (si un nœud de l'arbre est inconsistant avec les contraintes, on ne remonte pas de 1 mais d'un nombre aléatoire de nœuds). Ceci accélère la convergence.
2006 : Constitution d'une base de données de NCM.
2008 : MC-Fold : Le pipeline MC-Fold/MC-Sym permet d'obtenir des structures 3D par

assemblage de 15 classes de NCM. À partir de la séquence, le programme choisit tous les sites possibles d'initiation d'une HL (complexité $o(N^2)$) puis énumère pour chaque site en les empilant récursivement, tous les empilements possibles de NCM à partir de cette HL d'initiation (complexité en o(15N/2) puisqu'il y a 15 classes de NCMs). En comptant une contribution énergétique par NCM, MC-Fold peut renvoyer la structure de plus faible énergie. La fonction d'évaluation est une pseudo-énergie dépendante de la probabilité d'observation des caractéristiques d'une structure sachant la séquence (NCMs | séquence x jonctions | NCMs x appariements fermants | jonctions). Sa formulation dans le document supplémentaire accompagnant l'article reste cependant cryptique (à mes yeux). Des dispositions spécifiques modélisent les pseudonoeuds de type H et les empilements coaxiaux d'hélices.
2012 : RNA-MoIP (x Waldispühl lab) : Utilise une formulation ILP pour raffiner une ou plusieurs structures secondaires connues ou prédites pour un ARN. Il propose en sortie une structure secondaire dans laquelle sont insérés des modules 3D de RNA3dMotifs [101]. Le programme génère aussi un script de contraintes pour MC-Sym, ce qui aboutit à des prédictions 3D avec une bonne performance.
2015 : MC-FlashFold : Le modèle de MC-Fold est traduit dans une syntaxe de SCFG. Un algorithme de programmation dynamique énumère maintenant les assemblages de NCM.
2017 : RNA-MoIP : Devient un webservice.

Mathews group (Etats-Unis, Rochester, NY)

1981 : Mfold : Première implémentation du modèle du plus proche voisin et de l'algorithme de Zuker-Stiegler.
1984 : Mfold : Supporte les contraintes externes (contacts connus, probing).
1989 : Mfold : Peut renvoyer des solutions sous-optimales (suffisamment différentes).
1994 : Paramètres thermodynamiques pour les empilements coaxiaux.
1997-1999 : Plusieurs mises à jour de paramètres thermodynamiques.
1999 : Mfold : Amélioration technique.
1999 : RNAstructure : Alternative à Mfold pour Windows (mais sans les empilements coaxiaux ?).
2004 : RNAstructure (Fold) : Supporte les empilements coaxiaux, les nucléotides modifiés, et le calcul des probabilités est utilisé pour donner des intervalles de confiance sur chaque appariement.
2007 : Un algorithme d'optimisation permet d'apprendre les paramètres de Turner sur un jeu de données en plus d'expériences thermodynamiques.
2008 : UNAFold : Succède à Mfold et combine en un seul exécutable la prédiction d'une ou plusieurs structures secondaires, le calcul des probabilités d'appariement, et les calculs de température de dissociation des appariements.
2009 : MaxExpect : Utilise l'estimateur MEA.
2010 : La NNDB est lancée, une base de données de paramètres thermodynamiques, abandonnée aussitôt lancée (ce qui est bien dommage, l'idée était pertinente).
2010 : ProbKnot : Prétend identifier des structures avec pseudonoeuds sur la base de structures MEA sous-optimales. Mauvaise performance (~60% d'accuracy).

2010 : RNAstructure : Devient un « software package », comme ViennaRNA, et remplace UNAFold. Il contient les programmes Fold, MaxExpect, partition, efn2, AllSub, stochastic, RemovePseudoknots, ProbablePair, draw, NAPSS, DynAlign, OligoWalk, OligoScreen, Bifold, PARTS, Bipartition et DuplexFold.
2011 : Prédiction 3D all-atom par un recuit simulé (MC) dans AMBER-ff99 (AA, TB) en solvant implicite à partir d'une structure linéaire, soumis à des contraintes calculées à partir d'une prédiction de structure secondaire, DCA, prédiction d'empilements coaxiaux des hélices, et d'autres.
2011 : TurboFold : Propose une structure secondaire consensus pour plusieurs séquences, sur la base des probabilités d'appariements pour chaque séquence. Pas de support des pseudonoeuds.
2012 : TurboKnot : Infère des pseudonoeuds dans la structure consensus, à partir des structures MEA de l'ensemble des séquences.
2016 : ProbScan : Définit une fonction de partition des structures estimant la probabilité qu'une portion de séquence forme un élément de structure secondaire particulier.
2017 : TurboFold II : Utilise des HMM pour prédire une structure secondaire consensus sur la base de critères internes à chaque séquence (MEA/MFE) et externes (probabilités, covariation).
2017 : CycleFold : Similaire à MC-FlashFold, énumère des assemblages de NCM et renvoie des structures secondaires augmentées. Le programme supporte les « contraintes évolutives » en intégrant TurboFold.
2018 : « Design » : Permet de proposer une séquence dont l'ensemble de structures est proche d'une cible.
2018 : Rsample : Propose l'ensemble des structures probables à partir de données SHAPE. ShapeKnots est une version supportant les pseudonoeuds.
2019 : LinearFold : Utilise un schéma de programmation dynamique approché. La prédiction de structures secondaires sans pseudonoeuds est faisable en temps linéaire en gardant une bonne performance.
2020 : LinearPartition : Étend LinearFold pour renvoyer aussi les probabilités d'appariement.

… et aussi des outils d'alignement de séquences (deux à deux, ou multiples), de prédiction de structure secondaire consensus (deux à deux, ou multiples), des outils de design, et des contributions au champ de forces AMBER (en 3D). Des tentatives de traitement des boucles multiples plus élaborés ont été tentés mais concluant systématiquement que le modèle le plus simple est le meilleur.

Pyle group (Etats-Unis, Yale, CT)

1998 : AMIGOS : Calcule les angles de torsion eta/theta du squelette.
2003 : PRIMOS : Permet de rechercher un motif conformationnel dans un ensemble de structures.
2004 : COMPADRES : Automatise la recherche et découverte de nouveau motifs dans un jeu de données de structures.
2007 : Etude détaillée des clusters eta/theta.

2008 : Nomenclature des conformations du squelette du nucléotide.
2012 : Bibliothèque de nucléotides échantillonnés dans l'espace eta/theta.
2012 : RCrane : Plugin pour Coot déterminant les cartes de densité électronique pour faciliter la résolution des structures d'ARN par cristallographie aux rayons X.

… et aussi de nombreuses études approfondies des introns de groupe II, et d'ARN particuliers (ARNncl, Sars-Cov-2, …) ou de leurs mécanismes d'interaction.

Schlick group (Etats-Unis, New York, NY)

1988 : Algorithme pour générer des hélices d'ADN en 3D (formes A, B ou Z).
1994 : TNPACK pour CHARMM : Minimise l'énergie de structures dans CHARMM (par un algorithme de quasi-Newton).
2003 : Le modèle de RNA-As-Graphs est proposé, et la taille de l'espace conformationnel associé est estimée.
2004 : La base de données RAG en elle-même est lancée, Elle décrit toutes les topologies possibles des structures d'ARN de deux à sept SSE, et liste celles découvertes dans la nature, s'il y en a.
2007 : RAG-Pools : Design de populations d'ARN.
2011 : RNA Junction Explorer : Prédit l'empilement coaxial des hélices et la topologie des ML avec des Random-Forests.
2013 : RNAJAG : Propose un graphe RAG-3D en plus de la simple classe de topologie.
2014 : RAGTOP : Un potentiel statistique (KB) et un algo de recuit simulé (MC) sont proposés pour échantillonner des graphes 3D à partir d'une structure secondaire et de la prédiction de RNAJAG pour les ML. Ceci aboutit à une prédiction finale de topologie 3D (gros grains) de l'ARN.
2015 : RAG devient RAG-3D : Organise les hélices et boucles dans l'espace selon un graphe 3D. RAG-3D inclut un outil de recherche de sous structures via une recherche de sous-graphe dans les graphes à explorer.
2017 : F-RAG : Assemble des fragments. Transforme un graphe de topologie 3D en modèle all-atom. RAGTOP en profite et reconstitue désormais des modèles all-atom après la prédiction du graphe RAG-3D.
2018 : Le pipeline est adapté pour le design de séquences.

Tahi group (France, Evry Génopôle)

2002 : DCFold : Prédit des structures secondaires par approche comparative à partir de plusieurs séquences. Les hélices sont détectées par la détection de palindromes dans les séquences. L'algorithme retient les hélices conservées.
2003 : P-DCFold : Sur la même base que DC-Fold, mais les pseudonoeuds sont supportés et détectés.
2007 : SSCA : Choisit des séquences dans un ensemble de séquences proposées pour prédire une structure consensus.
2010 : T-Fold : Renvoie plusieurs solutions. Les hélices sont cherchées individuellement dans

les séquences par DC-Fold.
2012 : miRNAFold : Détecte les micro-ARN dans les génomes, par une méthode d'apprentissage supervisé.
2012 : ncRNAclassifier : Classifie les éléments transposables dans les micro-ARN.
2014 : piRPred : Classifie les ARN en Pi-ARN ou non, à l'aide de SVM à kernels multiples.
2014 : miRBoost : Une amélioration qui utilise des SVM boostés.
2016 : miRNAFold : Devient un webservice.
2017 : IpiRId : Classifie des ARN en Pi-ARN ou non, à partir de nombreuses caractéristiques.
2018 : Biokop : Prédit des structures secondaires par ILP (supporte donc les pseudonoeuds), et renvoie un ensemble de Pareto étendu entre les estimateurs MFE et MEA. Biokop peut aussi renvoyer des ensembles de Pareto sous-optimaux.
2018 : IRSOM & CRSOM : Classifient des séquences en ANR codant ou non codant, puis en familles d'ARN non codants. Ils sont basés sur les cartes auto-organisatrices.
2019 : BiORSEO : Outil de prédiction de structure secondaire basé sur l'ILP (et supportant les pseudonoeuds) utilisant la détection de modules dans les séquences comme critère supplémentaire en plus de MEA et renvoyant un ensemble de Pareto.
2019 : RCPred & C-RCPred : Prédisent la structure secondaire des complexes d'ARN.
2020 : RNANet : Jeu de données de structures 3D d'ARN réalignées avec les modèles de covariance de Rfam.

Waldispühl group (Canada, McGill Montréal, QC)

2012 : RNA-MoIP (x Major lab) : Utilise une formulation ILP pour raffiner une ou plusieurs structures secondaires connues ou prédites pour un ARN. Il propose en sortie une structure secondaire dans laquelle sont insérés des modules 3D de RNA3dMotifs[101]. Le programme génère aussi un script de contraintes pour MC-Sym, ce qui aboutit à des prédictions 3D avec une bonne performance.
2017 : RNA-MoIP : Devient un webservice.
2018 : CaRNAval : Détecte et organise une collection de réseaux d'interactions récurrents (les RIN), notamment avec des « modules d'interaction », c'est-à-dire des RIN faisant intervenir des interactions distantes entre deux SSE.
2019 : BayesPairing : Outil qui transforme les modules en réseaux bayésiens et utilise ces réseaux pour les rechercher dans les séquences.
2020 : BayesPairing 2 : Grosse amélioration de performance, la détection utilise maintenant un échantillon stochastique de structures secondaires de l'ARN cible, et éventuellement la détection à partir d'un alignement de séquences.
2020 : CaRNAval 2 : Extension des « modules d'interaction » à plus de deux SSE à la fois. Un changement d'annotation des graphes d'interactions et d'algorithme de recherche de sous-graphes améliore grandement la vitesse.
2020 : VeRNAl : Extension de CaRNAval à des réseaux d'interactions « flous » et non parfaitement isomorphes.

… et également des études sur le design, la ré-optimisation de structures, le calcul des effets des mutations ou modifications volontaires sur les paysages énergétiques, et plus

récemment le docking ligand-ARN.

Xiao group (Chine)

2012 : 3dRNA : Assemble des fragments à partir de templates all-atom d'éléments de structure secondaire. Pas d'échantillonnage ou évaluation, mais sélection des fragments sur leur identité de séquence. Si égalité, sélection de celui dont la séquence est le plus longtemps identique (dans le sens 5'->3'). Minimisation finale dans AMBER 98.
2015 : 3dRNAscore : Évalue les structures all-atom par une fonction KB basée sur les distances interatomiques et les 6 angles de torsion.
2017 : 3dRNA 2.0 : Prédit des structures 3D all-atom par recuit simulé (MC), les solutions étant évaluées par 3dRNAscore (KB) plus de l'analyse des covariations (DCA).
2020 : 2dRNA : Prédit des structures secondaires par un réseau de neurones LSTM couplé à un U-net.

Zhang group (Etats-Unis, Orlando, FL)

2010 : RNAMotifScan : Recherche des occurrences d'une sous-structure dans un ensemble de structures, en prenant en compte l'isostéricité et les contraintes symboliques des appariements non canoniques.
2012 : RNAMSC : Clusterise les motifs identifiés par RNAMotifScan.
2012 : RNASLOpt : Enumère des solutions sous optimales correspondant à des minima d'un modèle énergétique prenant en compte les empilements (et renvoie moins de solutions que RNAsubopt par exemple).
2015 : RNAMotifScanX : Rajoute la prise en compte des empilements dans la recherche de sous-structures et retourne aussi des solutions partielles.
2015 : STAR3D : Aligne des structures 3D sur la base des empilements.
2016 : WebSTAR3D : Un webservice pour STAR3D.
2017 : CompAnnotate : Annote des structures 3D en utilisant des structures homologues pour inférer les appariements plutôt qu'une seule structure 3D.
2018 : Base de données de modules RNAMotifClusters.
2020 : LocalSTAR3D : Permet des alignements locaux de structures d'ARN de taille différente
2021 : RNAMotifContrast : Les clusters ne sont plus obtenus sur la métrique de distance de RNAMotifScanX uniquement, mais prennent en compte une nouvelle mesure de distance, qui utilise elle-même l'alignement de séquences et le RMSD en 3D. Le jeu de données obtenu classe les motifs en quelques très grandes familles de boucles IL et HL.

université PARIS-SACLAY
ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Titre : Algorithmes multicritères pour la prédiction de structures d'ARN

Mots clés : Optimisation mathématique, Décision multicritère, ARN, Structure moléculaire, 3D, Bioinformatique

Résumé : Les méthodes informatiques de prédiction des structures d'ARN reposent sur deux étapes algorithmiques : proposer des structures (l'échantillonnage), et les trier par pertinence (l'évaluation). Une grande diversité de méthodes d'évaluation existe. Certaines reposent sur des modèles physiques, d'autres sur la similarité à des données déjà observées. Cette thèse propose des méthodes de prédiction de structure combinant deux ou plusieurs critères de tri des solutions, divers d'un point de vue de l'échelle de modélisation (structure secondaire, tertiaire), et du type (theory-based, data-based, compatibilité avec des données expérimentales de sondage chimique). Les méthodes proposées identifient le front de Pareto du problème d'optimisation multi-objectif formé par ces critères. Ceci permet d'identifier des solutions (structures) bien notées selon tous les modèles, et également d'étudier la corrélation entre critères. Les approches présentées exploitent les dernières avancées, comme l'identification de modules ou de réseaux d'interactions récurrents, ainsi que les algorithmes d'apprentissage profond. Deux architectures de réseaux de neurones (un RNN et un CNN) sont adaptées des protéines à l'ARN. Un jeu de données d'ARN est proposé pour entrainer ces réseaux : RNANet. Deux outils logiciels sont proposés : BiORSEO, qui prédit la structure secondaire des ARN sur la base de deux critères (l'un énergétique, l'autre relatif à la présence de modules connus). MOARNA, qui propose des structures 3D gros grains sur la base de 4 critères : l'énergie de la structure secondaire, l'énergie en 3D, la compatibilité avec des données expérimentales de sondage chimique, ou la forme d'une famille connue d'ARN si une famille est identifiée.

Title : Multicriteria algorithms for RNA structure prediction

Keywords : Mathematical optimization, Multicriteria decision, RNA, Molecular structure, 3D, Bioinformatics

Abstract : Computational RNA structure prediction methods rely on two major algorithmic steps : a sampling step to propose new structure solutions, and a scoring step to sort the solutions by relevance. A wide diversity of scoring methods exists. Some rely on physical models, some on the similarity to already observed data. This thesis proposes structure prediction methods combining two or more scoring criterions, diverse regarding the modelling scale (secondary structure, tertiary structure), their type (theory-based, knowledge-based, compatibility with experimental chemical probing results). The methods describe the Pareto front of the multi-objective optimization problem formed by these criteria. This allows to identify solutions (structures) well scored on each criterion, and to study the correlation between criterions. The presented approaches exploit the latest progress in the field, like the identification of modules or recurrent interaction networks, and the use of deep learning algorithms. Two neural network architectures (a RNN and a CNN) are adapted from proteins to RNA. A dataset is created to train these networks: RNANet. Two software tools are proposed: BiORSEO, which predicts the secondary structure based on two criterions (one relative to the structure's energy, the other relative to the presence of known modules). MOARNA, which predicts coarse-grained 3D structures based on four criterions: energy in 2D and 3D, compatibility with experimental probing results, and with the shape of a known RNA family if one has been identified.